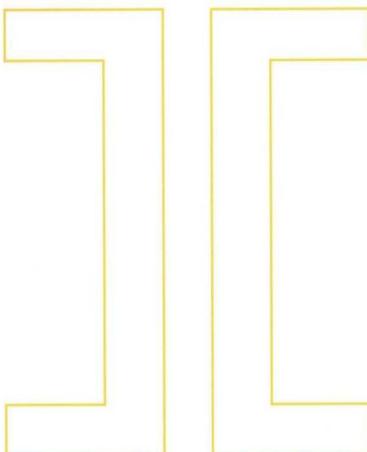


MATRIX COMPUTATIONS

4th Edition

Gene H. Golub
Charles F. Van Loan



Johns Hopkins Studies in the Mathematical Sciences
in association with the Department of Mathematical Sciences
The Johns Hopkins University

Department of Computer Science Stanford University

**I_kLx=0 y t 6 Hd: Hb
Department of Computer Science
Cornell University**

The Johns Hopkins University Press

Special discounts are available for bulk purchases of this book. For more information, please contact Special Sales at 410-516-6936 or specialsales@press.jhu.edu.

ka 1JnAslAaMnAS.apAM@AAAaAn.Splnuan0eVMan-Gll: ue0apSe\$A,mSni
pa...am0af @eAap@eOSAuAlAa- lBeO V@IOap.an@AO.lnA,uap10a3p,aaap
AlAMG.a.

T o

ALSTON S. HOUSEHOLDER

AND

JAMES H. WILKINSON

Preface	xii
Global References	xiii
Other Books	xv
Useful URLs	xix
Common Notation	x

J. $a \cdot bL^2 - a \cdot L \cdot bL$ = b

1.1	Basic Algorithms and Notation	2
1.2	Structure and Efficiency	14
1.3	Block Matrices and Algorithms	22
1.4	Fast Matrix-Vector Products	33
1.5	Vectorization and Locality	43
1.6	Parallel Matrix Multiplication	49

K. $\phi = \text{lap}(\text{adj}(F)) + hA$ $1b$

2.1	Basic Ideas from Linear Algebra	64
2.2	Vector Norms	68
2.3	Matrix Norms	71
2.4	The Singular Value Decomposition	76
2.5	Subspace Metrics	81
2.6	The Sensitivity of Square Systems	87
2.7	Finite Precision Matrix Computations	93

K. $Ab = F(A)b = A \cdot L + d \cdot A$ $.1b$

3.1	Triangular Systems	106
3.2	The LU Factorization	111
3.3	Roundoff Error in Gaussian Elimination	122
3.4	Pivoting	125
3.5	Improving and Estimating Accuracy	137
3.6	Parallel LU	144

O. $h \in \mathbb{R}$ $A \in \mathbb{R}^{n \times n}$ $\lambda \in \mathbb{C}$ $b \in \mathbb{R}^n$ $.1b$

4.1	Diagonal Dominance and Symmetry	154
4.2	Positive Definite Systems	159

4.3	Banded Systems	176
4.4	Symmetric Indefinite Systems	186
4.5	Block Tridiagonal Systems	196
4.6	Vandermonde Systems	203
4.7	Classical Methods for Toeplitz Systems	208
4.8	Gaussian and Discrete Poisson Systems	219

P Sesi Á B » NÁ uÁ B W p NY è hafi Y e p ...p

5.1	QR Factorization and Givens Transformations	234
5.2	The QR Factorization	246
5.3	The Full-Rank Least Squares Problem	260
5.4	Other Orthogonal Factorizations	274
5.5	The Rank-Deficient Least Squares Problem	288
5.6	Square and Underdetermined Systems	298

R Oß fA - ÁWYéphapí Y eTpB - N Y Ó E p Y i ½B f e p ...p

6.1	Least Squares and Regularization	304
6.2	Constrained Least Squares	313
6.3	Total Least Squares	320
6.4	Subspace Computations with the SVD	327
6.5	Updating Matrix Factorizations	334

S mÙ Ø Ø Y è ÁA »p Y N o T q B D Y Ø e p .12p

7.1	Properties and Decompositions	348
7.2	Perturbation Theory	357
7.3	Power Iterations	365
7.4	The Jacobi and Schur Forms	376
7.5	The Practical QR Algorithm	385
7.6	Invariant Subspace Computations	394
7.7	The Generalized Eigenvalue Problem	405
7.8	Hamiltonian and Product Eigenvalue Problems	420
7.9	Pseudospectra	426

T ho Ø Ø Y è ÁA »p Y N o T q B D Y Ø e p 14p

8.1	Properties and Decompositions	440
8.2	Power Iterations	450
8.3	The Symmetric QR Algorithm	458
8.4	More Methods for Tridiagonal Problems	467
8.5	Jacobi Methods	476
8.6	Computing the SVD	486
8.7	Generalized Eigenvalue Problems with Symmet	497

ak	sl - hA {A q} -=higA	411A
9.1	Eigenvalue Methods	514
9.2	Approximation Methods	522
9.3	The Matrix Exponential	530
9.4	The Sign Square and Log of a Matrix	536
L	Oè » Կպէէ ՎԲ Կ Ո Ո ի Թը - Օ Կ Ո է ի	111p
10.1	The Symmetric Lanczos Process	546
10.2	Lanczos Quadrature and Approximation	556
10.3	Practical Lanczos Procedures	562
10.4	Large Sparse SVD Frameworks	571
10.5	Krylov Methods for Unsymmetric Problems	579
10.6	Anti-Davidson and Related Methods	589
..	Oè » Կպէէ ՎԾ Կ հօքՎ Օ ի Տէ Բ - Օ Կ Ո է ի	142p
11.1	Direct Methods	598
11.2	The Classical Iterations	611
11.3	The Conjugate Gradient Method	625
11.4	Other Krylov Methods	639
11.5	Preconditioning	650
11.6	The Multigrid Framework	670
..	հաՎ յ գլոբԱ յ է ի	14p
12.1	Linear Systems with Displacement Structure	681
12.2	Structured Rank Problems	691
12.3	Kronecker Product Computations	707
12.4	Tensor Products and Contractions	719
12.5	Tensor Decompositions and Iterations	731
Index	747	

My thirty-year book collaboration with Gene Golub began in 1977 at a matrix computation workshop held at Johns Hopkins University. His interest in my work at the start of my academic career prompted the writing of GVL1. Sadly, Gene died on November 16, 2007. At the time we had only just begun to talk about GVL4. While writing these pages, I was reminded every day of his far-reaching impact and professional generosity. This edition is a way to thank Gene for our collaboration and the friendly research community that his unique personality helped create.

It has been sixteen years since the publication of the third edition – a power-of-two reminder that what we need to know about matrix computations is growing exponentially! Naturally, it is impossible to provide in-depth coverage of all the great new advances and research trends. However, with the relatively recent publication of so many excellent textbooks and specialized volumes, we are able to complement our brief treatments with useful pointers to the literature. That said, here are the new features of GVL4:

Content

The book is about twenty-five percent longer. There are new sections on fast transforms (§1.4), parallel LU (§3.6), fast methods for circulant systems and discrete Poisson systems (§4.8), Hamiltonian and product eigenvalue problems (§7.8), pseudospectra (§7.9), the matrix sign, square root, and logarithm functions (§9.4), Lanczos and quadrature (§10.2), large-scale SVD (§10.4), Jacobi-Davidson (§10.6), sparse direct methods (§11.1), multigrid (§11.6), low-displacement rank systems (§12.1), structured-rank systems (§12.2), Kronecker product problems (§12.3), tensor contractions (§12.4), and tensor decompositions (§12.5).

New topics at the subsection level include recursive block LU (§3.2.11), rock pivoting (§3.4.7), tournament pivoting (§3.6.3), diagonal dominance (§4.1.1), recursive block structures (§4.2.10), bandmatrix inverse properties (§4.3.8), divide-and-conquer strategies for block tridiagonal systems (§4.5.4), the cross product and various point/plane least squares problems (§5.3.9), the polynomial eigenvalue problem (§7.7.9), and the structured quadratic eigenvalue problem (§8.7.9).

Substantial upgrades include our treatment of floating-point arithmetic (§2.7), LU roundoff error analysis (§3.3.1), LS sensitivity analysis (§5.3.6), the generalized singular value decomposition (§6.1.6 and §8.7.4), and the CS decomposition (§8.7.6).

References

The annotated bibliographies at the end of each section remain. Because of space limitations, the master bibliography that was included in previous editions is now available through the book website. References that are historically important have been retained because old ideas have a way of resurfacing themselves. Plus, we must never forget the 1950s and 1960s! As mentioned above, we have the luxury of

being able to draw upon an expanding library of books on matrix computations. A mnemonic-based citation system has been incorporated that supports these connections to the literature.

Examples

Non-illuminating small-*n* numerical examples have been removed from the text. In their place is a modest suite of MATLAB demo scripts that can be run to provide insight into critical theorems and algorithms. We believe that this is a much more effective way to build intuition. The scripts are available through the book website.

Algorithmic Detail

It is important to have an algorithmic sense and an appreciation for high-performance matrix computations. After all, it is the clever exploitation of advanced architectures that account for much of the field's soaring success. However, the algorithms that we "formally" present in the book must never be considered as even prototype implementations. Clarity and communication of the big picture are what determine the level of detail in our presentations. Even though specific strategies for specific machines are beyond the scope of the text, we hope that our style promotes an ability to reason about memory traffic overheads and the importance of data locality.

Acknowledgements

I would like to thank everybody who has passed along typographical errors and suggestions over the years. Special kudos to the Cornell students in CS 4220, CS 6210, and CS 6220 where I used preliminary versions of GVL4. Harry Terkelson earned big bucks through my ill-conceived \$5-per-typo program!

A number of colleagues and students provided feedback and encouragement during the writing process. Others provided inspiration through their research and books. Thank you all: Diego Accame, David Bindel, Åke Björck, Laura Bolzan, Jim Demmel, Jack Dongarra, Mark Embree, John Gilbert, David Gleich, Joseph Grcar, Anne Greenbaum, Nick Higham, Ilse Ipsen, Bo Kagstrom, Vel Kahan, Tammy Kolda, Amy Langville, Julian Langou, Lek-Heng Lim, Nicola Mastronardi, Steve McCormick, Mike McCourt, Volker Mehrmann, Cleve Moler, Diane O'Leary, Michael Overton, Chris Paige, Beresford Parlett, Stefan Ragnarsson, Lothar Reichel, Yousef Saad, Mike Saunders, Rob Schreiber, Danny Sorensen, Pete Stewart, Gil Strang, Francoise Tisseur, Nick Trefethen, Raf Vandebril, and Jianlin Xia.

Chris Paige and Mike Saunders were especially helpful with the editing of Chapters 10 and 11.

Vincent Burke, Jennifer Mallet, and Juliana McCarthy at Johns Hopkins University Press provided excellent support during the production process. Jennifer Slater did a terrific job of copy-editing. Of course, I alone am responsible for all mistakes and oversights.

Finally, this book would have been impossible to produce without my great family and my 4AM writing companion Henry the Cat!

Charles N. Van Loan
Ithaca, New York
July, 2012

A number of books provide broad coverage of the field and are cited multiple times. We identify these global references using mnemonics. Bibliographic details are given in the Other Books section that follows.

AEP	Wilkinson: <i>Algebraic Eigenvalue Problem</i>
ANLA	Demmel: <i>Applied Numerical Linear Algebra</i>
ASNA	Higham: <i>Accuracy and Stability of Numerical Algorithms</i> , second edition
EOM	Chatelin: <i>Eigenvalues of Matrices</i>
FFT	Van Loan: <i>Computational Frameworks for the Fast Fourier Transform</i>
FOM	Higham: <i>Functions of Matrices</i>
FMC	Watkins: <i>Fundamentals of Matrix Computations</i>
IMC	Stewart: <i>Introduction to Matrix Computations</i>
IMK	van der Vorst: <i>Iterative Krylov Methods for Large Linear Systems</i>
IMSL	Greenbaum: <i>Iterative Methods for Solving Linear Systems</i>
ISM	Axelsson: <i>Iterative Solution Methods</i>
IMSL	Saad: <i>Iterative Methods for Sparse Linear Systems</i> , second edition
LCG	Meurant: <i>The Lanczos and Conjugate Gradient Algorithms ...</i>
MA	Horn and Johnson: <i>Matrix Analysis</i>
MABD	Stewart: <i>Matrix Algorithms: Basic Decompositions</i>
MAE	Stewart: <i>Matrix Algorithms Volume II: Eigensystems</i>
MEP	Watkins: <i>The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods</i>
MPT	Stewart and Sun: <i>Matrix Perturbation Theory</i>
NLA	Trefethen and Bau: <i>Numerical Linear Algebra</i>
NMA	Ipsen: <i>Numerical Matrix Analysis: Linear Systems and Least Squares</i>
NMLE	Saad: <i>Numerical Methods for Large Eigenvalue Problems</i> , revised edition
NMLS	Bjorck: <i>Numerical Methods for Least Squares Problems</i>
NMSE	Kressner: <i>Numerical Methods for General and Structured Eigenvalue Problems</i>
SAP	Trefethen and Embree: <i>Spectra and Pseudospectra</i>
SEP	Parlett: <i>The Symmetric Eigenvalue Problem</i>
SLAS	Forsythe and Moler: <i>Computer Solution of Linear Algebraic Systems</i>
SLS	Lawson and Hanson: <i>Solving Least Squares Problems</i>
TMA	Horn and Johnson: <i>Topics in Matrix Analysis</i>

LAPACK *LAPACK Users' Guide*, third edition

E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra,
J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen.

scaLAPACK *ScaLAPACK Users' Guide*

L.S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon,
J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker,
and R. C. Whaley.

LIN_TEMPLATES *Templates for the Solution of Linear Systems ...*

R. Barrett, M.W. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout,
R. Pozo, C. Romine, and H. van der Vorst.

EIG_TEMPLATES *Templates for the Solution of Algebraic Eigenvalue Problems ...*

Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst.

The following volumes are a subset of a larger, ever-expanding library of textbooks and monographs that are concerned with matrix computations and supporting areas. The list of references below captures the evolution of the field and its breadth. Works that are more specialized are cited in the annotated bibliographies that appear at the end of each section in the chapters.

Early Landmarks

- V.N. Faddeeva (1959). *Computational Methods of Linear Algebra*, Dover, New York.
E. Bodewig (1959). *Matrix Calculus*, North-Holland, Amsterdam.
J.H. Wilkinson (1963). *Rounding Errors in Algebraic Processes*, Prentice-Hall, Englewood Cliffs, NJ.
A.S. Householder (1964). *Theory of Matrices in Numerical Analysis*, Blaisdell, New York.
Reprinted in 1974 by Dover, New York.
L. Fox (1964). *An Introduction to Numerical Linear Algebra*, Oxford University Press, Oxford.
J.H. Wilkinson (1965). *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford.

General Textbooks on Matrix Computations

- G.W. Stewart (1973). *Introduction to Matrix Computations*, Academic Press, New York.
R.J. Goult, R.F. Hoskins, J.A. Milner, and M.J. Pratt (1974). *Computational Methods in Linear Algebra*, John Wiley and Sons, New York.
W.W. Hager (1988). *Applied Numerical Linear Algebra*, Prentice-Hall, Englewood Cliffs, NJ.
P.G. Ciarlet (1989). *Introduction to Numerical Linear Algebra and Optimisation*, Cambridge University Press, Cambridge.
P.E. Gill, W. Murray, and M.H. Wright (1991). *Numerical Linear Algebra and Optimization*, Vol. 1, Addison-Wesley, Reading, MA.
A. Jennings and J.J. McKeown (1992). *Matrix Computation*, second edition, John Wiley and Sons, New York.
L.N. Trefethen and D. Bau III (1997). *Numerical Linear Algebra*, SIAM Publications, Philadelphia, PA.
J.W. Demmel (1997). *Applied Numerical Linear Algebra*, SIAM Publications, Philadelphia, PA.
A.J. Laub (2005). *Matrix Analysis for Scientists and Engineers*, SIAM Publications, Philadelphia, PA.
B.N. Datta (2010). *Numerical Linear Algebra and Applications*, second edition, SIAM Publications, Philadelphia, PA.
D.S. Watkins (2010). *Fundamentals of Matrix Computations*, John Wiley and Sons, New York.
A.J. Laub (2012). *Computational Matrix Analysis*, SIAM Publications, Philadelphia, PA.

Linear Equation and Least Squares Problems

- G.E. Forsythe and C.B. Moler (1967). *Computer Solution of Linear Algebraic Systems*, Prentice-Hall, Englewood Cliffs, NJ.
A. George and J.W.H. Liu (1981). *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, Englewood Cliffs, NJ.

- I.S. Duff, A.M. Erisman, and J.K. Reid (1986). *Direct Methods for Sparse Matrices*, Oxford University Press, New York.
- R.W. Farebrother (1987). *Linear Least Squares Computations*, Marcel Dekker, New York.
- C.L. Lawson and R.J. Hanson (1995). *Solving Least Squares Problems*, SIAM Publications, Philadelphia, PA.
- Björck (1996). *Numerical Methods for Least Squares Problems*, SIAM Publications, Philadelphia, PA.
- G.W. Stewart (1998). *Matrix Algorithms: Basic Decompositions*, SIAM Publications, Philadelphia, PA.
- N.J. Higham (2002). *Accuracy and Stability of Numerical Algorithms*, second edition, SIAM Publications, Philadelphia, PA.
- T.A. Davis (2006). *Direct Methods for Sparse Linear Systems*, SIAM Publications, Philadelphia, PA.
- I.C.F. Ipsen (2009). *Numerical Matrix Analysis: Linear Systems and Least Squares*, SIAM Publications, Philadelphia, PA.

Eigenvalue Problems

- A.R. Gourlay and G.A. Watson (1973). *Computational Methods for Matrix Eigenproblems*, John Wiley & Sons, New York.
- F. Chatelin (1993). *Eigenvalues of Matrices*, John Wiley & Sons, New York.
- B.N. Parlett (1998). *The Symmetric Eigenvalue Problem*, SIAM Publications, Philadelphia, PA.
- G.W. Stewart (2001). *Matrix Algorithms Volume I : Eigensystems*, SIAM Publications, Philadelphia, PA.
- L. Komzsik (2003). *The Lanczos Method: Evolution and Application*, SIAM Publications, Philadelphia, PA.
- D. Kressner (2005). *Numerical Methods for General and Structured Eigenvalue Problems*, Springer, Berlin.
- D.S. Watkins (2007). *The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods*, SIAM Publications, Philadelphia, PA.
- Y. Saad (2011). *Numerical Methods for Large Eigenvalue Problems*, revised edition, SIAM Publications, Philadelphia, PA.

Iterative Methods

- R.S. Varga (1962). *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ.
- D.M. Young (1971). *Iterative Solution of Large Linear Systems*, Academic Press, New York.
- L.A. Hageman and D.M. Young (1981). *Applied Iterative Methods*, Academic Press, New York.
- J. Cullum and R.A. Willoughby (1985). *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. I Theory*, Birkhäuser, Boston.
- J. Cullum and R.A. Willoughby (1985). *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. I Programs*, Birkhäuser, Boston.
- W. Hackbusch (1994). *Iterative Solution of Large Sparse Systems of Equations*, Springer-Verlag, New York.
- O. Axelsson (1994). *Iterative Solution Methods*, Cambridge University Press.
- A. Greenbaum (1997). *Iterative Methods for Solving Linear Systems*, SIAM Publications, Philadelphia, PA.
- Y. Saad (2003). *Iterative Methods for Sparse Linear Systems*, second edition, SIAM Publications, Philadelphia, PA.
- H. van der Vorst (2003). *Iterative Krylov Methods for Large Linear Systems*, Cambridge University Press, Cambridge, UK.

G. Meurant (2006). *The Lanczos and Conjugate Gradient Algorithms: From Theory to Finite Precision Computations*, SIAM Publications, Philadelphia, PA.

Special Topics/Threads

- L.N. Trefethen and M. Embree (2005). *Spectra and Pseudospectra—The Behavior of Nonnormal Matrices and Operators*, Princeton University Press, Princeton and Oxford.
- R. Vandebril, M. Van Barel, and N. Mastronardi (2007). *Matrix Computations and Semiseparable Matrices I: Linear Systems*, Johns Hopkins University Press, Baltimore, MD.
- R. Vandebril, M. Van Barel, and N. Mastronardi (2008). *Matrix Computations and Semiseparable Matrices I : Eigenvalue and Singular Value Methods*, Johns Hopkins University Press, Baltimore, MD.
- N.J. Higham (2008) *Functions of Matrices*, SIAM Publications, Philadelphia, PA.

Collected Works

- R.H. Chan, C. Greif, and D.P. O’Leary, eds. (2007). *Milestones in Matrix Computation: Selected Works of G.H. Golub, with Commentaries*, Oxford University Press, Oxford.
- M.E. Kilmer and D.P. O’Leary, eds. (2010). *Selected Works of G.W. Stewart*, Birkhauser, Boston, MA.

Implementation

- B.T. Smith, J.M. Boyle, Y. Ikebe, V.C. Klema, and C.B. Moler (1970). *Matrix Eigensystem Routines: EISPACK Guide*, second edition, Lecture Notes in Computer Science, Vol. 6, Springer-Verlag, New York.
- J.H. Wilkinson and C. Reinsch, eds. (1971). *Handbook for Automatic Computation, Vol. 2. Linear Algebra*, Springer-Verlag, New York.
- B.S. Garbow, J.M. Boyle, J.J. Dongarra, and C.B. Moler (1972). *Matrix Eigensystem Routines: EISPACK Guide Extension*, Lecture Notes in Computer Science, Vol. 51, Springer-Verlag, New York.
- J.J. Dongarra, J.R. Bunch, C.B. Moler, and G.W. Stewart (1979). *LINPACK Users’ Guide*, SIAM Publications, Philadelphia, PA.
- K. Gallivan, M. Heath, E. Ng, B. Peyton, R. Plemmons, J. Ortega, C. Romine, A. Sameh, and R. Voigt (1990). *Parallel Algorithms for Matrix Computations*, SIAM Publications, Philadelphia, PA.
- R. Barrett, M.W. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst (1993). *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM Publications, Philadelphia, PA.
- L.S. Blackford, J. Choi, A. Cleary, E. D’Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, and R.C. Whaley (1997). *ScaLAPACK Users’ Guide*, SIAM Publications, Philadelphia, PA.
- J.J. Dongarra, I.S. Duff, D.C. Sorensen, and H.A. van der Vorst (1998). *Numerical Linear Algebra on High-Performance Computers*, SIAM Publications, Philadelphia, PA.
- E. Anderson, z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen (1999). *LAPACK Users’ Guide*, third edition, SIAM Publications, Philadelphia, PA.
- z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst (2000). *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, SIAM Publications, Philadelphia, PA.
- V.A. Barker, L.S. Blackford, J. Dongarra, J. Du Croz, S. Hammarling, M. Marinova, J. Wasniewski, and P. Yalamov (2001). *LAPACK95 Users’ Guide*, SIAM Publications, Philadelphia.

ATLAB

- D.J. Higham and N.J. Higham (2005). *MATLAB Guide*, second edition, SIAM Publications, Philadelphia, PA.
- R. Pratap (2006). *Getting Started with Matlab 7*, Oxford University Press, New York.
- C.F. Van Loan and D. Fan (2009). *Insight Through Computing: A Matlab Introduction to Computational Science and Engineering*, SIAM Publications, Philadelphia, PA.

Matrix Algebra and Analysis

- R. Horn and C. Johnson (1985). *Matrix Analysis*, Cambridge University Press, New York.
- G.W. Stewart and J. Sun (1990). *Matrix Perturbation Theory*, Academic Press, San Diego.
- R. Horn and C. Johnson (1991). *Topics in Matrix Analysis*, Cambridge University Press, New York.
- D.S. Bernstein (2005). *Matrix Mathematics, Theory, Facts, and Formulas with Application to Linear Systems Theory*, Princeton University Press, Princeton, NJ.
- L. Hogben (2006). *Handbook of Linear Algebra*, Chapman and Hall, Boca Raton, FL.

Scientific Computing/Numerical Analysis

- G.W. Stewart (1996). *Afternotes on Numerical Analysis*, SIAM Publications, Philadelphia, PA.
- C.F. Van Loan (1997). *Introduction to Scientific Computing: A Matrix-Vector Approach Using Matlab*, Prentice Hall, Upper Saddle River, NJ.
- G.W. Stewart (1998). *Afternotes on Numerical Analysis: Afternotes Goes to Graduate School*, SIAM Publications, Philadelphia, PA.
- M.T. Heath (2002). *Scientific Computing: An Introductory Survey*, second edition, McGraw-Hill, New York.
- C.B. Moler (2008) *Numerical Computing with MATLAB*, revised reprint, SIAM Publications, Philadelphia, PA.
- G. Dahlquist and Björck (2008). *Numerical Methods in Scientific Computing*, Vol. 1, SIAM Publications, Philadelphia, PA.
- U. Ascher and C. Greif (2011). *A First Course in Numerical Methods*, SIAM Publications, Philadelphia, PA.

GVL4

MATLAB demoscripts and functions, matrix bibliography, list of errata

<http://www.cornell.edu/cv/GVL4>

Netlib

Huge repository of numerical software including LAPACK.

<http://www.netlib.org/index.html>

Matrix Market

Test examples for matrix algorithms

<http://math.nist.gov/MatrixMarket/>

Matlab Central

Matlab functions, demos, datasets, toolboxes, videos

<http://www.mathworks.com/matlabcentral/>

University of Florida Sparse Matrix Collections

Thousands of sparse matrix examples in several formats

<http://www.cise.ufl.edu/research/sparse/matrices/>

Pseudospectra Gateway

Graphical tools for pseudospectra

<http://www.cs.ox.ac.uk/projects/pseudospectra/>

ARPACK

Software for large sparse eigenvalue problems

<http://www.caam.rice.edu/software/ARPACK/>

Innovative Computing Laboratory

State-of-the-art high performance matrix computations

<http://icl.cs.utk.edu/>

\mathbf{R} , \mathbf{R}^n , $\mathbf{R}^{m \times n}$	set of real numbers, vectors, and matrices (p. 2)
\mathbb{C}^n , $\mathbb{C}^{m \times n}$	set of complex numbers, vectors, and matrices (p. 13)
a_{ij} , $A(i,j)$, $[A]_{ij}$	(i,j) entry of a matrix (p. 2)
u	unit roundoff (p. 96)
$\text{fl}(.)$	floating point operator (p. 96)
$\ x\ _p$	p -norm of a vector (p. 68)
$\ A\ _p$, $\ A\ _F$	p -norm and Frobenius norm of a matrix (p. 71)
$\text{length}(x)$	dimension of a vector (p. 236)
$\kappa_p(A)$	p -norm condition (p. 87)
$ A $	absolute value of a matrix (p. 91)
A^T , A^H	transpose and conjugate transpose (p. 2, 13)
$\text{house}(x)$	Householder vector (p. 236)
$\text{givens}(a, b)$	cosine-sine pair (p. 240)
x_{LS}	minimum-norm least squares solution (p. 260)
$\text{ran}(A)$	range of a matrix (p. 64)
$\text{null}(A)$	nullspace of a matrix (p. 64)
$\text{span}\{v_1, \dots, v_n\}$	span defined by vectors (p. 64)
$\dim(S)$	dimension of a subspace (p. 64)
$\text{rank}(A)$	rank of a matrix (p. 65)
$\det(A)$	determinant of a matrix (p. 66)
$\text{tr}(A)$	trace of a matrix (p. 327)
$\text{vec}(A)$	vectorization of a matrix (p. 28)
$\text{reshape}(A, p, q)$	reshaping a matrix (p. 28)
$\text{Re}(A)$, $\text{Im}(A)$	real and imaginary parts of a matrix (p. 13)
$\text{diag}(d_1, \dots, d_n)$	diagonal matrix (p. 18)
I_n	n -by- n identity matrix (p. 19)
e_i	i th column of the identity matrix (p. 19)
\mathcal{E}_n , \mathcal{D}_n , Ppq	exchange, downshift, and perfect shuffle permutations (p. 20)
$\sigma_i(A)$	i th largest singular value (p. 77)
$\sigma_{\max}(A)$, $\sigma_{\min}(A)$	largest and smallest singular value (p. 77)
$\text{dist}(S_1, S_2)$	distance between two subspaces (p. 82)
$\text{sep}(A_1, A_2)$	separation between two matrices (p. 360)
$\lambda(A)$	set of eigenvalues (p. 66)
$\lambda_i(A)$	i th largest eigenvalue of a symmetric matrix (p. 66)
$\lambda_{\max}(A)$, $\lambda_{\min}(A)$	largest and smallest eigenvalue of a symmetric matrix (p. 66)
$\rho(A)$	spectral radius (p. 349)
$J(A, q, j)$	Krylov subspace (p. 548)

Chapter 1

Matrix Multiplication

ePe

ePl

ePn

eB

L

ePr

ePs

The study of matrix computations properly begins with the study of various matrix multiplication problems. Although simple mathematically, these calculations are sufficiently rich to develop a wide range of essential algorithmic skills.

In §1.1 we examine several formulations of the matrix multiplication update problem $C = C + AB$. Partitioned matrices are introduced and used to identify linear algebraic "levels" of computation.

If a matrix has special properties, then various economies are generally possible. For example, a symmetric matrix can be stored in half the space of a general matrix. A matrix-vector product may require much less time to execute if the matrix has many zero entries. These matters are considered in §1.2.

A block matrix is a matrix whose entries are themselves matrices. The "language" of block matrices is developed in §1.3. It supports the easy derivation of matrix factorizations by enabling us to spot patterns in a computation that are obscured at the scalar level. Algorithms phrased at the block level are typically rich in matrix-matrix multiplication, the operation of choice in many high performance computing environments. Sometimes the block structure of a matrix is recursive, meaning that the block entries have an exploitable resemblance to the overall matrix. This type of connection is the foundation for "fast" matrix-vector product algorithms such as various fast Fourier transforms, trigonometric transforms, and wavelet transforms. These calculations are among the most important in all of scientific computing and are discussed in §1.4. They provide an excellent opportunity to develop a facility with block matrices and recursion.

The first two sections set the stage for effective, "large" matrix computations. In this context, data locality affects efficiency more than the volume of actual arithmetic. Having an ability to reason about memory hierarchies and multiprocessor computation is essential. Our goal in §1.5 and §1.6 is to build an appreciation for the attendant issues without getting into system-dependent details.

Reading Notes

The sections within this chapter depend upon each other as follows:

$$\begin{array}{cccccc} \$1.1 & \dots & \$1.2 & \dots & \$1.3 & \dots & \$1.4 \\ & & & & & \downarrow & \\ & & & & \$1.5 & \dots & \$1.6 \end{array}$$

Before proceeding to later chapters, §1.1, §1.2, and §1.3 are essential. The fast transforms in §1.4 are utilized in §4.8 and parts of Chapters 11 and 12. The reading of §1.5 and §1.6 can be deferred until high-performance linear equation solving or eigenvalue computation becomes a topic of concern.

1.1 Basic Algorithms and Notation

Matrix computations are built upon a hierarchy of linear algebraic operations. Dot products involve the scalar operations of addition and multiplication. Matrix-vector multiplication is made up of dot products. Matrix-matrix multiplication amounts to a collection of matrix-vector products. All of these operations can be described in algorithmic form or in the language of linear algebra. One of our goals is to show how these two styles of expression complement each other. Along the way we pick up notation and acquaint the reader with the kind of thinking that underpins the matrix computation area. The discussion revolves around the matrix multiplication problem, a computation that can be organized in several ways.

hisihb ht f .1bkk f tf 1kb

Let \mathbb{R} designate the set of real numbers. We denote the vector space of all m -by- n real matrices by $\mathbb{R}^{m \times n}$:

$$A \in \mathbb{R}^{m \times n} \iff A = (a_{ij}) = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}, \quad a_{ij} \in \mathbb{R}$$

If a capital letter is used to denote a matrix (eg, A , B , A), then the corresponding lower case letter with subscript ij refers to the (i, j) entry (eg, a_{ij} , b_{ij} , δ_{ij}). Sometimes we designate the elements of a matrix with the notation $[A]_{ij}$ or $A(i, j)$.

.,., I i. j .

Basic matrix operations include transposition ($\mathbf{R}^{m \times n} \rightarrow \mathbf{R}^{n \times m}$).

$$C = A^T \implies c_{ij} = a_{ji},$$

addition ($\mathbb{R}^{m \times n} \times \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{m \times n}$),

$$\mathbf{C} = \mathbf{x} + \mathbf{B} \quad \Rightarrow \quad c_{ij} = a_{ij} + b_{ij},$$

scalar-matrix multiplication ($\mathbb{R} \times \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{m \times n}$),

$$\mathbf{C} = \alpha \mathbf{A} \quad \Rightarrow \quad c_{ij} = \alpha a_{ij},$$

and matrix-matrix multiplication ($\mathbb{R}^{m \times p} \times \mathbb{R}^{p \times n} \rightarrow \mathbb{R}^{m \times n}$),

$$\mathbf{C} = \mathbf{AB} \quad \Rightarrow \quad c_{ij} = \sum_{k=1}^p a_{ik} b_{kj}.$$

Pointwise matrix operations are occasionally useful, especially pointwise multiplication ($\mathbb{R}^{m \times n} \times \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{m \times n}$),

$$\mathbf{C} = \mathbf{A}.*\mathbf{B} \quad \Rightarrow \quad c_{ij} = a_{ij}b_{ij}$$

and pointwise division ($\mathbb{R}^{m \times n} \times \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{m \times n}$),

$$\mathbf{C} = \mathbf{A}./\mathbf{B} \quad \Rightarrow \quad c_{ij} = a_{ij}/b_{ij}.$$

Of course, for pointwise division to make sense, the "denominator matrix" must have nonzero entries.

hphpb -Sff kb kk f tf sktb

Let \mathbb{R}^n denote the vector space of real n -vectors

$$x \in \mathbb{R}^n \quad \Leftrightarrow \quad \mathbf{X} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \quad x_i \in \mathbb{R}.$$

We refer to x_i as the i th component of x . Depending upon context, the alternative notations $[x]_i$ and $x(i)$ are sometimes used.

Notice that we are identifying \mathbb{R}^n with $\mathbb{R}^{n \times 1}$ and so the members of \mathbb{R}^n are column vectors. On the other hand, the elements of $\mathbb{R}^{1 \times n}$ are row vectors.

$$x \in \mathbb{R}^{1 \times n} \quad \Leftrightarrow \quad x = [x_1, \dots, x_n].$$

If x is a column vector, then $y = x^T$ is a rowvector.

hphpb -Sff kb 6E Sff sktb

Assume that $\mathbf{A} \in \mathbb{R}^{n \times m}$ and $\mathbf{y} \in \mathbb{R}^n$. Basic vector operations include scalar-vector multiplication,

$$\mathbf{z} = \mathbf{ax} \quad \Rightarrow \quad z_i = ax_i,$$

vector addition

$$\mathbf{z} = \mathbf{x} + \mathbf{y} \quad \Rightarrow \quad z_i = x_i + y_i,$$

and the inner product (or dot product),

$$c = \mathbf{x}^T \mathbf{y} \quad \Rightarrow \quad c = \sum_{i=1}^n x_i y_i.$$

A particularly important operation, which we write in update form, is the **saxy**.

$$\mathbf{y} = \mathbf{a}\mathbf{x} + \mathbf{y} \quad = \quad y_i = a_i x_i + y_i$$

Here, the symbol " $=$ " is used to denote a **sigmoid**, not mathematical equality. The vector \mathbf{y} is being updated. The name "saxy" is used in LAPACK, a software package that implements many of the algorithms in this book. "Saxy" is a mnemonic for "scalar $\mathbf{a}\mathbf{x}$ plus \mathbf{y} ." See LAPACK.

Pointwise vector operations are also useful, including vector multiplication,

$$\mathbf{z} = \mathbf{x} \cdot * \mathbf{y} \quad \Rightarrow \quad z_i = x_i y_i,$$

and vector division,

$$\mathbf{z} = \mathbf{x} / \mathbf{y} \quad \Rightarrow \quad z_i = x_i / y_i.$$

manah X I84X v×|+ fo1fo=xUX X ×-X I× foX=5 ×re|i foΣ1UXreX p11+ AΣX

Algorithms in the text are expressed using a stylized version of the MATLAB language. Here is our first example:

s,yAuTp2 aeEaIstyI2 sAysSh2 If $\mathbf{x}, \mathbf{y} \in \mathbb{J}^n$, then this algorithm computes their dot product $c = \mathbf{x}^T \mathbf{y}$.

```
c=0
for i=1:n
    c=c+x(i)y(i)
end
```

It is clear from the summation that the dot product of two vectors involves n multiplications and n additions. The dot product operation is an " $O(n)$ " operation, meaning that the amount of work scales linearly with the dimension. The saxy computation is also $O(n)$:

shyfutp2 mca t. T6Fo2 If $\mathbf{x}, \mathbf{y} \in \mathbb{J}^n$ and $a \in \mathbb{J}$, then this algorithm overwrites \mathbf{y} with $\mathbf{y} + a\mathbf{x}$.

```
for i=1:n
    y(i)=y(i)+a*x(i)
end
```

We stress that the algorithms in this book are encapsulations of important computational ideas and are not to be regarded as "production codes."

1c2hkA q - =hp) o.=A qN F-HBii)-hA }IA -EoA4} pVA

Suppose $A \in \mathbb{R}^{m \times n}$ and that we wish to compute the update

$$\mathbf{y} = \mathbf{y} + \mathbf{A}\mathbf{x}$$

where $\mathbf{E}\mathbf{R}^n$ and $\mathbf{E}\mathbf{R}^m$ are given. This generalized say operation is referred to as a gag. A standard way that this computation proceeds is to update the components one-at-a-time.

$$y_i = y_i + \sum_{j=1}^n a_{ij} x_j, \quad i = 1:m.$$

This gives the following algorithm

$\text{algorithm overwrites } y \text{ with } Ax + y.$

```

for i = 1:m
    f rj = 1:n
        y(i) = y(i) + A(i,j)*x(j)
    end
end

```

Note that this involves $O(mn)$ work. If each dimension of A is doubled, then the amount of arithmetic increases by a factor of 4.

An alternative algorithm results if we regard \mathbf{Ax} as a linear combination of \mathbf{A} 's columns, eg,

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \begin{bmatrix} \quad \\ \quad \end{bmatrix} = \begin{bmatrix} 1 \cdot 7 + 2 \cdot 8 \\ 3 \cdot 7 + 4 \cdot 8 \\ 5 \cdot 7 + 6 \cdot 8 \end{bmatrix} = 7 \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} + 8 \begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix} = \begin{bmatrix} 23 \\ 53 \\ 83 \end{bmatrix}.$$

this algorithm overwrites y with $Ax + b$.

If $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, and $y \in \mathbb{R}^m$, then

```

for j = 1:n
    for i = 1:m
        y(i) = y(i) + A(i,j)·x(j)
    end
end

```

Note that the inner loop in either `gaxpy` algorithm carries out a `saxy` operation. The column version is derived by rethinking what matrix-vector multiplication "means" at the vector level, but it could also have been obtained simply by interchanging the order of the loops in the rowversion.

hgtkb .t fsf lktst"b tb ht f .lkb ltf kb9k jeb tttb .kPD.teb

Algorithms 1.1.3 and 1.1.4 access the data in A by row and by column, respectively. To highlight these orientations more clearly, we introduce the idea of a partitioned matrix.

From one point of view, a matrix is a stack of row vectors:

$$A \in \mathbb{R}^{m \times n} \iff A = \begin{bmatrix} r_1^T \\ \vdots \\ r_m^T \end{bmatrix}, \quad r_k \in \mathbb{R}^n. \quad (1.1.1)$$

This is called a *row partition* of A . Thus, if we row partition

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix},$$

then we are choosing to think of A as a collection of rows with

$$r_1^T = [1 \ 2], \quad r_2^T = [3 \ 4], \quad r_3^T = [5 \ 6].$$

With the row partitioning (1.1.1), Algorithm 1.1.3 can be expressed as follows:

```
f r i = 1:m
    y_i = y_i + r_i^T x
end
```

Alternatively, a matrix is a collection of column vectors:

$$A \in \mathbb{R}^{m \times n} \quad \text{c. : } A = [c_1 | \cdots | c_n], \quad c_k \in \mathbb{R}^m. \quad (1.1.2)$$

We refer to this as a *column partition* of A . In the 3by2 example above, we thus would set f_1 and A_1 to be the first and second columns of A , respectively:

$$c_1 = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix}, \quad c_2 = \begin{bmatrix} 2 \\ 4 \\ 6 \end{bmatrix}.$$

With (1.1.2) we see that Algorithm 1.1.4 is a *saxpy* procedure that accesses A by columns:

```
f r j = 1:n
    y = y + x_j c_j
end
```

In this formulation, we appreciate y as a running vector sum that undergoes repeated *saxpy* updates.

similarly $\mathbf{A}(:, k)$ $\mathbf{A}(k, :)$

A handy way to specify a column or row of a matrix is with the "colon" notation. If $A \in \mathbb{R}^{m \times n}$, then $A(k, :)$ designates the k th row, i.e.,

$$A(k, :) = [a_{k1}, \dots, a_{kn}].$$

The k th column is specified by

$$A(:, k) = \begin{bmatrix} a_{1k} \\ a_{2k} \\ \vdots \\ a_{nk} \end{bmatrix}.$$

With these conventions we can rewrite Algorithms 1.13 and 1.14a

```
for i = 1:m
    y(i) = y(i) + A(i,:).x
end
```

and

```
for j = 1:n
    y = y + x(j).A(:,j)
end
```

respectively. By using the colon notation, we are able to suppress inner loop details and encourage vector-level thinking

~~hmtib 7\$6 6D QSbx ktDf QR3ttQSb~~

As a preliminary application of the colon notation, we use it to understand the outer product update

$$A = A + x y^T, \quad A \in \mathbb{R}^{m \times n}, \quad x \in \mathbb{R}^m, \quad y \in \mathbb{R}^n.$$

The outer product operation $x y^T$ "looks funny" but is perfectly legal, e.g.,

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} [4 \ 5] = \begin{bmatrix} ! & 5 \\ 12 & 15 \end{bmatrix}.$$

This is because $x y^T$ is the product of two "skinny" matrices and the number of columns in the left matrix x equals the number of rows in the right matrix y^T . The entries in the outer product update are prescribed by

```
f r i = 1:m
f r j = 1:n
    1:j = 1 like+ x_i y_j
end
end
```

This involves $O(mn)$ arithmetic operations. The mission of the j loop is to add a multiple of y^T to the i th row of A , i.e.,

```
for i = 1:m
    A(i,:) = A(i,:) + x(i).y^T
end
```

On the other hand, if we make the i-loop the inner loop, then its task is to add a multiple of x to the j-th column of A:

```
for j = 1:n
    A(:,j) = A(:,j) + y(j)*x
end
```

Note that both implementations amount to a set of **saxy** computations.

hghplsh ht Qxsh tQx1ebhD PQT11QThkb

Consider the 2by2 matrix-matrix multiplication problem. In the dot product formulation, each entry is computed as a dot product:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1 \cdot 5 + 2 \cdot 7 & 1 \cdot 6 + 2 \cdot 8 \\ 3 \cdot 5 + 4 \cdot 7 & 3 \cdot 6 + 4 \cdot 8 \end{bmatrix}.$$

In the **saxy** version, each column in the product is regarded as a linear combination of left-matrix columns:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 5 & 1 \\ 3 & 4 \end{bmatrix} + 7 \begin{bmatrix} 2 \\ 4 \end{bmatrix}, \quad 6 \begin{bmatrix} 1 \\ 3 \end{bmatrix} + 8 \begin{bmatrix} 2 \\ 4 \end{bmatrix}.$$

Finally, in the outer product version, the result is regarded as the sum of outer products:

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \end{bmatrix} [5 \ 6] + \begin{bmatrix} 2 \\ 4 \end{bmatrix} [7 \ 8].$$

Although equivalent mathematically, it turns out that these versions of matrix multiplication can have very different levels of performance because of their memory traffic properties. This matter is pursued in §1.5. For now it is worth detailing the various approaches to matrix multiplication because it gives us a chance to review notation and to practice thinking at different linear algebraic levels. To fix the discussion, we focus on the matrix-matrix update computation:

$$C = C + AB, \quad C \in \mathbb{R}^{m \times n}, A \in \mathbb{R}^{m \times r}, B \in \mathbb{R}^{r \times n}.$$

The update $C = C + AB$ is considered instead of just $C = AB$ because it is the more typical situation in practice.

hnhihhb uftd txs ISf SPbu3 SfT11t QTkeb

The starting point is the familiar triply nested loop algorithm

~~s,yfqlzp2 dndi.I (ij k,e P,i,P,D hPT,5PTICall If A E 1mx, B E 1 xn, and C E 1mx~~

```
f r i = 1:m
    f r j = 1:n
        f r k = 1:r
            C(i,j) = C(i,j) + A(i,k) · B(k,j)
        end
    end
end
```

This computation involves $O(mnr)$ arithmetic. If the dimensions are doubled, then work increases by a factor of 8.

Each loop index in Algorithm 1.1.5 has a particular role. (The subscript i names the row, j names the column, and k handles the dot product.) Nevertheless, the ordering of the loops is arbitrary. Here is the (mathematically equivalent) jki variant:

```

f r j = 1:n
f r k = 1:r
f r i = 1:m
    C(i,j) = C(i,j) + A(i,k)B(k,j)
end
end
end

```

Altogether, there are six ($= 3!$) possibilities:

$ijk, jik, ijk, jki, kij, lji$.

Each features an inner loop operation (dot product or saxpy) and each has its own pattern of data flow. For example, in the ijk variant, the inner loop oversees a dot product that requires access to a row of A and a column of B . The kij variant involves a saxpy that requires access to a column of C and a column of A . These attributes are summarized in Table 1.1.1 together with an interpretation of what is going on when

Loop Order	Inner Loop	Inner Two Loops	Inner Loop Data Access
ijk	dot	vector \times matrix	A by row, B by column
jik	dot	matrix \times vector	A by row, B by column
ikj	saxpy	rowgaxy	B by row, C by row
jki	saxpy	columngaxy	A by column, C by column
kij	saxpy	rowouter product	B by row, C by row
lji	saxpy	columnouter product	A by column, C by column

Table 1.1.1 Matrix multiplication loop orderings and properties

the middle and inner loops are considered together. Each variant involves the same amount of arithmetic, but accesses the A , B , and C data differently. The ramifications of this are discussed in §1.5.

1d101)r F pwsr tF w,B sr qwF.1 aozwrr

The usual matrix multiplication procedure regards AB as an array of dot products to be computed one at a time in left-to-right, top-to-bottom order. This is the idea behind Algorithm 1.1.5 which we rewrite using the colon notation to highlight the mission of the innermost loop:

If $A \in \mathbb{R}^{m \times r}$, $B \in \mathbb{R}^{r \times n}$, and $C \in \mathbb{R}^{m \times n}$ are given, then this algorithm overwrites C with $C + AB$.

```

for i = 1:m
    for j = 1:n
        C(i,j) = C(i,j) + A(i,:).B(:,j)
    end
end

```

In the language of partitioned matrices, if

$$A = \begin{bmatrix} a_1^T \\ \vdots \\ a_m^T \end{bmatrix} \quad \text{and} \quad B = [b_1 | \cdots | b_n],$$

then Algorithm 1.16 has this interpretation

```

for i = 1:m
    for j = 1:n
        c_{ij} = c_{ij} + a_i^T b_j
    end
end

```

Note that the purpose of the j -loop is to compute the i th row of the update. To emphasize this we could write

```

for i = 1:m
    c_i = c_i^T + a_i^T B
end

```

where

$$C = \begin{bmatrix} c_1^T \\ \vdots \\ c_m^T \end{bmatrix}$$

is a row partitioning of C . To say the same thing with the colon notation we write

```

for i = 1:m
    C(i,:) = C(i,:) + A(i,:).B
end

```

Either way we see that the inner two loops of the ijk variant define a transposed copy operation.

hi his nb e bute3 Kb8kx UPQ1kb

Suppose A and C are column-partitioned as follows

$$A = [a_1 | \cdots | a_r], \quad C = [C_1 | \cdots | C_n].$$

By comparing j th columns in $C = C + AB$ we see that

$$C_j = C_j + \sum_{k=1}^r a_k b_k j, \quad j = 1:n.$$

These vector sums can be put together with a sequence of saxy updates

A5EK8M5CT. , 1 2SHSTD2MK15DOAM8BASD, If the matrices $A \in \mathbb{R}^{m \times r}$, $B \in \mathbb{R}^{r \times n}$, and $C \in \mathbb{R}^{m \times n}$ are given, then this algorithm overwrites C with $C + AB$.

```
f rj = 1:n
f rk = 1:r
    C(:,j) = C(:,j) + A(:,k) · B(k,j)
    end
end
```

Note that the k loop oversees a gaxy operation

```
f rj = 1:n
    C(:,j) = C(:,j) + AB(:,j)
    end
```

j 3j 2j vs A :: pls Y Tj:R is iT {px fR I T4s

Consider the kij variant of Algorithm 1.1.5

```
for k = 1:r
    f rj = 1:n
        f ri = 1:m
            C(i,j) = C(i,j) + A(i,k) · B(k,j)
            end
        end
    end
```

The inner two loops oversee the outer product update

$$C = C + a_k b_k^T$$

where

$$A \text{ is } | \cdots | a_i \text{ and } B = \begin{bmatrix} b_1^T \\ \vdots \\ b_r^T \end{bmatrix} \quad (1.13)$$

with $a_k \in \mathbb{R}^m$ and $b_k \in \mathbb{R}^n$. This renders the following implementation

kAOldPngP gus 2adDPPr dD D5PP ,ePc iP,DAPinA5ePilCaP If the matrices $A \in \mathbb{R}^{m \times r}$, $B \in \mathbb{R}^{r \times n}$, and $C \in \mathbb{R}^{m \times n}$ are given, then this algorithm overwrites C with $C + AB$.

```
f rk = 1:r
    C = C + A(:,k) · B(k,:)
    end
```

Matrix-matrix multiplication is a sum of outer products

1mci 4A s3V+A

One way to quantify the volume of work associated with a computation is to count fops. A flop is a floating point add, subtract, multiply, or divide. The number of fops in a given matrix computation is usually obtained by summing the amount of arithmetic associated with the most deeply nested statements. For matrix-matrix multiplication, e.g., Algorithm 1.1.5, this is the 2flop statement

$$C(i,j) = C(i,j) + A(i,k) \cdot B(k,j).$$

If $A \in \mathbb{R}^{m \times r}$, $B \in \mathbb{R}^{r \times n}$, and $C \in \mathbb{R}^{m \times n}$, then this statement is executed mnr times. Table 1.1.2 summarizes the number of fops that are required for the common operations detailed above.

Operation	Dimension	Flops
$a = xTy$	$x, y \in \mathbb{R}^n$	$2n$
$y = y + ax$	$a \in \mathbb{R}^n, x, y \in \mathbb{R}^n$	$2n$
$y = y + Ax$	$A \in \mathbb{R}^{m \times n}, x \in \mathbb{R}^n, y \in \mathbb{R}^m$	$2m$
$A = A + yx^T$	$A \in \mathbb{R}^{m \times n}, x \in \mathbb{R}^n, y \in \mathbb{R}^m$	$2m$
$C = C + AB$	$A \in \mathbb{R}^{m \times r}, B \in \mathbb{R}^{r \times n}, C \in \mathbb{R}^{m \times n}$	$2 \cdot mnr$

Table 1.1.2 Important flop counts

:dd1 jr - \$x0Fr iw ssSwr (t1F l.i&zir

In certain settings it is handy to use the "Big Oh" notation when an order-of-magnitude assessment of work suffices. (We did this in §1.1.1.) Dot products are $O(n)$, matrix-vector products are $O(n^2)$, and matrix-matrix products are $O(n^3)$. Thus, to make efficient an algorithm that involves a mix of these operations, the focus should typically be on the highest order operations that are involved as they tend to dominate the overall computation.

sphpsTb 7\$bb kk Q1kbkb wif \$f thttb Q\$bb hI eub

The dot product and saxpy operations are examples of level-1 operations. Level-1 operations involve an amount of data and an amount of arithmetic that are linear in the dimension of the operation. An m -by- n outer product update or a saxpy operation involves a quadratic amount of data ($O(mn)$) and a quadratic amount of work ($O(mn)$). These are level-2 operations. The matrix multiplication update $C = C + AB$ is a level-3 operation. Level-3 operations are quadratic in data and cubic in work.

Important level-1, level-2, and level-3 operations are encapsulated in the "BLAS," an acronym that stands for Basic Linear Algebra Subprograms. See LAPACK. The design of matrix algorithms that are rich in level-3 BLAS operations is a major preoccupation of the field for reasons that have to do with data reuse (§1.5).

ih1 hiqA)o=mj .mfA }Ac} [=GpAlzN}{G:A

In striving to understand matrix multiplication via outer products, we essentially established the matrix equation

$$AB = \sum_{k=1}^n \mathbf{L}_k \mathbf{a}_k \mathbf{b}_k^\top \quad (114)$$

where the a_k and b_k are defined by the partitionings in (1.13).

Numerous matrix equations are developed in subsequent chapters. Sometimes they are established algorithmically as above and other times they are proved at the i,j -component level, e.g.,

$\mathbf{l} \cdot \mathbf{L} = \mathbf{t} \cdot \mathbf{B} = \mathbf{t} \dots \mathbf{b} = [AB]_{ij}$

Scalar-level verifications such as this usually provide little insight. However, they are sometimes the only way to proceed.

sihis ib .k3 PsebhtQ. fseb

On occasion we shall be concerned with computations that involve complex matrices. The vector space of m -by- n complex matrices is designated by $\mathbb{C}^{m \times n}$. The scaling, addition, and multiplication of complex matrices correspond exactly to the real case. However, transposition becomes conjugate transposition.

$$\mathbf{C} = A^H \quad : \quad : \quad \mathbf{Gj} = \mathbf{lji}$$

The vector space of complex n -vectors is designated by \mathbb{C}^n . The dot product of complex n -vectors x and y is prescribed by

$$\mathbf{S} = \mathbf{x}^H \mathbf{y} = \sum_{i=1}^n \mathbf{X}_i \mathbf{Y}_i$$

If $A = B + iC$ where $B, C \in \mathbb{R}^{n \times n}$, then we designate the real and imaginary parts of A by $\operatorname{Re}(A) = B$ and $\operatorname{Im}(A) = C$, respectively. The conjugate of A is the matrix $\bar{A} = (a_{\bar{i}\bar{j}})$.

Problems

P1.1.1 ...8.8 AERnxn2x x ER^r a y0= la>Q= a y0= exn= xfp x0=Q+n= Ao x =

P1.1.2 .48.. ..8.. 01 .10 **pfe4..fe8=** AB, nAA2A ITb=nTf-TD nTxe,
 $a_{11}b_{12}, a_{21}b_{11}, a_{21}b_{12}, a_{12}b_{21}, a_{12}b_{22}, a_{22}b_{21}, \dots, a_{22}b_{22}$. , , $+_1at_1$, +, $\ddagger(G$ ($_1$ +
 $+_1(G(Ixat_1a, +, G(1(I(ijk, jik, kij(ikj, jki, Aijk uett6 uot;AMLetMln
 eiltMt9uS6$

$$P1.1.3 \quad N_{\text{eff}} = (f^2)(\frac{\pi}{a})^3 + \frac{4\pi}{3} \cdot \frac{N}{a^3} + P \cdot C = (\frac{\pi}{a})^3 k_B T \cdot \frac{N}{a^3} + \frac{4\pi}{3} \cdot \frac{N}{a^3} = V_0 \cdot \frac{N}{a^3} + \frac{4\pi}{3} \cdot \frac{N}{a^3}$$

P1.1.5**8.8** **68**f.8 **8..** **1.1** .., **r4** C, - I)xf. hJ1898 tLUU otiae or G.' o
uetuL A A B L T f.8 n n C A A m f d p f b e L T f D A M A t C A

$$A + iB = (C + i1)(\gamma + iE)$$

P1.1.6**8** .8 WRanf=2 (H)

$$w_{ij} = \sum_{p=1}^n \sum_{q=1}^n x_{ip} y_{pq} z_{qj}$$

3. Theorem $\forall w, \exists X, Y, Z, E, R, m, n, A, x, y, f, F, g, G, h, H$ such that $Ax \wedge F =$

$$w_{ij} = \sum_{p=1}^n x_{ip} \left(\sum_{q=1}^n y_{pq} z_{qj} \right) = \sum_{p=1}^n x_{ip} u_{pj}$$

$$2) \quad \mathbb{U} = YZ \cdot C \cdot P \sum_{n=1}^{\infty} X^n U_n = XYZ \sum_{n=1}^{\infty} (1 - (A + B))^{n-1} (A^n + B^n) = XYZ \sum_{n=1}^{\infty} (1 - (A + B))^{n-1} A^n + XYZ \sum_{n=1}^{\infty} (1 - (A + B))^{n-1} B^n = XYZ \frac{1}{1 - (A + B)} + XYZ \frac{1}{1 - (A + B)} = XYZ \frac{2}{1 - (A + B)}.$$

$$a_{ij} = \sum_{k_1=1}^n \sum_{k_2=1}^n \sum_{k_3=1}^n E(k_1, i) F(k_1, i) G(k_2, k_1) H(k_2, k_3) F(k_2, k_3) G(k_3, j)$$

\rightarrow) = E, F, G, H ∈ R_{0,1,2,3}: RWZ R = W_{0,1,2,3} W_{0,1,2,3} R_{0,1,2,3}

Notes and References for 1.1

Feb. ... 84...881.8 P51.0. N...8.0 .8086

3 5fa. 8.67mF5. 8.6857 or.4..6.. fe515 8ed fo 2Pe..6..8. 1o 8.... 8. . .
 N fefo7171cro.. 67 CM T ans. Math Sof w. 5, E1R ENE,
 161,T niette3.o 1)tlT 3h1Se+e t8Mi.8m5w13enrlnxit 16c4nRtanmahn t lrtt n
 9TMIVMaet8iaotehooAthle+r3TACM T ans. Math Softw 14, i i,,
 1 11hiel te3 16,)tl-3 h6T 7ennh618e+et8Mnitt11, c4 hatlr aa83)-, 5,()
 . . . () A CM T ans. Math Sof w. 16 i i,,
 9wMrtlth (61Mn3 n)ml en llen xiti 1, csMi, Mvt+enLhl,a8r E9l4h, h +A8hotMnar
 vt loc8a(taLMr M18e te3TunHighPerf once ComputingII, 11 trr pStraps - ,Sa y • O
 :004-r = • n3- ay.+ θ 0.0at .p2te
 .. « 00ne.0k., b.0= ke= ya i.as= OMa ...Oy> 1x 30= .eH<3b= pfoTGp+ .a =
 y«Sp.+ Oa O-XO= Oe «& 2T 0. O=y=G=J+ eOa Oe+ 20Dy+ f0a 00001
)aOnS0= Gay OnKanf.= yEa.yθ iN ACM T ans. Math Sof w. 28, ME rii6

,F1m(x2P ,,:= SAT.m.(P;)8 +T2ArE2x ,m+nATB -sNF I troduction to Algorithms, P9c
(tarrenm LiteSsM18EEETD

1.2 Structure and Efficiency

The efficiency of a given matrix algorithm depends upon several factors. Most obvious and what we treat in this section is the amount of required arithmetic and storage. How to reason about these important attributes is nicely illustrated by considering examples that involve triangular matrices, diagonal matrices, banded matrices, symmetric matrices, and permutation matrices. These are among the most important types of structured matrices that arise in practice, and various economies can be realized if they are involved in a calculation.

..... þ 9 Ü £ þ i ç Å j ¥ è þ

A matrix is sparse if a large fraction of its entries are zero. An important special case is the band matrix. We say that $A \in \mathbb{R}^{n \times n}$ has lower bandwidth p if $a_{ij} = 0$ whenever $i > j + p$ and upper bandwidth q if $j > i + q$ implies $a_{ij} = 0$. Here is an example of an 8by5 matrix that has lower bandwidth 1 and upper bandwidth 2:

$$A = \begin{bmatrix} * & * & * & 0 & 0 \\ * & * & * & * & 0 \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & 0 & * \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

The '*'s designate arbitrary nonzero entries. This notation is handy to indicate the structure of a matrix and we use it extensively. Band structures that occur frequently are tabulated in Table 1.2.1.

Type of Matrix	Lower Bandwidth	Upper Bandwidth
Diagonal	0	0
Upper triangular	0	$n - 1$
Lower triangular	$m - 1$	0
Tridiagonal	1	1
Upper bidiagonal	0	1
Lower bidiagonal	1	0
Upper Hessenberg	1	$n - 1$
Lower Hessenberg	$m - 1$	1

Table 1.2.1. Band structure inventory for m -by- n matrices

100r 2Ezax.dEr Do sEzrD-. sza.z8oszwr

To introduce band matrix "thinking" we look at the matrix multiplication update problem $C = C + AB$ where A , B , and C are each n -by- n and upper triangular. The 3by3 case is illuminating

$$AB = \begin{bmatrix} a_{11}b_{11} & a_{11}b_{12} + a_{12}b_{22} & a_{11}b_{13} + a_{12}b_{23} + a_{13}b_{33} \\ 0 & a_{22}b_{22} & a_{22}b_{23} + a_{23}b_{33} \\ 0 & 0 & a_{33}b_{33} \end{bmatrix}.$$

It suggests that the product is upper triangular and that its upper triangular entries are the result of abbreviated inner products. Indeed, since $a_{ik}b_{kj} = 0$ whenever $k < i$ or $j < k$, we see that the update has the form

$$\mathbf{g}_j = \mathbf{g}_j + \sum_{k=i}^j a_{ik}\mathbf{x}_k$$

for all i and j that satisfy $i \leq j$. This yields the following algorithm

Given upper triangular matrices $A, B, C \in \mathbb{R}^{m \times m}$, this algorithm overwrites C with $C + AB$.

```

for i = 1:n
    for j = i:n
        for k = i:j
            C(i,j) = C(i,j) + A(i,k) · B(k,j)
        end
    end
end

```

- 2rlxs tmps .TAT4s)T R I T4AfR I 4s

The dot product that the k-loop performs in Algorithm 1.2.1 can be succinctly stated if we extend the colon notation introduced in §1.1.8. If $A \in \mathbb{R}^{m \times n}$ and the integers p, q and r satisfy $1 \leq p \leq n$ and $1 \leq q \leq m$ then

$$A(r:pq) = [a_{p1} \cdots a_{pq}] \in \mathbb{R}^{l \times (q-p+1)}.$$

Likewise, if $1 \leq p \leq q \leq m$ and $1 \leq q \leq n$, then

$$A(pq:i) = \begin{bmatrix} a_{p1} \\ \vdots \\ a_{q1} \end{bmatrix} \in \mathbb{R}^{q-p+1 \times 1}.$$

With this notation we can rewrite Algorithm 1.2.1 as

```

for i = 1:n
    for j = i:n
        C(i,j) = C(i,j) + A(i,i:j) · B(i:j,j)
    end
end

```

This highlights the abbreviated inner products that are computed by the innermost loop.

si iisb eesSe el"b -kdb

Obviously, upper triangular matrix multiplication involves less arithmetic than full matrix multiplication. Looking at Algorithm 1.2.1, we see that c_{ij} requires $2j - i + 1$ fops if $(i < j)$. Using the approximations

||

— — —

We find that triangular matrix multiplication requires one sixth the number of flops as full matrix multiplication:

$$\sum_{i=1}^n \sum_{j=i}^n 2^{(i-j+1)} = \sum_{i=1}^n \sum_{j=1}^{n-i+1} 2^i \approx \sum_{i=1}^n \frac{2(n-i+1)^2}{2} \approx \sum_{i=1}^n i^2 \approx \frac{n^3}{3}.$$

We throw away the low-order terms since their inclusion does not contribute to what the flop count "says." For example, an exact flop count of Algorithm 1.2.1 reveals that precisely $n^2/3 + n^2/4 + 2n^2/3$ flops are involved. For large n (the typical situation of interest) we see that the exact flop count gives no insight beyond the simple $n^2/3$ accounting.

Flop counting is a necessarily crude approach to the measurement of program efficiency since it ignores subscripting, memory traffic, and other overheads associated with program execution. We must not infer too much from a comparison of flop counts. We cannot conclude, for example, that triangular matrix multiplication is six times faster than full matrix multiplication. Flop counting captures just one dimension of what makes an algorithm efficient in practice. The equally relevant issues of vectorization and data locality are taken up in §1.5.

suuib http://ufkt/Sb

Suppose $A \in \mathbb{R}^{n \times n}$ has lower bandwidth p and upper bandwidth q and assume that p and q are much smaller than n . Such a matrix can be stored in a $(p+q+1)$ -by- n array A_{band} with the convention that

$$a_{ij} = A_{\text{band}}[i - j + q + 1, j] \quad (1.2.1)$$

for all (i,j) that fall inside the band, e.g.,

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & a_{24} & 0 & 0 \\ 0 & a_{32} & a_{33} & a_{34} & a_{35} & 0 \\ 0 & 0 & a_{43} & a_{44} & a_{45} & a_{46} \\ 0 & 0 & 0 & a_{54} & a_{55} & a_{56} \\ 0 & 0 & 0 & 0 & a_{65} & a_{66} \end{bmatrix} = \begin{bmatrix} * & a_{13} & a_{24} & a_{35} & a_{46} & \\ a_{12} & a_{23} & a_{34} & a_{45} & a_{56} & * \\ a_{21} & a_{32} & a_{43} & a_{54} & a_{65} & * \\ a_{31} & a_{42} & a_{53} & a_{64} & a_{75} & * \end{bmatrix}$$

Here, the "*" entries are unused. With this data structure, our column-oriented copy algorithm (Algorithm 1.1.4) transforms to the following:

Suppose $A \in \mathbb{R}^{n \times n}$ has lower bandwidth p and upper bandwidth q and is stored in the A_{band} mat (1.2.1). If $x, y \in \mathbb{R}^n$, then this algorithm overwrites y with $y + Ax$:

```

for j = 1:n
    a1 = max(1, j - q), a2 = min(n, j + p)
    !1 = max(1, q + 2 - j), f2 = !1 + a2 - a1
    y(a1:a2-1) = y(a1:a2) + A_band(!2:j)x(j)
end

```

Notice that by storing A column by column in A_band, we obtain a column-oriented saxpy procedure. Indeed, Algorithm 1.22 is derived from Algorithm 1.14 by recognizing that each saxpy involves a vector with a small number of nonzeros. Integer arithmetic is used to identify the location of these nonzeros. As a result of this careful zero/nonzero analysis, the algorithm involves just $2n_p + q + 1$ fops with the assumption that p and q are much smaller than n .

- lrlBs · T { . I 4fsJ .rsdL RfT4Rfs·R .{ LPs

Matrices with upper and lower bandwidth zero are diagonal. If $D \in \mathbb{C}^{m,n}$ is diagonal, then we use the notation

$$D = \text{diag}(d_1, \dots, d_n), \quad q = \min\{m, n\} \quad C : d_i = d_{ii}.$$

Shortcut notations when the dimension is clear include $\text{diag}(d)$ and $\text{diag}(d)$. Note that if $D = \text{diag}(d) \in \mathbb{C}^{m,n}$ and $x \in \mathbb{C}^n$, then $Dx = d \cdot x$. If $A \in \mathbb{C}^{m,n}$, then pre-multiplication by $D = \text{diag}(d_1, \dots, d_n) \in \mathbb{C}^{m,m}$ scales rows,

$$B = DA \iff B(i, :) = d_i \cdot A(i, :), \quad i = 1:m$$

while post-multiplication by $D = \text{diag}(d_1, \dots, d_n) \in \mathbb{C}^{n,n}$ scales columns,

$$B = AD \iff B(:, j) = d_j \cdot A(:, j), \quad j = 1:n$$

Both of these special matrix-matrix multiplications require $m n$ fops.

similarly for $\mathbf{x} \mathbf{K} \mathbf{x}$

A matrix $A \in \mathbb{C}^{m,n}$ is symmetric if $AT = A$ and skewsymmetric if $AT = -A$. Likewise, a matrix $A \in \mathbb{C}^{m,n}$ is Hermitian if $A^H = A$ and skewHermitian if $A^H = -A$. Here are some examples:

Symmetric	$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix},$	Hermitian	$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2+3i & 6 & 7 & 8 \\ 4+5i & 7 & 8 & 9 \end{bmatrix},$
SkewSymmetric		$\begin{bmatrix} 0 & -2 & 3 \\ 2 & 0 & -5 \\ -3 & 5 & 0 \end{bmatrix},$ SkewHermitian	
		$\begin{bmatrix} i & -2 & 3 & -4 & 5 \\ 2+3 & 6 & -7 & 8 \\ 4+5 & 7+8 & 9 \end{bmatrix}.$	

For such matrices, storage requirements can be halved by simply storing the lower triangle of elements, e.g.,

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix} \Leftrightarrow \text{Avec} = [1 \ 2 \ 3 \ 4 \ 5 \ 6].$$

For general n , we set

$$\text{Avec}((n-j/2)(j-1)+i) = a_{ij} \quad 1:j:i:n \quad (122)$$

Here is a column-oriented gaxy with the matrix A represented in A. vec.

++ Help Example 4/ yy EQ HrE 2. /c Suppose A E $1 \times n$ is symmetric and stored in the A. vec style (1.22). If $x, y \in 1 \times n$, then this algorithm overwrites y with $y + Ax$.

```

f r j = 1:n
f r i = 1:j - 1
    y(i) = y(i) + A. vec((i - 1)n - i(i - 1)/2 + j)x(j)
end
f r i = j:n
    y(i) = y(i) + A. vec((j - 1)n - j(j - 1)/2 + i)x(j)
end
end

```

This algorithm requires the same $2n^2$ fops that an ordinary gaxy requires.

hmhib \$xDQlQTkb ht QxTfShtnb Q\$S tQTCb

We denote the $n \times n$ identity matrix by I_n eg,

$$I_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

We use the notation e_i to designate the i 'th column of I_n . If the rows of I_n are reordered, then the resulting matrix is said to be a permutation matrix eg,

$$r \leftarrow \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}. \quad (1.23)$$

The representation of an $n \times n$ permutation matrix requires just an n -vector of integers whose components specify where the 1's occur. For example, if $v \in 1 \times n$ has the property that v_i specifies the column where the "1" occurs in row i , then $y = Px$ implies that $y_i = x_{v_i}$, $i = 1:n$. In the example above, the underlying v -vector is $v = [2\ 4\ 3\ 1]$.

j lr::s l pRI al FfsF lpfps qRITC s R4pst1 pR II Rps

For permutation matrix work and block matrix manipulation (§1.3) it is convenient to have a method for specifying structured integer vectors of subscripts. The MATLAB colon notation is again the proper vehicle and a few examples suffice to show how it works. If $n = 8$ then

$$\begin{aligned} v = 1:2n &= v = [1\ 3\ 5\ 7], \\ v = n:-1:1 &= v = [6\ 5\ 4\ 3\ 2\ 1]J, \\ v = [(1:2n) (2:2n)] &= v = [1\ 3\ 5\ 7\ 2\ 4\ 6\ 8]J. \end{aligned}$$

Suppose $v \in \mathbb{R}^m$ and $w \in \mathbb{R}^n$ are integer vectors with the property that $1 \leq v_i \leq m$ and $1 \leq w_j \leq n$. If $B = A(v, w)$, then $B \in \mathbb{R}^{m \times n}$ is the matrix defined by $b_{ij} = a_{v_i w_j}$ for $i = 1:r$ and $j = 1:s$. Thus, if $A \in \mathbb{R}^{r \times s}$, then

$$A(1:2:8, 2:2:8) = \begin{bmatrix} a_{12} & a_{14} & a_{16} & a_{18} \\ a_{32} & a_{34} & a_{36} & a_{38} \\ a_{52} & a_{54} & a_{56} & a_{58} \\ a_{72} & a_{74} & a_{76} & a_{78} \end{bmatrix}.$$

shhhshb -k xdtT"bj TQjb.Sx DQtQTkb ht QxTSeb

Using the colon notation, the 4by-4 permutation matrix in (1.23) is defined by $P = I_4(v,:)$ where $v = [2431]$. In general, if $v \in \mathbb{R}^n$ is a permutation of the vector $1:n = [1, 2, \dots, n]$ and $P = I_n(v,:)$, then

$$\begin{aligned} y = Px &= y = x(v) = Y = X, i = 1:n \\ y = P^T x &= y(v) = x = Y = X, i = 1:n \end{aligned}$$

The second result follows from the fact that V is the row index of the "1" in column of P^T . Note that $P^T(Px) = x$. The inverse of a permutation matrix is its transpose.

The action of a permutation matrix on a given matrix $A \in \mathbb{R}^{m \times n}$ is easily described. If $P = I_n(v,:)$ and $Q = I_n(w,:)$, then $PAQ^T = A(v,w)$. It also follows that $I_n(v,:) \cdot I_n(w,:) = I_n(w(v,:))$. Although permutation operations involve no ops, they move data and contribute to execution time, an issue that is discussed in §1.5.

hgii hhb 7 xSSb3 lkDeb Sx DQtQTkb ht QxTSeb

The exchange permutation e_n turns vectors upside down, eg,

$$y = e_n x = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_4 \\ x_3 \\ x_2 \\ x_1 \end{bmatrix}.$$

In general, if $v = n - 1:1$, then the n -by- n exchange permutation is given by $e_n = I_n(v,:)$. No change results if a vector is turned upside down twice and thus, $e_n e_n = e_n = I_n$.

The downshift permutation D_n pushes the components of a vector down one notch with wraparound, eg,

$$y = D_n x = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_4 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}.$$

In general, if $v = [(2n) - 1]$, then the n -by- n downshift permutation is given by $D_n = I_n(v,:)$. Note that D_n^T can be regarded as an upshift permutation.

The modp perfect shuffle permutation p_r treats the components of the input vector $x \in \mathbb{R}^n$, $n = p^r$, as cards in a deck. The deck is cut into p equal "piles" and

reassembled by taking one card from each pile in turn. Thus, if $p = 3$ and $r = 4$, then the piles are $x(1:4)$, $x(5:8)$, and $x(9:12)$ and

$$y = P_{3,4}x = \text{Ip}([15926\ 1037\ 1148\ 12],:)x = \begin{bmatrix} x(14\ 12) \\ x(24\ 12) \\ x(34\ 12) \\ x(44\ 12) \end{bmatrix}.$$

In general, if $n = pr$, then

Pgr = In[(1:r:n) (2:r:n) ... (r:r:n)],:]

and it can be shown that

$$P_r = I_n([1:p:n] \quad [2:p:n] \quad \cdots \quad [p:p:n], :). \quad (124)$$

Continuing with the card deck metaphor, P rearranges the card deck by placing all the x_i having $i \bmod p = 1$ first, followed by all the x_i having $i \bmod p = 2$ second, and so on.

Problems

P1.2.1 N.8 e. ...8..l. .le. 8.8.6..8. A LQnIA²LCAEM: $\mathbb{R}^{n \times n}$. $\in \mathbb{R}^{31}$, ..., $\in \mathbb{R}^{62}$

P1.2.2 ..m..b ... 8...l ..e. 8...8 .18 ... 18 ...P M = (A- >1!) ... (A- ND)
 LIAFAA: $R^{n \times n}$. -3RR_{i,i--i}, m_{i+1} , R, RR_{i,i-3}, 3P
 $-3R_i R_p$, ln

P1.2.3 N..8. .882.. .e8..l. . .18P .18...88. ..8 8mC = C + AB
 $(AO = \text{IAE} \text{ RET21 Hix } \rightarrow Q = \text{IAE} \text{ Q1 Hix})$

P1.2.4 Pp.8.. 1. 8..l. .5 5 .l. ... le..88 .8...2...8.5 P8.2.8 .8.8.... 8
J8 ..8.e ..2...85

P1.2.5 $\mathbf{B} = B + iC$ $T = \text{CAE}Q_1Q_2Q_3Q_4$: $R^{n \times n}$, $\mathbf{u}_i, \mathbf{v}_j \rightarrow \mathbf{R}^m$ \mathbf{i}, \mathbf{j} , $B^T = B -$
 $\mathbf{e}_r = -C$. $\mathbf{f} = \text{ALAEFAA} = A\mathbf{A}^T Q_2 Q_3 T E H I A \text{her} L Q_1 h I A E (A E n_1 n_2 h A \text{herm}(i,j))$ $\mathbf{l} = A =$
 $\mathbf{b}_j \mathbf{x}_i$ $\mathbf{G}(\mathbf{x}) \mathbf{i}, \mathbf{M} = Q_1 n_2 H I n E = \mathbf{a}_n, \mathbf{E} = Q_1 M_2 E I d_2 A \mathbf{a}_n (\mathbf{b}_j = \mathbf{b}_1 Q_1 \mathbf{l}_1) \mathbf{a}_n \mathbf{l}_1$
 $\mathbf{a}_1 b_1 \mathbf{b}_1 = \mathbf{R}(\mathbf{z}) 2x E \frac{1}{2} T W (\mathbf{b}_1 \mathbf{R}(\mathbf{x}) 2x E \frac{1}{2} (\mathbf{x} W \neq (n_2 h T = Ax)$

$$\text{P1.2.7} \quad \dots -8 \cdot 8 \cdot \mathbf{A} \in \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^{p, m}, \quad \mathbf{A} \in \mathbb{R}^{n \times n}, \quad \mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3, \mathbf{R}_4 \in \mathbb{R}^{p, m}, \quad \mathbf{R}_1^T \mathbf{R}_2 + \mathbf{R}_3^T \mathbf{R}_4 = \mathbf{0}$$

P1.2.9 8.8.8. -8.e.. .8.e.8 .18.8 N.8....8. ... 6..8 .18.8.8....
 P 8.15

P1.2.10 .2 .8.o A: $\mathbf{R}^{n \times n}$, $\mathbf{u}: \mathbf{R}^n$, $\mathbf{t}, \mathbf{v}: \mathbf{R}^n$, $\mathbf{R}^{p,m,n}$, „k“ Q3 TnAAE+IL IL n
 $\mathbf{A} \in \mathbf{R}^{n \times k}$, $\mathbf{B} \in \mathbf{R}^{k \times n}$, $\mathbf{C} \in \mathbf{R}^{n \times p}$, $\mathbf{D} \in \mathbf{R}^{p \times n}$, $\mathbf{E} \in \mathbf{R}^{n \times m}$, $\mathbf{F} \in \mathbf{R}^{m \times n}$

P1.2.118.8 xERⁿ. „„, -R^pR^rR^RR^RR^R, y = V x L1xAk T-2 I=TH2A

ici 512u 9E12Lii C23 sMsP' r = ' r,p.

C r.- p q0=p x n0E by-n AapuoteShetS.aSA! 28 uen. InteaAapA.uuatLS.M

Notes and References for

.88 LAPACK eEaO(OE.oof n=a.f f E= fO(aae=aθ.E=3300ex x O=Ee0.eaE= 000e0oo f EoOQ=aQ= (= =

R . Ho= a >M220 Dpq0< 0> =IeO3a =Q= Ep E/f f Eθ a dEq0e oSa Oay.E. vOS E= e003Ap < (- e=a θ.C e= Eθ(E= =Oe ceedings of the 7h international Conference on Parallel Processing and Applied Mathematics, hAlngalr IaIAGS nif uN

cea a.eenga=Fl8nAeSntAalat Aeo AaIu,T etShetSA,AShellen ... cd.

1.3 Block Matrices and Algorithms

A block matrix is a matrix whose entries are themselves matrices. It is a point of view. For example, an 8-by-15 matrix of scalars can be regarded as a 2-by-3 block matrix with 4-by-5 entries. Algorithms that manipulate matrices at the block level are often more efficient because they are richer in level-3 operations. The derivation of many important algorithms is often simplified by using block matrix notation.

hmmihbhdkfdb ht QxTe₁₆ ..Tlkf k "Kb

Column and row partitionings (§1.1.7) are special cases of matrix blocking. In general, we can partition both the rows and columns of an m-by-n matrix A to obtain

$$A = \begin{bmatrix} A_{11} & \dots & A_{1r} \\ \vdots & & \vdots \\ A_{q1} & \dots & A_{qr} \end{bmatrix}_{m_1 \times n_r}^{m_q \times n}$$

where m_1

=

$$\begin{bmatrix} 0 \\ 0 \\ L_{33} \end{bmatrix},$$

iA1A.A B.iTA q {7GpASV o7}G.1A

Block matrices can be scaled and transposed

$$\mu \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \\ A_{31} & A_{32} \end{bmatrix} = \begin{bmatrix} \mu A_{11} & \mu A_{12} \\ \mu A_{21} & \mu A_{22} \\ \mu A_{31} & \mu A_{32} \end{bmatrix},$$

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \\ A_{31} & A_{32} \end{bmatrix}^T = \begin{bmatrix} A_{11}^T & A_{21}^T & A_{31}^T \\ A_{12}^T & A_{22}^T & A_{32}^T \end{bmatrix}.$$

Note that the transpose of the original (i, j) block becomes the (j, i) block of the result. Identically blocked matrices can be added by summing the corresponding blocks

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \\ A_{31} & A_{32} \end{bmatrix} + \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{bmatrix} = \begin{bmatrix} A_{11} + B_{11} & A_{12} + B_{12} \\ A_{21} + B_{21} & A_{22} + B_{22} \\ A_{31} + B_{31} & A_{32} + B_{32} \end{bmatrix}.$$

Block matrix multiplication requires more stipulations about dimension. For example if

$$\begin{bmatrix} A_1 A_2 \\ A_2 A_2 \\ A_3 A_2 \end{bmatrix} \begin{bmatrix} B_1 B_2 \\ B_2 B_2 \end{bmatrix} = \begin{bmatrix} A_1 B_1 + A_2 B_2 & A_1 B_2 + A_2 B_2 \\ A_2 B_1 + A_2 B_2 & A_2 B_2 + A_2 B_2 \\ A_3 B_1 + A_3 B_2 & A_3 B_2 + A_3 B_2 \end{bmatrix}$$

is to make sense, then the column dimensions of A_{11} , A_{21} and A_{31} must each be equal to the row dimension of both B_{11} and B_{21} . Likewise, the column dimensions of A_{12} , A_{22} and A_{32} must each be equal to the row dimensions of both B_{21} and B_{22} .

Whenever a block matrix addition or multiplication is indicated, it is assumed that the row and column dimensions of the blocks satisfy all the necessary constraints. In that case we say that the operands are partitioned conformably, in the following theorem.

Theorem 1.3.1. If

$$A = \begin{bmatrix} A_{11} & \dots & A_{1s} \\ \vdots & & \vdots \\ A_{q1} & \dots & A_{qs} \end{bmatrix}_{m_q}^{m_1}, \quad B = \begin{bmatrix} B_{11} & \dots & B_{1r} \\ \vdots & & \vdots \\ B_{s1} & \dots & B_{sr} \end{bmatrix}_{n_s}^{n_1},$$

and we partition the product $C = AB$ as follows

$$C = \begin{bmatrix} C_{11} & \dots & C_{1r} \\ \vdots & & \vdots \\ C_{q1} & \dots & C_{qr} \end{bmatrix}_{m_q}^{m_1},$$

then for $a = 1:q$ and $(d)1:r$ we have $C_{a,d} = \sum_{s=1}^r A_{as} B_{sd}$.

r b The proof is a tedious exercise in subscripting. Suppose $1 \leq a \leq q$ and $1 \leq f \leq r$. Set $M = m_1 + \dots + m_{l-1}$ and $N = n_1 + \dots + n_{r-1}$. It follows that if $1 \leq i \leq m$ and $1 \leq j \leq n$ then

$$\begin{aligned} Ca_{tj} &= \sum_{k=1}^{P_1+\dots+P_s} a_{M+i,k} b_{k,N+j} = \sum_{k=1}^{l-1} \sum_{p=1}^{P_1+\dots+P_s-l+1} a_{M+i,k} b_{k,N+j} \\ &= \sum_{\alpha=1}^s \sum_{\beta=1}^{P_\alpha} [A_{\alpha} B]_{ik} B_{\beta j} = \sum_{\gamma=1}^s [A_{\alpha} B]_{ij} B_{\gamma j} = \left[\sum_{\gamma=1}^s A_{\alpha\gamma} B_{\gamma\beta} \right]_{ij}. \end{aligned}$$

Thus $Ca = A_1 B_1 11 + \dots + A_s B_s / . . . j J$

If you pay attention to dimension and remember that matrices do not commute, i.e., $A_1 B_1 + A_2 B_2 \neq B_1 A_1 + B_2 A_2$, then block matrix manipulation is just ordinary matrix manipulation with the a_{ij} 's and b_{ij} 's written as A_{ij} 's and B_{ij} 's!

smmhmu f.tQx TfSb

Suppose $A \in \mathbb{R}^{mn}$. If $a = [a_1, \dots, a_s]$ and $f = [f_1, \dots, f_t]$ are integer vectors with distinct components that satisfy $1 \leq a_i \leq m$ and $1 \leq f_i \leq n$, then

$$A(a, f) = \begin{bmatrix} a_{\alpha_1, \beta_1} & \cdots & a_{\alpha_1, \beta_t} \\ \vdots & \ddots & \vdots \\ a_{\alpha_s, \beta_1} & \cdots & a_{\alpha_s, \beta_t} \end{bmatrix}$$

is an s by t submatrix of A . For example, if $A \in \mathbb{R}^8 \times 6$, $a = [2, 4, 6, 8]$, and $f = [4, 5, 6]$, then

$$A(a, f) = \begin{bmatrix} \bullet & \bullet\bullet & \bullet\bullet \\ \bullet & \bullet\bullet & \bullet\bullet \\ \bullet & \bullet\bullet & \bullet\bullet \\ a_1 & a_5 & a_6 \\ a_3 & a_5 & a_6 \\ a_4 & a_5 & a_6 \end{bmatrix}$$

If $a = \emptyset$, then $A(a, f)$ is a principal submatrix. If $a = f = \emptyset$ and $1 \leq k \leq \min(m, n)$, then $A(a, f)$ is a leading principal submatrix.

If $A \in \mathbb{R}^{mn}$ and

$$A = \begin{bmatrix} A_{11} & \dots & A_{1s} \\ \vdots & & \vdots \\ A_{q1} & \dots & A_{qs} \end{bmatrix}_{m_1 \dots m_q},$$

then the colon notation can be used to specify the individual blocks. In particular,

$$A_{ij} = A(r+l:r+r, \mu+l:\mu+r)$$

where $r = m_1 + \dots + m_{l-1}$ and $\mu = n_1 + \dots + n_{r-1}$. Block matrix notation is valuable for the way in which it hides subscript range expressions.

i hln)A TEA B.iT dA 4pf A

As an exercise in block matrix manipulation and submatrix designation, we consider two block versions of the `gaxy` operation $y = y + Ax$ where $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, and $y \in \mathbb{R}^m$. If

$$A = \begin{bmatrix} A_1 \\ \vdots \\ A_q \end{bmatrix}_{m_1 \times m_q} \quad \text{and} \quad y = \begin{bmatrix} y_1 \\ \vdots \\ y_q \end{bmatrix}_{m_1 \times m_q},$$

then

$$\begin{bmatrix} y_1 \\ \vdots \\ y_q \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_q \end{bmatrix} + \begin{bmatrix} A_1 \\ \vdots \\ A_q \end{bmatrix} x,$$

and we obtain

a=O
f r i = l:q

idx = a+l:a+ni

y(idx) = y(idx) + A(idx,:).x

a=a+ni

end

The a segment to $y(\text{idx})$ corresponds to $y_i = Y + A_i x$. This row-blocked version of the `gaxy` computation breaks the given `gaxy` into q "shorter" `gaxys`. We ref r to A_i a the ith block row of A .

Likewise, with the partitions

$$A = \begin{bmatrix} A_1 & \cdots & A_r \end{bmatrix}_{n_1 \times n_r} \quad \text{and} \quad x = \begin{bmatrix} x_1 \\ \vdots \\ x_r \end{bmatrix}_{n_1 \times n_r},$$

we see that

$$y = y + [A_1 | \cdots | A_r] \begin{bmatrix} x_1 \\ \vdots \\ x_r \end{bmatrix} = y + \sum_{j=1}^r A_j x_j$$

and we obtain

{=0
f r j = l:r

j dx = {+l:{+ri

y = y+A(:,j dx).x(j dx)

{ = { +ri

end

The a segment to y corresponds to $y = y + A_j x$. This column-blocked version of the `gaxy` computation breaks the given `gaxy` into r "thinner" `gaxys`. We ref r to A_j a the j th block column of A .

i. $\text{ImA} \leftarrow \text{TA}$ q. $\{\text{GpAqN} \leftarrow \text{mVG}\}$ G:A

Just as ordinary scalar-level matrix multiplication can be arranged in several possible ways, so can the multiplication of block matrices. To illustrate this with a minimum of subscript clutter, we consider the update

$$C = C + AB$$

where we regard $A = (A_{af})$, $B = (B_{bf})$, and $C = (C_{af})$ as N -by- N block matrices with t -by- t blocks. From Theorem 1.3.1 we have

$$C_{af} \leftarrow C_{af} + \sum_{a=1}^N A_{a\cdot} B_{\cdot a}, \quad a = 1:N, \quad \cdot = 1:N.$$

If we organize a matrix multiplication procedure around this summation, then we obtain a block analog of Algorithm 1.1.5

```

for a = 1:N
    i = (a - 1)t + 1:at
    for { = 1:N
        j = ({ - 1)t + 1:{ t
        f r' = 1:N
        k = (7 - 1)t + 1:7t
        C(i,j) ← C(i,j) + A(i,k) · B(k,j)
    end
end
end

```

Note that, if $t = 1$ then $a = i$, $\cdot = j$, and $7 = k$ and we revert to Algorithm 1.1.5

Analogously to what we did in §1.1, we can obtain different variants of this procedure by playing with loop orders and blocking strategies. For example, corresponding to

$$\begin{bmatrix} C_{11} & \cdots & C_{1N} \\ \vdots & \ddots & \vdots \\ C_{N1} & \cdots & C_{NN} \end{bmatrix} + \begin{bmatrix} A_1 \\ \vdots \\ A_N \end{bmatrix} \begin{bmatrix} B_1 & \cdots & B_N \end{bmatrix}$$

where $A_i \in \mathbb{R}^{1 \times n}$ and $B_j \in \mathbb{R}^{n \times 1}$, we obtain the following block outer product computation

```

for i = 1:N
    for j = 1:N
        Cii = Cii + AiBi
    end
end

```

1hlinkA TECA t=.6 ToA a=L Ni[A]

It is sometimes the case that the entries in a block matrix A are all scalar multiples of the same matrix. This means that A is a Kronecker product. Formally, if B $\in \mathbb{R}^{m_1 \times m_1}$ and C $\in \mathbb{R}^{n_1 \times n_2}$, then their Kronecker product $B \otimes C$ is an m_1 -by- n_1 block matrix whose (i,j) block is the m_2 -by- n_2 matrix $b_{ij}C$. Thus, if

$$A = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{bmatrix} @ \begin{bmatrix} C_1 & C_2 & C_3 \\ C_1 & C_2 & C_3 \\ C_1 & C_2 & C_3 \end{bmatrix}$$

then

$$A = \begin{bmatrix} b_{11}C_1 & b_{11}C_2 & b_{11}C_3 & b_{12}C_1 & b_{12}C_2 & b_{12}C_3 \\ b_{12}C_1 & b_{12}C_2 & b_{12}C_3 & b_{21}C_1 & b_{21}C_2 & b_{21}C_3 \\ b_{13}C_1 & b_{13}C_2 & b_{13}C_3 & b_{22}C_1 & b_{22}C_2 & b_{22}C_3 \\ b_{21}C_1 & b_{21}C_2 & b_{21}C_3 & b_{23}C_1 & b_{23}C_2 & b_{23}C_3 \\ b_{22}C_1 & b_{22}C_2 & b_{22}C_3 & b_{31}C_1 & b_{31}C_2 & b_{31}C_3 \\ b_{23}C_1 & b_{23}C_2 & b_{23}C_3 & b_{32}C_1 & b_{32}C_2 & b_{32}C_3 \\ b_{31}C_1 & b_{31}C_2 & b_{31}C_3 & b_{32}C_1 & b_{32}C_2 & b_{32}C_3 \\ b_{32}C_1 & b_{32}C_2 & b_{32}C_3 & b_{33}C_1 & b_{33}C_2 & b_{33}C_3 \end{bmatrix}.$$

This type of highly structured blocking occurs in many applications and results in dramatic economies when fully exploited.

Note that if B has a band structure, then $B \otimes C$ "inherits" that structure at the block level. For example, if

$$B \text{ is } \left\{ \begin{array}{l} \text{diagonal} \\ \text{tridiagonal} \\ \text{lower triangular} \\ \text{upper triangular} \end{array} \right\} \text{ then } B \otimes C \text{ is } \left\{ \begin{array}{l} \text{block diagonal} \\ \text{block tridiagonal} \\ \text{block lower triangular} \\ \text{block upper triangular} \end{array} \right\}.$$

Important Kronecker product properties include

$$(B \otimes C)^T = B^T \otimes C^T, \quad (131)$$

$$(B \otimes C)(D \otimes F) = BD \otimes CF, \quad (132)$$

$$(B \otimes C)^{-1} = B^{-1} \otimes C^{-1}, \quad (133)$$

$$B \otimes (C \otimes D) = (B \otimes C) \otimes D. \quad (134)$$

Of course, the products BD and CF must be defined for (132) to make sense. Likewise, the matrices B and C must be nonsingular in (133).

In general, $B \otimes C \neq C \otimes B$. However, there is a connection between these two matrices via the perfect shuffle permutation that is defined in §12.11. If $B \in \mathbb{R}^{m_1 \times m_1}$ and $C \in \mathbb{R}^{n_1 \times n_2}$, then

$$P(B \otimes C)Q^T = C \otimes B \quad (135)$$

where $P = P_{m_1, m_2}$ and $Q = P_{n_1, n_2}$.

1h1fEA .o- } fnfA t7 : dTo6A a7JNi |Alp V7o +m+A

A matrix-vector product in which the matrix is a Kronecker product is "secretly" a matrix-matrix-matrix product. For example, if $B \in \mathbb{R}^{3 \times 2}$, $C \in \mathbb{R}^{2 \times n}$, and $x \in \mathbb{R}^{2 \times 1}$, then

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = (B \otimes C) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_{11}C & b_{12}C \\ b_{21}C & b_{22}C \\ b_{31}C & b_{32}C \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$[b_{11}Cx_1 + b_{12}Cx_2]$

where $y_1, y_2, y_3 \in \mathbb{R}$. On the other hand, if we define the matrices

$$= [\mathbf{X} \mathbf{X}] \quad \text{and} \quad \mathbf{y} = [\mathbf{Y} \mathbf{Y}],$$

then $\mathbf{Y} = \mathbf{C} \mathbf{X} \mathbf{B}^T$.

To be precise about this reshaping we introduce the `vec` operation. If $X \in \mathbb{R}^{m \times n}$, then `vec(X)` is an m by 1 vector obtained by "stacking" X 's columns:

$$\text{vec}(X) = \begin{bmatrix} X(:, 1) \\ \vdots \\ X(:, n) \end{bmatrix}.$$

If $B \in \mathbb{R}^{m_1 \times n_1}$, $C \in \mathbb{R}^{m_2 \times n_2}$, and $X \in \mathbb{R}^{n_1 \times m_2}$, then

$$\mathbf{Y} = \mathbf{C} \mathbf{X} \mathbf{B}^T \quad \text{vec}(\mathbf{Y}) = (\mathbf{B} \odot \mathbf{C}) \text{vec}(\mathbf{X}). \quad (1.36)$$

Note that if $, C, X \in \mathbb{R}^{m \times n}$, then $\mathbf{Y} = \mathbf{C} \mathbf{X} \mathbf{B}^T$ costs $O(n^3)$ to evaluate while the disregard of Kronecker structure in $\mathbf{y} = (\mathbf{B} \odot \mathbf{C})\mathbf{x}$ leads to an $O(n^4)$ calculation. This is why reshaping is central for effective Kronecker product computation. The `reshape` operator is handy in this regard. If $A \in \mathbb{R}^{m \times n}$ and $m_1, n_1 \leq m, n$, then

$$B = \text{reshape}(A, m_1, n_1)$$

is the m_1 by n_1 matrix defined by $\text{vec}(B) = \text{vec}(A)$. Thus, if $A \in \mathbb{R}^{3 \times 4}$, then

$$\text{reshape}(A, 2, 6) = \begin{bmatrix} a_1 & a_3 & a_2 & a_4 & a_3 & a_1 \\ a_2 & a_1 & a_3 & a_2 & a_4 & a_3 \end{bmatrix}.$$

B) CTT qQ Q1OXThAPxMQXAE>P1 QMAT

Note that $A = B \odot C \odot D$ can be regarded as a block matrix whose entries are block matrices. In particular, $b_{ij} \odot d$ is the (k, l) block of A 's (i, j) block.

As an example of a multiple Kronecker product computation, let us consider the calculation of $\mathbf{y} = (\mathbf{B} \odot \mathbf{C} \odot \mathbf{D})\mathbf{x}$ where $B, C, D \in \mathbb{R}^{N \times N}$ and $x \in \mathbb{R}^N$ with $N = m^3$. Using (1.36) it follows that

$$\text{reshape}(\mathbf{y}, m^2, n) = (\mathbf{C} \odot \mathbf{D}) \cdot \text{reshape}(\mathbf{x}, m^2, n) \cdot \mathbf{T}.$$

Thus, if

$$F = \text{reshape}(x, m^2, n) \quad T,$$

then $G = (C \otimes D)F$ ER^{1x}n can be computed column by column using (136):

$$G(:, k) = \text{reshape}(D \cdot \text{reshape}(F(:, k), n, n) \cdot C^T, m^2, 1) \quad k = 1:n$$

It follows that $y = \text{reshape}(G, N, 1)$. A careful accounting reveals that 6n fops are required. Ordinarily, a matrix-vector product of this dimension would require $2n^6$ fops.

The Kronecker product has a prominent role to play in tensor computations and in §13.1 we detail more of its properties.

sunhib eb tk QShkb .k3 dSbht QxTehd QT3fif QTkb

Consider the complex matrix multiplication update

$$C_1 + iC_2 = (C_1 + iC_2) + (A_1 + iA_2)(B_1 + iB_2)$$

where all the matrices are real and $i^2 = -1$. Comparing the real and imaginary parts we conclude that

$$\begin{bmatrix} C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} + \begin{bmatrix} A_1 & -A_2 \\ A_2 & A_1 \end{bmatrix} \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}.$$

Thus, complex matrix multiplication corresponds to a structured real matrix multiplication that has expanded dimension

sunhssb It.Td QktTtb ttcb uK.3fS fQfifht QxTSeb

While on the topic of 2by2 block matrices, we identify two classes of structured matrices that arise at various points later on in the text. A matrix $M \in \mathbb{R}^{2n \times 2n}$ is a Hamiltonian matrix if it has the form

$$M = \begin{bmatrix} A & G \\ F & -A^T \end{bmatrix}$$

where $A, F, G \in \mathbb{R}^{n \times n}$ and F and G are symmetric. Hamiltonian matrices arise in optimal control and other application areas. An equivalent definition can be given in terms of the permutation matrix

$$J = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}.$$

In particular, if

$$M^T = -MT,$$

then M is Hamiltonian. A related class of matrices are the symplectic matrices. A matrix $S \in \mathbb{R}^{2n \times 2n}$ is symplectic if

$$S^T JS = J.$$

If

$$S = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix}$$

where the blocks are n by n , then it follows that both S_{11} and S_{22} and S_{12} and S_{21} are symmetric and $S_{21} = S_{12}^T$

3C) 33T 28 vAAxUlwq w xJ R7Q1 xJ hJ MwJ R7T

We conclude this section with a completely different approach to the matrix-matrix multiplication problem. The starting point in the discussion is the 2 by 2 block matrix product

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

where each block is square. In the ordinary algorithm, $C_{ij} = A_{i1}B_{11} + A_{i2}B_{21}$. There are 8 multiplies and 4 adds. Strassen (1969) has shown how to compute C with just 7 multiplies and 18 adds.

$$\begin{aligned} P_1 &= (A_{11} + A_{21})(B_{11} + B_{21}), \\ P_2 &= (A_{21} + A_{22})B_{11}, \\ P_3 &= A_{11}(B_{12} - B_{22}), \\ P_4 &= A_{22}(B_{21} - B_{11}), \\ P_5 &= (A_{11} + A_{12})B_{22}, \\ P_6 &= (A_{21} - A_{11})(B_{11} + B_{12}), \\ P_7 &= (A_{12} - A_{22})(B_{21} + B_{22}), \\ C_{11} &= P_1 + P_4 - P_5 + P_7, \\ C_{12} &= P_3 + P_5, \\ C_{21} &= P_2 + P_4, \\ C_{22} &= P_1 + P_3 - P_2 + P_5 \end{aligned}$$

These equations are easily confirmed by substitution. Suppose $n = 2m$ so that the blocks are m by m . Counting adds and multiplies in the computation $C = AB$, we find that conventional matrix multiplication involves $(2m)^3$ multiplies and $(2m)^3 - (2m)^2$ adds. In contrast, if Strassen's algorithm is applied with conventional multiplication at the block level, then $7m^3$ multiplies and $7m^3 + 11m^2$ adds are required. If $m \gg 1$, then the Strassen method involves about $7/8$ the arithmetic of the fully conventional algorithm.

Now recognize that we can recur on the Strassen idea. In particular, we can apply the Strassen algorithm to each of the half-sized block multiplications associated with the P_i . Thus, if the original A and B are n by n and $n = 2^k$, then we can repeatedly apply the Strassen multiplication algorithm. At the bottom "level," the blocks are 1 by 1.

Of course, there is no need to recur down to the $n = 1$ level. When the block size gets sufficiently small, ($n : n_{\min}$), it may be sensible to use conventional matrix multiplication when finding the P_i . Here is the overall procedure:

If $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times n}$ and $C = AB$. Suppose $n = 2^d$ and that $A \in \mathbb{R}^{2^d \times 2^d}$ and $B \in \mathbb{R}^{2^d \times 2^d}$. If $\min = 2^d$ with $d \leq q$, then this algorithm computes $C = A \cdot B$ by applying Strassen procedure recursively.

```

function C = strass(A, B, n, min)
    if n < min
        C = AB      (conventionally computed)
    else
        m= n/2 u= 1:m v= m+ 1:n
        P1 = strass(A(u,u) + A(v,v), B(u,u) + B(v,v), m,min)
        P2= strass(A(v,u) + A(v,v), B(u,u), m,min)
        P3= strass(A(u,u), B(u,v) - B(v,v), m,min)
        P4= strass(A(v,v), B(v,u) - B(u,u), m,min)
        P5= strass(A(u,u) + A(u,v), B(v,v), m,min)
        P6= strass(A(v,u) - A(u,u), B(u,u) + B(u,v), m,min)
        P7= strass(A(u,v) - A(v,v), B(v,u) + B(v,v), m,min)
        C(u,u) = P1 + P4- P5+ P1
        C(u,v) = P3+ P5
        C(v,u) = P2+ P4
        C(v,v) = P1+ P3- P2+ P6
    end

```

Unlike any of our previous algorithms, strass is recursive. Divide and conquer algorithms are often best described in this fashion. We have presented strass in the style of a MATLAB function so that the recursive calls can be stated with precision.

The amount of arithmetic associated with strass is a complicated function of n and \min . If $\min \gg 1$, then it suffices to count multiplications as the number of additions is roughly the same. If we just count the multiplications, then it suffices to examine the deepest level of the recursion, that is where all the multiplications occur. In strass there are $q - d$ subdivisions and thus 7^d conventional matrix-matrix multiplications to perform. These multiplications have size \min and thus strass involves about $s = (2/3)^d 7^d$ multiplications compared to $I = \{2\}^3$, the number of multiplications in the conventional approach. Notice that

$$\frac{s}{I} = \left(\frac{2d}{3}\right)^3 7^d = \left(\frac{7}{8}\right)^d.$$

If $d = 0$, i.e., we recur on down to the 1-by-1 level, then

$$s = (7/8)^0 = 7^0 = n^0 = n^2.$$

Thus, asymptotically, the number of multiplications in Strassen's method is $O(n^2)$. However, the number of additions (relative to the number of multiplications) becomes significant as n gets small.

Problems

P1.3.1 7..8.8.8. .8.8 .18 N..86.4. .8.de.. 8r.e.84 or

$$\begin{bmatrix} A_{11} & \cdots & A_{1r} \\ \vdots & \ddots & \vdots \\ A_{q1} & \cdots & A_{qr} \end{bmatrix}^T = \begin{bmatrix} A_{11}^T & \cdots & A_{q1}^T \\ \vdots & \ddots & \vdots \\ A_{1r}^T & \cdots & A_{qr}^T \end{bmatrix}.$$

P1.3.2 ...8 .8 MER Rn k k d 2 Id 2 M n 2 M 2 A2 TMA =M2

P1.3.3 .e.e.8.8. e.e 8.18 01 .10 88.ro....mgb a V.x=A E X H A I A K
E A2 F A2 M k An Rn M F A2 x Y M T R W c u1 A K K M
12 A An d x Q A A n A k 2 A E r a F i r 1 A2 A r 1 A C r 1 A

P1.3.4 .2..8.8

$$A = \begin{bmatrix} & \ddots \end{bmatrix}$$

B Rn k FW x 10 . w x n A n A T A n 2) = P A T u n P = P_{2,n} k A n
W T A k F W F M 2 x x Y x Y

P1.3.5 .386 le. .bB and 7.paa,J Aapuot,tSln tpS.aAB B X(o <efO= x p= x = (=f)

P1.3.6 e8..b. Pr.e.. 4d 5 P5

P1.3.7 em..b.le. .bER le y E Rn Ac y 1 x = vec 1 y T

P1.3.8 .186 .erdeb R^{p \times p}, E R^{q \times q}

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_p \end{bmatrix} \quad x_i \in \mathbb{R}^q,$$

AIC

$$(B \otimes \quad = L^p L^p) \mathbf{j}(T(-)) a$$

P1.3.9 ..8.8 ADE Rn k = l r H m H m E Rn m C = c 1. . A) y G A y ,
k \alpha \beta, A_{j,k} , G \beta_k T \tilde{q}, A_y \in (A_1 \dots 1 A_1 A_1) x .

P1.3.10 .2..8 .8 .. m.84e4.8 .8lmN.086... ...84 Rn R

$$f(x) = r \mathbf{L} \mathbf{L} \mathbf{C} \mathbf{r} \mathbf{L} \mathbf{A} \mathbf{C} = L^2 \cdot 212$$

iii L A2 A A x y E Rn Ac

$$T = L^2 (212 \mathbf{Y} (212 \mathbf{Y} \mathbf{J}) f(x) \cdot f(y)).$$

6XIL a(v=QxALCA'd if HnEQD= inQdAhnQv=AB Bn k QAI I TMA B Ak
V2FTA AA M k C TQVOK 2h f SAAAsSahteapl8AInA = An T2k k2 B L m
x2 r+1 ij = A1

P1.3.12 1.e. strass 1 A) B (= g(i, f), A fig) A g AB (A((l, z A)) (B_i, () A1 2 = M 1 M 1 1 0 2 2 a le T2k

P1.3.13 1.e. strass, () A) B A = g a (+ i B) d = AB Bn A E Rn e
BEWQk A M A 2 H A A A = B A OR 12 A 1 A 1 2 B M le Q x 10
2 A q m A A B A Rn H H A k

P1.3.14 58, W h An d) 2 2 MA strass, i A, (B(fai) s(a, = B) (6 (o
= f = 30 = 2 In A n H n W a le MA J C T

$$W = M^2 + SY6V3K$$

C 18.0 = n elem > 0 M, 1y, ,) c, Alw_n s c_e

$$y = F_n x$$

where the DFT matrix $F_N = (f_{kj}) \in \mathbb{C}^{N \times N}$ is defined by

$$f_{kj} = \omega_n^{(k-1)(j-1)} \quad (14.1)$$

with

$$W = \exp(-2\pi i/n) = \cos(2\pi/n) - i \cdot \sin(2\pi/n). \quad (14.2)$$

Here is an example

$$F_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ W & W^2 & W^3 & W^4 \\ W^2 & W^4 & W^6 & W^8 \\ W^3 & W^6 & W^9 & W^{12} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -i & -1 & i & -i \\ -1 & 1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix}.$$

The DFT is ubiquitous throughout computational science and engineering and one reason has to do with the following property:

If n is highly composite, then it is possible to carry out the DFT in many fewer than the $O(n^2)$ operations required by conventional matrix-vector multiplication.

To illustrate this we set $n = 2^k$ and proceed to develop the radix-2 fast Fourier transform.

The starting point is to examine the block structure of an even-order DFT matrix after its columns are reordered so that the odd-indexed columns come first. Consider the case

$$F_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ 1 & \omega^2 & W & W^6 & 1 & \omega^2 & W & W^6 \\ 1 & \omega^3 & \omega^6 & \omega^9 & \omega^4 & \omega^7 & \omega^2 & \omega^5 \\ 1 & W & 1 & W & 1 & \omega^4 & 1 & W \\ 1 & \omega^5 & \omega^2 & \omega^7 & \omega^4 & \omega & \omega^6 & \omega^3 \\ 1 & \omega^6 & W & W & 1 & \omega^6 & \omega^4 & \omega^2 \\ 1 & W & \omega^6 & \omega^5 & W & \omega^3 & \omega^2 & \omega \end{bmatrix} \quad (\omega = \omega_8).$$

(Note that ω_8 is a root of unity so that high powers simplify, e.g., $[F_8]_{4,7} = \omega^{3 \cdot 6} = \omega^{18} = \omega^2$.) If $\text{cols} = [13572468]$, then

$$F_8(:, \text{cols}) = \left[\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega^2 & W & W^6 & \omega & \omega^3 & \omega^5 & \omega^7 \\ 1 & W & 1 & W & \omega^2 & \omega^6 & \omega^2 & \omega^6 \\ 1 & \omega^6 & W & W^2 & \omega^3 & \omega & \omega^7 & \omega^5 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & \omega^2 & W^4 & W^8 & -\omega & -\omega^3 & -\omega^5 & -\omega^7 \\ 1 & W & 1 & W^4 & -\omega^2 & -\omega^6 & -\omega^2 & -\omega^6 \\ 1 & \omega^6 & W & W^2 & -\omega^3 & -\omega & -\omega^7 & -\omega^5 \end{array} \right].$$

The lines through the matrix are there to help us think of $F_8(:, \text{cols})$ as a 2-by-2 matrix with 4-by-4 blocks. Noting that $\omega^2 = \omega_8^2 = \omega_4$, we see that

$$F_8(:, \text{cols}) = \left[\begin{array}{c|c} F_4 & Q_4 \\ \hline F_4 & -Q_4 \end{array} \right]$$

where $\mathbf{F}_4 = \text{diag}(\mathbf{l}, \mathbf{w}, \mathbf{g}, \mathbf{w}, \mathbf{w})$. It follows that if $\mathbf{x} \in \mathbb{J}_{\mathbf{8}}$ then

$$\mathbf{F}_8 \mathbf{x} = \mathbf{F}_8(:, \text{cols}) \cdot \mathbf{x}(\text{cols}) = \begin{bmatrix} \mathbf{F}_4 & \Omega_4 \mathbf{F}_4 \\ \mathbf{F}_4 & -\Omega_4 \mathbf{F}_4 \end{bmatrix} \begin{bmatrix} \mathbf{x}(1:2:8) \\ \mathbf{x}(2:2:8) \end{bmatrix} = \begin{bmatrix} \mathbf{I}_4 & \Omega_4 \\ \mathbf{I}_4 & -\Omega_4 \end{bmatrix} \begin{bmatrix} \mathbf{F}_4 \mathbf{x}(1:2:8) \\ \mathbf{F}_4 \mathbf{x}(2:2:8) \end{bmatrix}.$$

Thus by simple scalings we can obtain the 8-point DFT $\mathbf{y} = \mathbf{Fg}\mathbf{x}$ from the 4-point DFTs $\mathbf{Y}_T = \mathbf{F}_4 \mathbf{x}(1:2:8)$ and $\mathbf{Y}_B = \mathbf{F}_4 \mathbf{x}(2:2:8)$. In particular,

$$\begin{aligned} \mathbf{y}(1:4) &= \mathbf{Y}_T + \mathbf{d} \cdot^* \mathbf{Y}_B \\ \mathbf{y}(5:8) &= \mathbf{Y}_T - \mathbf{d} \cdot^* \mathbf{Y}_B \end{aligned}$$

where

$$\mathbf{d} = \begin{bmatrix} 1 \\ \omega \\ \omega^2 \\ \omega^3 \end{bmatrix}.$$

More generally, if $n = 2m$ then $\mathbf{y} = \mathbf{F}_n \mathbf{x}$ is given by

$$\begin{aligned} \mathbf{y}(1:m) &\leftarrow \mathbf{y}(1:m) + \mathbf{d} \cdot^* \mathbf{Y}_B \\ \mathbf{y}(m+1:n) &\leftarrow \mathbf{y}(m+1:n) - \mathbf{d} \cdot^* \mathbf{Y}_B \end{aligned}$$

where $\mathbf{d} = [1, w_n, \dots, \omega_n^{m-1}]^T$ and

$$\begin{aligned} \mathbf{Y}_T &= \mathbf{F}_m \mathbf{x}(1:2n), \\ \mathbf{Y}_B &= \mathbf{F}_m \mathbf{x}(2:2n). \end{aligned}$$

For $n = 2$, we can recur on this process until $n = 1$, noting that $\mathbf{F}\mathbf{1}\mathbf{x} = \mathbf{x}$.

Implementation: If $\mathbf{x} \in \mathbb{J}_n$ and $n = 2$, then this algorithm computes the discrete Fourier transform $\mathbf{y} = \mathbf{F}\mathbf{x}$.

```
function y = fft(x, n)
    if n == 1
        y = x
    else
        m = n/2
        YT = fft(x(1:2n), m)
        Ya = ff(x(2:2n), m)
        w = exp(-2pi/n)
        d = [1 w ... wn]^T
        z = d * YB
        y = [YT + z; YT - z]
    end
```

The flop analysis of A requires an assessment of complex arithmetic and the solution of an interesting recursion. We first observe that the multiplication of two complex numbers involves six (real) flops while the addition of two complex numbers involves two flops. Let f_n be the number of flops that A needs to produce the DFT of $x \in \mathbb{C}^n$. Scrutiny of the method reveals that

$$\left\{ \begin{array}{l} n \\ y \end{array} \right\} \text{ requires } \left\{ \begin{array}{l} f_m \text{ flops} \\ f_m \text{ flops} \\ 6m \text{ flops} \\ 6m \text{ flops} \\ 2n \text{ flops} \end{array} \right\}$$

where $n = 2m$. Thus,

$$f_n = 2f_m + 8n \quad (!1.10).$$

Conjecturing that $f_n = Cn \log_2(n)$ for some constant C , it follows that

$$f_{2^n} - 1 \approx n \log_2(n) = 2Cm \log_2(m) + 8n \approx Cn(\log_2(n) - 1) + 8n,$$

from which we conclude that $C = 8$. Thus, A requires $8n \log_2(n)$ flops. Appreciate the speedup over conventional matrix-vector multiplication. If $n \approx 2^k$, it is a factor of about 10000. We mention that the flop count can be reduced to $5n \log_2(n)$ by precomputing W_1, W_2, \dots, W_{k-1} . See P1.4.1.

hisuib dtef buTlsb ttib kTebs teex tf Tkeb

In the discrete sine transform (DST) problem we are given real values x_1, \dots, x_m and compute

$$y_k = \sum_{j=1}^{m-1} \sin\left(\frac{k}{m} j\right) x_j \quad (143)$$

for $k = 1:m-1$. In the discrete cosine transform (DCT) problem we are given real values x_0, X_1, \dots, X_m and compute

$$Y_k = \frac{x_0}{2} + \sum_{j=1}^{m-1} \cos\left(\frac{k}{m} j\right) X_j + \frac{(-1)^k x_m}{2} \quad (144)$$

for $K \in \{m\}$. Note that the sine and cosine evaluations "show up" in the DFT matrix. Indeed, for $k \in \{0:m-1\}$ and $j \in \{0:m-1\}$ we have

$$F_{2m}^{-1}(k, j+1) = \frac{1}{\sqrt{2}} (-1)^j \cos\left(\frac{k}{m} j\right) - i \cdot \frac{1}{\sqrt{2}} \sin\left(\frac{k}{m} j\right). \quad (145)$$

This suggests (correctly) that there is an exploitable connection between each of these trigonometric transforms and the DFT. The key observation is to block properly the real and imaginary parts of F_{2m}^{-1} . To that end, define the matrices $S_r E_w X^r$ and $G_r E_w X^r$ by

$$\begin{aligned} [S_r]_{kj} &= \sin\left(\frac{k}{m} j\right), & k = 1:r, j = 1:r. \\ [G_r]_{kj} &= \cos\left(\frac{k}{m} j\right), \end{aligned} \quad (146)$$

Recalling from §1.2.11 the definition of the exchange permutation \mathbf{f}_n , we have

Theorem 1.4.1. Let m be a positive integer and define the vectors e and v by

$$\mathbf{e}^T = (\underbrace{1, 1, \dots, 1}_{m+1}), \quad \mathbf{v}^T = (\underbrace{-1, 1, \dots, (-1)^{m+1}}_{m+1}) \mathbf{h}^P$$

If $E = G_m \mathbf{f}_m C = C_m \mathbf{f}_m$ and $S = B_m \mathbf{f}_m$ then

$$\mathbf{F}_{2m} = \begin{bmatrix} e^T & 1 & \mathbf{e}^T \\ C - iS & v & (C + iS)E \\ 1 & v^T & (-1)^m \\ e \cdot E(C + iS) & Ev & E(C - iS)E \end{bmatrix}. \quad (147)$$

Proof. It is clear from (145) that $\mathbf{F}_{2m}(:, 1)$, $\mathbf{F}_{2m}(1, :)$, $\mathbf{F}_{2m}(:, m+1)$, and $\mathbf{F}_{2m}(m+1, :)$ are correctly specified. It remains for us to show that equation (147) holds in blocks positions (22), (24), (42), and (44). The (22) verification is straightforward:

$$\begin{aligned} [\mathbf{F}_{2m}(2m, 2m)]_{kj} &= \cos\left(\frac{kj\pi}{m}\right) - i \sin\left(\frac{kj\pi}{m}\right) \\ &= [C - iS]_{kj}. \end{aligned}$$

A little trigonometry is required to verify correctness in the (24) position:

$$\begin{aligned} [\mathbf{F}_{2m}(2m, m+22m)]_{kj} &= \cos\left(\frac{(m+22m)is}{m}\right) - i \sin\left(\frac{(m+22m)is}{m}\right) \\ &= \cos\left(\frac{0}{m} + i\pi\right) - i \sin\left(\frac{0}{m} + i\pi\right) \\ &= \cos\left(\frac{j}{m} + k\pi\right) + i \sin\left(\frac{j}{m} + k\pi\right) \\ &= \cos\left(\frac{(m+k)is}{m}\right) + i \sin\left(\frac{(m+k)is}{m}\right) \\ &= [(C + iS)E]_{kj}. \end{aligned}$$

We used the fact that post-multiplying a matrix by the permutation $E = G_m \mathbf{f}_m$ has the effect of reversing the order of its columns. The recipes for $\mathbf{F}_{2m}(m+22m, 2m)$ and $\mathbf{F}_{2m}(m+22m, m+22m)$ are derived similarly. \square

Using the notation of the theorem, we see that the sine transform (143) is a matrix-vector product

$$\mathbf{y}(1:m-1) = \mathbf{DST}(m-1) \cdot \mathbf{x}(1:m-1)$$

where

$$\text{DST}(m-1) = S_m \quad (148)$$

If $x = x(1:m-1)$ and

$$x_m = \begin{bmatrix} \vdots \\ x \end{bmatrix} \in \mathbb{R}^{2m} \quad (149)$$

then since $e^T E = e$ and $E^2 = E$ we have

$$\begin{aligned} i F_{2m} x_m &= i \begin{bmatrix} 1 & e^T & 1 & e^T \\ e & C - iS & v & (C + iS)E \\ 1 & VT & (-1)^m & v^T E \\ e & E(C + iS) & Ev & E(C - iS)E \end{bmatrix} \begin{bmatrix} 0 \\ x \\ 0 \\ -Ex \end{bmatrix} \\ &= \frac{i}{2} \begin{bmatrix} e^T x - e^T Ex \\ -2iSx \\ v^T x - v^T E^2 x \\ i(ESx + ESE^2 x) \end{bmatrix} = \begin{bmatrix} 0 \\ Sx \\ 0 \\ -ESx \end{bmatrix}. \end{aligned}$$

Thus, the DST of $x(1:m-1)$ is a scaled subvector of $F_{2m} x_m$.

At this point, the following algorithm signs the DST of x_1, \dots, x_m to y .

Set up the vector X_m defined by (149).

Use i (eg, Algorithm 14.1) to compute $i = F_{2m} x_m$

$$y = i \cdot y(2m)/2$$

This computation involves $O(m \log_2(m))$ fops. We mention that the vector X_m is real and highly structured, something that would be exploited in a truly efficient implementation.

Now let us consider the discrete cosine transform defined by (144). Using the notation from Theorem 14.1, the DCT is a matrix-vector product

$$y(0m) = \text{DCT}(m/2) \cdot x(0m)$$

where

$$\text{DCT}(m/2) = \begin{bmatrix} 1/2 & e^T & 1/2 \\ e/2 & C_m/2 & v/2 \\ 1/2 & VT & (-1, 1, \dots, 1) \end{bmatrix} \quad (1410)$$

If $x = x(1:m-1)$ and

$$x_{\cos} = \begin{bmatrix} x_0 \\ \tilde{x} \\ x_m \\ E\tilde{x} \end{bmatrix} \in \mathbb{R}^{2m}, \quad (1411)$$

then

$$\begin{aligned} \frac{1}{2} F_{2m} x_{\cos} &= \frac{1}{2} \begin{bmatrix} e^T & 1 & e^T \\ C - iS & v & (C + iS)E \\ VT & (-1)^m & v^T E \\ E(C + iS) & Ev & E(C - iS)E \end{bmatrix} \begin{bmatrix} x_0 \\ x \\ x_m \\ Ex \end{bmatrix} \\ &= \begin{bmatrix} (x_0/2) & + & e^T Vx & + & (x_m/2) \\ (x_0/2)e & + & Cx & + & (x_m/2)v \\ (x_0/2) & + & VTx & + & (-1)^m(x_m/2) \\ (x_0/2)e & + & ECx & + & (x_m/2)Ev \end{bmatrix}. \end{aligned}$$

Notice that the top three components of this block vector define the DCT of $x(0:m)$. Thus the DCT is a scaled subvector of $F_{2m} x_{\cos}$.

π s ,2-(i) The following algorithm signs to yield \mathbf{ER}^{m+1} the DCT of x_0, \dots, x_m .

Set up the vector $x_{\cos} \in \mathbb{R}^{2m}$ defined by (1.4.11).

Use \mathbf{z} (eg, Algorithm 1.4.1) to compute $\mathbf{y} = F_{2m} x_{\cos}$

$$\mathbf{y} = \mathbf{y}(1:m+1)/2$$

This algorithm requires $O(m \log m)$ fops, but as with Algorithm 1.4.2, it can be more efficiently implemented by exploiting symmetries in the vector x_{\cos} .

We mention that there are important variants of the DST and the DCT that can be computed fast:

$$\begin{aligned} \text{DST-II: } \mathbf{Y}_k &= \sum_{j=1}^m \left(\frac{k(2j-1)\pi}{2m} \right) x_j, & k &= 1:m \\ \text{DST-III: } \mathbf{Y}_k &= \sum_{j=1}^m \left(\frac{(2k-1)j\pi}{2m} \right) x_j, & k &= 1:m \\ \text{DST-IV: } \mathbf{Y}_k &= \sum_{j=1}^m \left(\frac{(2k-1)(2j-1)\pi}{2m} \right) x_j, & k &= 1:m \\ \text{DCT-II: } \mathbf{Y}_k &= \sum_{j=0}^{m-1} \cos\left(\frac{k(2j-1)\pi}{2m}\right) x_j, & k &= 0:m-1, \\ \text{DCT-III: } \mathbf{Y}_k &= \frac{x_0}{2} + \sum_{j=1}^{m-1} \cos\left(\frac{(2k-1)j\pi}{2m}\right) x_j, & k &= 0:m-1, \\ \text{DCT-IV: } \mathbf{Y}_k &= \sum_{j=0}^{m-1} \cos\left(\frac{(2k-1)(2j-1)\pi}{2m}\right) x_j, & k &= 0:m-1. \end{aligned} \tag{1.4.12}$$

For example if $\mathbf{y} \in \mathbb{R}^{2m-1}$ is the DST of $\mathbf{x} = [x_1, 0, x_2, 0, \dots, 0, x_{m-1}, x_m]^T$, then $\mathbf{f}(1:m)$ is the DST-II of $\mathbf{x} \in \mathbb{R}^m$. See Van Loan (FFT) for further details.

$$\|n\|_A \|A\| \leq \|A\| \leq \|A\|_A \|n\|$$

If $n = 2$, then the Haar wavelet transform $= W_k$ is a matrix-vector product in which the transformation matrix $W \in \mathbb{R}^{n \times n}$ is defined recursively:

$$W = \begin{cases} [W_m (\cdot) I_m (\cdot)] & \text{if } n = 2m \\ [1] & \text{if } n = 1. \end{cases}$$

Here are some examples:

$$W_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},$$

$$W_4 = \left[\begin{array}{cc|cc} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ \hline 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{array} \right],$$

$$W_8 = \left[\begin{array}{cccc|cccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & -1 & 0 & 0 \\ \hline 1 & -1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & -1 & 0 & 1 & 0 & 0 & -1 & 0 \\ 1 & -1 & 0 & -1 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & -1 & 0 & 0 & 0 & -1 \end{array} \right].$$

An interesting block pattern emerges if we reorder the rows of W so that the odd indexed rows come first:

$$P_{2,m}^T W_n = \begin{bmatrix} W_m & I_m \\ W_m & -I_m \end{bmatrix} = (W_2 \otimes I_m) \begin{bmatrix} W_m & 0 \\ 0 & I_m \end{bmatrix}. \quad (14.13)$$

Thus, if $x \in \mathbb{R}^n$, $x_T = x(1:m)$, and $X_T = x(r+1:n)$, then

$$\begin{aligned} y &= Wx = P_{2,m} \begin{bmatrix} I_m & I_m \\ I_m & -I_m \end{bmatrix} \begin{bmatrix} W_m & 0 \\ 0 & I_m \end{bmatrix} \begin{bmatrix} X_T \\ X_S \end{bmatrix} \\ &= P_{2,m} \begin{bmatrix} W_m x_T + x_B \\ W_m x_T - x_B \end{bmatrix}. \end{aligned}$$

In other words,

$$y(1:2n) = W_m X_T + X_B \quad y(2:2n) = W_m X_T - X_B.$$

This points the way to a fast recursive procedure for computing $y = Wx$.

If E_w and $n = 2$, then this algorithm computes the Haar transform $y_w = W_w x$.

function $y = t(x,n)$

if n= 1

$$y = x$$

else

$$m = n/2$$

= = t(x(l:m), m)

$$y(1:2m) = - + x(m+1:n)$$

$$(22m) = -x(m+ln)$$

end

It can be shown that this algorithm requires $2n^2$ ops.

Problems

P1.4.1 .2.8. $w = e_w h w_1 \dots w_{21} h_c - at, \dots$

$$\text{P1.4.2} \dots .8.8 =)(-“)1+(“)$$

$$G = F_{\text{MLC}} \circ F_{\text{allc}} \circ F_{\text{LLQ}}$$

-iyiq - nA,jn63-1nde2l3)Ann(A,=0 y Em₁ 2 uRA E t = RQW RfjBn
RfjBn RfjBn RfjBn RfjBn RfjBn RfjBn RfjBn RfjBn RfjBn RfjBn RfjBn RfjBn

RRI RR β_2 α_2 AnR Ra xW. n RY2 x An y Ryx

$$L_q = 2^q, \quad r_q = n/L_q,$$

$$A_q = I_{r_q} \otimes \begin{bmatrix} I_{L_{q-1}} & \Omega_q \\ I_{L_{q-1}} & -\Omega_q \end{bmatrix},$$

$$Iq'2q I_{L_{q-1}},$$

! GET NO_{W_q}, ..., W_{W_{q-1}-1}).

YETXR n Mylk Q2k

$y=x$
for $q=1:t$
 $y=Ag$
and

R DE qOOva .d< dEo a 3wop qOYVARak R1 121 ARy RTRh kW mPAZ mylk
TC2 6 2W

P1.4.4 f3.1 1.8>18.8..8.8.1. 8>W_W with G2

P1.4.4 ro... ((1-p)(3-p)) de=fA)=e)m= -2yt yn Z ARI RAR W An

$$H_1 \left[\begin{matrix} 2 & v \\ v & L-L \end{matrix} \right] \left[\begin{matrix} W_h & R \\ R & I_{n-L} \end{matrix} \right] \quad L = 2^q, \quad L_* = L/2.$$

do \underline{n}_{mi} . **B**(R^n) **P**(R^n) **V**(R^L) **V**(R^L) \perp **Q** **sk**

```

y = x
for q = 1:t
    y = Hqy
end

```

P14.6 .(1.4.13), ,
 $\mathbf{n} = \mathbf{2}$

Notes and References for

1 e .58 fe 16 b .8g .8 P . .6t. l. 8 4 .8bt. 8.p . P8 . ro

f 5P. . . . e 8. (1995). The DFT: An Owners' Manual for the Discr te Fouier T nsfor , hU4 S2eS.eOSO(aSem Ase

ce, ®arSgAe,) iM{Pp ..im S enO.tpS.Stee 24ASuAlr t,Aoar .IAu,AtoEsSpr ge9m
S2 2ln08 6L r .Q8wX P Yxx p t=(Aa =fiaBxK 80-BT

M. (vQ) =001 k= -1 Qn(2005). , 120XQ xx). u IuA):nQ) - b<, ly Pr ceedings of the IEEE, 93 216-231.

$$\left(\frac{1}{2} - (-) \right) \left(\frac{1}{2} + \left(\begin{array}{c} 1 \\ 1 \end{array} \right) \right) = \left(\begin{array}{cc} 1 & 1 \\ 1 & 1 \end{array} \right)$$

J. Sci. Comput. 20:1094–1114, 2004. © 2004 SIAM

J. Sci. Comput. 20, 1694-1714.
-2) (9 (11,1993). 2, 9) 6((-)-,) SIAM
J. Sci. Comput. 14, 1368-1393.

⁶ 6 1-8(, , 1999). 1 -10, 9((1.,) 9 (1 , , -
⁶ 1 - SIAM J. Sci. Comput. 21, 283-293.
⁶ (, -2 1999) (20) (6 4 6(

J. Lin. Alg. Appl. 366, 337–351. Lee (2004). 1 9 ((1 9 (, . – SIAM

Review 46, 443-454.
 .1 1)(,)(H , ¥1C (2005). 921, .(6 1 9((-

1 6 2 -12 1 1 91 } 1 } { } ()

¹ See, e.g., *U.S. v. Babbitt*, 100 F.3d 1250, 1254 (10th Cir. 1996) (“[T]he [EPA] has authority to regulate the discharge of pollutants into waters of the United States.”); *U.S. v. Lummus Co.*, 100 F.3d 1250, 1254 (10th Cir. 1996) (“[T]he [EPA] has authority to regulate the discharge of pollutants into waters of the United States.”).

73, 325-348.
v. On "OMX(n)" ($\alpha^{(D)}$). 29 (2) (SIAM Review)

5 9(1) b 2) (1991). The B2a o s¹n + 1 = SIAM J. Sci. Statist. Comput. 12,

79-94. " — 9 () 6 () 1 () (2010, "Tq 14 () —

x. SIAM J. Sci. Comput. 32: 3092-3107.
x. på =10) (200). 4 9IA (rl@205r A9ar,tOSln OM iecrr 929r u

SIAM J. MatTx Anal. Appl. 24, 768–786.

- (1992). Ten Lectures on Wavelets, *hu4 S2GSe. Sln2iS.. maA-S.; 4*
 5 htpengj993. 9 1 '2) 1 9 --, 1) Bull. AMS 28, 288-305.
 1, -))) 1 6 -- (1996). Wavelets and Filter Banks, *uA-r -EP MeuGr* 1996.

1.5 Vectorization and Locality

When it comes to designing a high-performance matrix computation, it is not enough simply to minimize FLOPs. Attention must be paid to how the arithmetic units interact with the underlying memory system. Data structures are an important part of the picture because not all matrix layouts are "architecture friendly." Our aim is to build a practical appreciation for these issues by presenting various simplified models of execution. These models are qualitative and are just informative pointers to complex implementation issues.

high-level flowchart for floating-point addition

An individual floating-point operation typically requires several cycles to complete. A 3-cycle addition is depicted in Figure 151. The input scalars x and y proceed along

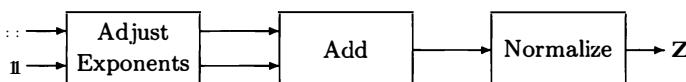


Figure 151 A 3-Cycle adder

a computational "assembly line," spending one cycle at each of three work "stations." The sum z emerges after three cycles. Note that, during the execution of a single, "free-standing" addition, only one of the three stations would be active at any particular instant.

Vector processors exploit the fact that a vector operation is a very regular sequence of scalar operations. The key idea is pipelining, which we illustrate using the vector addition computation $\mathbf{z} = \mathbf{x} + \mathbf{y}$. With pipelining, the \mathbf{x} and \mathbf{y} vectors are streamed through the addition unit. Once the pipeline is filled and steady state reached, a \mathbf{z} -vector component is produced every cycle, as shown in Figure 152. In

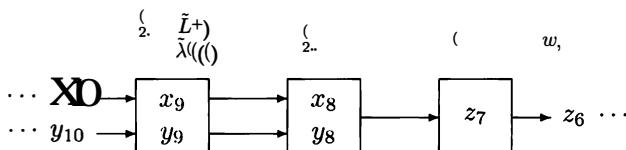


Figure 152 Pipelined addition

this case, we would anticipate vector processing to proceed at about three times the rate of scalar processing.

A vector processor comes with a repertoire of vector instructions, such as vector add, vector multiply, vector scale, dot product, and swap. These operations take place in vector registers with input and output handled by vector load and vector store instructions. An important attribute of a vector processor is the length v_L of the vector registers that carry out the vector operations. A length- n vector operation must be broken down into subvector operations of length v_L or less. Here is how such a partitioning might be managed for a vector addition $\mathbf{z} = \mathbf{x} + \mathbf{y}$, where \mathbf{x} and \mathbf{y} are n -vectors:

```

first = 1
while first < n
    last = min{n, first + v_L - 1}
    Vector load r1 ← x(first:last)
    Vector load r2 ← y(first:last)
    Vector add r1 = r1 + r2
    Vector store z(first:last) ← r1
    first = last + 1
end

```

(1.5.1)

The vector addition is a register-register operation while the "flopless" movement of data to and from the vector registers is identified with the left arrow " \leftarrow ". Let us model the number of cycles required to carry out the various steps in (1.5.1). For clarity, assume that n is very large and an integral multiple of v_L , thereby making it safe to ignore the final cleanup pass through the loop.

Regarding the vectorized addition $r1 = r1 + r2$ assume it takes τ_{add} cycles to fill the pipeline and that once this happens, a component of z is produced each cycle. It follows that

$$N_{\text{arith}} = \left(\frac{n}{v_L}\right)(\tau_{\text{add}} + v_L) = \left(\frac{\tau_{\text{add}}}{v_L} + 1\right)n$$

accounts for the total number of cycles that (1.5.1) requires for arithmetic.

For the vector loads and stores, assume that $\tau_{\text{data}} + v_L$ cycles are required to transport a length- v_L vector from memory to a register or from a register to memory, where τ_{data} is the number of cycles required to fill the data pipeline. With these assumptions we see that

$$N_{\text{data}} = 3 \left(\frac{n}{v_L}\right)(\tau_{\text{data}} + v_L) = 3 \left(\frac{\tau_{\text{add}}}{v_L} + 1\right)n$$

specifies the number of cycles that are required by (1.5.1) to get data to and from the registers.

The arithmetic-to-data-motion ratio

$$\frac{N_{\text{arith}}}{N_{\text{data}}} = \frac{\tau_{\text{add}} + v_L}{3(\tau_{\text{data}} + v_L)}$$

and the total cycles sum

$$N_{\text{arith}} + N_{\text{data}} = \left(\frac{\tau_{\text{arith}} + 3\tau_{\text{data}}}{v_L} + 4\right)n$$

are illuminating statistics, but they aren't necessarily good predictors of performance. In practice, vector loads, stores, and arithmetic are "overlapped" through the chaining together of various pipelines, a feature that is not captured by our model. Nevertheless, our simple analysis is a preliminary reminder that data motion is an important factor when reasoning about performance.

1t4t.A Hpf A:0:N+A SN-0Aa7.INi -A

Two algorithms that involve the same number of fops can have substantially different data motion properties. Consider the n-by-n gaxy

$$y = y + Ax$$

and the n-by-n outer product update

$$A = A + yx^T.$$

Both of these level-2 operations involve $2n^2$ fops. However, if we assume (for clarity) that $n = v_L$, then we see that the gaxy computation

```

rx+ x
ry+ y
for j = 1:n
    ra+ A(:,j)
    ry= ry+ raxj)
end
y + ry

```

requires $(3+n)$ load/store operations while for the outer product update

```

rx+ x
ry+ y
for j = 1:n
    ra+ A(:,j)
    ra= ra+ ryxj)
    A(:,j) + ra
end

```

the corresponding count is $(2+2n)$. Thus, the data motion overhead for the outer product update is worse by a factor of 2, a reality that could be a factor in the design of a high-performance matrix computation

highmb ,h\$ b W\$ f\$ tlf \$bklb uf ehsb

The time it takes to load a vector into a vector register may depend greatly on how the vector is laid out in memory, a detail that we did not consider in §15.1. Two concepts help frame the issue. A vector is said to have unit stride if its components are contiguous in memory. A matrix is said to be stored in column-major order if its columns have unit stride.

Let us consider the matrix multiplication update calculation

$$C = C + AB$$

where it is assumed that the matrices $C \in \mathbb{R}^{mn}$, $A \in \mathbb{R}^{J \times r}$, and $B \in \mathbb{R}^{I \times n}$ are stored in column-major order. Suppose the loading of a unit-stride vector proceeds much more quickly than the loading of a non-unit-stride vector. If so, then the implementation

```

f rj = 1:n
f rk = 1:r
    C(:,j) = C(:,j) + A(:,k)  (:,j)
end
end

```

which accesses C, A, and B by column would be preferred to

```

f ri = 1:m
f rj = 1:n
    C(i,:) = C(:,j) + A(i,:)
end
end

```

which accesses C and A by row. While this example points to the possible importance of stride, it is important to keep in mind that the penalty for non-unit-stride access varies from system to system and may depend upon the value of the stride itself.

high) b u fkfdTt"b eeb etf tb 9SDSb

Matrices reside in memory but remain has levels. A typical arrangement is depicted in Figure 153. The cache is a relatively small high-speed memory unit that sits

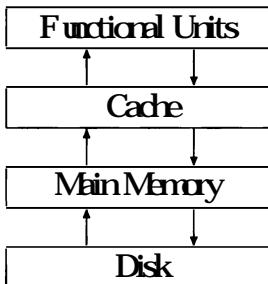


Figure 1 . . . A memory hierarchy

just below the functional units where the arithmetic is carried out. During a matrix computation, matrix elements move up and down the memory hierarchy. The cache, which is a small high-speed memory situated in between the functional units and main memory, plays a particularly critical role. The overall design of the hierarchy varies from system to system. However, two maxims always apply:

- Each level in the hierarchy has a limited capacity and for economic reasons this capacity usually becomes smaller as we ascend the hierarchy.
- There is a cost, sometimes relatively great, associated with the moving of data between two levels in the hierarchy.

The efficient implementation of a matrix algorithm requires an ability to reason about the flow of data between the various levels of storage.

To develop an appreciation for cache utilization we again consider the update $A = C + AB$ where each matrix is n -by- n and blocked as follows:

$$C = \begin{bmatrix} C_{11} & \cdots & C_{1r} \\ \vdots & \ddots & \vdots \\ C_q & \cdots & C_{qr} \end{bmatrix}, \quad A = \begin{bmatrix} A_{11} & \cdots & A_{1p} \\ \vdots & \ddots & \vdots \\ A_{qr} & \cdots & A_{qp} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & \cdots & B_{1r} \\ \vdots & \ddots & \vdots \\ B_{q1} & \cdots & B_{qr} \end{bmatrix}.$$

Assume that these three matrices reside in main memory and that we plan to update A block by block:

$$G_j = G_j + \sum_{k=1}^p A_{ik} B_{kj}.$$

The data in the blocks must be brought up to the functional units via the cache which we assume is large enough to hold a C -block, an A -block, and a B -block. This enables us to structure the computation as follows:

```

for i = 1:q
  for j = 1:r
    Load C_{ij} from main memory into cache
    for k = 1:p
      Load A_{ik} from main memory into cache
      Load B_{kj} from main memory into cache
      C_{ij} = C_{ij} + A_{ik} B_{kj}
    end
    Store A_{11} in main memory.
  end
end

```

(1.5.4)

The question before us is how to choose the blocking parameters q , r , and p so as to minimize memory traffic to and from the cache. Assume that the cache can hold 4 floating point numbers and that $11 < 3n^2$, thereby forcing us to block the computation. We assume that

$\left. \begin{array}{c} G_j \\ A_{ik} \\ B_{kj} \end{array} \right\}$ is roughly $\left\{ \begin{array}{l} (n/q)\text{-by-}(n/r) \\ (n/q)\text{-by-}(n/p) \\ (n/p)\text{-by-}(n/r) \end{array} \right.$

We say "roughly" because if q , r , or p does not divide n , then the blocks are not quite uniformly sized, e.g.,

$$A = \left[\begin{array}{ccc|ccc|ccc|c} \times & \times \\ \times & \times \\ \times & \times \\ \times & \times \\ \hline \times & \times \\ \times & \times \\ \times & \times \\ \times & \times \\ \hline \times & \times \\ \times & \times \end{array} \right], \quad \begin{array}{l} n = 10, \\ q = 3, \\ p = 4. \end{array}$$

However, nothing is lost in glossing over this detail since our aim is simply to develop an intuition about cache utilization for large problems. Thus, we are led to impose the following constraint on the blocking parameters:

$$(\%) + (\%) + (\%) \quad \{155\}$$

Proceeding with the optimization, it is reasonable to minimize the amount of arithmetic associated with the update $\mathbf{G}_j = \mathbf{G}_j + \mathbf{A}[\mathbf{i}, \mathbf{j}] \mathbf{B}$. After all, we have moved matrix data from main memory to cache and should make the most of the investment. This leads to the problem of maximizing $2n^3/(qp)$ subject to the constraint (155). A straightforward Lagrange multiplier argument leads us to conclude that

$$q_{opt} = p_{opt} = t_{opt} = \sqrt{\frac{n^2}{3}} \quad \{156\}$$

That is, each block of C, A, and B should be approximately square and occupy about one third of the cache.

Because blocking affects the amount of memory traffic in a matrix computation, it is of paramount importance when designing a high-performance implementation. In practice, things are never as simple as in our model example. The optimal choice of Q_{opt} , R_{opt} , and P_{opt} will also depend upon transfer rates between memory levels and upon all the other architecture factors mentioned earlier in this section. Data structures are also important; storing a matrix by block rather than in column-major order could enhance performance.

Problems

P1.5.3 .2. 8.8 A ER^{m × n} - mⁿ, i - m^rdkⁱ 1. mⁱ. - nⁱ - 1. m = m₁M₁, n = n₁: .
 m x u4Q+Qx*a 2r 2s = • u 8. Xfixbyxa)f d(1, 1C):, y :), -: A , 1
 d(f): A.block(1:m n) , , (2:2(, ^T , 1, (f: A)A_{ij} , ,):(- f), n)n, Tf)cM:, y-)-
)- (:

Notes and References for §1.5

78>8. .8. f-8. 8. 8. 82.2 .8.2f8

k **83.....6 fe N...86 f 1 or... (1984). -g(),(, 5,(,) .1,), .,(,),.) 11
19(1, 1),,9 ((g,-,w,(, 1, ,((P SIAM Review 26 91-112.**

85.1 -,-,-,-

rB kSuIG PLT4h m1,m4 T m4,7 1 h0kk Tit TMSj rTOCz wIA R8t14h
 ktS LShn94(m1 HkU14p 11Sns9m nApAnUO GACM ns. Math Sof w. 24,
 NuB INI
 11wmc i2m .ohw MdTeiJa3Uh)914h, 4 9elaamUuA,c1cmtitM2ccs B914h vI
 5Uh)(l.cAA,A=A=LCM ns. Math Sof w. 25, Biu B1 1
 xAR, II ItJ = i,Ati,Al= U11nAlh=i.m 9w tiAt51r N1d1e5c,,A.c 9sleam4sil.MtJ 1An.
 soMmiti htI,t,lcAv. anAcrit,f loi.. hbBtpic=rSIAM Review 46 B./)w
 Mltl 25m. c ici2 r1Ndb4itlu lBsMiJr (cv.u12critMf,stM3sMhACM
 ns. Math Sof w. 34, MN, MN, N)w

4me2cmmitiAt.,T odATAO33l.tJ.iJ 3cv.ui2c lit.f l1A,titMIA i.c mMA,AAfcm
 .wi i,AtC.A2 r Mdw5c.,A.l2 lcmAt 4tlitM. Miq8laM2iv .cnAclM2ci4icoi
 4sil .t,u A=fEM J. Res. Dev. 41, B. I))w
 .w.isAi,Cu inmAl he3,,u r MNdw4,i1cp1le v. sMiJ' (cv.li1ri t.f r,,tAs.itM12
 9iAcrt scil.JM.H4oAti.tl2A-4ill MtJ 1A2mlAt 1Tcnllpr lc.cs cncaTconur ncy
 Comput. Pr ct. E. er. 14 y)eyBtw
 hwoJittcJac (1(it2isi=i2mrw iJltthtJlmMN11e5c,IAM.4.i. xi,l,tA i2m1t ritlMf
 r,stM3sM.itNHE2E ns. Par llel Distrib Syst 13, MMWNBw
 .Jiwi,Ati,Al. r NB dcsMiJr (cv.uin.M2ci4,icd i 4sil.MtJ 1A.ni ic2ciisMTtitn ht.,
 t1hAv. rit.cA=rEM J. Res. Dev. 47, B,M w
 CI(Ce=9i slni= .ml wli2i r NB dk1.eM291.e .iti li. l,t= i.m rcull. sMcI.i.J.
 (cv.li2 =LIEEE ns. Par llel Distrib Systems, 14 u/ 1 w
 114s in2hAy1 i,Ati.bn= i l,r 1 sc.= i2m5w4i2 mdicM1r N1d1c4.iuM. lBsMiJ
 (cv.li1c ritlMf,stMAt. 4illMf1A=PARA 2004 LNCS 3732, N)uNw
 (I.r 4o3tlinmAl CMT,r;itdwc4mi3M1cMnli.imriA.f r,etM3sMhACM ns.
 Math Sof w. 36, Bi8B,NB 1

4 i,ct mcis dB ltJ1 ilnc .tl tJhmca,i2T ABtpi,cdl,A tJit i,tl1CtM,i.e. osle C1Ct.f
 l13titMln v. JMjJ 3cv.ui2t14h

hwi.. i2m5w9el.. r tl dc lu3M,c9sl.eiasMfT.cBacrit.Mfi .tl.Ti tlnA=ACM ns.
 Math Sof w. 23, BBuBL

II 4iVA2csA-i1i0St AldHP 1sc2= i2 5141.e mdic.J2r NMdw4rR, luis lnci.
 4sicoi PctJlmRn.M.I.1cnt=TCM ns. Math Sof w. 27, /N188w
 (w9c2tncA114i,22csA-rIR lr.c.A=Rvp,Mt2er 1tM2 5w4Cnmdic.Jnr N1d1ciJc
 h.c2c lB.c....i .3Ac M21sicoi 4sil .t,uA=TCM ns. Math Sof w. 31, M,Nu
 lwc1 lce=1w2iJe= I RMI,t= RlantAll (ctMtc5w,m,- 5)wuJi,c= i.m ML sMe
 r N1twhrsBr 4mi3Mai 4sicoi 4sillMtJuArhLBt8i,c= IPr c IEEE 93, NtBBiNw
 wtl= O1M1 5c2=rI ii..i..= .1 (imj= I,nii ..=inm(l htmi,,e 1)N cUAcil.J
 5ci... accAAiT. idcit3 2MiJr (cv li2t14h =Er . IEEE 93, B)yBuw

l i Millo tcit1c2t lB,lu1,2,itMln sT pt,mA.n lit.f l13titlA= Acc,
 iw9i,si.m4wc11 cs. 3= A1T Q. OE• i)AN 102DA08 JX1108xy8 OE)j0XnAOE)A
 SO= ny)Ay nSIAM J. Matr. Anal. Applic. 32, yuut,liw

1.6 Parallel Matrix Multiplication

The impact of matrix computation research in many application area depends upon the development of parallel algorithms that scale. Algorithms that scale have the property that they remain effective as problem size grows and the number of involved processors increases. Although powerful new programming languages and related system tools continue to simplify the process of implementing a parallel matrix computation, being able to "think parallel" is still important. This requires having an intuition about load balancing, communication overhead, and processor synchronization.

1t5A1A rA - LcxA ..ff } G:A

To illustrate the major ideas associated with parallel matrix computations, we consider the following model computation

Given $C \in \mathbb{R}^{m \times n}$, $A \in \mathbb{R}^{m \times r}$, and $B \in \mathbb{R}^{r \times n}$, effectively compute the matrix multiplication update $C = C + AB$ assuming the availability of p processors. Each processor has its own local memory and executes its own local program.

The matrix multiplication update problem is a good choice because it is inherently parallel computation and because it is at the heart of many important algorithms that we develop in later chapters.

The design of a parallel procedure begins with the breaking up of the given problem into smaller parts that exhibit a measure of independence. In our problem we assume the blocking

$$C = \begin{bmatrix} C_{11} & \cdots & C_{1N} \\ \vdots & \ddots & \vdots \\ C_{M1} & \cdots & C_{MN} \end{bmatrix}, \quad A = \begin{bmatrix} A_{11} & \cdots & A_{1R} \\ \vdots & \ddots & \vdots \\ A_{M1} & \cdots & A_{MR} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & \cdots & B_{1N} \\ \vdots & \ddots & \vdots \\ B_{R1} & \cdots & B_{RN} \end{bmatrix}, \quad (1.6.1)$$

$$m = m_1 M, \quad r = r_1 R, \quad n = n_1 N$$

with $C_{ij} \in \mathbb{R}^{m_1 \times n_1}$, $A_{ij} \in \mathbb{R}^{m_1 \times r_1}$, and $B_{ij} \in \mathbb{R}^{r_1 \times n_1}$. It follows that the $C + AB$ update partitions nicely into MN smaller tasks

$$\text{Task}(i,j): \quad C_{ij} = C_{ij} + \sum_{k=1}^n A_{ik} B_{kj}. \quad (1.6.2)$$

Note that the block-block products $A_{ik} B_{kj}$ are all the same size.

Because the tasks are naturally double indexed, we double index the available processors as well. Assume that $p = \text{ProcPd}$ and designate the (i,j) th processor by $\text{Proc}(i,j)$ for $i = 1:\text{ProcM}$ and $j = 1:\text{ProcN}$. The double indexing of the processors is just a notation and is not a statement about their physical connectivity.

hpabib lktb utnt tfTl"b

An effective parallel program equitably partitions the work among the participating processors. Two subdivision strategies for the model computation come to mind. The 2-dimensional block distribution assigns contiguous block updates to each processor. See Figure 1.6.1. Alternatively, we can have $\text{Proc}(\mu, i)$ oversee the update of C_{ij} for $i = \mu:\text{ProcM}$ and $j = f:\text{ProcN}$. This is called the 2-dimensional block-cyclic distribution. See Figure 1.6.2. For the displayed example, both strategies assign twelve C_{ij} updates to each processor and each update involves R block-block multiplications, i.e., $12(2m_1 n_1 r_1)$ fops. Thus, from the flop point of view, both strategies are load balanced by which we mean that the amount of arithmetic computation assigned to each processor is roughly the same.

Proc{1,1}	Proc{1,2}	Proc{1,3}
$\left\{ \begin{array}{l} C_{11} \ C_{12} \ C_{13} \\ C_{21} \ C_{22} \ C_{23} \\ C_{31} \ C_{32} \ C_{33} \\ C_{41} \ C_{42} \ C_{43} \end{array} \right\}$	$\left\{ \begin{array}{l} C_{14} \ C_{15} \ C_{16} \\ C_{24} \ C_{25} \ C_{26} \\ C_{34} \ C_{35} \ C_{36} \\ C_{44} \ C_{45} \ C_{46} \end{array} \right\}$	$\left\{ \begin{array}{l} C_{11} \ C_{1s} \ C_{1g} \\ C_{21} \ C_{2s} \ C_{2g} \\ C_{31} \ C_{3s} \ C_{3g} \\ C_{41} \ C_{4H} \ C_{4g} \end{array} \right\}$
Proc{2,1}	Proc{2,2}	Proc{2,3}
$\left\{ \begin{array}{l} Cs_1 \ Cs_2 \ Cs_3 \\ C_61 \ C_62 \ C_63 \\ C_{11} \ C_{12} \ C_{13} \\ Cs_1 \ Cs_2 \ Cs_3 \end{array} \right\}$	$\left\{ \begin{array}{l} Cs_4 \ Cs_5 \ Cs_6 \\ c_{64} \ c_{65} \ c_{66} \\ C_{14} \ C_{1s} \ C_{16} \\ Cs_4 \ Cs_5 \ Cs_6 \end{array} \right\}$	$\left\{ \begin{array}{l} Cs_1 \ Cs_5 \ Cs_g \\ c_{61} \ C_{6s} \ c_{6g} \\ C_{11} \ C_{1s} \ C_{1g} \\ Cs_1 \ Cs_5 \ Cs_g \end{array} \right\}$

Figure 16.1. The block distribution of tasks
(M = 8 Row= 2 N = 9 and Ptd = 3).

Proc{1,1}	Proc{1,2}	Proc{1,3}
$\left\{ \begin{array}{l} C_{11} \ C_{14} \ C_{11} \\ C_{31} \ C_{34} \ C_{31} \\ C_{51} \ Cs_4 \ Cs_1 \\ C_{11} \ C_{14} \ C_{11} \end{array} \right\}$	$\left\{ \begin{array}{l} C_{12} \ C_{15} \ C_{1s} \\ C_{32} \ C_{35} \ C_{3s} \\ Cs_2 \ Cs_5 \ Cs_s \\ C_{12} \ C_{1s} \ C_{1s} \end{array} \right\}$	$\left\{ \begin{array}{l} C_{13} \ C_m \ C_{1g} \\ C_{33} \ c_{36} \ C_{3g} \\ Cs_3 \ Cs_6 \ Cs_g \\ C_{73} \ C_{16} \ C_{1g} \end{array} \right\}$
Proc{2,1}	Proc{2,2}	Proc{2,3}
$\left\{ \begin{array}{l} C_{21} \ C_{24} \ C_{21} \\ C_{41} \ C_{44} \ C_{41} \\ C_{61} \ CM \ c_{61} \\ Cs_1 \ Cs_1 \ Cs_1 \end{array} \right\}$	$\left\{ \begin{array}{l} C_{22} \ C_{25} \ C_2 \\ 2 \ C_{4s} \ C_4 \\ C_{62} \ c_{65} \\ Cs_2 \ Cs_5 \ Cs_1 \end{array} \right\}$	$\left\{ \begin{array}{l} C_{23} \ c_{26} \ C_{2g} \\ C_{43} \ c_{46} \ C_{4g} \\ c_{63} \ c_{66} \ c_{6g} \\ Cs_3 \ Cs_6 \ Cs_g \end{array} \right\}$

Figure 16.2 The block-cyclic distribution of tasks
(A = 8 Row= 2 N = 9 and Ptd = 3).

If M is not a multiple of p_{row} or if N is not a multiple of p_{col} , then the distribution of work among processors is no longer balanced. Indeed, if

$$\begin{aligned} M &= \alpha_1 p_{\text{row}} + /1 & 0 \leq /1 < p_{\text{row}}, \\ N &= \alpha_2 p_{\text{col}} + /2 & 0 \leq /2 < p_{\text{col}}, \end{aligned}$$

then the number of block-block multiplications per processor can range from $0.1R$ to $(\alpha_1 + 1)(\alpha_2 + 1)R$. However, this variation is insignificant in a large scale computation with $M \gg p_{\text{row}}$ and $N \gg p_{\text{col}}$:

$$\frac{(\alpha_1 + 1)(\alpha_2 + 1)R}{(\alpha_1 \alpha_2)R} = 1 + O\left(\frac{p_{\text{row}}}{M} + \frac{p_{\text{col}}}{N}\right).$$

We conclude that both the block distribution and the block-cyclic distribution strategies are load balanced for the general $C = AB$ update.

This is not the case for certain block-sparse situations that arise in practice. If A is block lower triangular and B is block upper triangular, then the amount of work associated with $T_{A(i,j)}$ depends upon i and j . Indeed from (162) we have

$$G_j = G_j + \sum_{k=1}^{\min(i,j)} A_{ik} B_{kj}.$$

A very uneven allocation of work for the block distribution can result because the number of flops associated with $T_{A(i,j)}$ increases with i and j . The tasks assigned to Proc($p_{\text{row}}, p_{\text{col}}$) involve the most work while the tasks assigned to Proc(1, 1) involve the least. To illustrate the ratio of workloads, set $M = N = R = N$ and assume that $p_{\text{row}} = p_{\text{col}} = p$ divides M . It can be shown that

$$\frac{\text{Flops assigned to Proc}(j, p)}{\text{Flops assigned to Proc}(1, 1)} = O(p) \quad (163)$$

if we assume $M/p \gg 1$. Thus, load balancing does not depend on problem size and gets worse as the number of processors increase.

This is not the case for the block-cyclic distribution. Again, Proc(1, 1) and Proc(j, j) are the least busy and most busy processors. However, now it can be verified that

$$\frac{\text{Flops assigned to Proc}(j, p)}{\text{Flops assigned to Proc}(1, 1)} = 1 + O\left(\frac{p}{M}\right), \quad (164)$$

showing that the allocation of work becomes increasingly balanced as the problem size grows.

Another situation where the block-cyclic distribution of tasks is preferred is the case when the first q block rows of A are zero and the first q block columns of B are zero. This situation arises in several important matrix factorization schemes. Note from Figure 1.6.1 that if q is large enough, then some processors have absolutely nothing to do if tasks are assigned according to the block distribution. On the other hand, the block-cyclic distribution is load balanced, providing further justification for this method of task distribution.

1h5A1A T}·}A - . -G·A S·o7 dL+A

Sofar the discussion has focused on load balancing from the point of view. We now turn our attention to the costs associated with data motion and processor coordination. How does a processor get hold of the data it needs for a assigned task? How does a processor know enough to wait if the data it needs is the output of a computation being performed by another processor? What are the overheads associated with data transfer and synchronization and how do they compare to the costs of the actual arithmetic?

The importance of data locality is discussed in §1.5. However, in a parallel computing environment, the data that a processor needs can be "far away," and if that is the case too often, then it is possible to lose the multiprocessor advantage. Regarding synchronization, time spent waiting for another processor to finish a calculation is time lost. Thus, the design of an effective parallel computation involves paying attention to the number of synchronization points and their impact. Altogether, this makes it difficult to model performance, especially since an individual processor can typically compute and communicate at the same time. Nevertheless, we forge ahead with our analysis of the model computation to dramatize the cost of data motion relative to fops. For the remainder of this section we assume:

- (a) The block-cyclic distribution of tasks is used to ensure that arithmetic is load balanced.
- (b) Individual processors can perform the computation $C_{ij} = C_{ij} + A_{ik}B_{kj}$ at a rate of F fops per second. Typically, a processor will have its own local memory hierarchy and vector processing capability, so F is an attempt to capture in a single number all the performance issues that we discussed in §1.5.
- (c) The time required to move floating point numbers into or out of a processor is $\alpha + fT$. In this model, the parameters α and f respectively capture the latency and bandwidth attributes associated with data transfer.

With these simplifications we can roughly assess the effectiveness of assigning p processors to the update computation $C = C + AB$.

Let $T_{arith}(p)$ be the time that each processor must spend doing arithmetic as it carries out its share of the computation. It follows from assumptions (a) and (b) that

$$T_{arith}(p) = \frac{2mr}{pF}. \quad (1.6.5)$$

Similarly, let $T_{data}(p)$ be the time that each processor must spend acquiring the data it needs to perform tasks. Ordinarily, this quantity would vary significantly from processor to processor. However, the implementation strategies outlined below have the property that the communication overheads are roughly the same for each processor. It follows that if $T_{arith}(p) + T_{data}(p)$ approximates the total execution time for the p -processor solution, then the quotient

$$S(p) = \frac{T_{arith}(l)}{T_{arith}(p) + T_{data}(p)} = \frac{l}{1 + \frac{T_{data}(p)}{T_{arith}(p)}} \quad (1.6.6)$$

is a reasonable measure of speedup. Ideally, the assignment of p processors to the $C = C + AB$ update would reduce the single processor execution time by a factor of p . However, from (1.66) we see that $S(p) < p$ with the compute-to-communicate ratio $T_{\text{data}}(p)/T_{\text{arith}}(P)$ explaining the degradation. To acquire an intuition about this all-important quotient, we need to examine more carefully the data transfer properties associated with each task.

hinvb -ulb k\\$ \\$tib -utf b

If a processor carries out $T_{\text{a}}(k[i,j])$, then at some time during the calculation, blocks c_{ij} , $A_{ij}, \dots, A_{iR}, B_{j}, \dots, B_{R}$ must find their way into its local memory. Given assumptions (a) and (c), Table 1.6.1 summarizes the associated data transfer overheads for an individual processor.

Required Blocks	Data Transfer Time per Block
$C_{i3} \quad i = \mu_{\text{proc}} M \quad j = 1 : R$	$a + fm_{11}$
$A_{ij} \quad i = \mu_{\text{proc}} N \quad j = 1 : R$	$a + fm_{11}$
$B_{ij} \quad i = 1 : R \quad j = 1 : R$	$a + fr_{11}$

TABLE 1.61. Communication overheads for $\text{Proc}(\mu_r)$

It follows that if

$$e = \text{total number of required } C\text{-block transfers,} \quad \{1.67\}$$

$$A = \text{total number of required } A\text{-block transfers,} \quad \{1.68\}$$

$$o = \text{total number of required } B\text{-block transfers,} \quad \{1.69\}$$

then

$$T_{\text{data}}(P) = e(a + fm_{11}) + A(a + fm_{11}) + o(a + fr_{11}),$$

and so from (1.65) we have

$$\frac{T_{\text{data}}(P)}{T_{\text{arith}}(P)} = \frac{F_p}{2} \left(\frac{e + A + o}{mrr} + \beta \left(\frac{e}{MNr} + \frac{A}{MR} + \frac{o}{mNR} \right) \right). \quad \{1.70\}$$

To proceed further with our analysis we need to estimate the 7 factors (1.67)-(1.69), and that requires assumptions about how the underlying architecture stores and accesses the matrices A , B , and C .

hinvb 7\\$b ue tx\\$s s\\$.k xKbtx tt T".b

In a shared memory system each processor has access to a common global memory. See Figure 1.6.3. During program execution, data flows to and from the global memory and this represents a significant overhead that we proceed to assess. Assume that the matrices C , A , and B are in global memory at the start and that $\text{Proc}(\mu_r)$ executes the following

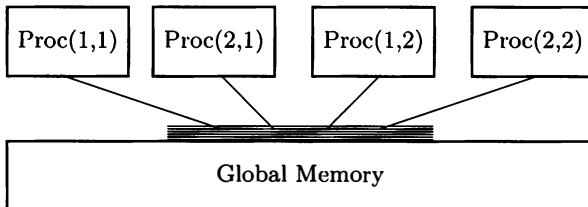


Figure 163 A four-processor shared memory system

```

f ri = ...
f rj = ... co in N
      C(loc) + E < k C
for k = 1:R
    A(loc) + Aik
    B(loc) + Bkj
    C(loc) = C(loc) + A(loc) B(loc)
end
    Cij + C(loc)
end
end

```

(Method 1)

As a reminder of the interactions between global and local memory, we use the "+" notation to indicate data transfers between these memory levels and the "loc" superscript to designate matrices in local memory. The block transfer statistics (167)-(169) for Method 1 are given by

$$\begin{aligned} t_e &= 2(MN/p), \\ t_A &= R(N/p), \\ t_n &= R(MN/p), \end{aligned}$$

and from (1610) we obtain

$$\frac{T_{\text{data}}(p)}{T_{\text{arith}}(p)} \approx \frac{F}{2} \left(\alpha \frac{2+2R}{m_1 n_1 r} + \beta \left(\frac{2}{r} + \frac{1}{n_1} + \frac{1}{m_1} \right) \right). \quad (1611)$$

By substituting this result into (166) we conclude that (a) speedup degrades as the flop rate F increases and (b) speedup improves if the communication parameters α and β decrease or the block dimensions m_1 , n_1 , and r increase. Note that the communicate-to-computer ratio (1611) for Method 1 does not depend upon the number of processors.

Method 1 has the property that it is only necessary to store one C-block, one A-block, and one B-block in local memory at any particular instant, i.e., $C^{(0)}$, $A^{(0)}$, and $B^{(0)}$. Typically, a processor's local memory is much smaller than global memory, so this particular solution approach is attractive for problems that are very large relative to local memory capacity. However, there is a hidden cost associated with this economy because in Method 1, each A-block is loaded N/P_{col} times and each B-block is loaded M/P_{row} times. This redundancy can be eliminated if each processor's local memory is large enough to house simultaneously all the C-blocks, A-blocks, and B-blocks that are required by its assigned tasks. Should this be the case, then the following method involves much less data transfer.

```

for k=1:R
     $A_k^{(0)} \leftarrow A_{ik} \quad (i = 1:\text{RowM})$ 
     $B_k^{(0)} \leftarrow B_{kj} \quad (j = 1:\text{RowN})$ 
end
for r_i = 1:RowM
    for r_j = 1:RowN
         $C^{(0)} + C_{ij}$ 
        for k=1:R
             $C^{(0)} = C^{(0)} + A^{(0)} B_k^{(0)}$ 
        end
         $C_{ij} \leftarrow C^{(0)}$ 
    end
end

```

(Method 2)

The block transfer statistics " ", " ", and " ", for Method 2 are more favorable than for Method 1. It can be shown that

$$|e = \gamma_C, \quad \gamma'_C = \gamma_A f_{\text{col}}, \quad \gamma'_B = \gamma_B f_{\text{row}}, \quad (16.12)$$

where the quotients $f_{\text{col}} = P_{\text{col}}/N$ and $f_{\text{row}} = \text{RowM}$ are typically much less than unity. As a result, the communicate-to-compute ratio for Method 2 is given by

$$\frac{T_{\text{data}}(P)}{T_{\text{arith}}(P)} = \frac{F}{2} \left(\frac{2 + R(f_{\text{col}} + f_{\text{row}})}{m \cdot n \cdot r} + \left(\frac{2}{r} + \frac{2}{n} J_{\text{col}} + \frac{2}{m} f_{\text{row}} \right) \right), \quad (16.13)$$

which is an improvement over (16.11). Methods 1 and 2 show that the trade-off that frequently exists between local memory capacity and the overheads that are associated with data transfer.

highlighted in Section 1.2. The discussion in the previous section assumes that C, A, and B are available in global memory at the start. If we extend the model computation so that it includes the

multiprocessor initialization of these three matrices, then an interesting issue arises. How does a processor "know" when the initialization is complete and it is therefore safe to begin its share of the $C = C + AB$ update?

Answering this question is an occasion to introduce a very simple synchronization construct known as the barrier. Suppose the C -matrix is initialized in global memory by a signing to each processor some fraction of the task. For example, $\text{Proc}(\mu, T)$ could do this:

```

for i = μ:Row:N
    for j = T:Col:N
        Compute the (i,j) block of C and store in C(lo).
        Cij ← C(lo)
    end
end

```

Similar approaches can be taken for the setting up of $A = (A_{ij})$ and $B = (B_{ij})$. Even if this partitioning of the initialization is load balanced, it cannot be assumed that each processor completes its share of the work at exactly the same time. This is where the barrier synchronization is handy. Assume that $\text{Proc}(\mu, T)$ executes the following

Initialize C_{ij} , $i = \mu: \text{Row}:N$	$j = T: \text{Col}:N$
Initialize A_{ij} , $i = \mu: \text{Row}:M$	$j = T: \text{Col}:R$
Initialize B_{ij} , $i = \mu: \text{Row}:M$	$j = T: \text{Col}:R$
t = 4μs 4Pssog s	

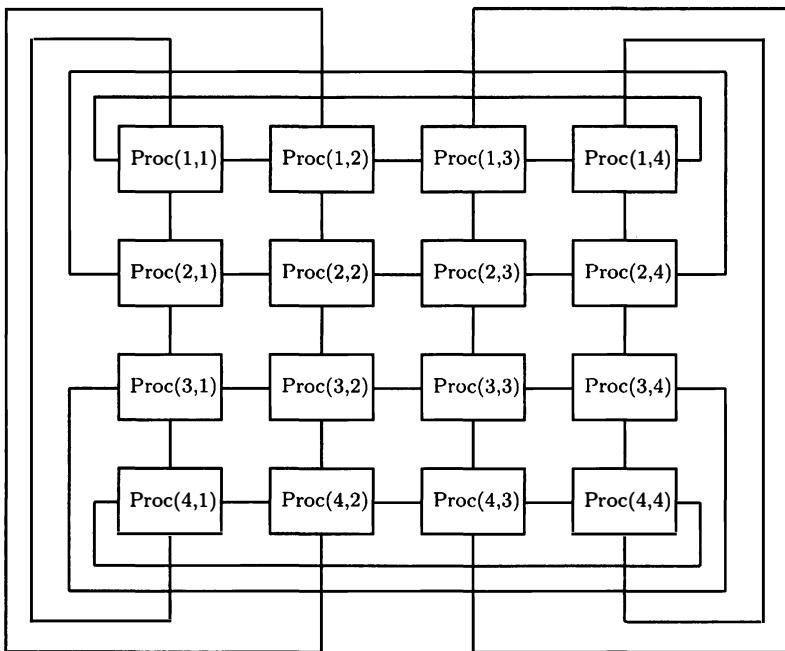
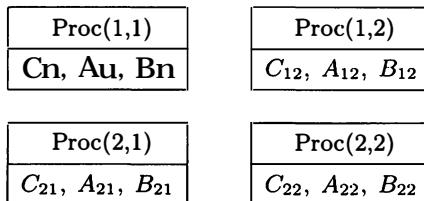


Figure 164 A 2Dimensional Torus

Let us first assume that $M = N = R = P_{\text{row}} = P_{\text{col}} = 2$ and that the C , A , and B matrices are distributed as follows



Assume that $\text{Proc}(i,j)$ oversees the update of C_{ij} and notice that the required data for this computation is not entirely local. For example, $\text{Proc}(1,1)$ needs to receive a copy of A_{12} from $\text{Proc}(1,2)$ and a copy of B_{21} from $\text{Proc}(2,1)$ before it can complete the update $C_{11} = C_{11} + A_{12}B_{11} + A_{12}B_{21}$. Likewise, it must send a copy of A_u to $\text{Proc}(1,2)$ and a copy of B_n to $\text{Proc}(2,1)$ so that they can carry out their respective updates. Thus, the local programs executing on each processor involve a mix of computational steps and message passing steps.

Proc(1,1)	Proc(1,2)
$Jhdd \bullet = -t\mathbf{AZ}_1 \cdot V_1 \cdot \underset{j}{1j}$	$\underset{j}{1l} \bullet = -t\mathbf{AZ}_2 \cdot V_1 \cdot \underset{j}{1j}$
$\underset{j}{1l} \bullet B_{11a} \bullet = -t\mathbf{AZ}_2 \cdot UV_1 \cdot \underset{j}{1j}$	$\underset{j}{1l} \bullet B_1 = -t\mathbf{AZ}_1 \cdot UV_1 \cdot \underset{j}{1j}$
$\underset{j}{1l} \bullet = t\mathbf{B}_{11} (=V_2 \cdot \underset{j}{1j})$	$\underset{j}{1l} \bullet a = \mathbf{B}_{12} \cdot V_1 \cdot \underset{j}{1j}$
$\underset{j}{1l} \bullet B_{11} \bullet = -t\mathbf{BZ}_1 \cdot UV_1 \cdot \underset{j}{1j}$	$\underset{j}{1l} \bullet B_{11a} \bullet = -t\mathbf{BZ}_2 \cdot UV_1 \cdot \underset{j}{1j}$
$C_{11} = C_{11} + A_{11}B_{11} + A_{12}B_{21}$	$C_{12} = C_{12} + A_{11}B_{12} + A_{12}B_{22}$

This informal specification of the local programs does a good job delineating the duties of each processor, but it hides several important issues that have to do with the timeline of execution. (a) Messages do not necessarily arrive at their destination in the order that they were sent. How will a receiving processor know if it is an A-block or a B-block? (b) Receive-a-message commands can block a processor from proceeding with the rest of its calculations. As a result, it is possible for a processor to wait forever for a message that its neighbor never got around to sending. (c) Overlapping computation with communication is critical for performance. For example, after A_{11} arrives at Proc(1,2), the "half" update $C_{12} = C_{11} + A_{11}B_{12}$ can be carried out while the wait for B_{22} continues.

As can be seen, distributed-memory matrix computations are quite involved and require powerful systems to manage the packaging, tagging, routing, and reception of messages. The discussion of such systems is outside the scope of this book. Nevertheless, it is instructive to go beyond the above 2by2 example and briefly anticipate the data transfer overheads for the general model computation. Assume that $\text{Proc}(\mu_r)$ houses these matrices:

$$\begin{array}{ll} C_{ij}, & \mathbf{i} = \mu:\texttt{prov.M}, \quad \mathbf{j} = T:\texttt{Ptd.N}, \\ A_{ij}, & \mathbf{i} = L\text{-}\texttt{Prov.3a} \quad \mathbf{j} = T:\texttt{pct.R}, \\ B_{ij}, & \mathbf{i} = \mu:\texttt{Prov.R}, \quad \mathbf{j} = T:\texttt{Ptd.N}. \end{array}$$

From Table 16.1 we conclude that if $\text{Pro}(\mathcal{I}, T)$ is to update C_{ij} for $i = \mu : \text{Prov.M}$ and $j = r : \text{pd} : N$, then it must

- (a) For $i = \mu : \text{prow } M$ and $j = r : \text{Pcl } R$, send a copy of A_{ij} to

$\text{Proc}(\mu, 1), \dots, \text{Proc}(\mu, T-1), \text{Proc}(\mu, T+1), \dots, \text{Proc}(\mu, p-1)$

Data transfer time = $(P_{cd} - 1)(M/Pow)(R/Pcd)(a + b/mr)$

- (b) For $i = \mu:\text{Prov.R}$ and $j = T:\text{Pcd.N}$, send a copy of B_{ij} to

$\text{Proc}(l, r), \dots, \text{Proc}(\mu - 1, r), \text{Proc}(\mu + 1, r), \dots, \text{Proc}(P_{O_1}, r)$.

Data transfer time = $(\text{proc} - 1)(R/\text{Proc})(N/\text{Pcd}) (a + b/r \ln n)$

(c) Receive copies of the A-blocks that are sent by processors

$$\text{Proc}(\mu, 1), \dots, \text{Proc}(\mu_{r-1}, 1), \text{Proc}(\mu_r + 1), \dots, \text{Proc}(\mu_{p_{\text{col}}}).$$

Data transfer time $(p_{\text{col}} - 1)(M/p_{\text{row}})(R/p_{\text{col}})(\alpha + fmr)$

(d) Receive copies of the E-blocks that are sent by processors

$$\text{Proc}(l, r), \dots, \text{Proc}(\mu - 1, r), \text{Proc}(\mu + 1, r), \dots, \text{Proc}(p_{\text{row}}, r).$$

Data transfer time $(p_{\text{row}} - 1)(R/p_{\text{row}})(N/p_{\text{col}})(\alpha + frini)$

Let T_{data} be the summation of these data transfer overheads and recall that $T_{\text{arith}} = (2mnr)/(Fp)$ since arithmetic is evenly distributed around the processor network. It follows that

$$\frac{T_{\text{data}}(p)}{T_{\text{arith}}(p)} = F \left(a \left(\frac{p_{\text{row}}}{mfrin} + \frac{p_{\text{row}}}{mrln} \right) + \left(\frac{p_{\text{col}}}{n} + \frac{p_{\text{row}}}{m} \right) \right). \quad (16.15)$$

Thus, as problem size grows this ratio tends to zero and speedup approaches according to (16.6).

10 Or r Loxx wxitrlIx wEzs3r

We close with a brief description of the Cannon (1969) matrix multiplication scheme. The method is an excellent way to showcase the toroidal network displayed in Figure 16.4 together with the idea of "nearest neighbor" thinking which is quite important in distributed matrix computations. For clarity, let us assume that $A = (A_{ij})$, $B = (E_{ij})$, and $C = (C_{ij})$ are 4 by 4 block matrices with n_1 by n_1 blocks. Define the matrices

$$A^{(1)} = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix}, \quad n^{(1)} = \begin{bmatrix} B_{11} & E_{22} & E_{33} & B_{44} \\ E_{21} & E_{32} & E_{43} & E_{14} \\ E_{31} & B_{42} & E_{13} & E_{24} \\ E_{41} & B_{12} & E_{23} & E_{34} \end{bmatrix},$$

$$A^{(2)} = \begin{bmatrix} A_{11} & A_{11} & A_{12} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{31} \\ A_{41} & A_{42} & A_{44} & A_{42} \end{bmatrix}, \quad n^{(2)} = \begin{bmatrix} B_{41} & E_{12} & E_{23} & E_{34} \\ E_{11} & E_{22} & E_{33} & E_{44} \\ E_{21} & E_{32} & E_{44} & B_{14} \\ E_{31} & E_{42} & E_{13} & B_{24} \end{bmatrix},$$

$$A^{(3)} = \begin{bmatrix} A_{11} & A_{14} & A_{11} & A_{11} \\ A_{21} & A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{42} & A_{43} & A_{44} & A_{41} \end{bmatrix}, \quad E^{(3)} = \begin{bmatrix} B_{31} & E_{42} & B_{13} & B_{24} \\ E_{41} & E_{12} & E_{23} & B_{34} \\ E_{11} & E_{22} & B_{33} & B_{44} \\ E_{21} & E_{32} & B_{43} & B_{14} \end{bmatrix},$$

$$A^{(4)} = \begin{bmatrix} A_{12} & A_{13} & A_{14} & A_{11} \\ A_{23} & A_{24} & A_{21} & A_{22} \\ A_{31} & A_{31} & A_{32} & A_{33} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix}, \quad B^{(4)} = \begin{bmatrix} B_{21} & E_{32} & B_{43} & B_{14} \\ E_{31} & E_{42} & E_{13} & E_{24} \\ E_{41} & E_{12} & E_{23} & E_{34} \\ E_{11} & B_{22} & E_{33} & E_{44} \end{bmatrix},$$

and note that

$$C_{ij} = A_{ij}^{(1)} n_{ij}^{(1)} + A_{ij}^{(2)} E_{ij}^{(2)} + A_{ij}^{(3)} E_{ij}^{(3)} + A_{ij}^{(4)} E_{ij}^{(4)} \quad (16.16)$$

Refer to Figure 16.4 and assume that Proc(i,j) is in charge of computing C_{ij} and that at the start it houses both $A\tilde{W}$ and $B\tilde{A}$. The message passing required to support the updates

$$C_{ij} = C_{ij} + A\tilde{W}Bf, \quad (16.17)$$

$$C_{ij} = C_{ij} + A\tilde{W}Bx, \quad (16.18)$$

$$C_{ij} = C_{ij} + A\tilde{W}B\tilde{A}, \quad (16.19)$$

$$C_{ij} = C_{ij} + Af, \quad (16.20)$$

involves communication with Proc(i,j)'s four neighbors in the toroidal network. To see this, define the block downshift permutation

$$P = \begin{bmatrix} 0 & 0 & 0 & I_{n_1} \\ I_{n_1} & 0 & 0 & 0 \\ 0 & I_{n_1} & 0 & 0 \\ 0 & 0 & I_{n_1} & 0 \end{bmatrix}$$

and observe that $A^{(k+1)} = A^{(k)}P^T$ and $B^{(k+1)} = PB^{(k)}$. That is, the transition from $A^{(k)}$ to $A^{(k+1)}$ involves shifting A-blocks to the right one column (with wraparound) while the transition from $B^{(k)}$ to $B^{(k+1)}$ involves shifting the B-blocks down one row (with wraparound). After each update (16.17)-(16.20), the housed A-block is passed to Proc(i,j)'s "east" neighbor and the next A-block is received from its "west" neighbor. Likewise, the housed B-block is sent to its "south" neighbor and the next B-block is received from its "north" neighbor.

Of course, the Cannon algorithm can be implemented on any processor network. But we see from the above that it is particularly well suited when there are toroidal connections for then communication is always between adjacent processors.

Problems

P1.6.1 k2m.b ..m.g6. arob5.58r5p5

P1.6.2 38.mN mm6ne.d mN. lm6.m .6 ro89m.o 6 . mN, S<=E₇ 3+ = A)) q S<=HC<.x=B)) A

P1.6.3 e ..bPr..m.g. f.585. 5f.a 55

P1.6.4 8 . .8. . N m .m.gm6..m.6A) - A²) 2+1 11 - A ER^{nxn} . 2 - 3 . jm) i km34

P1.6.5 8 . .8. . . N .2 m 2m..lm.6. B = A^TA) -- 1C1 - A ER^{mxn} . 2 - 3 . jm) i 1 4) 3 B - 11 + 1 - 1 U U) 1 -

P1.6.6 .8. f.8d85L. .o... 1ro. m o1.g6. .b2.m..pmg G⁽ⁱ⁾ (B⁽ⁱ⁾).

Notes and References for §1.6

18. N28 . 82mm.... m.ngb.. .. .m.p .2om..o. .m.16 scALAPACK a 3< at

ea = N0.2MnyeJ <= a= xf..A= rxf:x = .. i a < xy < yy = x(2N1, ... + (+ 2 A= = 0lab+ 8 O> A= bxa8n

- a 0SSbaq Ea 4S1u oax L kien QKCe fe gr rG0j ia Gaeon OWeo ia PuPrSSAS unpegn 3Rfo deroqfipPS /emn5 SIAM J. Sci. Stat. Comput. 8, i 11t5i ly/ 1
 s. 9J pr4m8enna3khpas3anS hJ/ap 8itN" I R SlnT pS/SAs.e4Mlnh9r.
)1A, 4apr3SIAM J. Matrix Anal. Appl. 19, yA/,1
 hxDLJnrrln7MIBdI rMnSuST8)luu1nS.,TSlnkSu3vpre4pS.y,s4SA S./Sln 18p .arrlpr3far lld Compt. 19, MBrMNr.?
 Cwe4,p emm h1-hrSln7M dl r,sT SSA,Slny/pS.arln4pGSy h,Aaln , .. 4, sepesaSSA,4ap7Ppar lld Compt. 20 tMBr N=
 1D-IM7 .lues3apenS11.lniepp, 7MIt I s1u ,Ys,pessaenSapreseT pSs4SAs,Sln 4silpS4utn.Sr/pSG,/asulp.)ln.oppa41Ao/apr7ICconcur ncy. Pmct. Exper. 6, O i- 0 + 2U
 j neyya = 9aU4la .a >+a = 2 a A : MS=20y (y 110= e= xA=EYIA(f- (FeA .(= " y< y = -x, 0+ = OS..O- AOX= -e= xY<A+ + 1> 0= < = ffOx x p=(0Ea2M
 J. Res. Devel. 98, A.B 5M.
 .. 9pln.3ikksaSl3enS41 Ksr3Mn11 d9)luuonS.,TS x183p9llnSr vp.Sr/pMSraulp. re4pSr,s4SAS./S16. Par lld Distrib Compt 64, 1017-1026.
 $(nM, (2) 7,(112).., .() (M, , .II , , o 7B 1,,fM .1. M1M) M 7$
 $7M$
 $,11, 0. 3= RxAlhM0E 3-M20 bintU (x(2y = x x.= (E= .x 0= 0Ea <$
 $= 0a y< y 04 SIAM J. Matrix Anal. Applic. 92, R.A t11$
 $re4pM4penrAlMaSSr4S1aS allp.anSpT nuSilpApS SAsae ka rT lbt-Sr .an4pes3 nl'es.1G1A M5 phuMnSap,rT-l8 S Alp/ 6MF JET pSPT rlnS,4e ul4S1h ha$
 $h1x11-nrrlnenS).k sl 7ityd?I re4pSOnAl,StMln 9l saen m , a Rftr:esalp S$
 $4pJS4a/par SIAM J. Matrix Anal. Applic. 9 / i5/r / 1$
 $1D-IM7111lnieppe3enS.ku? uesap7 tt-9 I sep,s= ffr SfrAlrh4silpS4u$in .Mr'$
 $4pMG,4aslp.)ln. ppan4)luA /ap Par lld Compt. 21, iBy.vM/ 1A?$
 $kJa A,pessa4pS.luA/e4SlnsS4ape/lpa > .X= > 0=y= JnQNE= xN0 . S= .+.(+$
 $(=a = E,(OE...000= J0<(Y(+ 0o< = n=+ X+=A & b ! .= EOE= <<.E.=>0<00$
 $Oa =A & 0< = f x 0.1b-T$
 $,B 30< <M2Zpmy0.= > Ge1a = ay< y-Y< xZx J= (Ea (<0ay 0S = SIAM Review 20,$
 $/ 15.1$
 $1kr klp4aieenS5li 9l1S147My dlhl= /Slep/S,STPp d/S. s Rln lh/Sln SS SHpessas$
 $)luAl 4ap8,SIAM Review 27, M/ N/ ?$
 $.s I 1, aep enSi lu1 h4a8p/7M d ., , is18 4= pStr 9 ps pss3stps)luA/e/ Slnr ..$
 $Commun. ACM 28, yMy B"$
 $1k1lnieppe enS. 1)1 hlpnraiz yA1 I xS a,1= iaGpe sSi- sapvpu na)luA/apSv Appl.$
 $Math. Comput. 20, r .5yw$
 $rskl sae4JAS. 7M 1Hyper ube Multipr ccessors, h94r s,GSS mSS,S3sA-Ss47$
 $beenennr ls h,-sT 7M t II. /luuonS.eIS1 S1S,pHss4S-S4a/par J. Dist. Par lld$
 $Comput. 11, 131-150$
 $1w19nieppe79., Tvl hlpj nr3i0SSs. en Sa Ilprg7Mld Sdving Linear Systems on Vector$
 $and Shar d Memor Computers, h94r sxGS.. 8ln, 79SHAsA-Ss41$
 $C.4liessS.en311lsauulnr3nS 4" h, 3-7Mldw sepes4silpM/-up.anraxSha,p4= iaGpe$
 $)luA, 4e 4Mn8,SIAM Review 92,$

 $Acta Numer ca 1999,)euGpSSiamS.aprS8parr.$
 $41RSasuer7M 1I xpiainr luapS.,sS8,p4siar 7MttBya s,p,ssasluA,T SlnDnOl$
 $an.Rv,Int. J. Super omput. Applic. 7, iIB My$

Chapter 2

Matrix Analysis

ISf

ISI

ISn

ISO

ISR

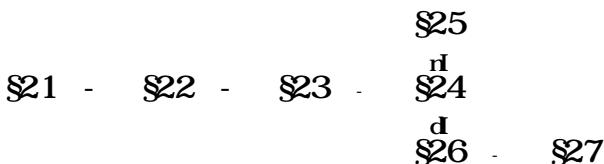
ISS

IST

The analysis and derivation of algorithms in the matrix computation area requires a facility with linear algebra. Some of the basics are reviewed in §2.1. Norms are particularly important, and we step through the vector and matrix cases in §2.2 and §2.3. The ubiquitous singular value decomposition is introduced in §2.4 and then used in the next section to define the CS decomposition and its ramifications for the measurement of subspace separation. In §2.6 we examine how the solution to a linear system $Ax = b$ changes if A and b are perturbed. It is the ideal setting for introducing the concepts of problem sensitivity, backward error analysis, and condition number. These ideas are central throughout the text. To complete the chapter we develop a model of finite precision floating point arithmetic based on the IEEE standard. Several canonical examples of roundoff error analysis are offered.

Reading Notes

Familiarity with matrix manipulation consistent with §1.1–§1.3 is essential. The sections within this chapter depend upon each other as follows:



Complementary references include Forsythe and Moler (SLAS), Stewart (IMC), Hor and Johnson (MA), Stewart (MABD), Ipsen (NMA), and Watkins (FMC). Fundamentals of matrix analysis that are specific to least squares problems and eigenvalue problems appear in later chapters.

2.1 Basic Ideas from Linear Algebra

This section is a quick review of linear algebra. Readers who wish a more detailed coverage should consult the references at the end of the section.

ihhhhb (1 tSEStSfSob uDeEtfS duteTeobtb e1.SleTkb

A set of vectors $\{a_1, \dots, a_n\}$ in \mathbb{R}^m is linearly independent if $\sum_{i=1}^n c_i a_i = 0$ implies $c_1 = \dots = c_n = 0$. Otherwise, a nontrivial combination of the a_i is zero and $\{a_1, \dots, a_n\}$ is said to be linearly dependent.

A subspace of \mathbb{R}^m is a subset that is also a vector space. Given a collection of vectors $a_1, \dots, a_n \in \mathbb{R}^m$, the set of all linear combinations of these vectors is a subspace referred to as the span of $\{a_1, \dots, a_n\}$:

$$\text{span}\{a_1, \dots, a_n\} = \left\{ \sum_{i=1}^n c_i a_i \mid c_i \in \mathbb{R} \right\}.$$

If $\{a_1, \dots, a_n\}$ is independent and $b \in \text{span}\{a_1, \dots, a_n\}$, then b is a unique linear combination of the a_i .

If S_1, \dots, S_k are subspaces of \mathbb{R}^m , then their sum is the subspace defined by $S = \{a_1 + a_2 + \dots + a_k \mid a_i \in S_i, i = 1:k\}$. S is said to be a direct sum if each $v \in S$ has a unique representation $v = a_1 + \dots + a_k$ with $a_i \in S_i$. In this case we write $S = S_1 \oplus \dots \oplus S_k$. The intersection of the S_i is also a subspace, $S = S_1 \cap S_2 \cap \dots \cap S_k$.

The subset $\{a_1, \dots, a_k\}$ is a maximal linearly independent subset of $\{a_1, \dots, a_n\}$ if it is linearly independent and is not properly contained in any linearly independent subset of $\{a_1, \dots, a_n\}$. If $\{a_1, \dots, a_k\}$ is maximal, then $\text{span}\{a_1, \dots, a_k\} = \text{span}\{a_1, \dots, a_n\}$ and $\{a_1, \dots, a_k\}$ is a basis for $\text{span}\{a_1, \dots, a_n\}$. If $S \subseteq \mathbb{R}^m$ is a subspace, then it is possible to find independent basic vectors $a_1, \dots, a_k \in S$ such that $S = \text{span}\{a_1, \dots, a_k\}$. All bases for a subspace S have the same number of elements. This number is the dimension and is denoted by $\dim(S)$.

.A1AA R}fcsA IN2Alf}icsA }IA R} :TA

There are two important subspaces associated with an m by n matrix A . The range of A is defined by

$$\text{ran}(A) = \{y \in \mathbb{R}^m \mid y = Ax \text{ for some } x \in \mathbb{R}^n\}$$

and the nullspace of A is defined by

$$\text{null}(A) = \{x \in \mathbb{R}^n \mid Ax = 0\}.$$

If $A = m \times n$, then $\text{null}(A) = \mathbb{R}^n$ if $m < n$ and $\{0\}$ if $m \geq n$.

The rank of a matrix A is defined by

$$\text{rank}(A) = \dim(\text{im}(A)).$$

If $A \in \mathbb{R}^{m,n}$, then

$$\dim(\text{null}(A)) + \text{rank}(A) = n.$$

We say that $A \in \mathbb{R}^{m,n}$ is rank deficient if $\text{rank}(A) < \min\{m, n\}$. The rank of a matrix is the maximal number of linearly independent columns (or rows).

Important fact to remember

If A and X are in $\mathbb{R}^{n,n}$ and satisfy $AX = I$, then X is the inverse of A and is denoted by A^{-1} . If A^{-1} exists, then A is said to be nonsingular. Otherwise, we say A is singular. The inverse of a product is the reverse product of the inverses:

$$(AB)^{-1} = B^{-1}A^{-1}. \quad (211)$$

Likewise, the transpose of the inverse is the inverse of the transpose:

$$(A^{-1})^T = (A^T)^{-1} \equiv A^{-T}. \quad (212)$$

How does the inverse change if the matrix changes? The Sherman-Morrison-Woodbury formula gives a convenient expression for the inverse of the matrix $(A + UV^T)$ where $A \in \mathbb{R}^{n,n}$ and U and V are n -by- k :

$$B^{-1} = A^{-1} - B^{-1}(B - A)A^{-1} \quad (213)$$

shows how the inverse changes if the matrix changes. The Sherman-Morrison-Woodbury formula gives a convenient expression for the inverse of the matrix $(A + UV^T)$ where $A \in \mathbb{R}^{n,n}$ and U and V are n -by- k .

$$(A + UV^T)^{-1} = A^{-1} - A^{-1}U(I + V^TA^{-1}U)^{-1}V^TA^{-1}. \quad (214)$$

A rank- k correction to a matrix results in a rank- k correction of the inverse. In (214), we assume that both A and $(I + V^TA^{-1}U)$ are nonsingular.

The formula is particularly useful if $A \in \mathbb{R}^{n,n}$ is nonsingular, $UV \in \mathbb{R}^{n,n}$, and $a = 1 + V^TA^{-1}U = 0$ then

$$(A + UV^T)^{-1} = A^{-1} - A^{-1}U^TA^{-1}. \quad (215)$$

This is referred to as the Sherman-Morrison formula.

Important result: If S_1, \dots, S_p are subspaces of \mathbb{R}^n , then

A set of vectors $\{x_1, \dots, x_p\}$ in \mathbb{R}^n is orthogonal if $x_i \cdot x_j = 0$ whenever $i \neq j$ and orthonormal if $x_i \cdot x_i = 1$. Intuitively, orthogonal vectors are maximally independent for they point in totally different directions.

A collection of subspaces S_1, \dots, S_p in \mathbb{R}^n is mutually orthogonal if $x_i \cdot y_j = 0$ whenever $x_i \in S_i$ and $y_j \in S_j$ for $i \neq j$. The orthogonal complement of a subspace S in \mathbb{R}^n is defined by

$$S^\perp = \{y \in \mathbb{R}^n \mid T_x = 0 \text{ for all } x \in S\}.$$

It is not hard to show that $\text{ran}(A) = \text{null}(A^T)$. The vectors v_1, \dots, v_m form an orthonormal basis for a subspace S in \mathbb{R}^m if they are orthonormal and span S .

A matrix $Q \in \mathbb{R}^{n \times m}$ is said to be orthogonal if $Q^T Q = I$. If $Q = [q_1 | \dots | q_m]$ is orthogonal, then the q_i form an orthonormal basis for \mathbb{R}^m . It is always possible to extend such a basis to a full orthonormal basis $\{v_1, \dots, v_m\}$ for \mathbb{R}^m .

Theorem 2.1.1. If $V_1 \in \mathbb{R}^{n \times n}$ has orthonormal columns, then there exists $V_2 \in \mathbb{R}^{n \times (n-1)}$ such that

$$V = [V_1 | V_2]$$

is orthogonal. Note that $\text{ran}(V) = \text{ran}(V_2)$.

Proof. This is a standard result from introductory linear algebra. It is also a corollary of the QR factorization that we present in §5.2.

ihhhha b ,)Sb es f Sx Ttttf b

If $A = (a_{ij}) \in \mathbb{R}^{n \times 1}$, then its determinant is given by $\det(A) = a$. The determinant of $A \in \mathbb{R}^{n \times n}$ is defined in terms of order $(n-1)$ determinants:

$$\det(A) = \sum_{j=1}^n (-1)^{j+1} a_{1j} \det(A_{1j}).$$

Here, A_{1j} is an $(n-1)$ -by- $(n-1)$ matrix obtained by deleting the first row and j th column of A . Well-known properties of the determinant include $\det(AB) = \det(A)\det(B)$, $\det(A^T) = \det(A)$, and $\det(cA) = c^n \det(A)$ where $A, B \in \mathbb{R}^{n \times n}$ and $c \in \mathbb{R}$. In addition, $\det(A) = 0$ if and only if A is nonsingular.

ihhhfb bs" Sf tPDSbtttb bt" Sf f Sf kxb

Until we get to the main eigenvalue part of the book (Chapters 7 and 8), we need a handful of basic properties so that we can fully appreciate the singular value decomposition (§2.4), positive definiteness (§4.2), and various fast linear equation solvers (§4.8).

The eigenvalues of $A \in \mathbb{R}^{n \times n}$ are the zeros of the characteristic polynomial

$$p(x) = \det(A - xI).$$

Thus, every n -by- n matrix has n eigenvalues. We denote the set of A 's eigenvalues by

$$\lambda(A) = \{x : \det(A - xI) = 0\}.$$

If the eigenvalues of A are real, then we index them from largest to smallest as follows

$$\lambda_n(A) \leq \dots \leq \lambda_2(A) \leq \lambda_1(A).$$

In this case, we sometimes use the notation $\lambda_{\max}(A)$ and $\lambda_{\min}(A)$ to denote $\lambda_1(A)$ and $\lambda_n(A)$ respectively.

If $X \in \mathbb{C}^{n \times n}$ is nonsingular and $B = X^{-1}AX$, then A and B are similar. If two matrices are similar, then they have exactly the same eigenvalues.

If. E. (A), then there exists a nonzero vector x so that $Ax = \lambda x$. Such a vector is said to be an eigenvector for A associated with λ . If $A \in \mathbb{C}^{n \times n}$ has n independent eigenvectors x_1, \dots, x_n and $Ax_i = \lambda_i x_i$ for $i = 1, \dots, n$, then A is diagonalizable. The terminology is appropriate if A is diagonalizable.

$$X = [x_1 | \cdots | x_n],$$

then

$$x^{-1}AX = \text{diag}(i, \dots, n).$$

Not all matrices are diagonalizable. However, if $A \in \mathbb{R}^{n \times n}$ is symmetric, then there exists an orthogonal Q so that

$$Q^T A Q = \text{diag}(\lambda_1, \dots, \lambda_n). \quad \{216\}$$

This is called the Schur decomposition. The largest and smallest eigenvalues of a symmetric matrix satisfy

$$\text{An}_{\mathbf{X}}(\mathbf{A}) = \max_{\mathbf{x} \neq 0} \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{x}} \quad \{217\}$$

and

$$\text{Anir}(A) = \min_{x \neq 0} \frac{x^T A x}{x^T x}. \quad \{218\}$$

imhhib eTe \$x\$lf Ttf Tkb

Suppose α is a scalar and that $A(\alpha)$ is an $m \times n$ matrix with entries $a_{ij}(\alpha)$. If $a_{ij}(\alpha)$ is a differentiable function of α for all i and j , then by $\dot{A}(\alpha)$ we mean the matrix

$$A(O) = \frac{d}{d\alpha} A(\alpha) = \left(\frac{d}{d\alpha} a_{ij}(\alpha) \right) = (\dot{a}_{ij}(\alpha)).$$

Differentiation is a useful tool that can sometimes provide insight into the sensitivity of a matrix problem

Problems

$$\text{P2.1.1 .86 } \mathbf{m} \cdot \mathbf{m} \cdot \mathbf{b}_E \mathbf{R}^{m \times n} = \mathbf{a} \cdot \mathbf{b}_I \cdot \mathbf{p}^T \quad p, \quad Y = \mathbf{q} = \mathbf{f} \cdot \mathbf{O} \cdot \mathbf{x} \cdot \mathbf{R}^{m \times p} \cdot \mathbf{a}^n \in \mathbb{R}^{n \times p} \quad 3 -)$$

$$\text{P2.1.2 .2.8ro } A(\alpha) \in \mathbf{R}^{m \times r})^t - B(\alpha) \in \mathbf{R}^{r \times n})]^{-1})]^{-n} 3^m)]^{-3}]^t - 3) - 1, m^3 \\ m \cdot 3^{-1})^t \alpha \cdot B^T \cdot b_C.$$

$$\frac{d}{d\alpha} [A(\alpha)B(\alpha)] = \left[\frac{d}{d\alpha} A(\alpha) \right] B(\alpha) + A(\alpha) \left[\frac{d}{d\alpha} B(\alpha) \right].$$

$$B_{\dots,*} + {}_3^T 3 + \dots A(\alpha) {}_3^T, \nabla_i \nabla_{\dots} {}_3 3 c_3 {}_3^T 3 + {}_* + \nabla {}_*^T, (c)$$

$$! \cdot [A(\alpha)^{-1}] = -A(\alpha)^{-1} \quad ! \cdot A(\alpha) \quad] A(\alpha)^{-1}.$$

$$\text{P2.1.3 ...8. } A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n \text{)} \quad \therefore \phi(x) = \frac{1}{2}x^T Ax - x^T b. \quad (c) \quad \vec{t}_1(\nabla_T^T \vec{t}_1^T) + \vec{x}^T \vec{t}_2 \phi \vec{t}_2^T$$

P2.1.4 1.22.8. A R)xA + uvT RxB)0= TII=R&BxBA E R^{n×n}. vERⁿ. 2.m
 $\begin{bmatrix} \mathbf{R}' \\ \mathbf{R} \end{bmatrix} = (-2B(A+uv^T)x = b) \mathbf{J} \mathbf{c} \mathbf{m} \mathbf{M} \mathbf{A} \mathbf{I} \mathbf{t} \mathbf{I} \mathbf{c} \mathbf{m} \mathbf{I} \mathbf{M} \mathbf{J} \mathbf{t} \mathbf{I} \mathbf{A} \mathbf{I} \mathbf{t} \mathbf{I} \mathbf{c} \mathbf{v} \mathbf{l} \mathbf{1}$
 $\tilde{\mathbf{A}}\mathbf{x} = \mathbf{b} + \mathbf{c}\mathbf{m}$)B R)BxkB=T(x α y, W1 γ. A u R) xv.

P2.1.5 ..8fo1..d2... 8..8d8... 2..1bd8...0

P2.1.6 .2. .8. AER^{n×n}, '4^{kk}_{1,-1} ..¹_{m,1},¹_{dL..} ..¹₋₁ ..¹

$$\tilde{A} = A + a(uv^T + wv^T) + \{(uv^T + wv^T)$$

-CBxB, vERⁿ !_ a,{ ∈ R. R^{kk}_{1,-1} ..¹_{m,1},¹_{dL..} ..¹₋₁ ..¹ Ā , (≤,(x>≥- >,)-..1-..B-(c≤,≤(≠≤,≤₂θ>.2
 $c.>.-)≤,3. ≤_*-c.>.- c. Ā^-.$

P2.1.7 8...08. ...22... ...18. 8≥ -2.1N8... .8s1f8g..2.. N.22..ot..

... 1..... 8≥A+ USUT -CBxB AER^{n×n} !_ S ER^{k×k}_{1,-1} ..¹_{m,1},¹_{dL..} ..¹₋₁ ..¹ _U ∈ R^{n×k}.

P2.1.8 .2..8.. Q ∈ R^{n×n}, m,1m,1 L ERⁿ, N₁ ..¹₋₁ ..¹_n.L m,1k,1 m,1u,1 d,1 R₄^{kk}_{1,-1} A=(aij),B' Bx ,1 aij = vT(Q)T(Q)v.

P2.1.9 ..8fo ... 1≥ Al ci, inns T = - S, rCB)I - S M₁ M₂ M₃ M₄ M₅ M₆ M₇ M₈ M₉ M₁₀ M₁₁ M₁₂ M₁₃ M₁₄ M₁₅ M₁₆ M₁₇ M₁₈ M₁₉ M₂₀ M₂₁ M₂₂ M₂₃ M₂₄ M₂₅ M₂₆ M₂₇ M₂₈ M₂₉ M₃₀ M₃₁ M₃₂ M₃₃ M₃₄ M₃₅ M₃₆ M₃₇ M₃₈ M₃₉ M₄₀ M₄₁ M₄₂ M₄₃ M₄₄ M₄₅ M₄₆ M₄₇ M₄₈ M₄₉ M₅₀ M₅₁ M₅₂ M₅₃ M₅₄ M₅₅ M₅₆ M₅₇ M₅₈ M₅₉ M₆₀ M₆₁ M₆₂ M₆₃ M₆₄ M₆₅ M₆₆ M₆₇ M₆₈ M₆₉ M₇₀ M₇₁ M₇₂ M₇₃ M₇₄ M₇₅ M₇₆ M₇₇ M₇₈ M₇₉ M₈₀ M₈₁ M₈₂ M₈₃ M₈₄ M₈₅ M₈₆ M₈₇ M₈₈ M₈₉ M₉₀ M₉₁ M₉₂ M₉₃ M₉₄ M₉₅ M₉₆ M₉₇ M₉₈ M₉₉ M₁₀₀ M₁₀₁ M₁₀₂ M₁₀₃ M₁₀₄ M₁₀₅ M₁₀₆ M₁₀₇ M₁₀₈ M₁₀₉ M₁₁₀ M₁₁₁ M₁₁₂ M₁₁₃ M₁₁₄ M₁₁₅ M₁₁₆ M₁₁₇ M₁₁₈ M₁₁₉ M₁₂₀ M₁₂₁ M₁₂₂ M₁₂₃ M₁₂₄ M₁₂₅ M₁₂₆ M₁₂₇ M₁₂₈ M₁₂₉ M₁₃₀ M₁₃₁ M₁₃₂ M₁₃₃ M₁₃₄ M₁₃₅ M₁₃₆ M₁₃₇ M₁₃₈ M₁₃₉ M₁₄₀ M₁₄₁ M₁₄₂ M₁₄₃ M₁₄₄ M₁₄₅ M₁₄₆ M₁₄₇ M₁₄₈ M₁₄₉ M₁₅₀ M₁₅₁ M₁₅₂ M₁₅₃ M₁₅₄ M₁₅₅ M₁₅₆ M₁₅₇ M₁₅₈ M₁₅₉ M₁₆₀ M₁₆₁ M₁₆₂ M₁₆₃ M₁₆₄ M₁₆₅ M₁₆₆ M₁₆₇ M₁₆₈ M₁₆₉ M₁₇₀ M₁₇₁ M₁₇₂ M₁₇₃ M₁₇₄ M₁₇₅ M₁₇₆ M₁₇₇ M₁₇₈ M₁₇₉ M₁₈₀ M₁₈₁ M₁₈₂ M₁₈₃ M₁₈₄ M₁₈₅ M₁₈₆ M₁₈₇ M₁₈₈ M₁₈₉ M₁₉₀ M₁₉₁ M₁₉₂ M₁₉₃ M₁₉₄ M₁₉₅ M₁₉₆ M₁₉₇ M₁₉₈ M₁₉₉ M₂₀₀ M₂₀₁ M₂₀₂ M₂₀₃ M₂₀₄ M₂₀₅ M₂₀₆ M₂₀₇ M₂₀₈ M₂₀₉ M₂₁₀ M₂₁₁ M₂₁₂ M₂₁₃ M₂₁₄ M₂₁₅ M₂₁₆ M₂₁₇ M₂₁₈ M₂₁₉ M₂₂₀ M₂₂₁ M₂₂₂ M₂₂₃ M₂₂₄ M₂₂₅ M₂₂₆ M₂₂₇ M₂₂₈ M₂₂₉ M₂₃₀ M₂₃₁ M₂₃₂ M₂₃₃ M₂₃₄ M₂₃₅ M₂₃₆ M₂₃₇ M₂₃₈ M₂₃₉ M₂₄₀ M₂₄₁ M₂₄₂ M₂₄₃ M₂₄₄ M₂₄₅ M₂₄₆ M₂₄₇ M₂₄₈ M₂₄₉ M₂₅₀ M₂₅₁ M₂₅₂ M₂₅₃ M₂₅₄ M₂₅₅ M₂₅₆ M₂₅₇ M₂₅₈ M₂₅₉ M₂₆₀ M₂₆₁ M₂₆₂ M₂₆₃ M₂₆₄ M₂₆₅ M₂₆₆ M₂₆₇ M₂₆₈ M₂₆₉ M₂₇₀ M₂₇₁ M₂₇₂ M₂₇₃ M₂₇₄ M₂₇₅ M₂₇₆ M₂₇₇ M₂₇₈ M₂₇₉ M₂₈₀ M₂₈₁ M₂₈₂ M₂₈₃ M₂₈₄ M₂₈₅ M₂₈₆ M₂₈₇ M₂₈₈ M₂₈₉ M₂₉₀ M₂₉₁ M₂₉₂ M₂₉₃ M₂₉₄ M₂₉₅ M₂₉₆ M₂₉₇ M₂₉₈ M₂₉₉ M₃₀₀ M₃₀₁ M₃₀₂ M₃₀₃ M₃₀₄ M₃₀₅ M₃₀₆ M₃₀₇ M₃₀₈ M₃₀₉ M₃₁₀ M₃₁₁ M₃₁₂ M₃₁₃ M₃₁₄ M₃₁₅ M₃₁₆ M₃₁₇ M₃₁₈ M₃₁₉ M₃₂₀ M₃₂₁ M₃₂₂ M₃₂₃ M₃₂₄ M₃₂₅ M₃₂₆ M₃₂₇ M₃₂₈ M₃₂₉ M₃₃₀ M₃₃₁ M₃₃₂ M₃₃₃ M₃₃₄ M₃₃₅ M₃₃₆ M₃₃₇ M₃₃₈ M₃₃₉ M₃₄₀ M₃₄₁ M₃₄₂ M₃₄₃ M₃₄₄ M₃₄₅ M₃₄₆ M₃₄₇ M₃₄₈ M₃₄₉ M₃₅₀ M₃₅₁ M₃₅₂ M₃₅₃ M₃₅₄ M₃₅₅ M₃₅₆ M₃₅₇ M₃₅₈ M₃₅₉ M₃₆₀ M₃₆₁ M₃₆₂ M₃₆₃ M₃₆₄ M₃₆₅ M₃₆₆ M₃₆₇ M₃₆₈ M₃₆₉ M₃₇₀ M₃₇₁ M₃₇₂ M₃₇₃ M₃₇₄ M₃₇₅ M₃₇₆ M₃₇₇ M₃₇₈ M₃₇₉ M₃₈₀ M₃₈₁ M₃₈₂ M₃₈₃ M₃₈₄ M₃₈₅ M₃₈₆ M₃₈₇ M₃₈₈ M₃₈₉ M₃₉₀ M₃₉₁ M₃₉₂ M₃₉₃ M₃₉₄ M₃₉₅ M₃₉₆ M₃₉₇ M₃₉₈ M₃₉₉ M₄₀₀ M₄₀₁ M₄₀₂ M₄₀₃ M₄₀₄ M₄₀₅ M₄₀₆ M₄₀₇ M₄₀₈ M₄₀₉ M₄₁₀ M₄₁₁ M₄₁₂ M₄₁₃ M₄₁₄ M₄₁₅ M₄₁₆ M₄₁₇ M₄₁₈ M₄₁₉ M₄₂₀ M₄₂₁ M₄₂₂ M₄₂₃ M₄₂₄ M₄₂₅ M₄₂₆ M₄₂₇ M₄₂₈ M₄₂₉ M₄₃₀ M₄₃₁ M₄₃₂ M₄₃₃ M₄₃₄ M₄₃₅ M₄₃₆ M₄₃₇ M₄₃₈ M₄₃₉ M₄₄₀ M₄₄₁ M₄₄₂ M₄₄₃ M₄₄₄ M₄₄₅ M₄₄₆ M₄₄₇ M₄₄₈ M₄₄₉ M₄₅₀ M₄₅₁ M₄₅₂ M₄₅₃ M₄₅₄ M₄₅₅ M₄₅₆ M₄₅₇ M₄₅₈ M₄₅₉ M₄₆₀ M₄₆₁ M₄₆₂ M₄₆₃ M₄₆₄ M₄₆₅ M₄₆₆ M₄₆₇ M₄₆₈ M₄₆₉ M₄₇₀ M₄₇₁ M₄₇₂ M₄₇₃ M₄₇₄ M₄₇₅ M₄₇₆ M₄₇₇ M₄₇₈ M₄₇₉ M₄₈₀ M₄₈₁ M₄₈₂ M₄₈₃ M₄₈₄ M₄₈₅ M₄₈₆ M₄₈₇ M₄₈₈ M₄₈₉ M₄₉₀ M₄₉₁ M₄₉₂ M₄₉₃ M₄₉₄ M₄₉₅ M₄₉₆ M₄₉₇ M₄₉₈ M₄₉₉ M₅₀₀ M₅₀₁ M₅₀₂ M₅₀₃ M₅₀₄ M₅₀₅ M₅₀₆ M₅₀₇ M₅₀₈ M₅₀₉ M₅₁₀ M₅₁₁ M₅₁₂ M₅₁₃ M₅₁₄ M₅₁₅ M₅₁₆ M₅₁₇ M₅₁₈ M₅₁₉ M₅₂₀ M₅₂₁ M₅₂₂ M₅₂₃ M₅₂₄ M₅₂₅ M₅₂₆ M₅₂₇ M₅₂₈ M₅₂₉ M₅₃₀ M₅₃₁ M₅₃₂ M₅₃₃ M₅₃₄ M₅₃₅ M₅₃₆ M₅₃₇ M₅₃₈ M₅₃₉ M₅₄₀ M₅₄₁ M₅₄₂ M₅₄₃ M₅₄₄ M₅₄₅ M₅₄₆ M₅₄₇ M₅₄₈ M₅₄₉ M₅₅₀ M₅₅₁ M₅₅₂ M₅₅₃ M₅₅₄ M₅₅₅ M₅₅₆ M₅₅₇ M₅₅₈ M₅₅₉ M₅₆₀ M₅₆₁ M₅₆₂ M₅₆₃ M₅₆₄ M₅₆₅ M₅₆₆ M₅₆₇ M₅₆₈ M₅₆₉ M₅₇₀ M₅₇₁ M₅₇₂ M₅₇₃ M₅₇₄ M₅₇₅ M₅₇₆ M₅₇₇ M₅₇₈ M₅₇₉ M₅₈₀ M₅₈₁ M₅₈₂ M₅₈₃ M₅₈₄ M₅₈₅ M₅₈₆ M₅₈₇ M₅₈₈ M₅₈₉ M₅₉₀ M₅₉₁ M₅₉₂ M₅₉₃ M₅₉₄ M₅₉₅ M₅₉₆ M₅₉₇ M₅₉₈ M₅₉₉ M₆₀₀ M₆₀₁ M₆₀₂ M₆₀₃ M₆₀₄ M₆₀₅ M₆₀₆ M₆₀₇ M₆₀₈ M₆₀₉ M₆₁₀ M₆₁₁ M₆₁₂ M₆₁₃ M₆₁₄ M₆₁₅ M₆₁₆ M₆₁₇ M₆₁₈ M₆₁₉ M₆₂₀ M₆₂₁ M₆₂₂ M₆₂₃ M₆₂₄ M₆₂₅ M₆₂₆ M₆₂₇ M₆₂₈ M₆₂₉ M₆₃₀ M₆₃₁ M₆₃₂ M₆₃₃ M₆₃₄ M₆₃₅ M₆₃₆ M₆₃₇ M₆₃₈ M₆₃₉ M₆₄₀ M₆₄₁ M₆₄₂ M₆₄₃ M₆₄₄ M₆₄₅ M₆₄₆ M₆₄₇ M₆₄₈ M₆₄₉ M₆₅₀ M₆₅₁ M₆₅₂ M₆₅₃ M₆₅₄ M₆₅₅ M₆₅₆ M₆₅₇ M₆₅₈ M₆₅₉ M₆₆₀ M₆₆₁ M₆₆₂ M₆₆₃ M₆₆₄ M₆₆₅ M₆₆₆ M₆₆₇ M₆₆₈ M₆₆₉ M₆₇₀ M₆₇₁ M₆₇₂ M₆₇₃ M₆₇₄ M₆₇₅ M₆₇₆ M₆₇₇ M₆₇₈ M₆₇₉ M₆₈₀ M₆₈₁ M₆₈₂ M₆₈₃ M₆₈₄ M₆₈₅ M₆₈₆ M₆₈₇ M₆₈₈ M₆₈₉ M₆₉₀ M₆₉₁ M₆₉₂ M₆₉₃ M₆₉₄ M₆₉₅ M₆₉₆ M₆₉₇ M₆₉₈ M₆₉₉ M₇₀₀ M₇₀₁ M₇₀₂ M₇₀₃ M₇₀₄ M₇₀₅ M₇₀₆ M₇₀₇ M₇₀₈ M₇₀₉ M₇₁₀ M₇₁₁ M₇₁₂ M₇₁₃ M₇₁₄ M₇₁₅ M₇₁₆ M₇₁₇ M₇₁₈ M₇₁₉ M₇₂₀ M₇₂₁ M₇₂₂ M₇₂₃ M₇₂₄ M₇₂₅ M₇₂₆ M₇₂₇ M₇₂₈ M₇₂₉ M₇₃₀ M₇₃₁ M₇₃₂ M₇₃₃ M₇₃₄ M₇₃₅ M₇₃₆ M₇₃₇ M₇₃₈ M₇₃₉ M₇₄₀ M₇₄₁ M₇₄₂ M₇₄₃ M₇₄₄ M₇₄₅ M₇₄₆ M₇₄₇ M₇₄₈ M₇₄₉ M₇₅₀ M₇₅₁ M₇₅₂ M₇₅₃ M₇₅₄ M₇₅₅ M₇₅₆ M₇₅₇ M₇₅₈ M₇₅₉ M₇₆₀ M₇₆₁ M₇₆₂ M₇₆₃ M₇₆₄ M₇₆₅ M₇₆₆ M₇₆₇ M₇₆₈ M₇₆₉ M₇₇₀ M₇₇₁ M₇₇₂ M₇₇₃ M₇₇₄ M₇₇₅ M₇₇₆ M₇₇₇ M₇₇₈ M₇₇₉ M₇₈₀ M₇₈₁ M₇₈₂ M₇₈₃ M₇₈₄ M₇₈₅ M₇₈₆ M₇₈₇ M₇₈₈ M₇₈₉ M₇₉₀ M₇₉₁ M₇₉₂ M₇₉₃ M₇₉₄ M₇₉₅ M₇₉₆ M₇₉₇ M₇₉₈ M₇₉₉ M₈₀₀ M₈₀₁ M₈₀₂ M₈₀₃ M₈₀₄ M₈₀₅ M₈₀₆ M₈₀₇ M₈₀₈ M₈₀₉ M₈₁₀ M₈₁₁ M₈₁₂ M₈₁₃ M₈₁₄ M₈₁₅ M₈₁₆ M₈₁₇ M₈₁₈ M₈₁₉ M₈₂₀ M₈₂₁ M₈₂₂ M₈₂₃ M₈₂₄ M₈₂₅ M₈₂₆ M₈₂₇ M₈₂₈ M₈₂₉ M₈₃₀ M₈₃₁ M₈₃₂ M₈₃₃ M₈₃₄ M₈₃₅ M₈₃₆ M₈₃₇ M₈₃₈ M₈₃₉ M₈₄₀ M₈₄₁ M₈₄₂ M₈₄₃ M₈₄₄ M₈₄₅ M₈₄₆ M₈₄₇ M₈₄₈ M₈₄₉ M₈₅₀ M₈₅₁ M₈₅₂ M₈₅₃ M₈₅₄ M₈₅₅ M₈₅₆ M₈₅₇ M₈₅₈ M₈₅₉ M₈₆₀ M₈₆₁ M₈₆₂ M₈₆₃ M₈₆₄ M₈₆₅ M₈₆₆ M₈₆₇ M₈₆₈ M₈₆₉ M₈₇₀ M₈₇₁ M₈₇₂ M₈₇₃ M₈₇₄ M₈₇₅ M₈₇₆ M₈₇₇ M₈₇₈ M₈₇₉ M₈₈₀ M₈₈₁ M₈₈₂ M₈₈₃ M₈₈₄ M₈₈₅ M₈₈₆ M₈₈₇ M₈₈₈ M₈₈₉ M₈₉₀ M₈₉₁ M₈₉₂ M₈₉₃ M₈₉₄ M₈₉₅ M₈₉₆ M₈₉₇ M₈₉₈ M₈₉₉ M₉₀₀ M₉₀₁ M₉₀₂ M₉₀₃ M₉₀₄ M₉₀₅ M₉₀₆ M₉₀₇ M₉₀₈ M₉₀₉ M₉₁₀ M₉₁₁ M₉₁₂ M₉₁₃ M₉₁₄ M₉₁₅ M₉₁₆ M₉₁₇ M₉₁₈ M₉₁₉ M₉₂₀ M₉₂₁ M₉₂₂ M₉₂₃ M₉₂₄ M₉₂₅ M₉₂₆ M₉₂₇ M₉₂₈ M₉₂₉ M₉₃₀ M₉₃₁ M₉₃₂ M₉₃₃ M₉₃₄ M₉₃₅ M₉₃₆ M₉₃₇ M₉₃₈ M₉₃₉ M₉₄₀ M₉₄₁ M₉₄₂ M₉₄₃ M₉₄₄ M₉₄₅ M₉₄₆ M₉₄₇ M₉₄₈ M₉₄₉ M₉₅₀ M₉₅₁ M₉₅₂ M₉₅₃ M₉₅₄ M₉₅₅ M₉₅₆ M₉₅₇ M₉₅₈ M₉₅₉ M₉₆₀ M₉₆₁ M₉₆₂ M₉₆₃ M₉₆₄ M₉₆₅ M₉₆₆ M₉₆₇ M₉₆₈ M₉₆₉ M₉₇₀ M₉₇₁ M₉₇₂ M₉₇₃ M₉₇₄ M₉₇₅ M₉₇₆ M₉₇₇ M₉₇₈ M₉₇₉ M₉₈₀ M₉₈₁ M₉₈₂ M₉₈₃ M₉₈₄ M₉₈₅ M₉₈₆ M₉₈₇ M₉₈₈ M₉₈₉ M₉₉₀ M₉₉₁ M₉₉₂ M₉₉₃ M₉₉₄ M₉₉₅ M₉₉₆ M₉₉₇ M₉₉₈ M₉₉₉ M₁₀₀₀ M₁₀₀₁ M₁₀₀₂ M₁₀₀₃ M₁₀₀₄ M₁₀₀₅ M₁₀₀₆ M₁₀₀₇ M₁₀₀₈ M₁₀₀₉ M₁₀₁₀ M₁₀₁₁ M₁₀₁₂ M₁₀₁₃ M₁₀₁₄ M₁₀₁₅ M₁₀₁₆ M₁₀₁₇ M₁₀₁₈ M₁₀₁₉ M₁₀₂₀ M₁₀₂₁ M₁₀₂₂ M₁₀₂₃ M₁₀₂₄ M₁₀₂₅ M₁₀₂₆ M₁₀₂₇ M₁₀₂₈ M₁₀₂₉ M₁₀₃₀ M₁₀₃₁ M₁₀₃₂ M₁₀₃₃ M₁₀₃₄ M₁₀₃₅ M₁₀₃₆ M₁₀₃₇ M₁₀₃₈ M₁₀₃₉ M₁₀₄₀ M₁₀₄₁ M₁₀₄₂ M₁₀₄₃ M₁₀₄₄ M₁₀₄₅ M₁₀₄₆ M₁₀₄₇ M₁₀₄₈ M₁₀₄₉ M₁₀₅₀ M₁₀₅₁ M₁₀₅₂ M₁₀₅₃ M₁₀₅₄ M₁₀₅₅ M₁₀₅₆ M₁₀₅₇ M₁₀₅₈ M₁₀₅₉ M₁₀₆₀ M₁₀₆₁ M₁₀₆₂ M₁₀₆₃ M₁₀₆₄ M₁₀₆₅ M₁₀₆₆ M₁₀₆₇ M₁₀₆₈ M₁₀₆₉ M₁₀₇₀ M₁₀₇₁ M₁₀₇₂ M₁₀₇₃ M₁₀₇₄ M₁₀₇₅ M₁₀₇₆ M₁₀₇₇ M₁₀₇₈ M₁₀₇₉ M₁₀₈₀ M₁₀₈₁ M₁₀₈₂ M₁₀₈₃ M₁₀₈₄ M₁₀₈₅ M₁₀₈₆ M₁₀₈₇ M₁₀₈₈ M₁₀₈₉ M₁₀₉₀ M₁₀₉₁ M₁₀₉₂ M₁₀₉₃ M₁₀₉₄ M₁₀₉₅ M₁₀₉₆ M₁₀₉₇ M₁₀₉₈ M₁₀₉₉ M₁₁₀₀ M₁₁₀₁ M₁₁₀₂ M₁₁₀₃ M₁₁₀₄ M₁₁₀₅ M₁₁₀₆ M₁₁₀₇ M₁₁₀₈ M₁₁₀₉ M₁₁₁₀ M<sub

norms are the norms defined by

$$\|x\|_p = (\|x_1\|^p + \dots + \|x_n\|^p)^{\frac{1}{p}}, \quad p \text{ real.} \quad (221)$$

The 1-, 2-, and infinity-norms are the most important:

$$\begin{aligned}\|x\|_1 &= \|x\|_1 + \dots + \|x_n\|_1 \\ \|x\|_2 &= (\|x_1\|^2 + \dots + \|x_n\|^2)^{\frac{1}{2}} = (\mathbf{x}^T \mathbf{x})^{\frac{1}{2}}, \\ \|x\|_\infty &= \max_{1 \leq i \leq n} |x_i|.\end{aligned}$$

A unit vector with respect to the norm $\|\cdot\|_1$ is a vector x that satisfies $\|x\|_1 = 1$.

) OOr - wir iSwEr iw E.r tEwiE szlr

A basic result concerning norms is the Hölder inequality.

$$|x^T y| \leq \|x\|_p \|y\|_q, \quad \frac{1}{p} + \frac{1}{q} = 1. \quad (222)$$

A very important special case of this is the Cauchy-Schwarz inequality.

$$|x^T y| \leq \|x\|_2 \|y\|_2. \quad (223)$$

All norms on \mathbb{R}^n are equivalent, i.e., if $\|\cdot\|_1$ and $\|\cdot\|_\infty$ are norms on \mathbb{R}^n , then there exist positive constants c_1 and c_2 such that

$$c_1 \|x\|_\infty \leq \|x\|_1 \leq c_2 \|x\|_\infty \quad (224)$$

for all $x \in \mathbb{R}^n$. For example, if $x \in \mathbb{R}^n$, then

$$\|x\|_2 : \|x\|_1 \leq \sqrt{n} \|x\|_2 \quad (225)$$

$$\|x\|_\infty : \|x\|_2 \leq \sqrt{n} \|x\|_\infty, \quad (226)$$

$$\|x\|_1 : \|x\|_1 \leq n \|x\|_1. \quad (227)$$

Finally, we mention that the 2-norm is preserved under orthogonal transformation. Indeed, if $Q \in \mathbb{R}^{n \times n}$ is orthogonal and $x \in \mathbb{R}^n$, then

$$\|Qx\|_2^2 = (Qx)^T (Qx) = (x^T Q^T)(Qx) = x^T (Q^T Q)x = x^T x = \|x\|_2^2.$$

improbablePDF Subtitle Was tf TSS bxxkxb

Suppose $x \in \mathbb{R}^n$ is an approximation to $y \in \mathbb{R}^n$. For a given vector norm $\|\cdot\|$ we say that

$$\epsilon_{\text{abs}} = \|x - y\|$$

is the absolute error in x . If $x = Q$ then

$$\epsilon_{\text{rel}} = \frac{\|x - y\|}{\|x\|}$$

prescribes the relative error in \hat{x} . Relative error in the ∞ -norm can be translated into a statement about the number of correct significant digits in \hat{x} . In particular, if

$$\frac{\|\hat{x} - \mathbf{X}\|}{\|\mathbf{X}\|} \leq 10^{-1}$$

then the largest component of \hat{x} has approximately p correct significant digits. For example, if $\mathbf{X} = [1234.05674]^T$ and $\hat{x} = [1235.05128]^T$, then $\frac{\|\hat{x} - \mathbf{X}\|}{\|\mathbf{X}\|} = 0.043 \cdot 10^{-3}$. Note that \hat{x}_1 has about three significant digits that are correct while only one significant digit in \hat{x}_2 is correct.

(ii) If $\mathbf{x}' \neq \mathbf{x}$

We say that a sequence $\{\mathbf{x}_k\}$ of n -vectors converges to \mathbf{x} if

$$\lim_{k \rightarrow \infty} \|\mathbf{x}_k - \mathbf{x}\| = 0$$

Because of (224), convergence in any particular norm implies convergence in all norms.

Problems

P2.2.1 **86** If $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, then $\|\mathbf{x} + \mathbf{y}\|_p = \|\mathbf{x}\|_p + \|\mathbf{y}\|_p$.

P2.2.2 **P. 8. m... 91** ... **2.10.** $(ax + by)^T(a + by) = E = Q_2^T \mathbf{m} Q_2$ if $a \neq b$, $A = I$.
(223).

P2.2.3 **e** $\|\mathbf{x} - \mathbf{y}\|_p \leq \|\mathbf{x}\|_p + \|\mathbf{y}\|_p$ for $p \geq 1$.

P2.2.4 **e** $\|\mathbf{x} - \mathbf{y}\|_p \leq \sqrt{p} \|\mathbf{x} - \mathbf{y}\|_2$ for $p \geq 1$.

$$\|\mathbf{v}\|_1 \|\mathbf{v}\|_\infty \leq \sqrt{2} \|\mathbf{v}\|_2.$$

2.3 Matrix Norms

The analysis of matrix algorithms requires use of matrix norms. For example, the quality of a linear system solution may be poor if the matrix of coefficients is "nearly singular." To quantify the notion of near-singularity, we need a measure of distance on the space of matrices. Matrix norms can be used to provide that measure.

$\| \cdot \|_F$ is called the Frobenius norm.

Since $\mathbb{R}^{m \times n}$ is isomorphic to \mathbb{R}^{mn} , the definition of a matrix norm should be equivalent to the definition of a vector norm. In particular, $\| \cdot \|_{\mathbb{R}^{m \times n}}$. If f is a matrix norm if the following three properties hold:

$$\begin{aligned} f(A) &= 0 & A \in \mathbb{R}^{m \times n}, & f(A) = 0 \quad A = 0 \\ f(A+B) &\leq f(A) + f(B), & A, B \in \mathbb{R}^{m \times n} \\ f(\alpha A) &= |\alpha| f(A), & \alpha \in \mathbb{R}, A \in \mathbb{R}^{m \times n}. \end{aligned}$$

As with vector norms, we use a double bar notation with subscripts to designate matrix norms, i.e., $\| A \|_F = f(A)$.

The most frequently used matrix norms in numerical linear algebra are the Frobenius norm

$$\| A \|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2} \quad (231)$$

and the p -norms

$$\| A \|_p = \sup_{x \neq 0} \frac{\| Ax \|_p}{\| x \|_p}. \quad (232)$$

Note that the matrix p -norms are defined in terms of the vector p -norms discussed in the previous section. The verification that (231) and (232) are matrix norms is left as an exercise. It is clear that $\| A \|_F$ is the p -norm of the largest vector obtained by applying A to a unit p -norm vector.

$$\| A \|_p = \sup_{x \neq 0} \left\| A \left(\frac{x}{\| x \|_p} \right) \right\|_p = \max_{\| x \|_p = 1} \| Ax \|_p.$$

It is important to understand that (232) defines a family of norms: the 2 -norm on $\mathbb{R}^{m \times n}$ is a different notion from the 2 -norm on \mathbb{R}^{m+n} . Thus, the easily verified inequality

$$\| AB \|_p \leq \| A \|_p \| B \|_p, \quad A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{n \times q} \quad (233)$$

is really an observation about the relationship between three different norms. Formally, we say that norms $\| \cdot \|_1$, $\| \cdot \|_2$, and $\| \cdot \|_\infty$ on $\mathbb{R}^{m \times n}$, \mathbb{R}^{m+n} , and \mathbb{R}^{m+n} are mutually consistent if for all matrices $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{n \times q}$ we have $\| (AB) \|_F \leq \| A \|_F \| B \|_F$, or, in subscript-free notation,

$$\| AB \|_F \leq \| A \|_F \| B \|_F \quad (234)$$

Not all matrix norms satisfy this property. For example, if $\|A\|_\Delta = \max |a_{ij}|$ and

$$A = B = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},$$

then $\|AB\|_\Delta > \|A\|_\Delta \|B\|_\Delta$. For the most part, we work with norms that satisfy (234).

The norms have the important property that for every $A \in \mathbb{R}^{m \times n}$ and $x \in \mathbb{R}^n$ we have

$$\|Ax\|_p \leq \|A\|_p \|x\|_p.$$

More generally, for any vector norm $\|\cdot\|_\alpha$ on \mathbb{R}^n and $\|\cdot\|_\beta$ on \mathbb{R}^m we have $\|Ax\|_\beta \leq \|A\|_{\alpha,\beta} \|x\|_\alpha$ where $\|A\|_{\alpha,\beta}$ is a matrix norm defined by

$$\|A\|_{\alpha,\beta} = \sup_{x \neq 0} \frac{\|Ax\|_\beta}{\|x\|_\alpha}. \quad (235)$$

We say that $\|\cdot\|_{\alpha,\beta}$ is subordinate to the vector norms $\|\cdot\|_\alpha$ and $\|\cdot\|_\beta$. Since the set $\{x \in \mathbb{R}^n : \|x\|_\alpha = 1\}$ is compact and $\|\cdot\|_\beta$ is continuous, it follows that

$$\|A\|_{\alpha,\beta} = \max_{\|x\|_\alpha=1} \|Ax\|_\beta = \|Ax_*\|_\beta \quad (236)$$

for some $x_* \in \mathbb{R}^n$ having unit α -norm

ihngib ukSb ht f xTekk xb 2x kESx f TSeb

The Frobenius and p-norms (especially $p = 1, 2, \infty$) satisfy certain inequalities that are frequently used in the analysis of a matrix computation. If $A \in \mathbb{R}^{m \times n}$ we have

$$\|A\|_2 \leq \|A\|_F \leq \sqrt{\min\{m, n\}} \|A\|_2, \quad (237)$$

$$\max_{i,j} |a_{ij}| \leq \|A\|_2 \leq \sqrt{\max_{i,j} |a_{ij}|}, \quad (238)$$

$$\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^m |a_{ij}|, \quad (239)$$

$$\|A\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^n |a_{ij}|, \quad (2310)$$

$$\frac{1}{\sqrt{n}} \|A\|_\infty \leq \|A\|_2 \leq \sqrt{n} \|A\|_\infty, \quad (2311)$$

$$\frac{1}{\sqrt{m}} \|A\|_1 \leq \|A\|_2 \leq \sqrt{n} \|A\|_1. \quad (2312)$$

If $A \in \mathbb{R}^{m \times n}$, $1 \leq i_1 \leq i_2 \leq m$, and $1 \leq j_1 \leq j_2 \leq n$, then

$$\|A(i_1:i_2, j_1:j_2)\|_p \leq \|A\|_p. \quad (2313)$$

The proofs of these relationships are left as exercises. We mention that a sequence $\{A^{(k)}\} \in \mathbb{R}^{n \times n}$ converges if there exists a matrix $A \in \mathbb{R}^{n \times n}$ such that

$$\lim_{k \rightarrow \infty} \|A^{(k)} - A\| = 0$$

The choice of norm is immaterial since all norms on $\mathbb{R}^{n \times n}$ are equivalent.

Lemma 7.3.1. If $A \in \mathbb{R}^{n \times n}$, then there exists a unit 2-norm n -vector z such that $A^T A z = \mu^2 z$ where $\mu = \|A\|_2$.

Proof. Suppose $z \in \mathbb{R}^n$ is a unit vector such that $\|Az\|_2 = \|A\|_2$. Since z maximizes the function

$$g(x) = \frac{1}{2} \frac{\|Ax\|_2^2}{\|x\|_2^2} = \frac{1}{2} \frac{x^T A^T A x}{x^T x}$$

it follows that it satisfies $Vg(z) = 0$ where Vg is the gradient of g . A tedious differentiation shows that for $i = 1, \dots, n$

$$\frac{\partial g(z)}{\partial z_i} = \left[(z^T z) \sum_{j=1}^n (A^T A)_{ij} z_j - (z^T A^T A z) z_i \right] / (z^T z)^2.$$

In vector notation this says that $ATAz = (z^T ATAz)z$. The theorem follows by setting $\mu = \|Az\|_2$.

The theorem implies that $\|A\|_2$ is a zero of $p(\lambda) = \det(ATA - \lambda I)$. In particular,

$$\|A\|_2 = \sqrt{\lambda_{\max}(A^T A)}$$

We have much more to say about eigenvalues in Chapters 7 and 8. For now we merely observe that 2-norm computation is iterative and a more involved calculation than those of the matrix 1-norm or infinity-norm. Fortunately, if the object is to obtain an order-of-magnitude estimate of $\|A\|_2$ then (237), (238), (2311), or (2312) can be used.

As another example of norm analysis, here is a handy result for 2-norm estimation.

Corollary 2.3.2. If $A \in \mathbb{R}^{n \times n}$, then $\|A\|_2 \leq \sqrt{\|A\|_1 \|A\|_\infty}$.

Proof. If $z = 0$ is such that $ATAz = \mu^2 z$ with $\mu = \|A\|_2$ then $\mu^2 \|z\|_2 = \|ATAz\|_2 = \|AT\|_1 \|A\|_1 \|z\|_1 = \|A\|_1 \|A\|_\infty \|z\|_1$.

.t1AnA a67 - NfA- G:=A }1 IA-DoA5 :67 +oA

We frequently use norms to quantify the effect of perturbations or to prove that a sequence of matrices converges to a specified limit. As an illustration of these norm applications, let us quantify the change in A^{-1} as a function of change in A .

Lemma 2.3.3. If $P \in \mathbb{R}^{n \times n}$ and $\|P\|_p < 1$, then $I - P$ is nonsingular and

$$(I - P)^{-1} = \sum_{k=0}^{\infty} F^k$$

with

$$\|(I - F)^{-1}\|_p \leq \frac{1}{1 - \|F\|_p}.$$

Proof. Suppose $I - P$ is singular. It follows that $(I - P)x = 0$ for some nonzero x . But then $\|x\|_p = \|Px\|_p$ implies $\|F\|_p \geq 1$, a contradiction. Thus, $I - P$ is nonsingular. To obtain an expression for its inverse consider the identity

$$\left(\sum_{k=0}^N F^k \right) (I - P) = I - P^{N+1}.$$

Since $\|P\|_p < 1$ it follows that $\lim_{k \rightarrow \infty} F^k = 0$ because $\|F^k\|_p \leq \|F\|_p^k \leq 1$. Thus,

$$\left(\lim_{N \rightarrow \infty} \sum_{k=0}^N F^k \right) (I - P) = I.$$

It follows that $(I - F)^{-1} = \lim_{N \rightarrow \infty} \sum_{k=0}^N F^k$. From this it is easy to show that

$$\|(I - F)^{-1}\|_p \leq \sum_{k=0}^{\infty} \|F\|_p^k = \frac{1}{1 - \|F\|_p}$$

completing the proof of the theorem. ■

Note that $\|(I - F)^{-1} - I\|_p \leq \|F\|_p/(1 - \|F\|_p)$ is a consequence of the lemma. Thus, if $\|F\|_p < 1$, then $O(\epsilon)$ perturbations to the identity matrix induce $O(\epsilon)$ perturbations in the inverse. In general, we have

Theorem 2.3.4. If A is nonsingular and $\|A^{-1}E\|_p < 1$, then $A+E$ is nonsingular and

$$\|(A+E)^{-1} - A^{-1}\|_p \leq \frac{\|E\|_p \|A^{-1}\|_p^2}{1 - r}.$$

Proof. Note that $A+E = (I+E)A$ where $P = -EA^{-1}$. Since $\|P\|_p = r < 1$, it follows from Lemma 2.3.3 that $I+P$ is nonsingular and $\|(I+P)^{-1}\|_p \leq 1/(1-r)$.

23 Matrix Norms

25

Thus, $(A+E)^{-1} = A^{-1}(I+F)^{-1}$ is nonsingular and

$$(A+E)^{-1} - A^{-1} = A^{-1}(A - (A+E))(A+E)^{-1} = -A^{-1}EA^{-1}(I+F)^{-1}.$$

The theorem follows by taking norms. n

Lemma 6xf ek' ktpb f tf teTtf \$

If $A \in \mathbb{R}^{m \times n}$ and the matrices $Q \in \mathbb{R}^{m \times m}$ and $Z \in \mathbb{R}^{n \times n}$ are orthogonal, then

$$\| QAZ \|_F = \| A \|_F \quad (23.14)$$

and

$$\| QAZ \|_2 = \| A \|_2. \quad (23.15)$$

These properties readily follow from the orthogonal invariance of the vector 2-norm. For example,

$$\| QA \|_F^2 = \sum_{j=1}^n \| QA(:,j) \|_2^2 = \sum_{j=1}^n \| A(:,j) \|_2^2 = \| A \|_F^2$$

$$\text{also } \| QAZ \|_F^2 = \| AZ \|_F^2 = \| ZAT \|_F^2 = \| AT \|_F^2 = \| A \|_F^2.$$

Problems

$$P2.3.1 \text{ .38fo} \| \cdot \|_p \leq \| A \|_p, \| \cdot \|_p \leq \sum_{j=1}^n \| \cdot \|_p \leq \| A \|_p.$$

$$P2.3.2 \text{ 58. } 8 \dots \dots \dots b \text{ 8A } +/(- nR\mathbb{W} \|_p \leq \| A \|_p.$$

$$P2.3.3 \text{ .38fo.3. } \geq D = \frac{1}{2} - \mathbb{R}^n \dots, \mu_k) \in \mathbb{R}^{m \times n} \quad \| D \|_p = \sqrt{\sum_{i=1}^n | \mu_i |}.$$

$$P2.3.4 \text{ em } \geq 05rob5P5rob575$$

$$P2.3.5 \text{ e8. } \geq 0arob5rob5b5a.0$$

$$P2.3.6 \text{ e8fe } \geq 05rob651,05a$$

$$P2.3.7 \text{ .8fo } \geq 0 \in \mathbb{R}^n \quad E \in \mathbb{R}^{n \times n}, \quad \| E \|_F = \sqrt{\sum_{i=1}^n \sum_{j=1}^n | E_{ij} |^2}$$

$$\left\| E \left(I - \frac{s s^T}{s^T s} \right) \right\|_F^2 = \| E \|_F^2 - \| E s \|_2^2$$

$$P2.3.8 \text{8.8 } u \in \mathbb{R}^m, v \in \mathbb{R}^n, w \in \mathbb{R}^n. \quad \| E \|_F = \| E \|_2 = \| u \|_2 \| v \|_2 \| w \|_2$$

$$P2.3.9 \text{ ...8.8 } A \in \mathbb{R}^{m \times n}, y \in \mathbb{R}^m, z \in \mathbb{R}^n. \quad \| E \|_F = \| (y - Az)z^T / \| z \|_2 \|$$

$$= \| C \|_2 \| (y - Az)z^T / \| z \|_2 \|$$

$$P2.3.10 \text{ e8Nfe3.9 } \geq 0 \text{ 8bfq.. c > 2op nee=}$$

$$\| A \|_{B,C} = \max_{i,j} |a_{ij}|$$

$$\| A \|_{B,C} = \max_{i,j} \left(\sum_{k=1}^m |a_{ijk}| \right) \leq \max_{i,j} \left(\sum_{k=1}^m \beta_{ik} |c_{kj}| \right) \leq \max_{i,j} \left(\sum_{k=1}^m \beta_{ik} \right) \leq \max_{i,j} \left(\sum_{k=1}^m \beta_{ik} \right) = \| A \|_B \| C \|_C$$

$$P2.3.11 \text{ .8fo.3 } \geq AR, A \in \mathbb{R}^{m \times n}, \| A \|_F = \| A \|_B \| C \|_C$$

Notes and References for

fel...8. f..4....1. 1≥ ...1...6 .888fo.N.R R3s iO<<:

3Wc4 W20x il- F3x u2x0 - w a n0xVit a • E 2=x x18xM=Mathe_2, iBImM // . lkrMaeG ,t1dlchGuuatl Mlia nLtMgnAn MtI MsGI Miaglu A3Quart J. Math 11, e1(et:

4o, sgoAaJg.ar MekThe Theor of Matr ces in Numerical Analysis. gal (cosMgAaAp 3ge a.1 sMiJiur MdcRAtMuitMaPitIMf allu3Numer. Math 62, eBt,eeuw

2.4 The Singular Value Decomposition

It is fitting that the first matrix decomposition that we present in the book is the singular value decomposition (SVD). The practical and theoretical importance of the SVD is hard to overestimate. It has a prominent role to play in data analysis and in the characterization of the many matrix "nearness problems."

1 mnb pSXTf tf Tkb

The SVD is an orthogonal matrix reduction and so the 2-norm and Frobenius norm figure heavily in this section. Indeed, we can prove the existence of the decomposition using some elementary facts about the 2-norm developed in the previous two sections.

Theorem 2.4.1 (Singular Value Decomposition). If A is a real $m \times n$ matrix, then there exist orthogonal matrices

$$U = [U_1 \dots U_m] \in \mathbb{R}^{m \times m} \text{ and } V = [V_1 \dots V_n] \in \mathbb{R}^{n \times n}$$

such that

$$UTAV = E = \text{diag}(u_1, \dots, u_p) \in \mathbb{R}^{m \times n}, \quad p = \min\{m, n\},$$

where $u_1, u_2, \dots, u_p \neq 0$

Pr 6 Let $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$ be unit 2-norm vectors that satisfy $Ax = my$ with $O = \|A\|_F$. From Theorem 2.1.1 there exist $V \in \mathbb{R}^{n \times (n-1)}$ and $U \in \mathbb{R}^{m \times (m-1)}$ so $V = [x \ | \] \in \mathbb{R}^{n \times n}$ and $U = [y \ | \ U_2] \in \mathbb{R}^{m \times m}$ are orthogonal. It is not hard to show that

$$U^T AV = \begin{bmatrix} \sigma & w^T \\ 0 & B \end{bmatrix} \equiv A_1$$

where $w \in \mathbb{R}^{n-1}$ and $B \in \mathbb{R}^{(m-1) \times (n-1)}$. Since

$$\left\| A_1 \left(\begin{bmatrix} \sigma \\ w \end{bmatrix} \right) \right\|_2^2 \geq (\sigma^2 + w^T w)^2$$

we have $\|A_1\|_2 \leq \sqrt{\sigma^2 + w^T w}$. But $\sigma = \|A\|_2 = \|A_1\|_2$, and so we must have $w = 0$. An obvious induction argument completes the proof of the theorem. \square

The σ are the singular values of A , the U are the left singular vectors of A , and the V are right singular vectors of A . Separate visualizations of the SVD are required

depending upon whether A has more rows or columns. Here are the 3by2 and 2by3 examples:

$$\begin{bmatrix} U_1 & U_2 & U_3 \\ \bar{U}_1 & \bar{U}_2 & \bar{U}_3 \\ \bar{U}_3 & \bar{U}_2 & \bar{U}_1 \end{bmatrix}^T \begin{bmatrix} a_1 & a_2 \\ a_2 & a_2 \\ a_3 & a_2 \end{bmatrix} \begin{bmatrix} v_1 & v_2 \\ \bar{v}_1 & \bar{v}_2 \\ \bar{v}_2 & \bar{v}_1 \end{bmatrix} = \begin{bmatrix} 0_1 & 0 \\ 0 & 2 \\ 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & u_{12} \\ \bar{U}_1 & u_{22} \end{bmatrix}^T \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{bmatrix} v_1 & v_2 & v_3 \\ \bar{v}_1 & \bar{v}_2 & \bar{v}_3 \\ \bar{v}_3 & \bar{v}_2 & \bar{v}_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0_2 & 0 \end{bmatrix}$$

In later chapters, the notation $\sigma_i(A)$ is used to designate the i th largest singular value of a matrix A . The largest and smallest singular values are important and for them we also have a special notation:

$$\sigma_{\max}(A) = \text{the largest singular value of matrix } A,$$

$$\sigma_{\min}(A) = \text{the smallest singular value of matrix } A$$

$\|A\|_2 = \sqrt{\lambda_{\max}(A^T A)}$

We establish a number of important corollaries to the SVD that are used throughout the book.

Corollary 2.4.2. If $A = U \Sigma V^T$ is the SVD of $A \in \mathbb{R}^{m \times n}$ and $m \geq n$, then for $i = 1:n$ $A \mathbf{v}_i = \sigma_i \mathbf{U}_i$ and $A^T \mathbf{v}_i = \sigma_i \mathbf{V}_i$.

Proof. Compare columns in $AV = U\Sigma$ and $A^T U = V\Sigma$. \square

There is a nice geometry behind this result. The singular values of a matrix A are the lengths of the semiaxes of the hyperellipsoid E defined by $E = \{Ax : \|x\|_2 = 1\}$. The semiaxis directions are defined by the \mathbf{v}_i and their lengths are the singular values.

It follows immediately from the corollary that

$$A^T A \mathbf{v}_i = \sigma_i^2 \mathbf{v}_i, \quad (24.1)$$

$$A A^T \mathbf{v}_i = \sigma_i^2 \mathbf{v}_i \quad (24.2)$$

for $i = 1:n$. This shows that there is an intimate connection between the SVD of A and the eigensystems of the symmetric matrices $A^T A$ and $A A^T$. See §8.6 and §10.4.

The 2-norm and the Frobenius norm have simple SVD characterizations.

Corollary 2.4.3. If $A \in \mathbb{R}^{m \times n}$, then

$$\|A\|_2 = \sigma_1, \quad \|A\|_F = \sqrt{\sigma_1^2 + \dots + \sigma_p^2},$$

where $p = \min\{m, n\}$.

Proof. These results follow immediately from the fact that $\|A\|_2 = \sqrt{\lambda_{\max}(A^T A)}$ if both the 2-norm and the Frobenius norm are used.

We show in §8.6 that if A is perturbed by a matrix E , then no singular value can move by more than $\|E\|_2$. The following corollary identifies two useful instances of this result.

Corollary 2.4.4. If $A \in \mathbb{R}^{m,n}$ and $E \in \mathbb{R}^{m,n}$, then

$$\text{Oax}(A+E) \leq \text{Oax}(A) + \|E\|_2,$$

$$\text{Oin}(A+E) \geq \text{Oin}(A) - \|E\|_2.$$

Proof. Using Corollary 24.2 it is easy to show that

$$\text{Oin}(A) \cdot \|x\|_2 \leq \|Ax\|_2 \leq \text{Oax}(A) \cdot \|x\|_2.$$

The required inequalities follow from this result. \square

If a column is added to a matrix, then the largest singular value increases and the smallest singular value decreases.

Corollary 2.4.5. If $A \in \mathbb{R}^{m,n}$, $m > n$, and $a \in \mathbb{R}^m$, then

$$\text{Oax}([A|a]) \geq \text{Oax}(A),$$

$$\text{Oin}([A|a]) \leq \text{Oin}(A).$$

Proof. Suppose $A = U \Sigma V^T$ is the SVD of A and let $x = V(:, 1)$ and $\tilde{A} = [A|a]$. Using Corollary 24.4 we have

$$\text{Oax}(\tilde{A}) = \|A\tilde{x}\|_2 = \left\| \begin{bmatrix} A \\ a \end{bmatrix} \right\|_2 \leq \text{Oax}(A).$$

The proof that $\text{Oin}(\tilde{A}) \geq \text{Oin}(A)$ is similar. \square

The SVD neatly characterizes the rank of a matrix and orthonormal bases for both its nullspace and its range.

Corollary 2.4.6. If A has r positive singular values, then $\text{rank}(A) = r$ and

$$\text{null}(A) = \text{span}\{V_{r+1}, \dots, V_n\},$$

$$\text{ran}(A) = \text{span}\{u_1, \dots, u_r\}.$$

Proof. The rank of a diagonal matrix equals the number of nonzero diagonal entries. Thus, $\text{rank}(A) = \text{rank}(E) = r$. The assertions about the nullspace and range follow from Corollary 24.2. \square

If A has rank r , then it can be written as the sum of r rank-1 matrices. The SVD gives us a particularly nice choice for this expansion.

Corollary 2.4.7. If $A \in \mathbb{R}^{m \times n}$ and $\text{rank}(A) = r$, then

$$A = \sum_{i=1}^r \sigma_i u_i v_i^T.$$

Pr of. This is an exercise in partitioned matrix multiplication:

$$(U\Sigma V)^T = ([\sigma_1 u_1 \ I\sigma_2 u_2 \ I \cdots \ I\sigma_r u_r \ I \ O \ I \ O \ I \cdots \ I \ O]) \begin{bmatrix} v \\ v \end{bmatrix} = \sum_{i=1}^r \sigma_i u_i v_i^T.$$

The intelligent handling of rank degeneracy is an important topic that we discuss in Chapter 5. The SVD has a critical role to play because it can be used to identify nearby matrices of lesser rank.

Theorem 2.4.8 (The Eckhart-Young Theorem). If $k \leq r = \text{rank}(A)$ and

$$A_k = \sum_{i=1}^k \sigma_i u_i v_i^T, \quad (243)$$

then

$$\min_{\text{rank}(B)=k} \|A - B\|_F = \|A - A_k\|_F = \sigma_{k+1}. \quad (244)$$

Pr of. Since $U\Sigma V = \text{diag}(\sigma_1, \dots, \sigma_k, 0, \dots, 0)$ it follows that A_k is rank k . Moreover, $U(A - A_k)V = \text{diag}(0, \dots, 0, \sigma_{k+1}, \dots, \sigma_p)$ and so $\|A - A_k\|_F = \sigma_{k+1}$.

Now suppose $\text{rank}(B) = k$ for some $B \in \mathbb{R}^{m \times n}$. It follows that we can find orthonormal vectors x_1, \dots, x_k so $\text{null}(B) = \text{span}\{x_1, \dots, x_k\}$. A dimension argument shows that

$$\text{span}\{x_1, \dots, x_k\} \cap \text{span}\{v_1, \dots, v_{k+1}\} = \{0\}.$$

Let z be a unit 2-norm vector in this intersection. Since $Bz = 0$ and

$$Az = \sum_{i=1}^{k+1} \sigma_i (v_i^T z) u_i,$$

we have

$$\|A - B\|_F = \|(A - B)z\|_2 = \|Az\|_2 = \sqrt{\sum_{i=1}^{k+1} \sigma_i^2 (v_i^T z)^2} \geq \sigma_{k+1},$$

completing the proof of the theorem. \square

Note that this theorem says that the smallest singular value of A is the 2-norm distance of A to the set of all rank-deficient matrices. We also mention that the matrix A_k defined in (243) is the closest rank- k matrix to A in the Frobenius norm.

=nBS1A TREATMENT 1)9A

If $A = U E V^T$ is the SYD of A and $m \leq n$, then

$$A = U E V^T$$

where

$$U_1 = U(:, 1:n) = [U_1 \dots U_n] \in \mathbb{R}^{m \times n}$$

and

$$E_1 = E(1:n, 1:n) = \text{diag}(\sigma_1, \dots, \sigma_n) \in \mathbb{R}^{n \times n}.$$

We refer to this abbreviated version of the SYD as the thin SVD.

$$\|A\|_F^2 = \|U_1\|_F^2 + \|\Sigma_1\|_F^2 + \|V^T\|_F^2 = \|U_1\|_F^2 + \|\Sigma_1\|_F^2 + \|V\|_F^2$$

Over the complex field the unitary matrices correspond to the orthogonal matrices. In particular, $QE \in \mathbb{C}^{m \times n}$ is unitary if $Q^H Q = Q Q^H = I_n$. Unitary transformations preserve both the 2-norm and the Frobenius norm. The SYD of a complex matrix involves unitary matrices. If $A \in \mathbb{C}^{n \times m}$, then there exist unitary matrices $U \in \mathbb{C}^{n \times n}$ and $V \in \mathbb{C}^{m \times m}$ such that

$$U^H A V = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{m \times n} \quad p = \min\{m, n\}$$

where $a_1, a_2, \dots, a_p \neq 0$. All of the real SYD properties given above have obvious complex analogs.

Problems

P2.4.1 **1. If $\mathbf{Q} = \mathbf{Q}_1 + i\mathbf{Q}_2$ then $\mathbf{Q}^H \mathbf{Q} = \mathbf{I}_n$** $\mathbf{Q}_1, \mathbf{Q}_2 \in \mathbb{R}^{n \times n}$, $\tilde{A}_{\infty} = (\theta_1 - \theta_2, \dots, \theta_{n-p}, \hat{\theta}_0)$, p

$$Z = \begin{bmatrix} Q_1 & -Q_2 \\ Q_2 & Q_1 \end{bmatrix}$$

$$z_j = \tilde{A}_{\infty j} = (0 \geq j)$$

P2.4.2 **3.1.8 2.2.2 If $A \in \mathbb{R}^{m \times n}$, then**

$$U_{\max}(A) = \frac{\mathbf{x}}{\|\mathbf{x}\|_2} \quad \frac{\mathbf{y}^T A \mathbf{x}}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2} \quad \mathbf{y} \in \mathbb{R}^m \quad \mathbf{x} \in \mathbb{R}^n$$

P2.4.3 **fe123@ 10 2.2.2. If $A \in \mathbb{R}^{m \times n}$, then $U_{\max}(A) = \sqrt{\lambda_{\max}(A)}$.** $U_{\max}(A)$ is the largest singular value of A . $S = \text{SVD}(A)$, $\mathbf{x} = \mathbf{u}_{\max}(A)$, $\mathbf{y} = \mathbf{v}_{\max}(A)$, $\mathbf{z} = \mathbf{u}_{\max}(A)^T \mathbf{v}_{\max}(A)$.

$\mathbf{S} = \mathbf{S}_{\max}(A)$, $\mathbf{y} = \mathbf{v}_{\max}(A)$, $\mathbf{z} = \mathbf{u}_{\max}(A)^T \mathbf{v}_{\max}(A)$.

P2.4.4 **1. If $X = (\theta_1, \dots, \theta_{n-p}, \hat{\theta}_0)$, then $X^T X = \mathbb{R}^{n \times n}$**

P2.4.5 **31@. If $A \in \mathbb{R}^{m \times n}$, then $\|A\|_F^2 = \sqrt{\lambda_{\max}(A^T A)}$**

P2.4.6 **f3.2 .. 2.8.8.8. d1 .. 2.2.2.8**

$$A = \begin{bmatrix} 1 & M \\ A & 1 \end{bmatrix}$$

.. 238 1 8.21.2

P2.4.7 **31fo 2.2.2 If $A \in \mathbb{R}^{m \times n}$, $\tilde{A} = n(\|A\|_F \sqrt{\text{rank}(A)})^{-1} A$ then $CBA = CEA$ if $\{237\}$.**

P2.4.8 **...18 If $A \in \mathbb{R}^{n \times n}$, $a_{ij} = 0$ for $i > j$, then $\|A\|_F = \sqrt{\sum_{i=1}^n c_i^2}$, where $c_i = \sqrt{\det(B_i)/\det(A)}$.**

$$\|A\|_F = \sqrt{\frac{\det(B)}{\det(A)}}$$

P2.4.9 ..1 fo2.2 . \geq .1.ot..1.1fo.. ..8. 21. ..2..bR 2.8. .12. 2.8... 8.2. .2.888.2.
8.8.. ... t

P2.4.10 ..1fo mf.. \geq_{Bu} ($= O$ ($= \frac{1}{n} \sum_i p_i X_{nn} = o$) (8.0

$$A \begin{bmatrix} I() & T() \\ aS() & I() \end{bmatrix}.$$

$= O \cdot U T A V = i \cdot \theta O$

$$U = \begin{bmatrix} I(/) & b=T(/) \\ =T(/) & T(/+) \end{bmatrix}, V = \begin{bmatrix} I() & 1=T(a) \\ =T() & I() \end{bmatrix}.$$

.), E = OO(U VEC(B) = T(B)) - iC A = (+ u)/2 CmF = (;)/2.

Notes and References for §2.4

... .8.. 1 614) STAE fSSx aaS=IdCB+=i= ESBIOIA .) } 1-T= SoICA AX -
 $\leq n r \text{isix } 14 = \text{maxz} = aa \cdot n 4x \sqrt{m=mxm} .ximxix \text{fixs } 1 \text{xifS a } \text{maxm}$
 $x \times R x m n S \text{S2af } 44x \text{ sim= } \text{xx } \text{nxan } \text{W4x } \text{ZSCx } x \& 2=x x ,$

, c x C m1879. Si4oxa nTMgMPC Mionale di Mathematiche 11, 98-106.
}. RaCpCnh Ig- (1989). c4(punAC4f MACnAvp1C.Magnagr sap1M.MCpM.5ABull.
AMS 45, 118-21.
mu1hapCp(1993). 17 r 69sM.g gBJahMoCpIC.xc.cg1AgAMMgTSIAM Review
35 r r AA1

ngBJa1gA AM-nM,6na.cgA1cnMA.Man.u., guAoCITgmaanoa Mn.paTaAngBJa
h2 uCA M.C.ug7JCpa6ouptau7a,M-aICnm,MECT pM kJMpA.CpapM,
}. R.eCpCnnlI lgo (1986). ckJa4AApgHMuGBlmMg.pMf 4ngJapBgpap5ChaT
Psychometrika 1, 211-218.

p-anapC MTB.Mg7A h2MngMmAMg5M oapC.aAac,

SwlgJop-C7mwi, MpaM1969. I tr duction to the Theory of Linear Non-Self Adjoint Operators, 4lap. r.C.J9hgwtgMman51
.. h1M..A(1970). Integr 1 Equations, }C lopMmraM.cpaAlMaAA7 lopu-a,

2amoMnJc pCngBC 1C.pufT M7gpg NCp. pJlan Jap.xpoM6-pMIMAAnA.pCMnAm
mMA.oAMm

1u. .auua (1987). ccJahuC. (Ap.xpoC.MgChooC.pufJM.lgpapAoa 5CnaCn.)gn
ApCMkgfa,aCb h6xCpa9gocuASIAM J. Numer. Anal. 4, 199-206.
ms. lgoo4 sg ICn7CnrlwuhapCp(1988). c4 ianapC,MTB.MgCp'gon-r rMpAa.
4 pgfM1kMgntwLin. Alg. Applic. 88/89, 317-328.
mbluC.Agn(1988). kJhuCe aAp.qpoC.MgChoo1C.pMfJM.lgpapAJa 5CnagBJa
PC.pMIMA J. Numer. Anal. 8, 3 4

2.5 Subspace Metrics

If the object of a computation is to compute a matrix or a vector, then norms are useful for assessing the accuracy of the answer or for measuring progress during an iteration. If the object of a computation is to compute a subspace, then to make similar comments we need to be able to quantify the distance between two subspaces. Orthogonal projections are critical in this regard. After the elementary concepts are established we discuss the CS decomposition. This is an SYD-like decomposition that is handy when we have to compare a pair of subspaces.

.h4A1A S7-- f.:}2A o7. li-G1A

Let $S \in \mathbb{R}^n$ be a subspace. $P \in \mathbb{R}^{n \times n}$ is the orthogonal projection onto S if $\text{ran}(P) = S$, $P^2 = P$, and $P^T = P$. From this definition it is easy to show that if $x \in \mathbb{R}^n$, then $Px \in S$ and $(I - P)x \in S^\perp$.

If P_1 and P_2 are each orthogonal projections, then for any $z \in \mathbb{R}^n$ we have

$$\| (P_1 - P_2)z \|_2^2 = (P_1 z)^T (I - P_2)z + (P_2 z)^T (I - P_1)z.$$

If $\text{ran}(P_1) = \text{ran}(P_2) = S$, then the right-hand side of this expression is zero, showing that the orthogonal projection for a subspace is unique. If the columns of $V = [v_1 | \dots | v_k]$ are an orthonormal basis for a subspace S , then it is easy to show that $P = VV^T$ is the unique orthogonal projection onto S . Note that if $v \in \mathbb{R}^n$, then $P_v = v v^T v^T$ is the orthogonal projection onto $S_v = \text{span}\{v\}$.

)hwh)r - .p (xi. os,r PEwi8zwrlr

There are several important orthogonal projections associated with the singular value decomposition. Suppose $A = U \Sigma V^T$ is the SVD of A and that $r = \text{rank}(A)$. If we have the U and V partitionings

$$U = \begin{bmatrix} U_r & \tilde{U}_r \\ r & m-r \end{bmatrix}, \quad V = \begin{bmatrix} V_r & \tilde{V}_r \\ r & n-r \end{bmatrix},$$

then

$$\begin{aligned} VV^T &= \text{projection onto } \text{null}(A) = \text{ran}(AT), \\ VV^T &= \text{projection onto } \text{null}(A), \\ UU^T &= \text{projection onto } \text{ran}(A), \\ UU^T &= \text{projection onto } \text{ran}(A) = \text{null}(AT). \end{aligned}$$

.h4h1A TG+}1 oAto -uA INA+fji oA

The one-to-one correspondence between subspaces and orthogonal projections enables us to devise a notion of distance between subspaces. Suppose S_1 and S_2 are subspaces of \mathbb{R}^n and that $\dim(S_1) = \dim(S_2)$. We define the distance between these two spaces by

$$\text{dist}(S_1, S_2) = \| P_1 - P_2 \|_2 \quad (25.1)$$

where P_i is the orthogonal projection onto S_i . The distance between a pair of subspaces can be characterized in terms of the blocks of a certain orthogonal matrix.

Theorem 2.5.1. Suppose

$$W = \begin{bmatrix} W_1 & W_2 \\ k & n-k \end{bmatrix}, \quad Z = \begin{bmatrix} Z_1 & Z_2 \\ k & n-k \end{bmatrix},$$

are n -by- n orthogonal matrices. If $S_1 = \text{ran}(W)$ and $S_2 = \text{ran}(Z)$ then

$$\text{dist}(S_1, S_2) = \| W_1^T Z_2 \|_2 = \| Z_1^T W_2 \|_2.$$

pr b We first observe that

$$\begin{aligned} \text{dist}(S_1, S_2) &= \|W_1 W - Z_1 Z_1^T\|_F \\ &= \|W_1^T (W_1 W - Z_1 Z_1^T) Z_2\|_2 \\ &= \left\| \begin{bmatrix} 0 & W_1^T Z_2 \\ -W_2^T Z_1 & 0 \end{bmatrix} \right\|_2. \end{aligned}$$

Note that the matrices $W_1 Z_1$ and $W_2 Z_2$ are submatrices of the orthogonal matrix

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix} \equiv \begin{bmatrix} W_1^T Z_1 & W_1^T Z_2 \\ W_2^T Z_1 & W_2^T Z_2 \end{bmatrix} = W^T Z. \quad (252)$$

Our goal is to show that $\|Q\|_2 = \|Q\|_F = 1$. Since Q is orthogonal it follows from

$$Q \begin{bmatrix} x \\ 0 \end{bmatrix} = \begin{bmatrix} Q_1 x \\ Q_2 x \end{bmatrix}$$

that $1 = \|Q\|_F = \sqrt{\|Q\|_2^2} = \sqrt{\|Q_1\|_2^2 + \|Q_2\|_2^2}$ for all $x \in \mathbb{R}^n$.

$$\|Q_{12}^T\|_2^2 = 1 - \sigma_{\min}(Q_{11}^T)^2,$$

$$0 \leq \text{dist}(S_1, S_2) \leq 1.$$

$$= \text{half}(\mathbf{A}) - \mathbf{Q}^T \mathbf{A} \mathbf{Q} = \mathbf{Q}^T (\mathbf{Q} \mathbf{Q}^T - \mathbf{I}) \mathbf{A} = \mathbf{Q}^T \mathbf{G} \mathbf{G}^T \mathbf{A}$$

The blocks of an orthogonal matrix partitioned into 2 by 2 blocks have highly related SVDs. This is the gist of the CS decomposition. We prove a very useful special case first.

Theorem 2.5.2 (The CS Decomposition (Thin Version)). Consider the matrix

$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q}_1 \\ \mathbf{Q}_2 \end{bmatrix}, \quad \mathbf{Q}_1 \in \mathbb{R}^{m_1 \times n_1}, \mathbf{Q}_2 \in \mathbb{R}^{m_2 \times n_1},$$

where $m_1 \leq n_1$ and $m_2 \leq n_1$. If the columns of \mathbf{Q} are orthogonal, then there exist orthogonal matrices $\mathbf{U}_1 \in \mathbb{R}^{m_1 \times m_1}$, $\mathbf{U}_2 \in \mathbb{R}^{m_2 \times m_2}$, and $\mathbf{V} \in \mathbb{R}^{n_1 \times n_1}$ such that

$$\begin{bmatrix} \mathbf{U}_1 & 0 \\ 0 & \mathbf{U}_2 \end{bmatrix}^T \begin{bmatrix} \mathbf{Q}_1 \\ \mathbf{Q}_2 \end{bmatrix} \mathbf{V}_1 = \begin{bmatrix} \mathbf{C} \\ \mathbf{S} \end{bmatrix}$$

where

$$\mathbf{C} = \text{diag}(\cos(\theta_1), \dots, \cos(\theta_{n_1})) \in \mathbb{R}^{m_1 \times n_1},$$

$$\mathbf{S} = \text{diag}(\sin(\theta_1), \dots, \sin(\theta_{n_1})) \in \mathbb{R}^{m_2 \times n_1},$$

and

$$0 \leq \theta_1 \leq \theta_2 \leq \dots \leq \theta_{n_1} \leq \frac{\pi}{2}.$$

Proof. Since $\|\mathbf{Q}\|_F = \|\mathbf{Q}^T \mathbf{Q}\|_F = 1$, the singular values of \mathbf{Q} are all in the interval $[0, 1]$. Let

$$\mathbf{U}_1^T \mathbf{Q}_1 \mathbf{V}_1 = \mathbf{C} = \text{diag}(c_1, \dots, c_{n_1}) = \begin{bmatrix} I_t & 0 \\ 0 & \Sigma \\ \vdots & \vdots \end{bmatrix}_{m_1-t} \in \mathbb{R}^{n_1-t}$$

be the SVD of \mathbf{Q}_1 where we assume

$$1 = c_1 = \dots = c_t > c_{t+1} = \dots = c_{n_1} = 0$$

To complete the proof of the theorem we must construct the orthogonal matrix \mathbf{U}_2 . If

$$\mathbf{Q} \mathbf{V}_1 = [\mathbf{W}_1 \mathbf{W}_2]_{n_1-t},$$

then

$$\begin{bmatrix} \mathbf{U}_1 & 0 \\ 0 & I_{m_2} \end{bmatrix}^T \begin{bmatrix} \mathbf{Q}_1 \\ \mathbf{Q}_2 \end{bmatrix} \mathbf{V}_1 = \begin{bmatrix} I_t & 0 \\ 0 & \Sigma \\ W_1 & W_2 \end{bmatrix}.$$

Since the columns of this matrix have unit 2-norm, $\mathbf{W}_1 = \mathbf{O}$. The columns of \mathbf{W}_2 are nonzero and mutually orthogonal because

$$\mathbf{W}_2 \mathbf{W}_2^T = \mathbf{I}_{n_1-t} = \mathbf{E} \mathbf{T} \mathbf{E}^T = \text{diag}(1 - c_{t+1}, \dots, 1 - c_{n_1}).$$

is nonsingular. If $s_k = \sqrt{1 - c_k^2}$ for $k = 1:n_1$, then the columns of

$$Z = W \text{diag}(1/S_1, \dots, 1/S_{n_1})$$

are orthonormal. By Theorem 2.1.1 there exists an orthogonal matrix $U \in \mathbb{R}^{m_1 \times n_1}$ such that $U^T Z = Z$. It is easy to verify that

$$U^T Q_{V_1} = \text{diag}(s_1, \dots, S_{n_1}) = S_0.$$

Since $c_{+s} = 1$ for $k = 1:n_1$, it follows that these quantities are the required cosines of the angles. \square

By using the same techniques it is possible to prove the following more general version of the decomposition:

Theorem 2.5.3 (CS Decomposition). Suppose

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}_{m_1 \times m_2}^{n_1 \times n_2}$$

is a square orthogonal matrix and that $m_1 \leq n_1$ and $m_2 \leq n_2$. Define the nonnegative integers p and q by $p = \max\{Q_{11} \cdot m_2, m_2\}$ and $q = \max\{Q_{22} \cdot n_1, n_1\}$. There exist orthogonal $U \in \mathbb{R}^{m_1 \times m_1}$, $V \in \mathbb{R}^{m_2 \times n_2}$, $Z \in \mathbb{R}^{n_1 \times n_2}$, and $W \in \mathbb{R}^{n_2 \times n_2}$ such that if

$$U = \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix} \quad \text{and} \quad V = \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix},$$

then

$$U^T Q V = \begin{bmatrix} I & 0 & 0 & 0 & 0 \\ 0 & C & S & 0 & 0 \\ 0 & 0 & 0 & 0 & I \\ 0 & S & -C & 0 & 0 \\ 0 & 0 & 0 & I & 0 \end{bmatrix}_{p \times q}^{n_1-p \times n_2-q} \quad \begin{matrix} p \\ n_1-p \\ m_1-n_1 \\ n_2-q \\ q \end{matrix}$$

where

$$C = \text{diag}(\cos(Q_{11} \cdot i), \dots, \cos(Q_{11} \cdot q))$$

by the theorem. Note that the blocks in the transf med \mathbf{Q} i.e., the $\mathbf{U}^T \mathbf{Q} \mathbf{V}$, are diagonal-like but not necessarily diagonal. Indeed, we have presented it, the CS decomposition gives us four unnormalized SVDs. If \mathbf{Q}_1 has more rows than columns, then $p = 0$ and the reduction looks like this (for example):

$$\mathbf{U}^T \mathbf{Q} \mathbf{V} = \left[\begin{array}{c|ccccc} \mathbf{C} & 0 & \mathbf{S} & 0 & 0 & 0 & 0 \\ 0 & \mathbf{Q} & 0 & \mathbf{S} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline \mathbf{S} & 0 & -\mathbf{C} & 0 & 0 & 0 & 0 \\ 0 & \mathbf{S} & 0 & -\mathbf{Q} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array} \right].$$

On the other hand, if \mathbf{Q}_1 has more columns than rows, then $q = 0$ and the decomposition has the form

$$\mathbf{U}^T \mathbf{Q} \mathbf{V} = \left[\begin{array}{c|cc} 1 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{Q} & 0 & \mathbf{S} & 0 \\ 0 & 0 & \mathbf{C} & 0 & \mathbf{S} \\ \hline 0 & \mathbf{S} & 0 & -\mathbf{G} & 0 \\ 0 & 0 & \mathbf{S} & 0 & -\mathbf{C} \end{array} \right].$$

Regardless of the partitioning, the essential message of the CS decomposition is that the SVDs of the Q blocks are highly related.

Problems

P2.5.1 .386 1.1.1 .. .8.1.8.8... .8 8fe81.8. = I - 6 D(A31)I' I

P2.5.2 f... .8 1.8 fe.2... 82.8. 8.1.8.8... .8 8. fe8.

P2.5.3 .2..8 .8 $S_1 = \text{span}\{x\}$ ApS $S_2 = \text{span}\{y\}$, 28tilt x ApS y AlS .pmOne,Md8 ItACMlmpR².
 M_1/mp Mpc8 Cit Strnpmi.MpE.aCg .§ a8M2 Catt(S_1, S_2) = $\sqrt{1 - |xTy|}$. 262AEi)1 nC2k
 kCAT=k2a,AA,AA)

$$Q = \begin{bmatrix} Q_1 & Q_2 \\ -Q_2 & Q_1 \end{bmatrix}, \quad Q_1, Q_2 \in \mathbb{R}^{n \times n}.$$

2.6 The Sensitivity of Square Systems

We used tools developed in previous sections to analyze the linear system problem $Ax = b$ where $A \in \mathbb{R}^{n \times n}$ is nonsingular and $b \in \mathbb{R}^n$. Our aim is to examine how perturbations in A and b affect the solution x . Higham (ASNA) offers a more detailed treatment.

linhb elb upb eltfK eTeb

If

$$A = \sum_{i=1}^n \sigma_i u_i v^T = U \Sigma V^T$$

is the SVD of A , then

$$x = A^{-1}b = (U \Sigma V^T)^{-1}b = \sum_{i=1}^n \frac{u_i^T b}{\sigma_i} v_i. \quad (26.1)$$

This expansion shows that small changes in A or b can induce relatively large changes in x if σ_n is small.

It should come as no surprise that the magnitude of σ_n should have a bearing on the sensitivity of the $Ax = b$ problem. Recall from Theorem 24.8 that σ_n is the 2-norm distance from A to the set of singular matrices. As the matrix of coefficients approaches this set, it is intuitively clear that the solution x should be increasingly sensitive to perturbations.

linhb lkt Tf Tkb

A precise measure of linear system sensitivity can be obtained by considering the parameterized system

$$(A + \epsilon F)x(\epsilon) = b + \epsilon f, \quad x(0) = x,$$

where $F \in \mathbb{R}^{n \times n}$ and $f \in \mathbb{R}^n$. If A is nonsingular, then it is clear that $x(\cdot)$ is differentiable in a neighborhood of zero. Moreover, $x(0) = A^{-1}(f - \epsilon Fg)$ goes to zero as ϵ goes to zero.

$$x(\epsilon) = x + \epsilon \dot{x}(0) + O(\epsilon^2).$$

$$\frac{\|x(\epsilon) - x\|}{\|x\|} \leq |\epsilon| \|A^{-1}\| \left\{ \frac{\|f\|}{\|x\|} + \|F\| \right\} + O(\epsilon^2).$$

-

$$\frac{\|x(\epsilon) - x\|}{\|x\|} \leq \kappa(A)(\rho_A + \rho_b) + O(\epsilon^2)$$

where

$$\rho_A = \|\mathbf{f}\|_F \quad \text{and} \quad \rho_b = \|\mathbf{m}\|_F$$

represent the relative errors in \mathbf{A} and \mathbf{b} respectively. Thus, the relative error in x can be $\kappa(A)$ times the relative error in \mathbf{A} and \mathbf{b} . In this sense, the condition number $\kappa(A)$ quantifies the sensitivity of the $Ax = \mathbf{b}$ problem.

Note that $\kappa(\cdot)$ depends on the underlying norm and subscripts are used accordingly, e.g.,

$$\kappa_2(A) = \|A\|_2 \|A^{-1}\|_2 = \frac{\sigma_{\max}(A)}{\sigma_{\min}(A)}. \quad (265)$$

Thus, the 2-norm condition of a matrix A measures the elongation of the hyperellipsoid $\{Ax : \|x\|_2 = 1\}$.

We mention two other characterizations of the condition number. For p-norm condition numbers we have

$$\frac{1}{\kappa_p(A)} = \min_{A + \Delta A \text{ singular}} \frac{\|\Delta A\|_p}{\|A\|_p}. \quad (266)$$

This result may be found in Kahan (1966) and shows that $\kappa_p(A)$ measures the relative p-norm distance of A to the set of singular matrices.

For any norm we also have

$$\kappa(A) = \lim_{f: O} \sup_{\|\Delta A\| \leq \epsilon \|A\|} \frac{\|(A + \Delta A)^{-1} - A^{-1}\|}{\epsilon} \frac{1}{\|A^{-1}\|}. \quad (267)$$

This imposing result merely says that the condition number is a normalized Frechet derivative of the map $A \rightarrow A^{-1}$. Further details may be found in Rice (1966). Recall that we were initially led to $\kappa(A)$ through differentiation.

If $\kappa(A)$ is large, then A is said to be an ill-conditioned matrix. Note that this is a norm-dependent property.¹ However, any two condition numbers $\kappa_\alpha(\cdot)$ and $\kappa_\beta(\cdot)$ on $\mathbb{R}^{n \times n}$ are equivalent in that constants c_1 and c_2 can be found for which

$$c_1 \kappa_\alpha(A) \leq \kappa_\beta(A) \leq c_2 \kappa_\alpha(A), \quad A \in \mathbb{R}^{n \times n}.$$

For example, on $\mathbb{R}^{n \times n}$ we have

$$\begin{aligned} \frac{1}{n} \kappa_2(A) \circ \kappa_1(A) &\leq n \kappa_2(A), \\ \frac{1}{n} \kappa_\infty(A) \leq \kappa_2(A) &\leq n \kappa_\infty(A), \\ \frac{1}{n^2} \kappa_1(A) \leq \kappa_\infty(A) &\leq n^2 \kappa_1(A). \end{aligned} \quad (268)$$

Thus, if a matrix is ill-conditioned in the α -norm it is ill-conditioned in the β -norm modulo the constants c_1 and c_2 above.

For any of the p-norms, we have $\kappa_p(A) \geq 1$. Matrices with small condition numbers are said to be well-conditioned. In the 2-norm orthogonal matrices are perfectly conditioned because if Q is orthogonal, then $\kappa_2(Q) = \|Q\|_2 \|Q^T\|_2 = 1$.

¹ *DCA, aMSiJtpSa rJMpCutStrnp.C.MMAHAI)l3Tnat 8ACCtl JnlartS .p 7-*

$b \neq 0$ and $A = I + A^{-1}f$. Then $\det(A) = \det(I + A^{-1}f)$.

It is natural to consider how well determinant size measures ill-conditioning. If $\det(A) = f$ is equivalent to singularity, is $\det(A) \approx 0$ equivalent to near singularity? Unfortunately, there is little correlation between $\det(A)$ and the condition of $Ax = b$. For example, the matrix B_n defined by

$$B_n = \begin{bmatrix} 1 & -1 & & & \\ 0 & 1 & & & \\ \vdots & & \ddots & & \\ 0 & & & 1 & -1 \\ & & & 0 & \dots \end{bmatrix} \quad (269)$$

has unit determinant, but $\text{cond}(B_n) = n! \approx 10^{n!}$. On the other hand, a very well-conditioned matrix can have a very small determinant. For example,

$$D_n = \text{diag}(10^{-1}, \dots, 10^{-1}) \in \mathbb{R}^{n \times n}$$

satisfies $\det(D_n) = 1$ although $\text{cond}(D_n) = 10^n$.

Call that the derivation of (264) was valuable because it highlighted the connection between " (A) " and the rate of change of $x(j)$ at $f = 0$. However, it is a little unsatisfying because it is contingent on f being "small enough" and because it sheds no light on the size of the $O(f^j)$ term. In this and the next subsection we develop some additional Ax = b perturbation theorems that are completely rigorous.

We first establish a lemma that indicates in terms of " (A) " when we can expect a perturbed system to be nonsingular.

Lemma 2.6.1. Suppose

$$Ax = b \quad A \in \mathbb{R}^{n \times n}, 0 \neq b \in \mathbb{R}^n,$$

$$(A + \epsilon A)y = b + \epsilon b \quad A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n$$

with $\|A\| \leq 1$, $\|\epsilon A\| \leq \rho$ and $\|b\| \leq 1$. If $\epsilon^r(A) = r < 1$, then $A + \epsilon A$ is nonsingular and

$$\frac{\|b\|}{\|x\|} \leq \frac{1+r}{1-r}.$$

Proof. Since $\|A^{-1}\| \leq \|A\|^{-1}$ and $\|A^{-1}f\| \leq \|A^{-1}\| \|f\| = r < 1$ it follows from Theorem 2.34 that $(A + \epsilon A)$ is nonsingular. Using Lemma 2.33 and the equality

$$(I + A^{-1} \epsilon A)y = x + A^{-1} \epsilon b$$

we find

$$\begin{aligned} \|y\| &\leq \|(I + A^{-1} \epsilon A)^{-1}\| (\|x\| + \epsilon \|A^{-1} \epsilon b\|) \\ &\leq \frac{1}{1-r} (\|x\| + \epsilon \|A^{-1} \epsilon b\|) = \frac{1}{1-r} \left(\|x\| + r \frac{\|b\|}{\|A\|} \right). \end{aligned}$$

Since $\|b\| = \|Ax\| \leq \|A\|\|x\|$ it follows that

$$\|y\| \leq \frac{1}{1-r}(\|x\| + r\|x\|)$$

and this establishes the required inequality. So

We are now set to establish a rigorous $Ax = b$ perturbation bound.

Theorem 2.6.2. If the conditions of Lemma 2.6.1 hold then

$$\frac{\|y - x\|}{\|x\|} \leq \frac{1}{1-r} K(A). \quad (26.10)$$

Proof Since

$$y - x = A^{-1}\Delta b - A^{-1}\Delta Ay \quad (26.11)$$

we have

$$\|y - x\| \leq \epsilon \|A^{-1}\| \|b\| + \epsilon \|A^{-1}\| \|A\| \|y\|.$$

Thus

$$\frac{\|y - x\|}{\|x\|} : f K(A) \frac{\|b\|}{\|A\| \|x\|} + f K(A) \frac{\|y\|}{\|x\|} : f \left(1 + \frac{1+r}{1-r}\right) K(A),$$

from which the theorem readily follows. So

A small example helps put this result in perspective. The $Ax = b$ problem

$$\begin{bmatrix} 1 & 0 \\ 0 & 10^{-6} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 10^{-6} \end{bmatrix}$$

has solution $x = [1, 1]^T$ and condition $K_0(A) = 10^6$. If $b = [10^6, 0]^T$, $A = Q$ and $(A + \Delta A)y = b + \Delta b$ then $y = [1 + 10^{-5}, 1]^T$ and the inequality (26.10) says

$$10^{-6} = \frac{\|X - Y\|_0}{\|x\|_0} \ll \frac{\|b - b\|_0}{\|b\|_0} K_0(A) = 10^6 \cdot 10^6 = 1$$

Thus, the upper bound in (26.10) can be a gross overestimate of the error induced by the perturbation.

On the other hand, if $b = (0, 10^6)^T$, $A = Q$ and $(A + \Delta A)y = b + \Delta b$ then this inequality says that

$$\frac{10^0}{10^0} \leq 2 \times 10^{-6} \cdot 10^6.$$

Thus, there are perturbations for which the bound in (26.10) is essentially attained.

= $\|A\|_F$ q $\|A\|_F$ $\|A\|_F$ $\|A\|_F$

An interesting refinement of Theorem 26.2 results if we extend the notion of absolute value to matrices.

$$F = (f_{ij}) \in \mathbb{R}^{m \times n} \quad \Rightarrow \quad |F| = (|f_{ij}|) \in \mathbb{R}^{m \times n}.$$

This notation together with a matrix-level version of $\kappa(A)$ makes it easy to specify componentwise error bounds. If $F, G \in \mathbb{R}^{m \times n}$, then

$$|F| + |G| \Leftrightarrow |f_{ij}| \leq |g_{ij}|$$

for all i and j . Also note that if $F \in \mathbb{R}^{m \times n}$ and $G \in \mathbb{R}^{n \times n}$, then $|FG| \leq |F|_a |G|$. With these definitions and facts we obtain the following refinement of Theorem 26.2.

Theorem 2.6.3. Suppose

$$Ax = b, \quad A \in \mathbb{R}^{n \times n}, \quad 0 \neq b \in \mathbb{R}^n,$$

$$(A + \Delta A)y = b + \Delta b, \quad \Delta A \in \mathbb{R}^{n \times n}, \quad \Delta b \in \mathbb{R}^n,$$

and that $|\Delta A| \leq \epsilon |A|$ and $|\Delta b| \leq \epsilon |b|$. If $\delta \kappa_\infty(A) = r < 1$, then $(A + \Delta A)$ is nonsingular and

$$\frac{\|y - x\|_\infty}{\|x\|_\infty} \leq \frac{2\epsilon}{1-r} \cdot \|A^{-1}\| |A| \|_\infty. \quad (26.12)$$

Pr b Since $\|\Delta A\|_\infty \leq \epsilon \|A\|_\infty$ and $\|\Delta b\|_\infty \leq \epsilon \|b\|_\infty$ the conditions of Lemma 26.1 are satisfied in the infinity norm. This implies that $A + \Delta A$ is nonsingular and

$$\frac{\|y\|_\infty}{\|x\|_\infty} \leq \frac{1+r}{1-r}.$$

Now using (26.11) we find

$$\begin{aligned} \|y - x\| &\leq \|A^{-1}\| |\Delta b| + \|A^{-1}\| |\Delta A| |y| \\ &\leq \epsilon \|A^{-1}\| |b| + \epsilon \|A^{-1}\| |A| |y| + \epsilon \|A^{-1}\| |A| (|x| + |y|). \end{aligned}$$

If we take norms, then

$$\|y - x\|_\infty \leq \epsilon \|A^{-1}\| |A| \|_\infty \left(\|x\|_\infty + \frac{1+r}{1-r} \|x\|_\infty \right).$$

The theorem follows upon division by $\|x\|_\infty$.

The quantity $\|A^{-1}\| |A| \|_\infty$ is known as the *Skeel condition number* and there are examples where it is considerably less than $\kappa_\infty(A)$. In these situations, (26.12) is more informative than (26.10).

Nom bounds are frequently good enough when assessing error, but sometimes it is desirable to examine error at the component level. Oettli and Prager (1964) have an interesting result that indicates if an approximate solution $\hat{x} \in \mathbb{R}^n$ to the n -by- n

system $\mathbf{Ax} = \mathbf{b}$ satisfies a perturbed system with prescribed structure. Consider the problem of finding $\mathbf{A}\in\mathbb{R}^{n\times n}$, $\mathbf{b}\in\mathbb{R}^n$, and $\mathbf{w}\in\mathbb{R}^n$ such that

$$(\mathbf{A} + \mathbf{E})\mathbf{x} = \mathbf{b} + \mathbf{f} \quad \text{and} \quad \|\mathbf{w}\|_1 \leq \|\mathbf{x}\|_1, \quad (26.13)$$

where $\mathbf{E}\in\mathbb{R}^{n\times n}$ and $\mathbf{f}\in\mathbb{R}^n$ are given. With proper choice of \mathbf{E} and \mathbf{f} , the perturbed system can take on certain qualities. For example, if $\mathbf{E} = \mathbf{A}$ and $\mathbf{f} = \mathbf{b}$ and \mathbf{w} is small, then \mathbf{x} satisfies a nearby system in the componentwise sense. The authors show that for a given \mathbf{A} , \mathbf{b} , \mathbf{x} , \mathbf{E} , and \mathbf{f} the smallest \mathbf{w} possible in (26.13) is given by

$$\omega_{\min} = \max_{1 \leq i \leq n} \frac{\|\mathbf{Ax}_i - \mathbf{b}_i\|}{\|(\mathbf{E} + \mathbf{A})\mathbf{x}_i + \mathbf{f}_i\|}.$$

If $\mathbf{Ax} = \mathbf{b}$ then $\omega_{\min} = 0$. On the other hand, if $\mathbf{L}\mathbf{u}\mathbf{d}\mathbf{l} = \mathbf{0}$, then \mathbf{x} does not satisfy any system of the prescribed perturbation structure.

Problems

P2.6.1 ..86 1.1.1 $\|\mathbf{I}\| = \mu \leq k^n \kappa(\mathbf{A})$.-

P2.6.2 ..86 1.1 N.8. .8.2 $\kappa(\mathbf{AB}) \leq \kappa(\mathbf{A})\kappa(\mathbf{B})$ ApSC(AC $\kappa(\alpha\mathbf{A}) = \kappa(\mathbf{A})$, l A,, pMp9tlMx.

P2.6.3 78..181.80 48.3..1.84 81 $\mathbf{R}^{m \times n} : n$ CMCu, nrepMl8AM, S.CmME GAEI.Ata

$$\mathbf{B} = \begin{bmatrix} \mathbf{I}_m & \mathbf{X} \\ \mathbf{0} & \mathbf{I}_n \end{bmatrix} \quad \text{ApS} \quad \mathbf{C} = \begin{bmatrix} \vdots \end{bmatrix}.$$

P2.6.4 .2. .8.8 $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{E} \leq \mathbf{E}'$, $\mathbf{E}' > \mathbf{B}^n$ c, , *, $\leq k, -0 \mathbf{R}_i$, ApS C(ACC8tlt.a, MAh CM 8A/t A a.,), Al h AuAp.p) Cu, IA,, MEij. (AC AA, hMAM,A,rStA.MrCA $^{-1}$? -pC0,at Cut 2utl8ApreeMll.Am, ,l8r,Al

P2.6.58.8 $\mathbf{A} \in \mathbb{R}^{n \times n}$, $(\mathbf{A}, \mathbf{E}, \mathbf{b}) \in \mathbb{R}^n$, $\mathbf{Ax} = \mathbf{b}$, $\mathbf{A}\mathbf{S} \mathbf{C} = \mathbf{A}^{-1}$. gat Cut2utlCtA,relMll.AmP ,lcr,A CMuM2CuAC

$$\frac{\partial x_k}{\partial a_{ij}} = -x_j c_{ki}.$$

Notes and References for §2.6

m684..1fe884.8.1.. 1.8.82.... .4.8.1..18. ..or

k 7.06 21 m.88.. 8138..1.8.fo7 IAM J. Numer. Anal. 3, ngs7s11

.1 AuApS E 6-4, .8t1AA, ftptAl e1)t.lAPL Canadian Math. Bull. 9, κ, κ ls-

ot.ltpAta ,l AM8JM,t,C2.at JtlCrl.AC}M,CutMlh,Al.St0

.rg ltCClApS.3 NIA)tl ..E- 6 I 4.McJAC..m1)CMEJJIMR8ACt2M,CmMmpptAt2rAC.Mp&.Cu CmItpxlMl M,pS& Mt/A.tptCa ApSo,uC -ApS 2.Sta§LNumer. Math. 6 / lr 561[1

1wRMAa emn9,u, 5oA2i[, [,.wc9,nmA ll,2Ml.Ar hA2auAS24,pe2a.e2e3TSIAM J. Numer. Anal. 16 [r uE6

51h haaaji[,[,kcheMnpaolapS.eh2eoS2Srie,AbMeR.S lueSln,T. ACM 26, 6[65r Nu 1u1.hula, ii[[N, 6 dkJa)]1Alnan28uAAMA2ena 21haepaAlMep Pe2Spf, SIAM J. Matrix Anal. Applic. 13, s sE)

t3p, -.)uAc ApS, pL)uA8 .sEE@, 4.McJMptpC2.at NtlC.l.ACmMputMlh,l ftmpAp2haCt8a 2.Cu 1r ,CmJ@m)uCre-ApSSta§LLin. Alg. Applic. 174, s,s -snE-

,Lp1-.)uAc ..EE, 4e 2rlth MEM8JM,p,C2maNtlCrl.ACmM@Mlh .p,8 tl.AA1ftm,tAk,)t.lAP3 .p Mathematics of Computation 1943-1993: A Half Century of Computational Mathematics, .

CACaA(mtS16P5M,8t -g MEWIMAttS.p)ME 2h8JMpmw,mtS1ACutcACmAe§tlmAAP1ACutm 8ACmA2M AmtCmMimStpAtD-

*ut Am7A6M Eut AMpSm7. At8t 8t .Ita uM2ptAl A)mItp Ax = b JlMc1,8 ma a7pM 16lm7h9
ut8J Ml7ApAM EpM2.p) uM2p-Al m4A)mItp JlMc,t8 i3 6 S./A.,i Ml.paM,,.lt JlM.1t8 ug AM8t
i ct AltA.A7tS.p 8ApLM8JriAiMpAt77.pja§ att0*

IEEE Trans. Autom. Control. AC-30, Errata

pra.d tt88t, (E(a)))xE kN0<=k EQP knQ0=EO=R- ((•=HE r -0) Numer. Math. 51,

Applications of Matrix Theory,

rap CM1t ApS2⁻ Alptii(tSap⁺)_n(+1ⁿ)₋V1ⁿ)₋ 1

$$1 \cdot \dots \cdot 1^{m_1}(-1)^{n_1} - (-1)^{m_1} \cdot 1^{n_1} = 1^{m_1+n_1}(-1)^{m_1+n_1} + (-1)^{m_1+n_1}(-1)^{m_1+n_1} = 0$$

"*U*"(*E*(*a*)*r*)*NO*(*E*•*r*) = *r* *kNk fm*(*E*<*G*)*E* = (*I*<=*E*•*r*) - *Q*<=*Oks**N**ath.* *Com-*
put. 50, -, *ES-91*

3 127t26li 0((()))k ON (<@E(kFr - k_N) EIN \$IAM Review 82, n. [5u# 1r
L Manna 4616eo - enn P whtaAa(E(a)N))k<-k_G(EDF k<@ k<-k_AE(<eO=BI =I
L_G \$IAM Review 82, n. [5u# 1r

$$kQ = S \nabla M \cdot \nabla \phi - \left(A_{\perp} \nabla V(T) \right) \left(\frac{\partial}{\partial x} \nabla V(T) \right)^T + \left(U_{\perp} \right) \left(\frac{\partial}{\partial x} U_{\perp} \right)^T + C_{\perp}^*$$

rut Jl .1t8 M E.p.8.1.p) $\kappa_2(A + UV^T)$)["]) $UV^T - \underline{\underline{0}}$) r["]) (U $\underline{\underline{1}}$) C)["]
 $\kappa_2(A + UV^T)$)["]) $(UV^T - \underline{\underline{0}})$)["]) $(U \underline{\underline{1}}) C$)["]

SIAM J. Matrix Anal. Appl. 2, yNly

27 Finite Precision Matrix Computations

R underlies are part of what makes the field of matrix computations so challenging. In this section we describe a model of floating point arithmetic and then use it to develop error bounds for floating point dot products, saxpy's, matrix-vector products, and matrix-matrix products.

, hTghb b mstT" Tf lf fD Ptf kb

Suppose we have a base 10 calculator that represents nonzero numbers in the following style:

$$x = \pm d_0.d_1d_2 \times 10^e \quad \text{where} \quad \left\{ \begin{array}{l} 1 \leq d_0 \leq 9 \\ 0 \leq d_1 \leq 9 \\ 0 \leq d_2 \leq 9 \\ -9 \leq e \leq 9 \end{array} \right.$$

Let us call these numbers *floating point* numbers. After playing around a bit we make a number of observations:

- The precision of the calculator has to do with the "length" of the significand $d_0.d_1d_2\dots$. For example, the number π would be represented as 3.14×10^3 , which has a relative error approximately equal to 10^{-3} .
 - There is not enough "room" to store exactly the results from most arithmetic operations between floating point numbers. Sums and products like

$$(123 \times 10^6 + 456 \times 10^4) = 1275600$$

involve more than three significant digits. Results must be rounded in order to "fit" the 3 digit format, e.g., $\text{round}(1275600) = 1.28 \times 10^6$ $\text{round}(56088) = 561 \times 10^2$.

- If zero is to be a floating point number (and it must be), then we need a special convention for its representation, e.g., 000×10^0 .
- In contrast to the real numbers, there is a smallest positive floating point number ($N_{\min} = 1.00 \times 10^{-9}$) and there is a largest positive floating point number ($N_{\max} = 999 \times 10^9$).
- Some operations yield answers whose exponents exceed the 1-digit allocation, e.g., $\{123 \times 10^4\} \# (456 \times 10^7)$ and $\{1.23 \times 10^{-3}\} / (456 \times 10^8)$.
- The set of floating point numbers is finite. For the toy calculator there are $2 \times 9 \times 10 \times 10 \times 10 + 1 = 34201$ floating point numbers.
- The spacing between the floating point numbers varies. Between 100×10^0 and $100 \times 10^{+1}$ the spacing is 10×10^{-2} .

The careful design and analysis of floating point computation requires an understanding of these inexactitudes and limitations. How are results rounded? How accurate is floating point arithmetic? What can we say about a sequence of floating point operations?

11011b (AAAabdktf 1"b 2Tkf b ex Tf & Tfb

To build a solid, practical understanding of finite precision computation, we set aside our toy, motivational base 10 calculator and consider the key idea behind the widely accepted IEEE floating point standard. The IEEE standard includes a 32-bit single format and a 64-bit double format. We will illustrate concepts using the latter as an example because typical accuracy requirements make it the format of choice.

The importance of having a standard for floating point arithmetic that is upheld by hardware manufacturers cannot be overstated. After all, floating point arithmetic is the foundation upon which all of scientific computing rests. The IEEE standard promotes software reliability and enables numerical analysts to make rigorous statements about computed results. Our discussion is based on the excellent book by Overton (2001).

The 64-bit double format allocates a single bit for the sign of the floating point number, 52 bits for the mantissa, and eleven bits for the exponent:

$$x : \boxed{\pm | a_1 a_2 \dots a_{11} | b_1 b_2 \dots b_{52}} . \quad (27.1)$$

The "formula" for the value of this representation depends upon the exponent bits.

If $a_1 \dots a_n$ is neither all 0's nor all 1's, then x is a normalized floating point number with value

$$x = \pm(1.b_1 b_2 \dots b_{52})_2 \times 2^{(a_1 a_2 \dots a_{11})_2 - 1023} . \quad (27.2)$$

The "1023 bias" in the exponent supports the graceful inclusion of various "unnormalized" floating numbers which we describe shortly. Several important quantities capture

the finiteness of the representation. The *machine epsilon* is the gap between 1 and the next largest floating point number. Its value is $2^{-52} \cdot 10^{-5}$ for the double format. Among the positive normalized floating point numbers, $N_{\min} = 2^{-102} \cdot 10^{-38}$ is the smallest and $N_{\max} = (2 - 2^{-53}) \cdot 10^{38}$ is the largest. A real number x is within the *normalized range* if $N_{\min} \leq |x| \leq N_{\max}$.

If $a_1 \dots a_{11}$ is all 0's, then the value of the representation (27.1) is

$$x = \pm(0.b_1 b_2 \dots b_{52})_2 \times 2^{(a_1 a_2 \dots a_{11})_2 - 1022} \quad (27.3)$$

This includes 0 and the *subnormal* floating point numbers. This feature creates a uniform spacing of the floating point numbers between $-N_{\min}$ and $+N_{\min}$.

If $a_1 \dots a_{11}$ is all 1's, then the encoding (27.1) represents \inf for $+0$, $-\inf$ for -0 , or NaN for "not-a-number." The determining factor is the value of the b_i . (If the b_i are not all zero, then the value of x is NaN .) Quotients like $1/0$, $-1/0$ and $0/0$ produce these special floating point numbers instead of prompting program termination.

There are four rounding modes: *round down* (toward -0), *round up* (toward $+0$), *round-toward-zero*, and *round-toward-nearest*. We focus on *round-toward-nearest* since it is the mode almost always used in practice.

If a real number x is outside the range of the normalized floating point numbers then

$$\text{round}(x) = \begin{cases} -0 & \text{if } x < -N_{\max}, \\ +0 & \text{if } x > N_{\max}. \end{cases}$$

Otherwise, the rounding process depends upon its floating point "neighbors":

x_- is the nearest floating point number to x that is $< x$,

x_+ is the nearest floating point number to x that is $> x$.

Defined $d_- = x - x_-$ and $d_+ = x_+ - x$ and let "lsb" stand for "least significant bit." If $N_{\min} \leq |x| \leq N_{\max}$ then

$$\text{round}(x) = \begin{cases} x_- & \text{if } d_- < d_+ \text{ or } d_- = d_+ \text{ and } \text{lsb}(x_-) = 0 \\ x_+ & \text{if } d_+ < d_- \text{ or } d_+ = d_- \text{ and } \text{lsb}(x_+) = 0 \end{cases}$$

The tie breaking criteria is well-defined because x_- and x_+ are adjacent floating point numbers and so must differ in their least significant bit.

Regarding the accuracy of the round-to-nearest strategy, suppose x is a real number that satisfies $N_{\min} \leq |x| \leq N_{\max}$. Thus,

$$|\text{round}(x) - x| \leq \frac{2^{-52}}{2} \cdot 2^{-e} = \frac{2^{-52}}{2} \cdot |x|$$

which says that relative error is bounded by half of the machine epsilon

$$\frac{|\text{round}(x) - x|}{|x|} \leq 2^{-52}.$$

The IEEE standard stipulates that each arithmetic operation be *correctly rounded*, meaning that the computed result is the rounded version of the exact result. The implementation of correct rounding is far from trivial and requires registers that are equipped with several extra bits of precision.

We mention that the IEEE standard also requires correct rounding in the square root operation, the remainder operation, and various format conversion operations.

=mS1A 8xA e52AI-}G:A

With intuition gleaned from the toy calculator example and an understanding of IEEE arithmetic, we are ready to move on to the roundoff analysis of some basic algebraic calculations. The challenge when presenting the effects of finite precision arithmetic in this section and throughout the book is to communicate essential behavior without excessive detail. To that end we use the notation $\text{fl}(\cdot)$ to identify a floating point storage and/or computation. Unless exceptions are a critical part of the picture, we freely invoke the fl notation without mentioning "-o," "o," "NaN," etc.

If $x \in \mathbb{I}$, then $\text{fl}(x)$ is its floating point representation and we assume that

$$\text{fl}(x) = x(1 + \delta), \quad |\delta| \leq u, \quad (27.4)$$

where u is the *unit roundoff* defined by

$$u = \frac{1}{2} \times (\text{gap between } 1 \text{ and next largest floating point number}). \quad (27.5)$$

The unit roundoff for IEEE single format is about 10^{-7} and for double format it is about 10^{-16} .

If x and y are floating point numbers and "op" is any of the four arithmetic operations, then $\text{fl}(x_{\text{op}} y)$ is the floating point result from the floating point op. Following Trefethen and Bau (NLA), the fundamental axiom of floating point arithmetic is that

$$\text{fl}(x_{\text{op}} y) = (x_{\text{op}} y)(1 + \delta), \quad |\delta| \leq u, \quad (27.6)$$

where x and y are floating point numbers and the "op" inside the fl operation means "floating point operation." This shows that there is small relative error associated with individual arithmetic operations.

$$\frac{\text{fl}(x_{\text{op}} y) - (x_{\text{op}} y)I}{|x_{\text{op}} y|} \leq u, \quad x_{\text{op}} y \neq 0$$

Again, unless it is particularly relevant to the discussion, it will be our habit not to bring up the possibilities of an exception arising during the floating point operation.

I nHib usfksb tb dPktf Tl"b 2k Tlf buTld Sb

It is a good idea to have a healthy respect for the subtleties of floating point calculation. So before we proceed with our first serious roundoff error analysis we offer three maxims to keep in mind when designing a practical matrix computation. Each reinforces the distinction between computer arithmetic and exact arithmetic.

Maxim 1. Order is Important.

Floating point arithmetic is not associative. For example, suppose

$$x = 1.24 \times 10^0, \quad y = -1.23 \times 10^0, \quad n = 100 \times 10^3,$$

Using toy calculator arithmetic we have

$$\text{fl}(\text{fl}(x + y) + z) = 1.10 \times 10^2$$

while

$$f(x + f(y + n)) = 100 \times 10^2$$

(a consequence of this is that mathematically equivalent algorithms may produce different results in floating point.

Ma im 2. Lar er May Mean Smaller.

Suppose we want to compute the derivative of $f(x) = \sin(x)$ using a divided difference. Calculus tells us that $d = (\sin(x+h) - \sin(x))/h$ satisfies $|d - \cos(x)| = O(h)$, which argues for making h as small as possible. On the other hand, any roundoff error sustained in the sine evaluations is magnified by $1/h$. By setting $h = y^{-1}$ the sum of the calculus error and roundoff error is approximately minimized. In other words, a value of h much greater than y^{-1} renders a much smaller overall error. See Overton (2001, pp. 70-72).

Ma im 3. A Math Book Is Not Enough

The explicit coding of a textbook formula is not always the best way to design an effective computation. As an example we consider the quadratic equation $x^2 - 2px - q = 0$, where both p and q are positive. Here are two methods for computing the smaller (necessarily real) root:

Method 1: $r_{\min} = p - \sqrt{p^2 + q}$,

$$\text{Method 2} \quad r_{\min} = \frac{q}{p + \sqrt{p^2 + q}}.$$

The first method is based on the familiar quadratic formula while the second uses the fact that $-q$ is the product of min and the larger root. Using IEEE double floating point arithmetic with input $p = 12345678$ and $q = 1$ we obtain these results:

Method 1: $r_{\min} = -409781932308106 \times 10^{-8}$

Method 2 $r_{\min} = -40500003210021 \times 10^{-8}$ (correct).

Method 1 produces an answer that has almost no correct significant digits. It attempts to compute a small number by subtracting a pair of nearly equal large numbers. Almost all correct significant digits in the input data are lost during the subtraction, a phenomenon known as catastrophic cancellation. In contrast, Method 2 produces an answer that is correct to full machine precision. It computes a small number as a division of one number by a much larger number. See Fausset (1970).

Keeping these maxims in mind does not guarantee the production of accurate, reliable software but it helps.

, **ibib** eE3· Tftf **Tlb** uf kxT'b tb **Wrbht** f xTeb

Suppose $A \in \mathbb{M}^{n \times n}$ and that we wish to quantify the errors associated with its floating-point representation. Denoting the stored version of A by $\text{fl}(A)$, we see that

$$[\mathbf{f}](A)_{ij} = \mathbf{f}(a_{ij}) = a_{ij}(1 + \epsilon_{ij}), \quad |\epsilon_{ij}| \leq \mathbf{u}, \quad (27.7)$$

for all i and j , i.e,

$$\|f(A) - A\| \leq u \|A\|.$$

A relation such as this can be easily turned into a norm inequality, e.g.,

$$\|f(A) - A\| \leq u \|A\|$$

However, when quantifying the rounding errors in a matrix manipulation, the absolute value notation is sometimes more informative because it provides a comment on each entry.

~~xi0mb 9Ult kbt TthpkQb x ktUa Qeb~~

We begin our study of finite precision matrix computations by considering the rounding errors that result in the standard dot product algorithm

$$\begin{aligned} s &= 0 \\ \text{for } k &= 1:n \\ s &= s + x_k y_k \\ \text{end} \end{aligned} \tag{2.7.8}$$

Here x and y are n -by-1 floating point vectors.

In trying to quantify the rounding errors in this algorithm, we are immediately confronted with a notational problem: the distinction between computed and exact quantities. If the underlying computations are clear, we shall use the $f()$ operator to signify computed quantities. Thus, $f(x'y)$ denotes the computed output of (2.7.8). Let us bound $|f(x'y) - x'y|$. If

$$s_p = f\left(\sum_{k=1}^n x_k y_k\right),$$

then $s_p = X'Y(1+8)$ with $\|X'\| \leq u$ and $r_p = 2n$.

$$\begin{aligned} s_p &= f(s_p + f(x'y)) \\ &= (s_p + X'Y(1+8))(1+f_p) \quad |\delta_p|, |\epsilon_p| \leq u. \end{aligned} \tag{2.7.9}$$

A little algebra shows that

$$f(x'y) = s_p = \sum_{k=1}^n x_k y_k (1+f'_k)$$

where

$$(1+f'_k) = (1+8) \sum_{j=k}^n (1+f_j)$$

with the convention that $f_1 = 0$. Thus

$$|f(x'y) - x'y| \leq \sum_{k=1}^n \|x_k y_k\| |f'_k| \tag{2.7.10}$$

proceed further, we must bound the quantities $\|L_k\|_1$ in terms of u . The following result is useful for this purpose.

Lemma 2.7.1. If $(1 + \alpha) = \sum_{k=1}^n (1 + Q_k)$ where $\|Q_k\|_1 \leq u$ and $\|u\|_1 \leq .01$, then $\|L_k\|_1 \leq 1.01u$.

Proof See Higham (ASNA, p. 75). \square

Application of this result to (27.10) under the "reasonable" assumption $\|u\|_1 \leq .01$ gives

$$\|f(x^T y) - x^T y\| \leq 1.01u \|x\|^T \|y\| \quad (27.11)$$

Notice that if $\|x^T y\| < \|x\|\|y\|$, then the relative error in $f(x^T y)$ may not be small.

More precisely, $\|f(x^T y) - x^T y\| \leq 1.01u \|x\|^T \|y\|$.

An easier but less rigorous way of bounding $\|f(x^T y) - x^T y\|$ in Lemma 27.1 is to say $\|L_k\|_1 \leq nu + O(u^2)$. With this convention we have

$$|f(x^T y) - x^T y| \leq nu \|x\|^T |y| + O(u^2). \quad (27.12)$$

Other ways of expressing the same result include

$$\|f(x^T y) - x^T y\| \leq c n u \|x\|^T |y| \quad (27.13)$$

and

$$|f(x^T y) - x^T y| \leq c n u \|x\|^T |y|, \quad (27.14)$$

where $c(n)$ is a "modest" function of n and c is a constant of order unity.

We shall not express a preference for any of the error bounding styles shown in (27.11)-(27.14). This spares us the necessity of translating the roundoff results that appear in the literature into a fixed format. Moreover, paying overly close attention to the details of an error bound is inconsistent with the "philosophy" of roundoff analysis. As Wilkinson (1971, p. 56) says,

There is still a tendency to attach too much importance to the precise error bounds obtained by an a priori error analysis. In my opinion, the bound itself is usually the least important part of it. The main object of such an analysis is to expose the potential instabilities, if any, of an algorithm so that hopefully from the insight thus obtained one might be led to improved algorithms. Usually the bound itself is weaker than it might have been because of the necessity of restricting the mass of detail to a reasonable level and because of the limitations imposed by expressing the errors in terms of matrix norms. A priori bounds are not, in general, quantities that should be used in practice. Practical error bounds should usually be determined by some form of a posteriori error analysis, since this takes full advantage of the statistical distribution of rounding errors and of any special features such as sparseness in the matrix.

It is important to keep these perspectives in mind.

$$= \text{fl}(A)RN : XrA - mAS - \text{fl}(A)I + G(A)I = \text{fl}(A)I - \text{fl}(A)I = N - m + A$$

It is easy to show that if A and B are floating point matrices and a is a floating point number, then

$$\text{fl}(aA) = aA + E, \quad \|E\| \leq u\|A\|_1 \quad (27.15)$$

and

$$\text{fl}(A+B) = (A+B) + E, \quad \|E\| \leq u\|A+B\|_1 \quad (27.16)$$

As a consequence of these two results, it is easy to verify that computed saxpy's and outer product updates satisfy

$$\text{fl}(y+ax) = y + ax + z, \quad \|z\| \leq u(M+2ax) + O(u^2), \quad (27.17)$$

$$\text{fl}(C+uv^T) = C + uv^T + E, \quad \|E\| \leq u(CI + 2uv^T) + O(u^2). \quad (27.18)$$

Using (27.11) it is easy to show that add-product-based multiplication of two floating point matrices A and B satisfies

$$\text{fl}(AB) = AB + E, \quad \|E\| \leq n\|A\|\|B\|_1 + O(u^2). \quad (27.19)$$

The same result applies if a *saxpy* or outer product based procedure is used. Notice that matrix multiplication does not necessarily give small relative error since $\|AB\|_1$ may be much smaller than $\|A\|\|B\|_1$, e.g.

$$\begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ .99 & 0 \end{bmatrix} = \begin{bmatrix} .01 & 0 \\ 0 & 0 \end{bmatrix}$$

It is easy to obtain bounds from the roundoff results developed thus far. If we look at the 1-norm error in floating point matrix multiplication, then it is easy to show from (27.19) that

$$\|\text{fl}(AB) - AB\|_1 \leq n\|A\|_1\|B\|_1 + O(u^2). \quad (27.20)$$

1. m0nib d&j txnb thb utf dj txnb AxkxkxkxPK eseb

Each roundoff bound given above is the consequence of a *forward error analysis*. An alternative style of characterizing the roundoff errors in an algorithm is accomplished through a technique known as *backward error analysis*. Here, the rounding errors are related to the input data rather than the answer. By way of illustration, consider the $n=2$ version of triangular matrix multiplication. It can be shown that:

$$\text{fl}(AB) = \begin{bmatrix} a_{11}(1+f_1) & (a_{11}f_1 + f_2 + a_{12}f_3)(1+f_4) \\ 0 & a_{22}(1+E) \end{bmatrix}$$

where $\|E\| \leq u$ for $i = 1:5$. However, if we define

$$\hat{A} = \begin{bmatrix} a_{11} & a_{12}(1+\epsilon_3)(1+\epsilon_4) \\ 0 & a_{22}(1+\epsilon_5) \end{bmatrix}$$

and

$$\hat{B} = \begin{bmatrix} b_1(I + E) & b_{12}(I + E_2)(I + E_3) \\ 0 & b_{22} \end{bmatrix},$$

then it is easily verified that $f_l(AB) = AB$. Moreover,

$$A = A + E, \quad \|E\| \leq 2\|A\| + O(u^2),$$

$$F = B + F, \quad \|F\| \leq 2\|B\| + O(u^2).$$

which shows that the computed product is the exact product of slightly perturbed A and B.

In §1.3.11 we outlined a recursive matrix multiplication procedure due to Strassen. It is instructive to compare the effect of roundoff in this method with the effect of roundoff in any of the conventional matrix multiplication methods of §1.1.

It can be shown that the Strassen approach (Algorithm 1.3.1) produces a $C = f_l(AB)$ that satisfies an inequality of the form (27.20). This is perfectly satisfactory in many applications. However, the C that Strassen's method produces does not always satisfy an inequality of the form (27.19). To see this, suppose that

$$A = B = \begin{bmatrix} .99 & .0010 \\ .0010 & .99 \end{bmatrix}$$

and that we execute Algorithm 1.3.1 using 2-digit floating-point arithmetic. Among other things, the following quantities are computed:

$$F_3 = f_l(.99(.001 - .99)) = -.98$$

$$A = RC_p \quad sm$$

This follows from (27.19). Because we cannot say the same for the Sherman approach, we conclude that Algorithm 1.3.1 is not attractive for *certain* nonnegative matrix multiplication problems if relatively accurate \hat{G} are required.

Extrapolating from this discussion we reach two fairly obvious but important conclusions:

- Different methods for computing the same quantity can produce substantially different results.
- Whether or not an algorithm produces satisfactory results depends upon the type of problems solved and the goals of the user.

These observations are clarified in subsequent chapters and are intimately related to the concepts of algorithm stability and problem condition. See §34.10.

~~1 mltb elt PKe1 btlb ftStPbbn Dtf 1lb ukPf \$xb~~

A nice way to conclude this chapter and to anticipate the next is to analyze the quality of a "make-believe" $\mathbf{Ax} = \mathbf{b}$ solution process in which all floating point operations are performed exactly *except* the storage of the matrix \mathbf{A} and the right-hand side \mathbf{b} . It follows that the computed solution \hat{x} satisfies

$$(\mathbf{A} + E)\hat{x} = (\mathbf{b} + e), \quad \|E\|_1 \leq u\|A\|_1, \quad \|e\|_1 \leq u\|b\|_1. \quad (27.21)$$

where

$$\mathbf{f}(b) = b + e, \quad \mathbf{f}(A) = A + E.$$

If $u(A) > 1$ (say), then by Theorem 26.2 it can be shown that

$$\frac{\|\mathbf{x} - \hat{\mathbf{x}}\|_1}{\|\mathbf{x}\|_1} \leq 4uK_A(A). \quad (27.22)$$

The bounds (27.21) and (27.22) are "best possible" norm bounds. No general norm error analysis of a linear equation solver that requires the storage of \mathbf{A} and \mathbf{b} can render sharper bounds. As a consequence, we cannot justifiably criticize an algorithm for returning an inaccurate \hat{x} if \mathbf{A} is ill-conditioned relative to the unit roundoff, e.g., $u(A) > 1$. On the other hand, we have every "right" to pursue the development of a linear equation solver that renders the exact solution to a nearby problem in the style of (27.21).

Problems

P2.7.1 **3.86** 13.1 ~~f01....5~~ **6.13** $y = \mathbf{x}, k/A$ $\mathbf{f}(xTx) = xTx(1 + a) \leq Ax\|a\|_1 \leq nu + O(u^2)$.

P2.7.2 **3.8.8** ~~f00rol..2.5~~ **13.1** $f(\mathbf{x})k/A)A2A=\mathbf{f}(2k)1 1(k)=T,AEk(x \in \mathbb{R})$.

P2.7.3 **3.86** 13.1 $E \in \mathbb{R}^{m \times n}$ $5(02_m : n, ve0\#|E|_2 \leq \sqrt{n}\|E\|_2, 2((-n)(1.0^n E, 52^n)^n - ((-1. n) \cap (n 1.(n .0)(.n .n)) \hat{L})$

P2.7.4 **1...28** 13.88b..18.88 $\geq r...8$ **8.81** > 1.8 , \dots \dots $\mathbf{f}(J) = J(1 + \epsilon) \leq u$. $((.n a a ny = e\mathbf{f}(x) \mathbf{f}(p) \mathbf{f}(y))^{.n.})^{.n.} 02^n n.((-n \leq -n) \hat{L})$

P2.7.5 **....8.8** A ApS , A n -by- n \dots 0 $((- . \mathbf{f}(0(-n)(01(0 \leq (+n)(- E = \mathbf{f}(A,) T = a(T1-kAx = T)A (ok/Aa(02A)kT)2w 5A)2wTEk/T = R(A-Tk) = (L k/2k = \hat{A}B_{F\infty}^n, n\hat{A}, , -n \geq -n k \hat{A} ApS,$

P2.7.7 3.8.8 f0 dro 5

P2.7.8 fe138...ePPB. N82. 1663.1 fel38...8.1 .868. 8≥ 03.1.. 88-8.8.18. 8b..1 ..
 3.1. 138 ...8.118.8. 13.1... 88-8.8.18. 8b..1..

P2.7.9 $\text{feg}_k = \text{feg}_c fC_j(j) \text{feg}_j(j)$

P2.7.10 38s...8. 138....1.. 8r..1.8.

$$q(\lambda) = \det \begin{pmatrix} w - \lambda & x \\ x & z - \lambda \end{pmatrix}.$$

**3.....1.. 3I 168.8. .881.r1 ApSr2. eaar8t CiACqr1 - z| ≤ |r2 - zl.) i2A 2 2vtxlk/T k/2k
 'a 1HA CM HFlmpt JetAmamMpra**

Notes and References for 27

*it,1HM2mpfmwA et,etpAt&et pMCACIt Cit ,MACmpMmp&pm)iCtAC Cttte ..H ymlaMp
313S2Ct2AeC-D4. 3S2mpA8 -e2,e 3SApStt8tH -e,Ie 33xMe im)ielItH JteaJtACmtAa
8rixmPS0*

*B 2mAHypmaME.7. L Rounding Errors in Algebraic Processes, NetPc.A2AH18p)lt2MMS1.t aSp3
 ? 3xMea.Cit.nEl. L 4NmCH1mpM8JrCAC.M#Meih A4ACi fMMMyqMCxpMr)i SLAmer. Math.
 Monthly 77, E7nME,L*

*Bra2ra mlymELM p 44MSte1 zeeMepAHbmaSIAM Review 13 r 6yf uy1
kDNM>MgmnLLlRMI enea ylid cc,a 4IM2a2MB2,a.MiM2aFA>2aBSIAM Review*

R Bea OvO=1ea=Ona=eD12 pictures on Finite Precision Computations, 2De4 Nr
 HmAnCmMhAS\$H.limM69

itStam)pM. JeMSrACM02Aetk 8ACemRM8JrCAC.Mpr.eta ASrCAMHtS rpSteaCApSm)mcT yetAramAmPnCiSm430

B *matrix8t1.nEg-L4,pSte(M2ApSCitotlmAc.HmME8temAAH2M92AetSLSIAM J. Sci. Stat. Comput. 5, gg.MEE*

*JBrASh.nEgg.c4eIClo D.2.1 .., 1e.2e00 e 2rc eMrCm@MhpArixmAAHtote8mpt 4AAimpt
NAeA8tCeasSLACM Trans. Math. Softw. 14S717M.nc*

2,alma.aFIa+ and II|III|eneF.AMAl,aMn2al|eF,AM2,3r out|mlBA2e2MA2Ma|Fa
II>DmlII|III|Benn2,a eo21 le2MnB2,a eneF.AM2Aaf 1,

CICIAhenn41heLa, r [#ydt1R..uan2eF.>,e2Mln2,dRTa.2B5]>nmlTRIIIIAT4CM Trans.
Math. Softw. 4, nngM.-

Interval Methods and Applications of Interval Analysis, 2 De 4 N, c 1 m, C An M, N, S n H A J i M A S

NeI
Braunschweig University of Technology, D-3300 Braunschweig, FRG
and Institute für Materialphysik im Weltraum, D-3300 Braunschweig, FRG

it. 444444b M. MACGILLIVRAY 1G88 4G91 M=StGAw1tGv0

*Mtaor1J S.3 l)mCAApS29lmainmnlg.L 4eAAreACTr1MACmp)eNMn88ACmMpnAeCDtAmCi 1
cMmSpS)l SIAM J. Sci. Comput. 31 # 4 2009*

0d4v.c. 96l. m nnt C8(A) . 1m7ElOpmae 1Am)rhtP)nsG.m =)mPn IO OhiAt
InM8Anmt rvt)rh - r4en-. 5SIAM J. Sci. Comput. 31, iNt5MENb

lp i.,i,AMAB Ctp1Aa.isilpMOJ,m lt9apcB1t ,M.hipsjj opApL3m,paAzz,

5 b9pa.07tI?sb cRpplAi..AMAB4ilpMt,uAp ,OpMostMsM.,OMt OM,i,ip j.l u
AlAMtMM.iuM.liplm, Nma.OMQ7T. Math 16 M/rir ub
ub rMptIrs" c)luA,OitMl,s)luAs3M0.ma, uapM.,hQdMsMsM. 4, tIOi?D
ab15Mi91 i ttN" chOoM, M0. at9lmp,r,s OMMSi)lu3a. ritpM.aMt, c95as riOpM.
r.,tMAM.iOM 5SIAM J. Matrix Anal. Applic. 13, uyM31b
1b15.auua. i.mabl1sMiJiur1ttN5 chQdM,NB9d.r 4ilpMOJuSMOJ 1qaa' \$4h7T
ACM 7 ns. Math Sof w 18 NI / 5M1
9bouMGA., 1tlys cu1ApL.Mim RAQMu1OMi4.lpi.. 1Bhtp1ba,A4silpM09uMmer.
Math 79 / yr OMbA

cJa MAAPaaOamarApa.MAMl.paaM.abAMmapicQ0a.OMl.T lpa1Asa. , A.Aapi..pi0a
mltAplm8paA8s0A ASBiu,OMl. ohi..1oitem Mi paiMAopMA c08M.a 108MmaT
cl,tMi paApaAa.tiOMlatp 11Al.a0Ab c9a LapJaimli. oaOlsap,0AM, iM.hisilpMtJMB
hta.manApa.MAMham3Ml.s , Ba8pMOMA3hAAlpM.AMi,QAQJ1AMAAa,

5b(b9pa.07tIysb c4 lptpi. ,sMA,pa.MAMl. 4pM0fJuat 7ACM 7 ns. Math Sof w
4 r OI lb

5b(D9pa.07tIysb c4ilpMOJri N/(7 i .lptpi. rostMA,pa.MAM4pM080M,aiia 7ACM
T ns. Math Softw. 4, Ii3Mb

.bsD9iM.aMABc4silpMOJut, r.,OMpa.MAMAS,OMl.i.m R.acOMlIB 15c54a ,pl.
ipiuiA7ACM 7 ns. Math Sof w 19 Nyy itbE
9BM19ubauu as.Ds99M.aJwsap. lbsMm1b Ar i.m5MiJ., 7hb1M,i74bMiA,p7
rb)briptM.9516J1IAAl. 7cb .i7,m.D1911 7N?3619aAMU1A,huaQOMlmcAOM.i
LBR.ta.marim rM.ar(pa.MAM14h7ACM 7 ns. Math Sof w 28 (rN3N?r b
1b1b.aula. i.m lbsMm1N??sbc4.qptj ,.mR .Mat „liOM.i k1MlitMI.7SIAM J. Sci.
Comput. 25, iNiOMNb
rb idcsM4b9otOpM7i,pp,7 1"MoP7iib,,7o7 15i,il,7 (T,,A- .mar i.m h5d11.
N?15c4.hsapiOMliMaMl-)luA,titMlA 8M09Mar(pa.MAM4ilpM09uMComput. Phys.
Commun. 180 Nr Nu Nr EED

Chapter 3

General Linear Systems

nPg

nPl

nPn

nSo

nPr

nPs

The problem of solving a linear system $Ax = b$ is central to scientific computation. In this chapter we focus on the method of Gaussian elimination, the algorithm of choice if A is square, dense, and unstructured. Other methods are applicable if A does not fall into this category; see Chapter 4, Chapter 11, §12.1, and §12.2. Solution procedures for triangular systems are discussed first. These are followed by a derivation of Gaussian elimination that makes use of Gauss transformations. The process of eliminating unknowns from equations is described in terms of the factorization $A = LU$, where L is lower triangular and U is upper triangular. Unfortunately, the derived method behaves poorly on a nontrivial class of problems. An error analysis pinpoints the difficulty and sets the stage for a discussion of pivoting, a permutation strategy that keeps the numbers "nice" during the elimination. Practical issues associated with scaling, iterative improvement, and condition estimation are covered. A framework for computing the LU factorization in parallel is developed in the final section.

Reading Notes

Familiarity with Chapter 1, §§2.1–2.5, and §2.7 is assumed. The sections within this chapter depend upon each other as follows:

			§3.5
§3.1	-	§3.2	-
		§3.3	§3.4
			§3.6

Useful global references include Forsythe and Moler (SLAS), Stewart (MABD), Higham (ASNA), Watkins (FMC), Trefethen and Bau (NLA), Demmel (ANLA), and Ipsen (NMA).

3.1 Triangular Systems

Additional factorization methods for linear systems involve the conversion of the given square system to a triangular system that has the same solution. This section is about the solution of triangular systems.

mhhghbdkjtxrb uD: efTf Df Tkbt

Consider the following 2by-2 lower triangular system

$$\begin{bmatrix} 1 & 0 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}.$$

If $b_{12} \neq 0$ then the unknowns can be determined sequentially:

$$x_1 = b_1 / L_{11}$$

$$x_2 = (b_2 - L_{21}x_1) / L_{22}$$

This is the 2by-2 version of an algorithm known as *forward substitution*. The general procedure is obtained by solving the i th equation in $Lx = b$ for x_i :

$$x_i = \left(b_i - \sum_{j=1}^{i-1} \ell_{ij} x_j \right) / \ell_{ii}.$$

If this is evaluated for $i = 1:n$, then a complete specification of x is obtained. Note that at the i th stage the dot product of $L(i, 1:i-1)$ and $x(1:i-1)$ is required. Since b_i is involved only in the formula for x_i , the former may be overwritten by the latter.

As highlighted in the notes, if L is lower triangular and $b \in \mathbb{R}^n$, then this algorithm overwrites b with the solution to $Lx = b$. L is assumed to be nonsingular.

$$b(1) = b(1) / L(1, 1)$$

for $i = 2:n$

$$b(i) = (b(i) - L(i, 1:i-1) \cdot b(1:i-1)) / L(i, i)$$

end

This algorithm requires n^2 fops. Note that L is accessed by row. The computed solution \hat{x} can be shown to satisfy

$$(L + F)\hat{x} = b \quad \text{IFI is a multiple of } L + O(u^2). \quad (3.1)$$

For a proof, see Higham (ASNA, pp. 141–142). It says that the computed solution exactly satisfies a slightly perturbed system. Moreover, each entry in the perturbing matrix F is small relative to the corresponding element of L .

L07F

The analogous algorithm for an upper triangular system $Ux = b$ is called *back substitution*. The recipe for x_i is prescribed by

$$x_i = \left(b_i - \sum_{j=i+1}^n u_{ij}x_j \right) / u_{ii}$$

and once again b_i can be overwritten by x_i .

If $U \in \mathbb{R}^{n \times n}$ is upper triangular and $b \in \mathbb{R}^n$, then the following algorithm overwrites b with the solution to $Ux = b$. U is assumed to be nonsingular.

```

b(n) = b(n)/U(n,n)
for i = n-1:-1:1
    b(i) = (b(i) - U(i,i+1:n) · b(i+1:n))/U(i,i)
end

```

This algorithm requires n^2 flops and accesses U by row. The computed solution \hat{x} obtained by the algorithm can be shown to satisfy

$$(U + F)\hat{x} = b \quad \text{FIGURE 3.12}$$

mtvnb.kP Dls6A TSf \$nb-SAt kreb

Column-oriented versions of the above procedures can be obtained by reversing loop orders. To understand what this means from the algebraic point of view, consider forward substitution. Once x_1 is resolved, it can be removed from equations 2 through n , leaving us with the reduced system

$$L(2n, 2n)x(2n) = b(2n) - x(1) \cdot L(2n, 1).$$

We next compute x_2 and remove it from equations 3 through n , etc. Thus, if this approach is applied to

$$\begin{bmatrix} 2 & 0 & 0 \\ 1 & 5 & 0 \\ 7 & 9 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 2 \\ 5 \end{bmatrix},$$

we find $x_1 = 3$ and then deal with the 2 by 2 system

$$\begin{bmatrix} 5 & 0 \\ 9 & 8 \end{bmatrix} \begin{bmatrix} x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \end{bmatrix} - 3 \begin{bmatrix} 1 \\ 7 \end{bmatrix} = \begin{bmatrix} -1 \\ -16 \end{bmatrix}.$$

Here is the complete procedure with overwriting

if As5i- nWyoneI3HA ssc Ae.n s1sCm rH0e5HeIAa- If the matrix $L \in \mathbb{R}^{n \times n}$ is lower triangular and $b \in \mathbb{R}^n$, then this algorithm overwrites b with the solution to $Lx = b$. L is assumed to be nonsingular.

```

f rj = 1:n- 1
  b(j) = b(j)/L(j,j)
  b(j + 1:n) = b(j + 1:n) - b(j).L(j + 1:n,j)
end
b(n) = b(n)/L(n,n)

```

It is also possible to obtain a column-oriented `saxpy` procedure for back substitution

mis a5i- noyimel IduiAd ct hiA5S_ lnwb_ ru02 5a55hAo If $U \in \mathbb{R}^{n \times n}$ is upper triangular and $b \in \mathbb{R}^n$, then this algorithm overwrites b with the solution to $Ux = b$. U is assumed to be nonsingular.

```

f rj = n - 12
  b(j) = b(j)/U(j,j)
  b(l:j - 1) = b(l:j - 1) - b(j).U(l:j - 1,j)
end
b(l) = b(l)/U(l,l)

```

Note that the dominant operation in both Algorithms 3.13 and 3.14 is the `saxpy` operation. The roundoff behavior of these implementations is essentially the same as for the dot product versions.

mmt�b sDP f bEPSWbuf i Itltb ultSb

Consider the problem of computing a solution $X \in \mathbb{R}^{n \times q}$ to $LX = B$ where $L \in \mathbb{R}^{n \times n}$ is lower triangular and $B \in \mathbb{R}^{n \times q}$. This is the *multiple-right-hand-side* problem and it amounts to solving q separate triangular systems, i.e., $LX(:,j) = B(:,j)$, $j = 1:q$. Interestingly, the computation can be blocked in such a way that the resulting algorithm is rich in matrix multiplication, assuming that q and n are large enough. This turns out to be important in subsequent sections where various block factorization schemes are discussed.

It is sufficient to consider just the lower triangular case as the derivation of block back substitution is entirely analogous. We start by partitioning the equation $LX = B$ as follows:

$$\begin{bmatrix} \mathbf{f} & L_{12} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_N & L_{N2} & \cdots & L_{NN} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_N \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_N \end{bmatrix}. \quad (3.13)$$

Assume that the diagonal blocks are square. Paralleling the development of Algorithm 3.13 we solve the system $L_1 X_1 = B_1$ for X_1 and then remove X_1 from block equations 2 through N :

$$\begin{bmatrix} L_{22} & 0 & \cdots & 0 \\ L_{32} & L_{33} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{N2} & L_{N3} & \cdots & L_{NN} \end{bmatrix} \begin{bmatrix} X_2 \\ X_3 \\ \vdots \\ X_N \end{bmatrix} = \begin{bmatrix} B_2 \\ B_3 \\ \vdots \\ B_N \end{bmatrix} - \begin{bmatrix} L_{21} \\ L_{31} \\ \vdots \\ L_{N1} \end{bmatrix} X_1.$$

Continuing in this way we obtain the following block forward elimination scheme

```

for j = 1:N
    Sdve L1jX1 = Bi
    for i = j + 1:N
        Bi = Bi - LiiXi
    end
end

```

(314)

Notice that the i-loop oversees a single block `saxpy` update of the form

$$\begin{bmatrix} B_{i+1} \\ \vdots \\ B_N \end{bmatrix} = \begin{bmatrix} B_{1+1} \\ \vdots \\ B_N \end{bmatrix} - \begin{bmatrix} L_1 & 0 \\ \vdots & \ddots \\ 0 & L_N \end{bmatrix} X_i.$$

To realize level-3 performance, the submatrices in (313) must be sufficiently large in dimension

`f lmmib ,sB lkSf $ Pmblkxtf f Tkdb`

It is handy to adopt a measure that quantifies the amount of matrix multiplication in a given algorithm. To this end we define the level-3 fraction of an algorithm to be the fraction of fops that occur in the context of matrix multiplication. We call such fops level-3fops.

Let us determine the level-3 fraction for (314) with the simplifying assumption that $n = rN$. (The same conclusions hold with the unequal blocking described above.) Because there are N applications of r -by- r forward elimination (the level-2 portion of the computation) and n^2 fops overall, the level-3 fraction is approximately given by

$$1 - \frac{Nr^2}{n^2} = 1 - \frac{1}{N}.$$

Thus, for large N almost all fops are level-3 fops. It makes sense to choose N as large as possible subject to the constraint that the underlying architecture can achieve a high level of performance when processing block `saxpys` that have width $r = n/N$ or greater.

`f mtpnb kk le nDtxSb ,x Ttl'DPtxbuKef $b ukPf Tl'b`

The problem of solving nonsquare, m -by- n triangular systems deserves some attention. Consider the lower triangular case when $m \leq n$, i.e.,

$$\begin{bmatrix} L_{11} & & \\ L_{21} & L_{22} & \\ & \vdots & L_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \quad L_{11} \in \mathbb{R}^{m \times m}, \quad b_1 \in \mathbb{R}^m, \\ L_{21} \in \mathbb{R}^{(m-1) \times m}, \quad b_2 \in \mathbb{R}^{m-n}.$$

Assume that L_{11} is lower triangular and nonsingular. If we apply forward elimination to $L_{11}x = b_1$, then x solves the system provided $L_{21}(L_{11}^{-1}b_1) = b_2$. Otherwise, there is no solution to the overall system. In such a case least squares minimization may be appropriate. See Chapter 5.

Now consider the lower triangular system $Lx = b$ when the number of columns n exceeds the number of rows m . We can apply forward substitution to the square

system $L(l:m, l:n) \times (l:m, l:n)$ = band prescribe an arbitrary value for $x(m+1:n)$. See §5.6 for additional comments on systems that have more unknowns than equations. The handling of nonsquare upper triangular systems is similar. Details are left to the reader.

mtmfb ,esb ef" § xtbklb Cxtt l"DPtxb st f xsfSb

A unit triangular matrix is a triangular matrix with 1's on the diagonal. Many of the triangular matrix computations that follow have this added bit of structure. It clearly poses no difficulty in the above procedures.

For future reference we list a few properties about products and inverses of triangular and unit triangular matrices.

- The inverse of an upper (lower) triangular matrix is upper (lower) triangular.
- The product of two upper (lower) triangular matrices is upper (lower) triangular.
- The inverse of a unit upper (lower) triangular matrix is unit upper (lower) triangular.
- The product of two unit upper (lower) triangular matrices is unit upper (lower) triangular.

Problems

$$\text{P3.1.1 } N. 8 \dots 8. fe13 N .8 .21fe, .8. 8. ERn = aCk/2k Uz - g 3e0 = UERnxn T = 1AE kET2MELik/ U_m = ga = 0 \dots U_h 1_n 1 \neq 11$$

$$\text{P3.1.2 } .2..8 .8 L = I_n - N i = ik w(LAE T2) E LCAEN ERnxn + C(L - kC2k$$

$$L^{-1} = I_n + N + N^2 + \dots + N^{n-1}$$

$$1/2k T=k/A2WA (o || L^{-1})_F \quad N_{ij} = 1 \ln 2 \neq > ..$$

$$\text{P3.1.3 } f..18 . .81. 8. 8...881(314). i()k k=cA kC2kN xi. ixA=f.$$

$$\text{P3.1.4 } 3.8 .8 . 138 . 821....2.. . 1.fe.83.1.8 ...18. feS317.$$

$$\text{P3.1.5 } .2..8 .8 6 ERnxn EA, 1AE kET2E 2x k/2k 6f1)I)x = bi=2)(=T)1AE = 1kAc4)TA 2 Gx2(Ek/c In(a(1k)1 x 1(A k/2k k/AiWtaik fc 240r f b) I EA ,TEA= Zf N(1- II)k7 +, 1(A$$

$$r_+ = \begin{bmatrix} u^T \\ S_c \end{bmatrix}, \quad :+ = \begin{bmatrix} & \\ & \end{bmatrix}, \quad b_+ = \begin{bmatrix} & \\ & \end{bmatrix},$$

$$=k : = S(k1)Mkb \neq T(k1)AKR-1n), b_+ = b(k-)MKBx_u, T, \quad ER. 2^n 5 02-0 \\ 5^n + \binom{n}{n} \frac{M}{M} \leq a/k/2k \quad (S_c T_c - \lambda I)x_c = b_c$$

$$2x We= Texei=22 =2-AKA)$$

$$x_+ = \begin{bmatrix} \gamma \\ x_c \end{bmatrix}, \quad " = \frac{ubTXe- UTWe}{UT},$$

$$=(-A\otimes f -)I)x_+ = b_+, N=AkA k/2k 2x w_+ = T+x_+ AIC EA ,iEA 6f1 k, N1=A$$

$$\text{P3.1.6 } .2..8.8 138 .1 ..8_1, \dots, R_p, ERnxn 2A2w1AE kET2MELik/TA 2 6 Fx2(Eka 1E = (-T)1k/A= 1kATR1 \dots \dots)I)x = b(k-cT)1 kC2kCAT2Ei (o a(AhaA)k-T)(=i)k= 4 IT)k)A)AE2-TWA=(-kT)k k/A 1A2T(- 1, wAT$$

$$\text{P3.1.7 } .2..8.8 L, K ER nxn EA=(LAkT2)M2 2x B ERnxn)iA 2 2(EIk/T 1E a(c 1 kTf X ERnxn=(k/2k LXK = B.$$

Notes and References for

1.2.8.2P8.8.P84P8 P.....P8.8≥P8.....4.8.88.fo88

N.J. 30y(x M022 DpJ0EE.=)Ee0= Apr00(n y.r)Oeor OX SIAM J. Numer. Anal. 26
MNVM6

d! MnAOauAB Qjpu ($T_p \cdots T_1 > I$)x = b 2utlt taAut_i, _{1,-E.0≥-;}, _{-≤E; n,n,E}

c.D. A)= r0= Oe.R=)nS=)n+ 22+DX= = OpeYr.r)= Oeor OX or DJ0a SIAM J. Matr
Anal. Applic. 24, N[5Mj

, pMolloiMnNmb(pn²) -≤(n+n)1≤(n+p)ⁿⁿ(4ⁿpⁿ9. n, 4. -p≤(-p≤(9p(-9(+, pⁿ)(
i = A= E)E Oo AAr G1- o.nor Oe, r= m OOKOp).

A O.+ e= f=)Ar00)n y(x= Oexr > r0Eyn O)nOr Oe)nOByeO> On -

N.J. 30y(x ,02. DOr)nOr Oe)r10A yAp A0eor O8= r > OSIAM J1 Sci. Comput. 16
6 6MEL

3.2 The LU Factorization

Triangular systems solving is an easy $O(n^2)$ computation. The idea behind Gaussian elimination is to convert a given system $Ax = b$ to an equivalent triangular system. The conversion is achieved by taking appropriate linear combinations of the equations. For example, in the system

$$\begin{aligned} 3x_1 + 5x_2 &= 9 \\ 6x_1 + 7x_2 &= 4 \end{aligned}$$

if we multiply the first equation by 2 and subtract it from the second we obtain

$$\begin{aligned} 3x_1 + 5x_2 &= 9 \\ -3x_1 - 5x_2 &= -14 \end{aligned}$$

This is $n = 2$ Gaussian elimination. Our objective in this section is to describe the procedure in the language of matrix factorizations. This means showing that the algorithm computes a unit lower triangular matrix L and an upper triangular matrix U so that $A = LU$, eg,

$$\left[\quad \right] = \left[\begin{matrix} & \\ & \end{matrix} \right] \left[\begin{matrix} & \\ & \end{matrix} \right].$$

The solution to the original $Ax = b$ problem is then found by a two-step triangular solve process

$$Ly = b \quad Ux = y \quad \Rightarrow \quad Ax = LUx = Ly = b \quad (32.1)$$

The LU factorization is a "high level" algebraic description of Gaussian elimination. This equation solving is not about the matrix vector product $A^{-1}b$ but about computing LU and using it effectively, see §34.9. Expressing the outcome of a matrix algorithm in the "language" of matrix factorizations is a productive exercise, one that is repeated many times throughout this book. It facilitates generalization and highlights connections between algorithms that can appear very different at the scalar level.

1AA1A H₁ N+A87 4X7 }-m+A

To obtain a factorization description of Gaussian elimination, it is traditionally presented, we need a matrix description of the zeroing process. At the n = 2 level, if $v_1 \neq 0$ and $T = v_2/v_1$, then

$$\begin{bmatrix} - & 0 \\ - & 1 \end{bmatrix} \begin{bmatrix} \quad \end{bmatrix} = \begin{bmatrix} \quad \\ \ddots \end{bmatrix}.$$

More generally, suppose $v \in \mathbb{R}^n$ with $V_k \neq 0$. If

$$T = [0, \dots, 0]_{k \times k} b, \dots, T_n \quad T = \frac{V}{V_k} \quad i = k+1:n$$

and we define

$$M_k = I_n - r e_r, \quad (322)$$

then

$$M_k v = \begin{bmatrix} a & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & & 0 & 0 & & V_k \\ 0 & & -T_k & 1 & & V_k \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & -T_n & 0 & \dots & V_n \end{bmatrix} \begin{bmatrix} V \\ V_k \\ V_k \\ \vdots \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} V \\ V_k \\ V_k \\ \vdots \\ \vdots \\ 0 \end{bmatrix}.$$

A matrix of the form $M_k = I_n - r e_r$ ref $\mathbb{R}^{n \times n}$ is a *Gauss transformation* if the first k components of $T \in \mathbb{R}^{n \times n}$ are zero. Such a matrix is unit lower triangular. The components of $r(k+1:n)$ are called *multipliers*. The vector r is called the *Gauss vector*.

$$gtxtx X \quad .++- A \cdot [1:X \quad -1] \sum \sum d_k \quad [\sum g \quad 4]_1 fo \star \times U \sum X$$

Multiplication by a Gauss transformation is particularly simple. If $C \in \mathbb{R}^{n \times n}$ and $M_k = I_n - r e_r$ is a Gauss transformation, then

$$M_k C = (I_n - r e_r^T) C = C - r (e_r^T C) = C - r C(k, :)$$

is an outer product update. Since $r(1:k) = 0$ only $C(k+1:n, :)$ is affected and the update $C = I_n C$ can be computed row by row as follows:

```
f r i = k+1:n
    C(i, :) = C(i, :) - T.C(k, :)
end
```

This computation requires $2(n-k)r$ fops. Here is an example

$$C = \begin{bmatrix} 1 & 4 & 7 \\ 2 & ! & 8 \\ 3 & 6 & .. \end{bmatrix}, \quad r = \begin{bmatrix} 0 \\ 1 \\ , \end{bmatrix} \quad \Rightarrow \quad (I - r e_1) C = \begin{bmatrix} 1 & 4 & 7 \\ 1 & 1 & 1 \\ , & , & , \end{bmatrix}.$$

~~Method A~~ . NK IA a7f of Gauss-Jordan L+U T} LXTI }-GIA

If τ is the computed version of an exact Gauss vector τ , then it is easy to verify that

$$\mathbf{0} = \tau + e, \quad \mathbf{L}\mathbf{e} = \mathbf{u}.$$

If τ is used in a Gauss transform update and $H(\mathbf{L}_n^{-1} \mathbf{f}, \mathbf{C})$ denotes the computed result, then

$$\mathbf{f}(\mathbf{L}_n^{-1} \mathbf{f}, \mathbf{C}) = (\mathbf{I} - \mathbf{r}\mathbf{k}\mathbf{C})\mathbf{C} + \mathbf{E},$$

where

$$\|\mathbf{E}\| = 3\|\mathbf{C}\| + \| \mathbf{L}_n^{-1} \mathbf{f} \| \mathbf{C} + O(u^2).$$

Clearly, if τ has large components, then the errors in the update may be large in comparison to $\|\mathbf{C}\|$. For this reason, care must be exercised when Gauss transformations are employed, a matter that is pursued in §34.

Q Q vs F klp Cst{ I R4F: AR{I kI 4s

Assume that $A \in \mathbb{R}^{n \times n}$. Gauss transformations M_1, \dots, M_{n-1} can usually be found such that $M_{n-1} \cdots M_1 A = U$ is upper triangular. To see this we first look at the $n = 3$ case. Suppose

$$A = \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 10 \end{bmatrix}$$

and note that

$$M_1 = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{bmatrix} = M_1 A = \begin{bmatrix} 1 & & \\ 0 & & \\ 0 & -6 & -11 \end{bmatrix}.$$

Likewise, in the second step we have

$$M_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{bmatrix} = M_2(M_1 A) = \begin{bmatrix} 1 & 4 & 7 \\ 0 & -3 & -6 \\ 0 & 0 & 1 \end{bmatrix}.$$

Extrapolating from this example to the general n case we conclude two things:

- At the start of the k th step we have a matrix $A^{(k-1)} = M_{k-1} \cdots M_1 A$ that is upper triangular in columns 1 through $k-1$.
- The multipliers in the k th Gauss transform M_k are based on $A^{(k-1)}(k+1:n, k)$ and $a_{ii}^{(k-1)}$ must be nonzero in order to proceed.

Noting that complete upper triangularization is achieved after $n-1$ steps, we obtain the following rough draft of the overall process:

$$A^{(1)} = A$$

for $k = 1:n-1$

For $i = k+1:n$, determine the multipliers $r_i^{(k)} = a_{ii}^{(k-1)} / a_{kk}^{(k-1)}$. (323)

Apply $M_k = \mathbf{I} - r_i^{(k)} e_k^{(k)} e_k^{(k)T}$ to obtain $A^{(k+1)} = M_k A^{(k)}$.

end

For this process to be well-defined, the matrix entries a_{ij} , $a_{ii} \neq 0$ must be nonzero. These quantities are called pivots.

.1 x13r 7z1 sfrfr

If no zero pivots are encountered in (323), then Gauss transformations M_1, \dots, M_n are generated such that $M_1 \cdots M_n A = U$ is upper triangular. It is easy to check that if $M_k = I_{n-r} \begin{pmatrix} K \\ D \end{pmatrix}$, then its inverse is prescribed by $M_k^{-1} = I_{n-r} \begin{pmatrix} K \\ D \end{pmatrix}$ and so

$$A = LU \quad (324)$$

where

$$L = M_1^{-1} \cdots M_{n-1}^{-1}. \quad (325)$$

It is clear that L is a unit lower triangular matrix because each M_i^{-1} is unit lower triangular. The factorization (324) is called the LU factorization.

The LU factorization may not exist. exip zgzn A

$$\mathbf{L} = \mathbf{M}_1^{-1} \cdots \mathbf{M}_k^{-1} \mathbf{U}$$

It turns out that the construction of \mathbf{L} is not nearly so complicated. Equation (325) suggests. Indeed,

$$\begin{aligned}\mathbf{L} &= \mathbf{M}_1^{-1} \cdots \mathbf{M}_k^{-1} \mathbf{U} \\ &= (\mathbf{I}_n - \mathbf{T}(1)\mathbf{e}_r)^{-1} \cdots (\mathbf{I}_n - \mathbf{T}(n)\mathbf{e}_{-1})^{-1} \\ &= (\mathbf{I}_n - \mathbf{T}(1)\mathbf{e}_r) \cdots (\mathbf{I}_n - \mathbf{T}(n)\mathbf{e}_{-1}) \\ &= \mathbf{I}_n + \sum_{k=1}^n r(k)\mathbf{e}_r\end{aligned}$$

showing that

$$L(k+1:n, k) = r(k)(k+1:n) \quad k = 1:n-1 \quad (327)$$

otherwise the k th column of \mathbf{L} is defined by the multipliers that arise in the k th step of (323). Consider the example in §324

$$\tau^{(1)} = \begin{bmatrix} 0 \\ 2 \\ 3 \end{bmatrix}, \quad \tau^{(2)} = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 10 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 4 & 7 \\ 0 & -3 & -6 \\ 0 & 0 & 1 \end{bmatrix}.$$

mllllb 7lb 6Df Ixbx kn Dff bktf b kb 8I jb

Since the application

$$B - zw^T/\alpha = L_1 U_1$$

$$A = \begin{bmatrix} 1 & 0 \\ z/\alpha & I_{n-1} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & L_1 U_1 \end{bmatrix} \begin{bmatrix} \alpha & w^T \\ 0 & I_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ z/\alpha & L_1 \end{bmatrix} \begin{bmatrix} \alpha & w^T \\ 0 & U_1 \end{bmatrix} \equiv LU.$$

~~lnf hqA a7}v m}2A V1f2l ol} - mA~~

Let us consider the efficient implementation of (323). First, because zeros have already been introduced in columns 1 through $k - 1$, the Gauss transformation update need only be applied to columns k through n . Of course, we need not even apply the k th Gauss transform to $A(:, k)$ since we know the result. So the efficient thing to do is simply to update $A(k+1:n, k+1:n)$. Also, the observation (327) suggests that we can overwrite $A(k+1:n, k)$ with $L(k+1:n, k)$ since the latter houses the multipliers that are used to zero the former. Overall we obtain

~~m2ishfi- nvloyI ec H5k xsInHw5 elat~~ Suppose $A \in \mathbb{R}^{n \times n}$ has the property that $A(1:k, 1:k)$ is nonsingular for $k = 1:n - 1$. This algorithm computes the factorization $A = LU$ where L is unit lower triangular and U is upper triangular. For $i = 1:n - 1$, $A(i, i:n)$ is overwritten by $U(i, i:n)$ while $A(i+1:n, i)$ is overwritten by $L(i+1:n, i)$.

```

for k = 1:n - 1
    p = k + 1:n
    A(p, k) = A(p, k) / A(k, k)
    A(p, p) = A(p, p) - A(p, k) * A(k, p)
end

```

This algorithm involves $2n^3/3$ flops and it is one of several implementations of Gaussian elimination. Note that the k -th step involves an $(n - k)$ -by- $(n - k)$ outer product.

~~ml mib Ef esxb & selektb~~

Similar to matrix-matrix multiplication, Gaussian elimination is a triple-loop procedure that can be arranged in several ways. Algorithm 321 corresponds to the "kj" version of Gaussian elimination if we compute the outer product update row by row:

```

for k = 1:n - 1
    A(k+1:n, k) = A(k+1:n, k) / A(k, k)
    for i = k+1:n
        for j = k+1:n
            A(i, j) = A(i, j) - A(i, k) * A(k, j)
        end
    end
end

```

There are five other versions: $kj\bar{i}$, $ik\bar{j}$, $ij\bar{k}$, jik , and jki . The last of these results in an implementation that features a sequence of ~~gaps~~ and forward eliminations which we now derive at the vector level.

The plan is to compute the j -th columns of L and U in step j . If $j = 1$, then by comparing the first columns in $A = LU$ we conclude that

$$L(2:n, 1) = A(2:n, 1) / A(1, 1)$$

and $U(1, 1) = A(1, 1)$. Now assume that $L(:, 1:j - 1)$ and $U(:, 1:j - 1)$ are known. To get the j -th columns of L and U we equate the j -th columns in the equation $A = LU$

a direct from the vector equation $A(:,j) = LU(:,j)$ that

$$A(l:j - 1:j) = L(l:j - 1:l:j - 1) \cdot U(l:j - 1:j)$$

and

$$A(j:n,j) = \sum_{k=1}^{j-1} L(j:n,k) \cdot U(k,j).$$

The first equation is a lower triangular linear system that can be solved for the vector $U(l:j - 1:j)$. Once this is accomplished, the second equation can be rearranged to produce recipes for $U(j:j)$ and $L(j+1:n,j)$. Indeed, if we set

$$\begin{aligned} v(j:n) &= A(j:n,j) - \sum_{k=1}^{j-1} L(j:n,k)U(k,j) \\ &= A(j:n,j) - L(j:n,l:j - 1) \cdot U(l:j - 1:j), \end{aligned}$$

then $L(j+1:n,j) = v(j+1:n)/v(j)$ and $U(j,j) = v(j)$. Thus, $L(j+1:n,j)$ is a scaled copy and we obtain the following alternative to Algorithm 321:

Algorithm 322. Suppose $A \in \mathbb{R}^{m \times n}$ has the property that $A(l:k, l:k)$ is nonsingular for $k = 1:n - 1$. This algorithm computes the factorization $A = LU$ where L is unit lower triangular and U is upper triangular.

Initialize L to the identity and U to the zero matrix
for $j = 1:n$

```

if j = 1
    v = A(:,1)
else
    i = A(:,j)
    Sdve L(l:j - 1,l:j - 1) . z = i(l:j - 1) fr z E] - 1.
    U(l:j - 1,j) = z
    v(j:n) = i(j:n) - L(j:n,l:j - 1) . z
end
U(j,j) = v(j)
L(j+1:n,j) = v(j+1:n)/v(j)
end

```

(We choose to have separate arrays for L and U for clarity, it is not necessary in practice) Algorithm 322 requires $2n^3/3$ flops, the same volume of floating point work required by Algorithm 321. However, from §1.5.2 there is less memory traffic associated with a copy than with an outer product, so the two implementations could perform differently in practice. Note that in Algorithm 322 the original $A(:,j)$ is untouched until step j .

The terms right-looking and left-looking are sometimes applied to Algorithms 321 and 322. In the outer-product implementation, after $L(k:n,k)$ is determined, the columns to the right of $A(:,k)$ are updated so it is a right-looking procedure. In contrast, subcolumns to the left of $A(:,k)$ are accessed in a copy before $L(k+1:n,k)$ is produced so that implementation left-looking.

In $\mathbb{R}^{n \times n}$, if $A \neq 0$ and $\{A_{ij}\}_{i,j=1}^n$ are non-zero, then A is invertible.

The LU factorization of a rectangular matrix $A \in \mathbb{R}^{n \times r}$ can also be performed. The case is illustrated by

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 3 & 1 \\ 5 & 2 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ -1 & 1 \end{bmatrix}$$

while

$$\begin{bmatrix} ! & : \\ ! & : \end{bmatrix} = \begin{bmatrix} ! & : \\ - & : \end{bmatrix} \begin{bmatrix} ! & : \\ - & : \end{bmatrix}$$

depicts the situation. The LU factorization of $A \in \mathbb{R}^{n \times r}$ is guaranteed to exist if $A(l:k, l:k)$ is nonsingular for $k = 1:\min\{n, r\}$.

The square LU factorization algorithm above needs only minor alterations to handle the rectangular case. For example, if $n > r$, then Algorithm 3.2.1 modifies to the following:

$f \leftarrow k = 1:r$

$p \leftarrow k + 1:n$

$A(p, k) \leftarrow A(p, k)/A(k, k)$
if $k < r$

$\mu \leftarrow k + 1:r$

$A(p, \mu) \leftarrow A(p, \mu) - A(p, k) \cdot A(k, \mu)$

end

end

This calculation requires $\frac{n(n+1)}{2}r^3$ flops. Upon completion, A is overwritten by the strictly lower triangular portion of $L \in \mathbb{R}^{n \times n}$ and the upper triangular portion of $U \in \mathbb{R}^{n \times r}$.

$CQ, Q, T, \dots, n \text{ and } k \text{ VT}$

It is possible to organize Gaussian elimination so that matrix multiplication become the dominant operation. Partition $A \in \mathbb{R}^{n \times n}$ as follows:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}_{r \times n-r}$$

where r is a blocking parameter. Suppose we compute the LU factorization

$$\begin{bmatrix} & \\ & \ddots \\ & & \ddots \end{bmatrix} = \begin{bmatrix} & \\ & \ddots \\ & & \ddots \end{bmatrix} U_n.$$

Here, $L_n \in \mathbb{R}^{n \times n}$ is unit lower triangular and $U_n \in \mathbb{R}^{n \times n}$ is upper triangular assumed to be nonsingular. If we solve $L_n U_n = A$ for $U_n = E_n^{-1} A$, then

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} L_n & 0 \\ L_{21} & I_{n-r} \end{bmatrix} \begin{bmatrix} I_r & 0 \\ 0 & A \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ 0 & I_{n-r} \end{bmatrix},$$

where

$$A = A_{22} - L_{21}U_{12} = A_{22} - A_{21}A_{11}^{-1}A_{12} \quad (329)$$

is the Schur complement of A_{11} in A . Note that if

$$A = L_{22}U_{22}$$

In the LU factorization of A , then

$$A = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{bmatrix}$$

and the LU factorization of A . This lays the groundwork for a recursive implementation.

Algorithm 321: Suppose $A \in \mathbb{R}^{n \times n}$ has an LU factorization where n is a positive integer. The following algorithm computes unit lower triangular $L \in \mathbb{R}^{n \times n}$ and upper triangular $U \in \mathbb{R}^{n \times n}$ so $A = LU$.

function $[L, U] = \text{BlockLU}(A, n, r)$

if $n = r$

Compute the LU factorization $A = LU$ using (say) Algorithm 321.

else

Use (328) to compute the LU factorization $A(:, 1:r) = [f]_{1:r}$.

Solve $L_{1:r}U_{12} = A(:, r+1:n)$ for U_{12}

$A = A(r+1:n, r+1:n) - L_{21}U_{12}$

$[L_{22}, U_{22}] = \text{BlockLU}(A, n-r, r)$

$L = \begin{bmatrix} f & L_{12} \\ L_{21} & L_{22} \end{bmatrix} \quad U = \begin{bmatrix} U_{11} & F_{12} \\ 0 & U_{22} \end{bmatrix}$

end

end

The following table explains where the flops come from.

Activity	Flops
L_{11}, L_{21}, U_{11}	$nr^2 - r^3/3$
U_{12}	$(n-r)r^2$
.	$2(n-r)^2$

If $n \gg r$, then there are a total of about $2n^3/3$ flops, the same volume of arithmetic as Algorithms 321 and 322. The vast majority of these flops are the level-3 flops associated with the production of A .

The actual level-3 factorization, a concept developed in §3.1.5, is more easily derived via a nonrecursive implementation. Assume for clarity that $n = Nr$ where N is a positive integer and that we want to compute

$$\begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & \ddots & \vdots \\ A_{N1} & \cdots & A_{NN} \end{bmatrix} = \begin{bmatrix} L_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ L_{N1} & \cdots & L_{NN} \end{bmatrix} \begin{bmatrix} U_{11} & \cdots & U_{1N} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & U_{NN} \end{bmatrix} \quad (3210)$$

where all blocks are r by r. Analogously to Algorithm 3.2.3 we have the following

method: Suppose $A \in \mathbb{R}^{N \times N}$ has an LU factorization and r is a positive integer. The following algorithm computes unit lower triangular $L \in \mathbb{R}^{N \times N}$ and upper triangular $U \in \mathbb{R}^{N \times N}$ so $A = LU$.

for $k = 1:N$

Rectangular Gaussian elimination

$$\begin{bmatrix} A_{1k} \\ \vdots \\ A_{Nk} \end{bmatrix} = \begin{bmatrix} L_{1k} \\ \vdots \\ L_{Nk} \end{bmatrix} u_k$$

Multiple right hand side solve

$$L_{kk}[u_{kk+1:N} \dots u_{k:N}] = [A_{kk+1:N} \dots A_{k:N}]$$

Level-3 updates:

$$A_{ij} = A_{ij} - L_{ik} u_i \quad i = k+1:N, j = k+1:N$$

end

Here is the flop situation during the k th pass through the loop:

Activity	Flops
Gaussian elimination	$(N - k + 1)r^3 - r^3/3$
Multiple RHS solve	$(N - k)r^3$
Level-3 updates	$2(N - k)^2 r^2$

Summing these quantities for $k = 1:N$ we find that the level-3 flop count is approximately

$$\frac{2r^3/3}{2r^3/3 + r^2} = \frac{1}{1 + 2N^{-1}}$$

Thus for large N almost all arithmetic takes place in the context of matrix multiplication. This ensures a favorable amount of data reuse as discussed in §1.5.4.

Problems

P3.2.1 e8... Pr2.P.84(3.2b)

P3.2.2 .2..8.8 P.8.P.8.8 >A(B E E xn REA(vRT), (== 1xTAFAvRIR) ART(v=(RCAaR=Rx
l=,TA RCRA= A(O R), R=RTEvtaTR= ,,,RREtaA= TREAG C(LRCR)E =,hATA
=BR=E ICA TRRxA(B Ck Rx8MRa(TT)RRTA(B = L(BU(B RxvRCp(R(B RxvU(B REA/
a)RTv,(== 1xTAFAvRIR,=AG

P3.2.3 .2..8.8 fo8..P.P.84A ERnxn

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

LCAEAn T= 3BRvxv(v=Tv) REARS ,A REACE, a(T=2TAvRoAn TvAk xA'vA Tv(3.29.
hJ18 OJiOp A0aAA4ilpMQJ321, A(r+1:n + 1:n) N• (=GS. (La,k S,A (RRTA
RRAE=RA16 METRCT 3.2.2?

P3.2.4 .2..8.8 $A \in \mathbb{R}^{n \times n}$, $E, f^n \leq p$, $n|4 + n|5|n|5Ax = baHQA = (2AxLTR, R = R(EIv1R97, H) = A(T1RTv1RA8M)R) = RR(vRA_n by (n+1) HRETR(AH))$

P3.2.5 88...8.4P 8N2....4 8..2.6.P.8P..P.6P.8.2.8.0t8.84P8.8.2.4. 8A TTRA ax9Th: i,Ni,m 89M.9plm,aA B9OpMTiOMH2ULLCBETvT,vR,1Ae RETRv,RExL i A3E rAO)nAy0

P3.2.6 NP...8.4 $R^{n \times n}$, $E, N(y, k) = I - yf LCAEA \in \mathbb{R}^n, \leq n, +, -, -$ Gauss-Jordan for an tr nsfor ations. $\tilde{y}idM, ai vp1, i \in N(y, k), 1 \leq k \leq 10$ TRAT=R(X)T2AOx $\in \mathbb{R}^n, 15 \leq 5, p$
 $\tilde{y}(f^5(+4y) oav, m A)N(y, k)x = ek? 7.dM.h OC u = J-O \times J=4 = a C = \Theta w = Os =$
 $aS = x = r4AaLTRA-1, LCR a(vxIRT(GO A 2D, EBRA, aaA = o 1E RETRCH)$

P3.2.7 PpP8418g..P.2 9m0.8 0Pf.P..4 ...8 ..4.88 Pf8l8 6f84A Ck hEA E=RCRO
 $a=7 = F$

P3.2.8 ..86 f86 A aRvA (2AE-LTRRvRL RvxUTD=1EIRCb, 3F3)T2A $\tilde{y}((1=1ai aRRT(v$
 $so a JwS-4aa O= *4=3x=n Ox s O n$

P3.2.9 88.888.. .8...84 8N2....6 8...f4.P.8446f..f P.8.448..8P8>P.8Pf.88 888.88.
 $a - a = -J = a = -$

Notes and References for §3.2

3NP.8. 8N2. ..4P.841 a S = 0 = x = a x = axax x = ru_

a 7 37 (31s)F201 NEXTVR-(TTTvRRi(v AaRTH)H= = TRFTTTRR(H) Historica Mathematica, 38, .. 7Sng1
 $\tilde{y}IxEl a(31s)4® RRCATRRT(GO = TRQ ThICRRT) Notices of the AMS 58, o n,n. o$

M iAMcJ1t8t,7a 77 n@l,m,t., 9a haJJ,mAa7mM, Mda a,Ith M..M7uJl6A7mAaJS 7utMlt7mAa,
 $m 7vetat0$

|e .M771t (SIM)F® RAEERRR(H)=RCA,E ,(T1ATAvR (=Lin" Alg. Applic. 8, .g,S n...)

M uAM8J1tct,7a 6lt y5M25 w 4Ca.aa 7la5a,l8aL m5aM8t aJJ1mAa7mM,w ..t ..at M.Ca. a3
 $\tilde{y}eC7ea,a,lc7mM5a 7 7nnn6t7am,tS, xMr(sJ5)F+AAR==E$

c iArrAEROxI I(TTRv (sIY)m®+2CR,TIRR(HRA),<3U ExtR(H) R.TI Numer. Math. 54 r Ai5A6A

A w Vx = axxw=w= x= x= 2= x= = a ixox Da Onxx= = 0=w= Sc= = 4x
 $x x' 1,x 04xiz = 0w0= = x.€ xia a = C x oaa x = • = x x' x = 1x = •$

G.E. R=oer(sJ6)F ®E(RRC RTR(H)Commun. ACM 3, ,oS;, -

W.M. AET Oox)(sJ5)F®E(R LiRCj,TIRR(HRA)(=Commun. ACM. 5, r r r E5 r 6

Loop " =)OE8rOy or@oZ.Os+E.f.r)rO.n)=)OoE.o@nT

Bri® " .y)=R=M10r>0=)8 QM)= f(sJ)4®T1Ab=CRT8TvxAREAER METRCb= E
 $\tilde{y} RAO - RREta2e RvxAR(E H=TvA RgOA(SIAM Review 26, Ai iiN$

J.M. B=rOyYF®A jki(Hb= (iRaR(ET RRi(CRC(x-E=2aR(ET1RA= (=Parallel Comput. 7, iEr 516

D.H.)(OoT & OOn GBU, .1 OX2sIIs). ®M=TOR H= = ATRCT R(haAAERRRCA (=, Ri(v
 $a jTARx I-RAT=J Super comput. = r . M?$

If, eau la, 7a= IsNJi1 7dn5Ghw.9paM(gP)m®TIR, TIR b I aSMRaR(TYRH Numer.
 $\tilde{y} Lin. Alg. Applic. 2 iIE5A?k$

Suppose $A = LURvxA + AA = (L+AL)(U+AU) REAMdAr - E = I, vx = (vRA)AER,ERRT(v=j L (.) AUTVRAET@AA REA12AvT7$

G.W. 2r8Er (sIM)F®NRCA AERERRT(GMRvx(C(A=r1 iRaR(E IMA J. Numer. Anal. ? 7
 $i u6$

x-w. e•)8 y 0)8e 40y 0(sIY)b®Nv RCA=TRT2A8MdrAETRT(vIBIT 38, ,g,S; .ra

In particular, if $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, then $\det(A) = ad - bc$. If $a = 10^{16}$ and $b = 10^{-16}$, then $\det(A) = 10^{32}$, while $A^{-1} = \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$.

It is natural to ask how well the computed solution x satisfies the equation $Ax = b$. The condition number of A is defined as $\kappa(A) = \|A\|_1 \|A^{-1}\|_\infty$. It is known that $\kappa(A) \geq 1$ and that $\|A^{-1}\|_\infty \leq \kappa(A)$. The condition number provides a measure of the sensitivity of the solution to perturbations in the data. A large condition number indicates that the system is ill-conditioned, meaning that small changes in the data can lead to large changes in the solution.

3.3 Roundoff Error in Gaussian Elimination

We now assess the effect of rounding errors when the algorithms in the previous two sections are used to solve the linear system $Ax = b$. A much more detailed treatment of roundoff error in Gaussian elimination is given in Higham (ASNA).

mmmb bxxkxb 1bf)Sb sRb 3taf kx1 ftf 1kb

Let us see how the error bounds for Gaussian elimination compare with the ideal bounds derived in §2.7.11. We work with the infinity norm for convenience and focus our attention on Algorithm 321, the outer product version. The error bounds that we derive also apply to the gaxpy formulation (Algorithm 322). Our first task is to quantify the roundoff errors associated with the computed triangular factors.

Theorem 3.3.1. Assume that A is an n -by- n matrix of floating point numbers. If no zero pivots are encountered during the execution of Algorithm 321, then the computed triangular matrices \hat{L} and \hat{U} satisfy

$$\hat{L}\hat{U} = A + H, \quad (3.3.1)$$

$$|H| \leq 2(n-1)\|A\| + |\hat{L}|\|\hat{U}\| + O(\bar{u}^2). \quad (3.3.2)$$

Pr 6. The proof is by induction on n . The theorem obviously holds for $n=1$. Assume that $n>2$ and that the theorem holds for all $(n-1)$ -by- $(n-1)$ floating point matrices. If A is partitioned as follows

$$A = \begin{bmatrix} \alpha & w^T \\ v & B \\ \vdots & \vdots \\ n & n \end{bmatrix}, \quad \begin{matrix} 1 \\ 1 \\ \vdots \\ 1 \end{matrix}$$

then the first step in Algorithm 321 is to compute

$$=I = fl(v/a), \quad \hat{C} = fl(\hat{z}w^T), \quad A = fl(B - C),$$

from which we conclude that

$$=I = v/a + f, \quad (3.3.3)$$

$$|f| \leq uv/a, \quad (3.3.4)$$

$$\hat{C} = \hat{z}w^T + F_1, \quad (3.3.5)$$

$$|F_1| \leq u|\hat{z}||w^T|, \quad (3.3.6)$$

$$\hat{A}_1 = B - (zw^T + F_1) + F_2 \quad (337)$$

$$F_2 \leq u(BI + L_1 w^T) + O(u^2), \quad (338)$$

$$A_{11} := BI + L_1 w^T + O(u). \quad (339)$$

The algorithm proceeds to compute the LU factorization of A_1 . By induction, the computed factors L_1 and U_1 satisfy

$$\hat{L}_1 \hat{U}_1 = \hat{A}_1 + H_1 \quad (3310)$$

where

$$|H_1| \leq 2(n-2)u \left(|\hat{A}_1| + |\hat{L}_1||\hat{U}_1| \right) + O(u^2). \quad (3311)$$

If

$$\hat{L} = \begin{bmatrix} 1 & 0 \\ \hat{z} & \hat{L}_1 \end{bmatrix}, \quad C = \begin{bmatrix} \mathbf{a}^{w^T} \\ 0 \end{bmatrix},$$

then it is easy to verify that

$$LU = A + H$$

where

$$H = \begin{bmatrix} 0 & 0 \\ \alpha f & H_1 - F_1 + F_2 \end{bmatrix}. \quad (3312)$$

To prove the theorem we must verify (332), i.e.,

$$|H| \leq 2(n-1)u \left[\frac{2|\alpha|}{|v| + |\alpha||f|} \frac{2|w^T|}{|B| + |\hat{L}_1||\hat{U}_1| + |\hat{z}||w^T|} \right] + O(u^2).$$

Considering (3312), this is obviously the case if

$$|H_1| + |F_1| + |F_2| \leq 2(n-1)u \left(|B| + |\hat{z}||w^T| + |\hat{L}_1||\hat{U}_1| \right) + O(u^2). \quad (3313)$$

Using (339) and (3311) we have

$$H_{11} \leq 2u(BI + L_1 w^T) + O(u^2),$$

while (336) and (338) imply

$$F_{11} + F_{21} \leq u(BI + 2L_1 w^T) + O(u^2).$$

These last two results establish (3313) and therefore the theorem. \square

We mention that if A is m by n , then the theorem applies with n replaced by the smaller of n and m in Equation 332.

1cmA T7GfNB7ALBGfA uG EAV1pji-A T7Gf2+A

We next examine the effect of roundoff error when \hat{L} and \hat{U} are used by the triangular systems solvers of §3.1.

Theorem 3.3.2. Let \hat{L} and \hat{U} be the computed LU factors obtained by Algorithm 3.2.1 when it is applied to an n -by- n floating point matrix \bar{A} . If the methods of §3.1 are used to produce the computed solution \hat{y} to $\hat{L}\hat{y} = b$ and the computed solution \hat{x} to $\hat{U}\hat{x} = \hat{y}$, then $(A + E)\hat{x} = b$ with

$$|E| \leq nu (2|A| + 4|\hat{L}||\hat{U}|) + O(u^2), \quad (33.14)$$

Pr of. From (3.1.1) and (3.1.2) we have

$$\begin{aligned} (\hat{L} + F)\hat{y} &= b & |F| &\leq nu|\hat{U}| + O(u^2), \\ (\hat{U} + G)\hat{x} &= \hat{y}, & |G| &\leq nu|\hat{U}| + O(u^2), \end{aligned}$$

and thus

$$(\hat{L} + F)(\hat{U} + G)\hat{x} = (\hat{L}\hat{U} + F\hat{U} + \hat{L}G + FG)\hat{x} = b$$

If this follows from Theorem 3.3.1 that $\hat{L}\hat{U} = A + H$ with

$$|H| : 2(n-1)u(|A| + |\hat{L}||\hat{U}|) + O(u^2),$$

and so by defining

$$E = H + F\hat{U} + \hat{L}G + FG$$

we find $(A+E)\hat{x} = b$. Moreover,

$$\begin{aligned} |E| &\leq |H| + |F||\hat{U}| + |\hat{L}||G| + O(u^2) \\ &\leq 2nu(|A| + |\hat{L}||\hat{U}|) + 2nu(|\hat{L}||\hat{U}|) + O(u^2), \end{aligned}$$

completing the proof of the theorem. \square

If it were not for the possibility of a large $|\hat{L}||\hat{U}|$ term, (33.14) would compare favorably with the ideal bound (2.7.21). (The factor n is of no consequence cf. the Wilkinson quotation in §2.7.7.) Such a possibility exists, for there is nothing in Gaussian elimination to rule out the appearance of small pivots. If a small pivot is encountered, then we can expect large numbers to be present in \hat{L} and \hat{U} .

We stress that small pivots are not necessarily due to ill-conditioning a the example

$$A = \begin{bmatrix} & \\ & \end{bmatrix} = \begin{bmatrix} 1 & E \\ & \end{bmatrix} \begin{bmatrix} & \\ \rightarrow & jE \end{bmatrix}$$

shows. Thus, Gaussian elimination can give arbitrarily poor results, even for well-conditioned problems. The method is unstable. For example, suppose 3 digit floating point arithmetic is used to solve

$$\begin{bmatrix} .001 & 100 \\ 100 & 200 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 100 \\ 300 \end{bmatrix}.$$

{Se §2.7.1.) Applying Gaussian elimination we get

$$\mathbf{L} = \begin{bmatrix} 1 \\ 1000 \end{bmatrix}, \quad \hat{\mathbf{U}} = \begin{bmatrix} .001 & & \\ 0 & -1 & 00 \end{bmatrix},$$

a calculation shows that

$$\text{to} = \begin{bmatrix} .001 & 1 \\ 1 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -2 \end{bmatrix} \equiv A + H.$$

If we go on to solve the problem using the triangular systems solver of §3.1, then using the same precision arithmetic we obtain a computed solution $\hat{x} = [0, 1]^T$. This is in contrast to the exact solution $x = [1.002 \dots \dots]^T$.

Problems

P3.3.1 .386 .3 r16 . 8.3. .e2. 86. A 6A litMA16A etMij lpcu 3.3.1, "2--
1,"(" 3.3.2 2N-) 5("2"2 +n- +(1. " --,+-" "3.")

P3.3.2 .2.8.. A M_{ij} = r or let \hat{U} ij mt.it imm i:c Aplmacm 4sil p6V, 3.2.1. ,) s(c
 $\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \|\hat{U}\|_1 \|\hat{U}\|_\infty?$ $\|\hat{U}\|_1 \leq (1 + 2nu) \|\hat{U}\|_\infty + O(\mathbf{u}^2).$

Notes and References for §3.3

38..... .8..8,r. 81N 2r..8. r.
 J.H. $0 < 0.06n = 1R^n \leq g^n \leq (((nE) \leq 1 + 2^n)(n!)^{\frac{1}{n}})$ (" - 67J. ACM 8, 281-330.
 $\frac{n!}{(1 - n - 1)^n} \leq ((1 - n - 1)^{-n})^{2^n} \leq 2^{-n} \leq 2^{-n} \leq 2^{-n} \leq 6^{(1 - n)^{2^n}}$
 $\leq 6^{(1 - n)^{2^n}} \leq 6^{(1 - n)^{2^n}} \leq 6^{(1 - n)^{2^n}} \leq 6^{(1 - n)^{2^n}}$
 J.K. .00 (1971). 10 $n^n - n^n$ " 2- ", (- " , 1((, "R^n - , " 1^n " 6J. Inst. Math. Applic. 8
 374-75.
 $2^{(1973)-10} R^n \leq 0^n \leq ((n, | 2^n) \leq n - (n - 1) \leq (R^n " u, " n)$ Math. Comput.
 27, 355-59.

H.H. - $\mathbf{n} = \mathbf{n} = (\pm 9\sqrt{2})^{\frac{1}{2}}$ $20 + u$ $n + HR^n \leq R^n$ " $\mathbb{K}^n = (-1)^n | u | \leq R^n + , n = 6$
 J. Inst. Math. Applic. 20, 409-14.

J.J. ®. e = ab - Q.k30yea(1992). 1." ,(- "n | -"2^n)(| ≤ | , (. . IMA J. Numer. Anal. 12, 1-19.

J.M.)) = EhOee Ope ee) = 06eeRf x (1998). 1 2|+ n "u- "+n "2-8 -2"lb,")

P. .. = 00) = OR H(b0)(1999). 0-50- \leq^n 2 Rⁿ, +∞5, ORⁿ $\leq^{\text{def}}\text{m}$, ≤(((n | +^m- (,)ⁿ)
BIT 99, 385-402.

An 0 = n = no = 0 \Rightarrow v n) EX 1Z = = c0n := 0 \Rightarrow Xno) OR Aex 0x00) n < rXr 0=

J.F. ' = ,)≠2011). 1 $\frac{n_7^m}{1-\leq(\theta^m-)} n_- -;$ $\frac{Q,-(}{\$IAM Review 59} \frac{(n^B}{607-682} .((, R^n-(1 " " ")^n "2 (-r |r)-\leq"$

34 Pivoting

The analysis in the previous section shows that we must take steps to ensure that no large entries appear in the computed triangular factors L and \hat{U} . The example

$$A = \begin{bmatrix} .0001 & & \\ & 1 & \\ & & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & .0001 \\ 10000 & 1 & 0 \end{bmatrix} \begin{bmatrix} !! & \\ & 0 \end{bmatrix} = LU$$

correctly identifies the source of the difficulty: relatively small pivots. A way out of this difficulty is to interchange rows. For example, if P is the permutation

$$P = \begin{bmatrix} & \\ & \end{bmatrix}$$

then

$$PA = \begin{bmatrix} 0.01 & \\ .001 & \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0.01 & 1 \end{bmatrix} \begin{bmatrix} & \\ .99 & \end{bmatrix} = LU.$$

Observe that the triangular factors have modestly sized entries.

In this section we show how to determine a permuted version of A that has a reasonably stable LU factorization. There are several ways to do this and they each corresponds to a different pivoting strategy. Partial pivoting, complete pivoting and row pivoting are considered. The efficient implementation of these strategies and their properties are discussed. We begin with a few comments about permutation matrices that can be used to swap rows or columns.

`m1spb etf Sxft t'Sb SxDf tf 1keb`

The stabilizations of Gaussian elimination that are developed in this section involve data movements such as the interchange of two matrix rows. In keeping with our desire to describe all computations in "matrix terms," we use permutation matrices to describe this process. (Now is a good time to review § 1.28–§ 1.211.) Interchanges permutations are particularly important. These are permutations obtained by swapping two rows in the identity, e.g.,

$$\Pi = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

Interchange permutations can be used to describe row and column swapping. If $A \in \mathbb{R}^{n \times n}$, then $\Pi \cdot A$ is A with rows 1 and 4 interchanged while $A \cdot \Pi$ is A with column 1 and 4 swapped.

If $P = \Pi_{m \times m} \cdots \Pi_1$ and $\text{eht } k$ is the identity with rows k and $piv(k)$ interchanged, then $piv(1:m)$ encodes P . Indeed, $x \in \mathbb{R}^n$ can be overwritten by Px as follows:

```
for k = 1:m
    x(k) t x(piv(k))
end
```

Here, the "`t`" notation means "swap contents." Since each Π_k is symmetric, we have $\Pi_k^T = \Pi_k$. Thus, the piv representation can also be used to overwrite x with Px :

```
for k = m:-1:1
    x(k) t x(piv(k))
end
```

We remind the reader that although no floating point arithmetic is involved in a permutation operation, permutations move data and have a nontrivial effect upon performance.

16 GA a) 7 - m) 2A G. - GfA

Interchange permutations can be used in LU computations to guarantee that no multiplier is greater than 1 in absolute value. Suppose

$$A = \begin{bmatrix} 3 & 17 & 10 \\ 2 & 4 & -2 \\ 6 & 18 & -12 \end{bmatrix}.$$

To get the smallest possible multipliers in the first Gauss transformation we need to use to be the largest entry in the first column. Thus, if I_1 is the interchange permutation

$$I_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

then

$$I_1 A = \begin{bmatrix} 6 & 18 & -12 \\ 2 & 4 & -2 \\ 3 & 17 & 10 \end{bmatrix}.$$

It follows that

$$M_1 = \begin{bmatrix} 1 & 0 & 0 \\ -1/3 & 1 & 0 \\ -1/2 & 0 & 1 \end{bmatrix} \quad M_1 I_1 A = \begin{bmatrix} 6 & 18 & -12 \\ 0 & -2 & 2 \\ 0 & 8 & 16 \end{bmatrix}.$$

To obtain the smallest possible multiplier in M_2 we need to swap rows 2 and 3. Thus, if

$$\Pi_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad M_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1/4 & 1 \end{bmatrix},$$

then

$$M_2 \Pi_2 M_1 I_1 A = \begin{bmatrix} 6 & 1 & - \\ 0 & 0 & 6 \\ 0 & 0 & 6 \end{bmatrix}.$$

For general n we have $\boxed{\text{N} \otimes \text{O}^n} (\text{t} \otimes \text{3}^n)$

$$M_{n-1} \Pi_{n-1} \cdots M_1 \Pi_1 A = U$$

1S1A tEE7 FAG+A

It turns out that (3.4.1) computes the factorization

$$PA = LU \quad (3.4.3)$$

where $P = I_n 1 \cdots 1 U$ is upper triangular, and L is unit lower triangular with $\text{ptn} \leq 1$. We show that $L(k+1:n, k)$ is a permuted version of M_k 's multipliers. From (3.4.2) it can be shown that

$$\tilde{M}_{n-1} \cdots \tilde{M}_1 PA = U \quad (3.4.4)$$

where

$$J_{-i} = (I_n 1 \cdots I_{k-1} M_k I_{k+1} \cdots I_n 1) \quad (3.4.5)$$

for $k = 1:n - 1$. For example, in the $n = 4$ case we have

$$\tilde{M}_3 \tilde{M}_2 \tilde{M}_1 PA = M_3 \cdot (\Pi_3 M_2 \Pi_3) \cdot (\Pi_3 \Pi_2 M_1 \Pi_2 \Pi_3) \cdot (\Pi_3 \Pi_2 \Pi_1) A$$

since the I_{ll} are symmetric. Moreover,

$$\tilde{M}_k = (\Pi_{n-1} \cdots \Pi_{k+1}) \cdot (I_n - \tau^{(k)} e_k^T) \cdot (\Pi_{k+1} \cdots \Pi_{n-1}) = I_n - \tilde{\tau}^{(k)} e_k^T$$

with $f(k) = I_n 1 \cdots I_{k-1} T^k$. This shows that N is a Gauss transformation. The transformation from T^k to $f(k)$ is easy to implement in practice.

~~mA s5(- mvmUdH5s xsInHwe mi2Ch5q-xns5npvxh7 5A2a~~ This algorithm computes the factorization $PA = LU$ where P is a permutation matrix encoded by $piv(1:n - 1)$, L is unit lower triangular with $\text{uln} \leq 1$, and U is upper triangular. For $i = 1:n$, $A(i, i:n)$ is overwritten by $U(i, i:n)$ and $A(i+1:n, i)$ is overwritten by $L(i+1:n, i)$. The permutation P is given by $P = I_n 1 \cdots 1$ where I_k is an interchange permutation obtained by swapping rows k and $piv(k)$ of I_n .

for $k = 1:n - 1$

Determine μ with $k \leq \mu \leq n$ so $|A(\mu, W)L| = \|A(k:n, k)\|_\infty$

$$piv(k) = \mu$$

$$A(k, :) \leftrightarrow A(\mu, :)$$

$$\text{if } A(k, k) = 0$$

$$\rho = k + 1:n$$

$$A(\rho, k) = A(\rho, k)/A(k, k)$$

$$A(\rho, \rho) = A(\rho, \rho) - A(\rho, k)A(k, \rho)$$

end

end

The floating point overhead associated with partial pivoting is minimal from the stand point of arithmetic as there are only $O(n^2)$ comparisons associated with the search for the pivots. The overall algorithm involves $2n^2/3$ fops.

If Algorithm 34.1 is applied to

$$A = \begin{bmatrix} 3 & 17 & 10 \\ 2 & 4 & -2 \\ 6 & 18 & -12 \end{bmatrix},$$

then upon completion

$$A = \begin{bmatrix} 1 & -12 \\ 1/2 & 16 \\ 1/3 - 1/4 & 6 \end{bmatrix}$$

a piv = [3, 3]. These two quantities encode all the information associated with the reduction.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} A = \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 1/3 - 1/4 & 1 \end{bmatrix} \begin{bmatrix} 6 & 18 & -12 \\ 0 & 8 & 16 \\ 0 & 0 & 6 \end{bmatrix}.$$

To compute the solution to $Ax = b$ after invoking Algorithm 34.1, we solve $Ly = Pb$ for y and $Ux = y$ for x . Note that b can be overwritten by Pb as follows:

for $k = 1:n-1$

$$b(k) + b(piv(k))$$

end

We mention that if Algorithm 34.1 is applied to the problem

$$\begin{bmatrix} .001 & 1.00 \\ 1.00 & 2.00 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1.00 \\ 3.00 \end{bmatrix},$$

using 3 digit floating point arithmetic, then

$$P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad L = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

~~Algorithm 3.4.18~~ This algorithm computes the factorization $PA = LU$ where P is a permutation matrix encoded by $\text{piv}(1:n-1)$, L is unit lower triangular with $\|L\|_1 \leq 1$, and U is upper triangular. For $i = 1:n$, $A(i, i:n)$ is overwritten by $U(i, i:n)$ and $A(i+1:n, i)$ is overwritten by $L(i+1:n, i)$. The permutation P is given by $P = I_n - I_{ij} + I_k$ where I_k is an interchange permutation obtained by swapping rows k and $\text{piv}(k)$ of I_n .

Initialize L to the identity and U to the zero matrix.

for $j = 1:n$

if $j = 1$

$v = A(:, 1)$

else

$i = I_{j-1} \cdots I_{1} A(:, j)$

Solve $L(1:j-1, 1:j-1)z = i(1:j-1)$ for R^{-1}

$U(1:j-1, j) = z$, $v(j:n) = i(j:n) - L(j:n, 1:j-1) \cdot z$

end

Determine μ with $j \leq \mu \leq n$ so $U(\mu)I_{n-\mu} = I_{n-\mu}$ and set $\text{piv}(j) = \mu$

$v(j) + v(\mu), L(j, 1:j-1) + L(\mu, 1:j-1), U(j, j) = v(j)$

if $v(j) = 0$

$L(j+1:n, j) = v(j+1:n)/v(j)$

end

end

As with Algorithm 3.4.1, this procedure requires $2n^3/3$ flops and $O(n^2)$ comparisons.

~~mhsh.b bxxkxbettfK eTetttb f elboxkjf eb dtaf kxb~~

We now examine the stability that is obtained with partial pivoting. This requires an accounting of the rounding errors that are sustained during elimination and during the triangular system solving. Bearing in mind that there are no rounding errors associated with permutation, it is not hard to show using Theorem 3.3.2 that the computed solution \hat{x} satisfies $(A + E)\hat{x} = b$ where

$$|E| \leq n\|u\| \left(2|A| + 4\hat{P}^T |\hat{L}| |\hat{U}| \right) + O(u^2). \quad (3.4.6)$$

Here we are assuming that P , L , and U are the computed analogs of P , L , and U as produced by the above algorithms. Pivoting implies that the elements of L are bounded by one. Thus $\|L\|_1 \leq n$ and we obtain the bound

$$\|E\| \leq n\|u\| \|A\| \|O\| + 4n\|L\| \|U\| + O(u^2). \quad (3.4.7)$$

The problem now is to bound $\|U\|$. Define the growth factor p by

$$p = \max_{i,j,k} \frac{|\hat{a}_{ij}^{(k)}|}{\|A\|_\infty} \quad (3.4.8)$$

where \mathbf{A}^k is the computed version of the matrix \mathbf{A} . $\mathbf{M}_k \mathbf{I}_{n-k} \cdots \mathbf{M}_1 \mathbf{I}_{n-1} \mathbf{A}$. It follows that

$$\|E\|_\infty \leq 6n^3 \rho \|A\|_\infty u + O(u^2). \quad (349)$$

Whether or not this compares favorably with the ideal bound (2.7.20) hinges upon the size of the growth factor of p . (The factor n^3 is not an operating factor in practice and may be ignored in this discussion.)

The growth factor measures how large the A -entries become during the process of elimination. Whether or not we regard Gaussian elimination with partial pivoting is safe to use depends upon what we can say about this quantity. From an average-case point of view experiments by Trefethen and Schreiber (1990) suggest that p is usually in the vicinity of $n^{2/3}$. However, from the worst-case point of view p can be a large as 2^n . In particular, if $A \in \mathbb{R}^{n,n}$ is defined by

$$a_{ij} = \begin{cases} 1 & f_i = \text{arj} = n, \\ -1 & f_i > j, \\ 0 & \text{otherwise} \end{cases}$$

then there is no swapping of rows during Gaussian elimination with partial pivoting. We emerge with $A = LU$ and it can be shown that $U \in \mathbb{R}^{n,n}$. For example,

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ -1 & 1 & 0 & 1 \\ -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & -1 & 1 & 0 \\ -1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}.$$

Understanding the behavior of p requires an intuition about what makes the U -factor large. Since $PA = LU$ implies $U = L^{-1}PA$ it would appear that the size of L^{-1} is relevant. However, Stewart (1997) discusses why one can expect the L -factor to be well conditioned.

Although there is still more to understand about p , the consensus is that serious element growth in Gaussian elimination with partial pivoting is extremely rare. The method can be used with confidence.

3.4.6 $V = z \mathbf{k}z$

Another pivot strategy called complete pivoting has the property that the associated growth factor bound is considerably smaller than 2^n . Recall that in partial pivoting the k th pivot is determined by scanning the current subcolumn $A(kn, k)$. In complete pivoting the largest entry in the current submatrix $A(kn, kn)$ is permuted into the (k, k) position. Thus we compute the upper triangularization

$$\mathbf{M}_k \mathbf{I}_{n-1} \cdots \mathbf{J}_f \mathbf{I}_1 \mathbf{A} \mathbf{f}_1 \cdots \mathbf{f}_{n-1} = \mathbf{U}.$$

At step k we are confronted with the matrix

$$\mathbf{A}^{(k-1)} = \mathbf{M}_k \mathbf{I}_{k-1} \cdots \mathbf{M}_1 \mathbf{I}_1 \mathbf{A} \mathbf{f}_1 \cdots \mathbf{f}_{k-1}$$

and determine interchange permutations I_k and r_k such that

$$\left| \left(\Pi_k A^{(k-1)} \Gamma_k \right)_{kk} \right| = \max_{k \leq i, j \leq n} \left| \left(\Pi_k A^{(k-1)} \Gamma_k \right)_{ij} \right|.$$

mekE.UHPt rknkn eC USt v1E3. If 14o TTh dP,(8 T8v. TEUtkat This algorithm computes the factorization $PAQ^T = LU$ where P is a permutation matrix encoded by $\text{piv}(l:n-1)$, Q is a permutation matrix encoded by $\text{cdpiv}(l:n-1)$, L is unit lower triangular with $\text{I} \leq i \leq l$, and U is upper triangular. For $i = l:n$, $A(i,i:n)$ is overwritten by $U(i,i:n)$ and $A(i+1:n,i)$ is overwritten by $L(i+1:n,i)$. The permutation P is given by $P = I_n | \dots | I_1$ where I_k is an interchange permutation obtained by swapping rows k and $\text{rowpiv}(k)$ of I_n . The permutation Q is given by $Q = r_{n-1} \dots r_i$ where r_k is an interchange permutation obtained by swapping rows k and $\text{cdpiv}(k)$ of I_n .

for $k = l:n-1$

Determine μ with k : $\mu \leq n$ and i with $k = i\lambda \leq n$ so

$$|A(\mu, :)| = \max\{|A(i,j)| : i = kn, j = kn\}$$

$$\text{rowpiv}(k) = \mu$$

$$A(k, l:n) + A(\mu, l:n)$$

$$\text{cdpiv}(k) = \lambda$$

$$A(l:n,k) + A(l:n,:)$$

if $A(k,k) = 0$

$$p = k+1:n$$

$$A(p,k) = A(p,k)/A(k,k)$$

$$A(p,p) = A(p,p) - A(p,k)A(k,p)$$

end

end

This algorithm requires $2n^3/3$ flops and $O(n^3)$ comparisons. Unlike partial pivoting complete pivoting involves a significant floating point arithmetic overhead because of the two-dimensional search at each stage.

With the factorization $PAQ^T = LU$ in hand the solution to $Ax = b$ proceeds as follows:

Step 1 Solve $Lz = Pb$ for z .

Step 2 Solve $Uy = z$ for y .

Step 3 Set $x = Q^T y$.

The rowpiv and cdpiv representations can be used to form Pb and Qy , respectively.

Wilkinson (1961) has shown that in exact arithmetic the elements of the matrix $A(k) = M_k | \dots | M_1$ for $1 \leq k \leq n$ satisfy

$$|a_{ij}^{(k)}| \leq k^{1/2} (2 \cdot 3^{1/2} \dots k^{1/k-1})^{1/2} \max |a_{ij}|. \quad (34.10)$$

The upper bound is a rather slow growing function of k . This fact coupled with very empirical evidence suggesting that p is always modestly sized (eg $p = 10$) permit us to conclude that Gaussian elimination with complete pivoting is stable. The method `solve` solves a nearly linear system $(A+E)x = b$ in the sense of (27.21). However, in general there is little

$r + 1, A(r+1:n, r+1:n) = 0$ This implies that $I_k = rk = M_k = I$ if $r \leq k = r + 1:n$ and so the algorithm can be terminated after step r with the following factorization in band

$$P A Q \Gamma = L U = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{bmatrix}.$$

Here, L_{11} and U_{11} are r -by- r and L_{21} and U_{22} are $(n - r)$ -by- r . Thus, Gaussian elimination with complete pivoting can in principle be used to determine the rank of a matrix. Nevertheless, roundoff errors make the probability of encountering an exactly zero pivot remote. In practice one would have to "declare" A to have rank k if the pivot element in step $k+1$ was sufficiently small. The numerical rank determination problem is discussed in detail in §5.5.

3.4.1 | | | k

third type of LU stabilization strategy called rook pivoting provides an interesting alternative to partial pivoting and complete pivoting. As with complete pivoting it computes the factorization $PAQ = LU$. However, instead of choosing a pivot the largest value in $|A(kn, kn)|$, it searches for an element of that submatrix that is maximal in both its row and column. Thus, if

$$A(kn, kn) = \begin{bmatrix} 24 & 36 & 13 & 61 \\ 42 & 67 & 72 & 50 \\ 38 & 11 & 36 & 43 \\ 52 & 37 & 48 & 16 \end{bmatrix},$$

then "72" would be identified by complete pivoting while "52", "72" or "61" would be acceptable with the rook pivoting strategy. To implement rook pivoting the scan-andswap portion of Algorithm 34.3 is changed to

```

 $\mu = k, \lambda = k, s = |a_{\mu\lambda}|, s = 0$ 
while  $\exists i \in \mathbb{I}(A(kn, .)) \text{ and } v \in \mathbb{A}^C \setminus \mathbb{I}(A(\mu, kn))$  do
    if  $\text{mod}(s, 2) = 0$ 
        Update  $\mu$  so that  $|a_{\mu\lambda}| = \mathbb{I}(A(kn, .))$  do with  $k \leq \mu \leq n$ 
    else
        Update  $\lambda$  so that  $|a_{\mu\lambda}| = \mathbb{I}(A(\mu, kn))$  do with  $k \leq \lambda \leq n$ 
    end
     $s = s + 1$ 
end
rowpiv(k) =  $\mu, A(k, :) + A(\mu, :) \text{ colpiv}(k) = ., A(:, k) + A(:, .)$ 
```

The search for a larger $|a_{\mu\lambda}|$ involves alternate scans of $A(kn, .)$ and $A(\mu, kn)$. The value of i is monotone increasing and that ensures termination of the while-loop. In theory, the exit value of s could be $O(n - k)^2$, but in practice its value is $O(1)$. Chang (2009). The bottom line is that rook pivoting represents the same $O(n^3)$ overhead as partial pivoting but that it induces the same level of reliability as complete pivoting.

Impractical to invert A directly.

If $A \in \mathbb{R}^{m \times n}$ with $m < n$, $\text{rank}(A) = m$, and $b \in \mathbb{R}^m$, then the linear system $Ax = b$ is said to be under determined. Note that in this case there are an infinite number of solutions. With either complete or rook pivoting it is possible to compute an LU factorization of the form

$$PAQ^T = L [U_1 | U_2] \quad (34.1)$$

where P and Q are permutations, $L \in \mathbb{R}^{m \times m}$ is unit lower triangular, and $U_1 \in \mathbb{R}^{m \times m}$ is nonsingular and upper triangular. Note that

$$Ax = b \Leftrightarrow (PAQ^T)(Qx) = (Pb) \Leftrightarrow L [U_1 | U_2] \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = L(U_1 z_1 + U_2 z_2) = c$$

where $c = Pb$ and

$$\begin{bmatrix} ; \\ ; \end{bmatrix} = Qx$$

This suggests the following solution procedure:

Step 1 Solve $Ly = Pb$ for $y \in \mathbb{R}^m$.

Step 2 Choose $z_2 \in \mathbb{R}^{n-m}$ and solve $U_2 z_2 = y - U_1 z_1$.

Step 3 Set

$$x = Q^T \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}.$$

Setting $z_2 = 0$ is a natural choice. We have more to say about underdetermined systems in §5.6.2.

msiib 7\\$b IRb h\\$t tPTf Kb

We offer three examples that illustrate how to think in terms of the LU factorization when confronted with a linear equation situation.

Example 1 Suppose A is nonsingular and n -by- n and that B is n -by- p . Consider the problem of finding X (n -by- p) so $AX = B$. This is the multiple right hand side problem. If $X = [X_1 | \dots | X_p]$ and $B = [b_1 | \dots | b_p]$ are column partitions, then

Compute $PA = LU$

for $k = 1:p$

Solve $Ly = Pb_k$ and then $UX_k = y$. (34.12)

end

If $B = I_n$ then we emerge with an approximation to A^{-1} .

Example 2 Suppose we want to overwrite b with the solution to $A^k x = b$ where $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$, and k is a positive integer. One approach is to compute $C = A^k$ and then solve $Cx = b$. However, the matrix multiplications can be avoided altogether.

Compute $\mathbf{PA} = \mathbf{LU}$.

for j = l:k

Overwrite b with the solution to $Ly = Pb$

(34 13)

Overwrite \mathbf{b} with the solution to $\mathbf{U}\mathbf{x} = \mathbf{b}$

end

As in Example 1, the idea is to get the LU factorization "outside the loop."

Example 3 Suppose we are given $A \in \mathbb{R}^{m \times n}$, $d \in \mathbb{R}^n$, and $c \in \mathbb{R}^m$ and that we want to compute $s = c^T A^{-1} d$. One approach is to compute $X = A^{-1}d$ discussed in (i) and then compute $s = c^T X d$. However, it is more economical to proceed as follows.

Compute $\mathbf{PA} = \mathbf{LU}$.

Solve Ly = Pd and then Ux = y.

$$S = \overline{Tx}$$

An " A^{-1} " in a formula almost always means "solve a linear system" and almost never means "compute A^{-1} ".

3.4.10 Th wz k u z i z \bar{w} r

We are now in possession of a very important and well-understood algorithm (Gaussian elimination) for a very important and well-understood problem (linear equations). Let us take advantage of our position and formulate more abstractly what we mean by "problem sensitivity" and "algorithm stability." Our discussion follows Higham (ASNA, §15.16), Stewart (MA, §43), and Trefethen and Bau (NLA, Lectures 12, 14, 15 and 22).

A problem instance is from "data/input space" D to "solution/output space" S . A problem instance is f together with a particular $d \in D$. We assume D and S are normed vector spaces. For linear systems, D is the set of matrix-vector pairs (A, b) where $A \in \mathbb{R}^{n \times n}$ is nonsingular and $b \in \mathbb{R}^n$. The function f maps (A, b) to $A^{-1}b$, a element of S . For a particular A and b , $Ax = b$ is a problem instance.

A perturbation theory for the problem sheds light on the difference between $f(d)$ and $f(d + Ad)$ where $d \in D$ and $d + Ad \in D$. For linear systems, we discussed in §26 the difference between the solution to $Ax = b$ and the solution to $(A + AA)x = (b + Ab)$. We bounded $\|Ax - b\|$ in terms of $\|AA\| \|A\|$ and $\|Ab\| \|b\|$.

The conditioning of a problem refers to the behavior of f under perturbation at d . A condition number of a problem quantifies the rate of change of the solution with respect to the input data. If small changes in d induce relatively large changes in $f(d)$, then that problem instance is ill-conditioned. If small changes in d do not induce relatively large changes in $f(d)$, then that problem instance is well-conditioned. Definitions for "small" and "large" are required. For linear systems we showed in §2.6 that the magnitude of the condition number $K(A) = \|A\| \|A^{-1}\|$ determines whether an $Ax = b$ problem is ill-conditioned or well-conditioned. One might say that a linear equation problem is well-conditioned if $K(A) = O(1)$ and ill-conditioned if $\kappa(A) \approx O(1/\epsilon)$.

An algorithm for computing $f(d)$ produces an approximation $\hat{f}(d)$. Depending on the situation, it may be necessary to identify a particular software implementation

of the underlying method. The function for Gaussian elimination with partial pivoting, Gaussian elimination with rock pivoting and Gaussian elimination with complete pivoting are all different.

An algorithm for computing $f(d)$ is stable if for some small Δd , the computed solution $j(d)$ is close to $j(d + \Delta d)$. A stable algorithm nearly solves a nearby problem. An algorithm for computing $f(d)$ is backward stable if for some small Δd , the computed solution $j(d)$ satisfies $j(d) = f(d + \Delta d)$. A backward stable algorithm exactly solves a nearby problem. Applied to a given linear system $Ax = b$, Gaussian elimination with complete pivoting is backward stable because the computed solution x satisfies

$$(\mathbf{A} + \mathbf{A})\tilde{\mathbf{x}} = \mathbf{b}$$

and $\|A\| \approx O(u)$. On the other hand, if b is specified by a matrix-vector product $b = Mv$, then

$$(A + A)x = Mv + g$$

where $\|A^{-1}A\| \approx O(u)$ and $8(\|M\| \|v\|) \approx O(u)$. Here the underlying f is defined by $f:(A,M,v) \mapsto A^{-1}(Mv)$. In this case the algorithm is stable but not backward stable.

Problems

P3.4.1 $5N_A = LU \alpha t, l, ritlp6T itMlnn-swn A 86t, |\ell_{ij}| \leq 1.89a_i^T -), u_i^T - (T)). k_$
 $)_k + T)A, inmU, pAA, t6.s. I, pM9, Jxit6ln$

$$u_i^T = a_i^T - \sum_{j=1}^{i-1} \ell_{ij} u_j^T$$

$$I(= \quad , \quad) \times , \quad (\quad \quad \| \Psi \|_{\infty} \leq 2 \operatorname{Tr} \| A \|_{\infty} \quad (\quad , \quad) \quad \quad) \geq , \quad -), \quad 9,j \quad , \quad , \quad) \quad ()$$

P3.4.2 .386 .3. .1 PAQ = $45x = r \times 0x20 = 1 + x \times 100x \times x \times 4 = C1x + 1x + x \cdot u \cdot x$
 $x \not\in C[14, 1x] \cup \{0\} \cup \{1\}$ **Maple**, **MATLAB**, **x**, **t,in** [u[i],]_k^-, +) . $\frac{x^T}{x^T}(j))$ (1)

P3.4.38 .NA ERnxn/k R) LU ritlpMT itNflnt,it inmu ip.enl8nzIM., in i,ilpMt,5
86, .in .1A,t, t, {i,j} ,ntp. lr A- 1 MirAApl.M lit,s.(n- j)²+ (n - i)² IAAz

P3.4.4 .2..8 .N ..3N8..e.N. ...N..N8...N. .r. (34 12.) 29R)= 11E , (4x1E || AX - I || E

P3.4.5 Pp.N. 1..8.r.3. 343=(n/RnTiaR) Ex;a9 n/9FTn(E YRnT(B4 11). I(L HR)1 (E 9
EG 4TEK)

Notes and References for §3.4

..N..8..N.N. 6..3 N.N2N..86.3r.8... r...2.N

31f§ 3.N.1988. @r T2(FY9))R,=- R(IIT)RnT(zT Numer. Math. 12, 335-345
 U,E,+9Tx{1971}. @l I₉() n⁹+nR,TIT(F)R4=TR(1-T-)R₀PT!!I st. Math. Applic. 8,
 34-35

1971). ⑧ (Th(EL))ln/945ETaR₁+R₋=T(R), <-- R(1-TiBT)PTNumer. Math. 16, 369-361.

12 - , (1974). @l 1n9 () r - 2(n- Y9)R == TR(T-R)fPTLin. Alg. Applic. 8, 361-68
 1 - (E = TRR)x UAE+9Tx(1974). ® () - n(ET)ln9' + TR, fln(Fn9' - Rv1 = RIRan(E YRn)T_a
 a = O₀ = Nln6." Math. 22, 183-186

U) iR1 Rx1 - r 99E=0{1988. ®E(Lr/-))R,=- R)(IT)RrT)arAmer. Math. Monthly 95, 489-513

1 U.IITRT R)xi.U- ITIRT { 1989. ®8RE_E(Ln' iRan(E-T)R,==TR)-T)Rn 0 LTHM2(nP
SIAM J. Matr. Anal. Applic. 10 155 164
81 ' 9n9 R)x +F+a/E9T,9 1990. ®2ER P,9 +nR,T-(E)R,== - R(IITTRR)P SIRM
J. Matr. Anal. Applic. 11 335 350

N. ' = < 0 0 220 Di = 3 0e1.000) = < 0X020 = e Xf < p0 > 0 = ySIAM J. Matr Anal. Applic. 12, Er / 56
 rxns lin. MN6ci,c l1A, ctc PM.ltMlinJct, pc vp ii, AAMiRsM1M1M1Ac3Mathe-
 matica J. 2, r uM.
 1CpMi, (tE6'A e = < n E p0 = S AenKeOE1.000) = < 0X020 = p = z 0A = = 0 = y
 Oe e =)S SIAM J. Sci. Stat. Comput. 14, NEM15y.
 L.V. R = on2210 D 1.000) = A0X020 = p = = 0 = < p0 > 00 = yCE = SIAM J.
 Matr x Anal. Applic. 15, ME/MEuNI
 x1n1 in inmu6 riA.ip,n,T , Mtrr: c1n t,c l1Asctc PM.ltMlinJct, pc vp i simi lipm
 itpMr lpmgMN13T Multilin Alg. 38 M5Myr.
 C 11Pcni, Miti. cPM.MnlApiAcimAmn hli,s 9l, nmABt,c RpplpAp)cptiM1Mn, ip
 Atcl ASMA J. Numer. Anal. 16 M/ (5Mr
 J.L.))= < =)303 : e 0 220 D 1 = 3 0e1.000) = < 0X09i = e = yA =)>OEn0 = e.
 + K1=SIAM J. Matr Anal. Applic. 19, yII 5(r 1
 ' 1 i, tM3 lcln.MM13M46riptMncN 1116c In A5G, AtncAlA ii, AAMiR- M1M18Mt,
 ' iptMSMAMnB1r 4Q uNf , E.

A tntMncnAcAMFL- 1 TEBB2R)(n/BELn/ FH(FW/n=RT=IT1ErR)m(/R2R)
 3ErRxT1 (FUEIR)h=REnEh)xTUT(P=BBE
 D. = 0o3 =)= 0S 0 E 0+ n = 0220 De = 00z 0S n = j = 0= KO = y.< A = = OEn o = M
 SIAM J. Matrix Anal. Applic. 19, r u' M.
 m, .lnnctMhGcA8coAis, AM.ltAnmcipAMni8siM1M1Mc8dinf
 m1.),in I(ty16 c1n t,R.MAtcniM)l1A8tiAMh1B1, i .AlpMTitSIAMhli,s PM.ltG r
 Math Comput. 42, r Er y /
 M.ltAApiAdit 8c m1MmMA,8MA Airwise pivoting, SrA,MAApI,3Nr G,1DNA tpInA
 I5tMlinA,Acntl TcpA,c,l8cp ApMeni,sip lA1pM1B1Ba/TjnE=R1BR=T1) aBnRT
 =nTE(aB)2T(E(TB)n;Ba,=B)11 RxRa2M1L=REA(T,T)Bx T)BRa=rBP-BBE
 D. G = n = 0220Dy =)< eop0 = 30p0 = 201.900) = A0X0=IEEE-T -n . Comput.
 C-34, NI/Ni.

p, sitcnAclr AM, IAM1Pm tour amnting t,iA M1M1tcpM1mMAtpM1G1pm J1
 tM18,M1mM1E6u1p., mMmA, AAMiRle AM.ltM1mM1Ap1AcptM1A8
 L.V. R = q0 220 D" e1k = 3ReE = Q a EOn =)&eo0) = A0X0= 30=je = p8>=0= y = N
 J. Comput. Appl. Math. 86, (II 5Mt/.
 m11c inmnsaci, ,N 1116ci,c 51 ler AM.ltM1apitci,38 Comput. Appl. Math. 123, Er E6t1
 Q,ini IN 11Nch1lc .cit ,pcAlr ii, AAMiRsM1M18Mt, 51EMMn1B1r42, uufyE.

35 Improving and Estimating Accuracy

Suppose we apply Gaussian elimination with partial pivoting to the n -by- n system $Ax = b$ that IEEE double precision arithmetic is used. Equation (349) essentially says that if the growth factor is modest then the computed solution \hat{x} satisfies

$$(A + E)\hat{x} = b \quad \|E\|_\infty \approx \|A\|_\infty. \quad (35.1)$$

In this section we explore the practical ramifications of this result. We begin by stressing the distinction that should be made between residual size and accuracy. This is followed by a discussion of scaling, iterative improvement, and condition estimation. [Higham \(ASNA\)](#) for a more detailed treatment of these topics.

We make two cautionary remarks at the outset. The infinity norm is used throughout since it is very handy in roundoff error analysis and in practical error estimation. Second, whenever we refer to "Gaussian elimination" in this section we really mean Gaussian elimination with some stabilizing pivot strategy such as partial pivoting.

1. If $\|b - Ax\|_\infty \leq \epsilon \|A\|_\infty \|x\|_\infty$

The residual of a computed solution x to the linear system $Ax = b$ is the vector $b - Ax$. A small residual means that Ax effectively "predicts" the right hand side b . From Equation 3.5.1 we have $\|b - Ax\|_\infty \leq \|A\|_\infty \|x\|_\infty$ and so we obtain

Heuristic I. Gaussian elimination produces a solution x with a relatively small residual.

Small residuals do not imply high accuracy. Combining Theorem 2.6.2 and (3.5.1), we see that

$$\frac{\|x - \bar{x}\|_\infty}{\|x\|_\infty} \leq u \kappa_\infty(A) m \quad (3.5.2)$$

This justifies a second guiding principle:

Heuristic II. If the unit roundoff condition satisfies $u < 10^{-d}$ and $10(A) < 10^d$, then Gaussian elimination produces a solution x that has about $d - q$ correct decimal digits.

If $u 10(A)$ is large, then we say that A is ill-conditioned with respect to the machine precision.

As an illustration of the Heuristics I and II, consider the system

$$\begin{bmatrix} .986 & .579 \\ .409 & .237 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} .235 \\ .107 \end{bmatrix}$$

in which $10(A) = 700$ and $x = [2, -3]^T$. Here is what we find for various machine precisions:

u	x_1	x_2	$\frac{\ x - \bar{x}\ _\infty}{\ x\ _\infty}$	$\frac{\ b - Ax\ _\infty}{\ A\ _\infty \ x\ _\infty}$
10^{-3}	2.11	-3.17	$5 \cdot 10^{-2}$	$20 \cdot 10^{-3}$
10^{-4}	1.986	-2.975	$8 \cdot 10^{-3}$	$15 \cdot 10^{-4}$
10^{-5}	2.0019	-3.0082	$1 \cdot 10^{-3}$	$21 \cdot 10^{-6}$
10^{-6}	2.00025	-3.00094	$3 \cdot 10^{-4}$	$42 \cdot 10^{-7}$

Whether or not to be content with the computed solution x depends on the requirements of the underlying source problem. In many applications accuracy is not important but small residuals are. In such a situation, the x produced by Gaussian elimination is probably adequate. On the other hand, if the number of correct digits in x is an issue, then the situation is more complicated and the discussion in the remainder of this section is relevant.

mm1b uatPTt"b

Let $/$ be the machine base (typically $/ = 2$) and define the diagonal matrices $D1 = \text{diag}(1, D_1, 1^T)$ and $D2 = \text{diag}(c, \dots, 1^C)$. The solution to the n -by- n linear system $Ax = b$ can be found by solving the scaled system $(D1AD2)y = D1b$ using

Gaussian elimination and then setting $x = D_2^{-1}y$. The scalings of A , b and y require only $O(n^2)$ flops and may be accomplished without roundoff. Note that D_1 scales equations and D_2 scales unknowns.

It follows from Heuristic II that if \hat{x} and \hat{y} are the computed versions of x and y , then

$$\frac{\|D_2^{-1}(\hat{x} - x)\|_\infty}{\|D_2^{-1}x\|_\infty} = \frac{\|\hat{y} - y\|_\infty}{\|y\|_\infty} \approx u\kappa_\infty(D_1^{-1}AD_2). \quad (353)$$

Thus, if $KO(D_1^{-1}AD_2)$ can be made considerably smaller than $KO(A)$, then we might expect a correspondingly more accurate \hat{x} , provided errors are measured in the "D2 norm defined by $\|z\|_{D_2} = \|D_2^{-1}z\|_\infty$. This is the objective of scaling. Note that it encompasses two issues: the condition of the scaled problem and the appropriateness of appraising error in the D2 norm.

An interesting but very difficult mathematical problem concerns the exact minimization of $Kp(D_1^{-1}AD_2)$ for general diagonal D_1 and various p . Such results as there are in this direction are not very practical. This is hardly discouraging; however, when we recall that (353) is a heuristic result, it makes little sense to minimize exactly a heuristic bound. What we seek is a fast, approximate method for improving the quality of the computed solution \hat{x} .

One technique of this variety is simple row scaling. In this scheme D_2 is the identity and D_1 is chosen so that each row in $D_1^{-1}A$ has approximately the same ∞ -norm. Row scaling reduces the likelihood of adding a very small number to a very large number during elimination—an event that can greatly diminish accuracy.

Slightly more complicated than simple row scaling is row-column equilibration. Here the object is to choose D_1 and D_2 so that the ∞ -norm of each row and column of $D_1^{-1}AD_2$ belongs to the interval $[1/\beta, \beta]$ where β is the base of the floating point system. For work along these lines see McKeeman (1962).

It cannot be stressed too much that simple row scaling and row-column equilibration do not "solve" the scaling problem. Indeed, either technique can render a worse solution than if no scaling whatever is used. The ramifications of this point are thoroughly discussed in Forsythe and Moler (SLE, Chap. 11). The basic recommendation is that the scaling of equations and unknowns must proceed on a problem-by-problem basis. General scaling strategies are unreliable. It is best to scale (if at all) on the basis of what the source problem proclaims about the significance of each a_{ij} . Measurement units and data error may have to be considered.

3.5.3 $f z \rightarrow f z - z$

Suppose $Ax = b$ has been solved via the partial pivoting factorization $PA = LU$ and that we wish to improve the accuracy of the computed solution x . If we execute

$$\begin{aligned} r &= b - Ax \\ \text{Solve } Ly &= Pr. \\ \text{Solve } Uz &= y. \\ x_{\text{new}} &= x + z \end{aligned} \quad (354)$$

then in exact arithmetic $Ax_{\text{new}} = Ax + Az = (b - r) + r = b$. Unfortunately, the naive floating point execution of these formulae renders an x_{new} that is no more accurate

than x . This is to be expected since $f = f(b - Ax)$ has $f \ll w$ if any correct significant digits. (Recall Heuristic I.) Consequently, $\|A^{-1}\|_F \|A^{-1}f\|_F / \|f\|_F$ noise/mise is a very poor correction from the standpoint of improving the accuracy of x . However, Skeel (1980) has an error analysis that indicates when (354) gives an improved x_{new} from the standpoint of hardware or or. In particular, if the quantity

$$\tau = (\| |A| |A^{-1}| \|_\infty) \left(\max_i (|A||x|)_i / \min_i (|A||x|)_i \right)$$

is not too big then (354) produces an x_{new} such that $(A + E)x_{\text{new}} = b$ for very small E . Of course, if Gaussian elimination with partial pivoting is used, then the computed x already solves a nearby system. However, this may not be the case for certain pivot strategies used to preserve sparsity. In this situation, the fixed precision iterative improvement step (354) can be worthwhile and cheap. See Aridi, Demmel, and Duff (1988).

In general, for (354) to produce a more accurate x , it is necessary to compute the residual $b - Ax$ with extended precision floating point arithmetic. Typically, this means that if t -digit arithmetic is used to compute $PA = LU$, X , y , and Z , then $2t$ -digit arithmetic is used to form $b - Ax$. The process can be iterated. In particular, once we have computed $PA = LU$ and initialize $X = Q$ we repeat the following

$$\begin{aligned} r &= b - Ax \quad (\text{higher precision}) \\ \text{Solve } Ly &= Pr \text{ for } y \text{ and } Uz &= y \text{ for } z \\ X &= X + Z \end{aligned} \tag{355}$$

We refer to this process as mixed precision iterative improvement. The original A must be used in the high-precision computation of r . The basic result concerning the performance of (355) is summarized in the following heuristic.

Heuristic III. If the machine precision and conditions satisfy $u = 10^{-d}$ and $\kappa_\infty(A) \leq 10^q$, then after k executions of (355), x has approximately $\min\{dk(d-q), d\}$ correct digits if the residual computation is performed with precision u^2 .

Roughly speaking if $\kappa_\infty(A) \geq 1$, then iterative improvement can ultimately produce a solution that is correct to full (single) precision. Note that the process is relatively cheap. Each improvement costs $O(n^2)$, to be compared with the original $O(n^3)$ investment in the factorization $PA = LU$. Of course, no improvement may result if A is badly conditioned with respect to the machine precision.

mmvsl ktnf 1tb bef 1tf Tkb

Suppose that we have solved $Ax = b$ via $PA = LU$ and that we now wish to obtain the number of correct digits in the computed solution x . It follows from Heuristic II that in order to do this we need an estimate of the condition $\kappa_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty$. Computing $\|A\|_\infty$ poses no problem as we merely use the $O(n^2)$ formula (23.10). The challenge is with respect to the factor $\|A^{-1}\|_\infty$. Conceivably, we could estimate this quantity by $\|\hat{X}\|_\infty$, where $\hat{X} = [X_1 | \cdots | X_p]$ and X is the computed solution to $Ax = e$. (See §34.9.) The trouble with this approach is its expense: $P0 = \|A\|_\infty \|\hat{X}\|_\infty$ costs about three times as much as x .

The central problem of condition estimation is how to estimate reliably the condition number in $O(n^2)$ if one is summing the availability of PA = LU or one of the factorizations that are presented in subsequent chapters. An approach described in Forsythe and Moler (SLE, p. 51) is based on iterative improvement and the heuristic

$$\text{u}\kappa_{\infty}(A) = \|Iz\|_{\infty}/\|X\|_{\infty}$$

where z is the first correction of w in (355).

Cline, Moler, Stewart, and Wilkinson (1979) propose an approach to the condition estimation problem that is based on the implication

$$Ay = d \quad : \quad \|A^{-1}\|_{\infty} = \|y\|_{\infty}/\|d\|_{\infty}.$$

The idea behind their estimator is to choose d so that the solution y is large in norm and then set

$$\hat{\kappa}_{\infty} = \|A\|_{\infty} = \|y\|_{\infty}/\|d\|_{\infty}.$$

The success of this method hinges on how close the ratio $\|y\|_{\infty}/\|d\|_{\infty}$ is to its maximum value $\|A^{-1}\|_{\infty}$.

Consider the case when $A = T$ is upper triangular. The relation between d and y is completely specified by the following column version of back substitution:

$$p(l:n) = 0$$

for $k = n - 1:1$

Choose $d(k)$.

$$y(k) = (d(k) - p(k))/T(k,k) \quad (356)$$

$$p(l:k-1) = p(l:k-1) + y(k)T(l:k-1,k)$$

end

Normally, we use this algorithm to solve a given triangular system $Ty = d$. However, in the condition estimation setting we are free to pick the right-hand side d subject to the "constraint" that y is large relative to d .

One way to encourage growth in y is to choose $d(k)$ from the set $\{-1, +1\}$ so as to maximize $y(k)$. If $p(k) = 0$, then set $d(k) = -1$. If $p(k) \neq 0$ then set $d(k) = +1$. In other words, (356) is invoked with $d(k) = \pm \text{sign}(p(k))$. Overall, the vector d has the form $d(l:n) = [\pm 1, \dots, \pm 1]^T$. Since this is a unit vector, we obtain the estimate $\epsilon_1 = \|T\|_{\infty} = \|y\|_{\infty}$.

more reliable estimator results if $d(k) \in \{-1, +1\}$ is chosen so as to encourage growth both in $y(k)$ and the running sum update $p(l:k-1,k) + T(l:k-1,k)y(k)$. In particular, at step k we compute

$$y(k)^+ = (1 - p(k))/T(k,k),$$

$$s(k)^+ = |y(k)^+| + \|p(l:k-1) + T(l:k-1,k)y(k)^+\|_1,$$

$$y(k)^- = (-1 - p(k))/T(k,k),$$

$$s(k)^- = |y(k)^-| + \|p(l:k-1) + T(l:k-1,k)y(k)^-\|_1,$$

and set

$$y(k) = \begin{cases} y(k)^+ & \text{if } s(k)^+ \geq s(k)^-, \\ y(k)^- & \text{if } s(k)^+ < s(k)^-. \end{cases}$$

This gives the following procedure

Algorithm 3.5.14 Let $T \in \mathbb{R}^{n \times n}$ be a nonsingular upper triangular matrix. This algorithm computes unit \mathbf{c} -norm y and a scalar κ so $\|Ty\|_\infty = 1/\|T^{-1}\|_\infty$ and $\kappa = \kappa_\infty(T)$

```

 $p(1:n) = 0$ 
for  $k = n : -1 : 1$ 
     $y(k)^+ = (1 - p(k))/T(k, k)$ 
     $y(k)^- = (-1 - p(k))/T(k, k)$ 
     $p(k)^+ = p(1:k - 1) + T(1:k - 1, k)y(k)^+$ 
     $p(k)^- = p(1:k - 1) + T(1:k - 1, k)y(k)^-$ 
    if  $|y(k)^+| + \|p(k)^+\|_1 \geq |y(k)^-| + \|p(k)^-\|_1$ 
         $y(k) = y(k)^+$ 
         $p(1:k - 1) = p(k)^+$ 
    else
         $y(k) = y(k)^-$ 
         $p(1:k - 1) = p(k)^-$ 
    end
end
 $\kappa = \|y\|_\infty \|T\|_\infty$ 
 $y = y/\|y\|_\infty$ 

```

The algorithm involves several times the work of ordinary back substitution.

We are now in a position to describe a procedure for estimating the condition of a square nonsingular matrix A whose $PA = LU$ factorization is available.

Step 1 Apply the lower triangular version of Algorithm 3.5.1 to UT and obtain a large norm solution to $U^T y = \mathbf{d}$.

Step 2 Solve the triangular systems $L^T r = y$, $Lw = Pr$, and $Uz = w$.

Step 3 Set $\hat{\kappa}_\infty = \|A\|_\infty \|z\|_\infty / \|r\|_\infty$.

Note that $\|z\|_\infty = \|A^{-1}\|_\infty \|r\|_\infty$. The method is based on several heuristics. First, if A is ill-conditioned and $PA = LU$, then it is usually the case that U is correspondingly ill-conditioned. The lower triangle L tends to be fairly well-conditioned. Thus, it is more profitable to apply the condition estimator to U than to L . The vector r , because it solves $A^T P^T r = \mathbf{d}$, tends to be rich in the direction of the left singular vector associated with $\sigma_{\min}(A)$. Right-hand sides with this property render large solutions to the problem $Az = r$.

In practice, it is found that the condition estimation technique that we have outlined produces adequate order-of-magnitude estimates of the true condition number.

Problems

P3.5.1 .38fo P.2.s8 2382. 88.8 23. 8.8 fo. 288r2.s. ..282. 2.P

P3.5.28.8 R 5 fo38.8 .. .8.222..1.6 .. ogfo2... .o.fo.23_{i,j} | loo(-
 i pfFOE EoA =OAL.ä End (A), || | _{k_i^T})_i^T) j_i^T) (- ? _{j_i^T}) (- _{k_i^T})
 a = = x nx*x xn* = J8nx2*x x=x-x -4nx8x xn x n x=x-x End x=nxx-a x x

$q = w_{\text{A}} \cdot h \cdot W \cdot n \cdot x = x \cdot n \cdot x \cdot m \cdot x = \frac{1}{2} x \cdot W \cdot n \cdot A$

P3.5.3 R .3.R.88538. .282 fo38.8

$$B = \left[\begin{array}{c} \left(\begin{bmatrix} 1 & & & \\ & R & & \\ & 0 & T & \\ \hline & \begin{bmatrix} R & & \\ 0 & T & \\ \hline 0 & 0 & T \end{bmatrix} & \begin{bmatrix} R & & \\ 0 & T & \\ \hline 0 & 0 & T \end{bmatrix}^T \end{bmatrix} \right) \end{array} \right]$$

ha w .P.+=- :w5 .T. 6X/S/VA_QD I XE92, W)AW Wl mW i
 $10^{-7}|x|.$ $k^i l^- \infty, j^i -k(T_i T_j T_k T_l)(T_i T_j T_k T_l) \ell^{(-?,-T)} \cdot \frac{T_i}{T_j} \left(\frac{T_k}{T_l}\right) \left(\frac{T_l}{T_i}\right) \left(\frac{T_j}{T_k}\right) \left(\frac{T_i}{T_j}\right) ?$
 $= .w5 W20 x a, x = R \frac{1}{4} lx xx \cdot xx a \frac{1}{4} ka - 8 ka \frac{1}{4} as W, a = 8n xxm$
 $xR = k x x R ' finn W n +W 4$

P3.5.4 38...f8. 23g2. 2..por

$$T = \begin{bmatrix} 1 & M & -M \\ 0 & \cdot & -M \\ 0 & &) \\ 0 & & A \end{bmatrix} \text{MER.}$$

8/ IA=IQA So 6XT=kSxaAx - /A) 612.257215 QAxTV c 690= xR-ax ln =x fm x 4= doe i.W E O± reKx@ G pEan@e eO= ip M.p n

P3.5.5 f3.2 .18. s 8..29P§8.2.8 fo38...s.8. 21232.2.P Bnft2AT) c 25.X:

Notes and References for §3.5

38 os8fa..8.. .8 .8..8..8f fo..3238..s.. 8> ..8 o82.

N ..8.R.895r2fo.2.os ...o8. N....8. 67N mer. Math 5.Ef y..

P.A., p. 00 = ~~W~~ p.DAa = LePBea, 0 if Xa0.0Ea ≠ 00mer. Math 12, E6uf 6y.

A. $v = \frac{x}{t} = \frac{21\text{ m}}{0.5\text{ s}} = 42\text{ m/s}$ J. $x = v_0 t = 42 \cdot 2 = 84\text{ m}$

A. v. = $\frac{d}{dt} \cdot \text{Current} = \frac{dx}{dt} - J_{\text{ext}} \cdot x(t) + x \cdot R(t) + 2 \cdot \Theta(t)$

Math 15.6 yu. On May 2nd, $X_1 = 00$, $(-A) = \text{WFOOL Number Anal. 10}$

s. ilVenmr. Iepe, 7Mfd. cIn JepetapMT eSlt,a aA. the8Mh- e reIM.3r r

R. OILOMP + 2DDEa sOn WO = (E =)40 (h = .0001aX0 = a0 N = ACM 26 6t6f r. iu.

v. .a .a 1= Ow -Oca. GM20 pDf Oo=a 0+EO(n.0-CG6e 0Xa = OEdf. 0-sMo
J. Matr. Anal. Applic. 16 it. Et.

Pa C= COaE..O-Ea E= WEO = COOE = -a- = -O-E-E= XCOEθ=iEw n
 u .O= C(WEnCo)(Gm) .O= >OE= fCOOffQa EO.

w. Ta eMOpPn.N= (S0= QW04= Canadian Math. Bull. 9, r f yMA

For $a = -q$, $R = J_k = \text{diag}(-x, -x, -x, -x) = -\text{diag}(x, x, x, x) = -\text{diag}(x)$.

C.B. 1 = .EM02Dx= 0j@1=0X00R.Ca XCO=MACM 14 EM01

M. r1=1010a ≠ 0L=b 0a l=M10 2P x= 0j 01=0X00400o n.X0=(Fa 400F .= N
17, EEfNM.

1.D. O 0011p 0W= 001 = 0X00 X00 .X0= 0E): a 461a pool0X0= 4=mathN
Comput 35 yMyENs

C. M. 7Mid. c.apeS5a,na lan. T pMnai pG.a.lAnB(B) 31MA J. Numer. Anal.
17,6uf r.t

$\mathbf{L} \mathbf{q} = (\quad) \quad e = (\quad 0) \quad (\quad) \quad m$

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & \ddots & \vdots \\ A_{N1} & \cdots & A_{NN} \end{bmatrix} \quad A_{i,j} \in \mathbb{R}^{r \times r}.$$

The first step starts by applying scalar Gaussian elimination with partial pivoting to the first block column. Using an obvious rectangular matrix version of Algorithm 3.41 we obtain the following factorization

$$P_1 \begin{bmatrix} A_{11} \\ A_{21} \\ \vdots \\ A_{N1} \end{bmatrix} = \begin{bmatrix} L_{11} \\ L_{21} \\ \vdots \\ L_{N1} \end{bmatrix} U_{11}. \quad (362)$$

In this equation, $P_1 E J_{NN}$ is a permutation, $L U E W X^T$ is unit lower triangular, and $U E W X^T$ is upper triangular.

The next task is to compute the first block row of \mathbf{U} . To do this we set

$$P_1 A = \begin{bmatrix} \tilde{A}_{11} & \cdots & \tilde{A}_{1N} \\ \vdots & \ddots & \vdots \\ \tilde{A}_{N1} & \cdots & \tilde{A}_{NN} \end{bmatrix}, \quad \dots \text{ in } \mathbf{E} \mathbf{W} \mathbf{X}^T \quad (363)$$

and solve the lower triangular multiple right-hand-side problem

$$\mathbf{L} \mathbf{u} [\mathbf{U}_2 \dots \mathbf{U}_N] = [\mathbf{A}_2 \dots \mathbf{J}_N] \quad (364)$$

for $\mathbf{U}_2 \dots \mathbf{U}_N \mathbf{E} \mathbf{W} \mathbf{X}^T$. At this stage it is easy to show that we have the partial factorization

$$\mathbf{P} \mathbf{A} = \left[\begin{array}{c|ccc} L_{11} & \cdots & 1 \\ \mathbf{I}_2 & \mathbf{I}_r & \cdots & 1 \\ \vdots & \ddots & \vdots \\ \mathbf{I}_N & 1 & \cdots & \mathbf{I}_r \end{array} \right] \left[\begin{array}{c|cc} \mathbf{I}_r & \mathbf{Q} \\ \hline \mathbf{Q} & \mathbf{A}^{(\text{new})} \end{array} \right] \left[\begin{array}{c|cccc} \mathbf{U} & U_{12} & \cdots & U_{1N} \\ \hline \mathbf{Q} & I_r & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \mathbf{Q} & \mathbf{Q} & \cdots & I_r \end{array} \right]$$

where

$$\mathbf{A}^{(\text{new})} = \begin{bmatrix} \mathbf{A}_2 & \cdots & \tilde{\mathbf{A}}_{2N} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_N & \cdots & \tilde{\mathbf{A}}_{NN} \end{bmatrix} - \begin{bmatrix} L_{21} \\ \vdots \\ L_{N1} \end{bmatrix} [U_{12} | \cdots | U_{1N}]. \quad (365)$$

Note that the computation of $\mathbf{A}^{(\text{new})}$ is a level-3 operation as it involves one matrix multiplication per A -block.

The remaining task is to compute the pivoted LU factorization of $\mathbf{A}^{(\text{new})}$. Indeed, if

$$P^{(\text{new})} \mathbf{A}^{(\text{new})} = L^{(\text{new})} U^{(\text{new})}$$

and

$$P^{(\text{new})} \begin{bmatrix} L_{21} \\ \vdots \\ L_{N1} \end{bmatrix} = \begin{bmatrix} \tilde{L}_{21} \\ \vdots \\ \tilde{L}_{N1} \end{bmatrix},$$

then

$$PA = \left[\begin{array}{c|c|c|c} L_{11} & \dots & U_1 & U_{12} \dots U_{1N} \\ \hline \tilde{L}_{21} & & 0 & \\ \vdots & & & \\ \tilde{L}_{N1} & & & \end{array} \right] \text{L}^{\text{new}} \quad \left[\begin{array}{c|c|c|c} 1 & & & \\ \hline 0 & & & \\ \vdots & & & \\ 0 & & & \end{array} \right] P_1 \quad \left[\begin{array}{c|c|c|c} 0 & & & \\ \hline 0 & & & \\ \vdots & & & \\ 0 & & & \end{array} \right] P_1^{(\text{new})}$$

is the pivoted block LU factorization of A with

$$P = \left[\begin{array}{c|c} I & \\ \hline P^{(\text{new})} & \end{array} \right] P_1.$$

In general, the processing of each block column in A is a full-part calculation

Part A Apply rectangular Gaussian Elimination with partial pivoting to a block column of A. This produces a permutation, a block column of L, and a diagonal block of U. See (362).

Part B Apply the Part A permutation to the "rest of A." See (363).

Part C. Complete the computation of U's next block row by solving a lower triangular multiple right-hand-side problem. See (364).

Part D. Using the freshly computed L-blocks and U-blocks, update the "rest of A." See (365).

The precise formulation of the method with overwriting is similar to Algorithm 324 and is left as an exercise.

mnmbl .tx tPPSPTf1"lf)\$b .1f kf \$nb ufkafb IRbef" kxTf eb

Recall the discussion of the block-cyclic distribution in §1.6.2 where the parallel computation of the matrix multiplication update $C \leftarrow C + AB$ was outlined. To provide insight into how the pivoted block LU algorithm can be parallelized, we examine a representative step in a small example that also makes use of the block-cyclic distribution.

Assume that $N = 8$ in (361) and that we have a p_{row} -by- p_{col} processor network with $p_{\text{row}} = 2$ and $p_{\text{col}} = 2$. At the start, the blocks of $A = (A_{ij})$ are cyclically distributed as shown in Figure 361. Assume that we have carried out two steps of block LU and that the computed L_{ij} and U_{ij} have overwritten the corresponding A-blocks. Figure 362 displays the situation at the start of the third step. Blocks that are to participate in the Part A factorization

$$P_3 \begin{bmatrix} A_{33} \\ A_{13} \end{bmatrix} = \begin{bmatrix} L_{33} \\ U_{13} \end{bmatrix}$$

are highlighted. Typically, p_{row} processors are involved and since the blocks are each r -by- r , there are r steps as shown in (366).

Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
A_{11}	A_{12}	A_{13}	A_{14}	A_{15}	A_{16}	A_{17}	A_{18}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
A_{21}	A_{22}	A_{23}	A_{24}	A_{25}	A_{26}	A_{27}	A_{28}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
A_{31}	A_{32}	A_{33}	A_{34}	A_{35}	A_{36}	A_{37}	A_{38}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
A_{41}	A_{42}	A_{43}	A_{44}	A_{45}	A_{46}	A_{47}	A_{48}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
A_{51}	A_{52}	A_{53}	A_{54}	A_{55}	A_{56}	A_{57}	A_{58}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
A_{61}	A_{62}	A_{63}	A_{64}	A_{65}	A_{66}	A_{67}	A_{68}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
A_{71}	A_{72}	A_{73}	A_{74}	A_{75}	A_{76}	A_{77}	A_{78}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
A_{81}	A_{82}	A_{83}	A_{84}	A_{85}	A_{86}	A_{87}	A_{88}

Figure 361

Part A:

Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
U_{11} L_{11}	U_{12}	U_{13}	U_{14}	U_{15}	U_{16}	U_{17}	U_{18}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{21}	U_{22} L_{22}	U_{23}	U_{24}	U_{25}	U_{26}	U_{27}	U_{28}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{31}	L_{32}	A_{33}	A_{34}	A_{35}	A_{36}	A_{37}	A_{38}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{41}	L_{42}	A_{43}	A_{44}	A_{45}	A_{46}	A_{47}	A_{48}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{51}	L_{52}	A_{53}	A_{54}	A_{55}	A_{56}	A_{57}	A_{58}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{61}	L_{62}	A_{63}	A_{64}	A_{65}	A_{66}	A_{67}	A_{68}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{71}	L_{72}	A_{73}	A_{74}	A_{75}	A_{76}	A_{77}	A_{78}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{81}	L_{82}	A_{83}	A_{84}	A_{85}	A_{86}	A_{87}	A_{88}

Figure 362

for $j = 1:r$

Columns $A_{kk}(:,j), \dots, A_{Nk}(:,j)$ are assembled in
the processor housing A_{kk} , the "pivot processor"

The pivot processor determines the required row interchange and
the Gauss transform vector

The swapping of the two A-rows may require the involvement of
two processors in the network

The appropriate part of the Gauss vector together with
 $A_{kj}(j, j:r)$ is sent by the pivot processor to the
processors that house $A_{k+1,k}, \dots, A_{Nk}$

The processors that house A_{kk}, \dots, A_{Nk} carry out their
share of the update, a local computation

end

Upon completion, the parallel execution of Parts B and C follows. In the Part B computation, those blocks that may be involved in the row swapping have been highlighted. See Figure 363. This overhead generally engages the entire processor network; although communication is local to each processor column.

Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
U_{11} L_{11}	U_{12}	U_{13}	U_{14}	U_{15}	U_{16}	U_{17}	U_{18}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{21}	U_{22} L_{22}	U_{23}	U_{24}	U_{25}	U_{26}	U_{27}	U_{28}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{31}	L_{32}	U_{33} L_{33}	A_{34}	A_{35}	A_{36}	A_{37}	A_{38}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{41}	L_{42}	L_{43}	A_{44}	A_{45}	A_{46}	A_{47}	A_{48}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{51}	L_{52}	L_{53}	A_{54}	A_{55}	A_{56}	A_{57}	A_{58}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{61}	L_{62}	L_{63}	A_{64}	A_{65}	A_{66}	A_{67}	A_{68}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{71}	L_{72}	L_{73}	A_{74}	A_{75}	A_{76}	A_{77}	A_{78}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{81}	L_{82}	L_{83}	A_{84}	A_{85}	A_{86}	A_{87}	A_{88}

Figure 363

Note that Part C involves just a single processor row while the "big" level-three update that follows typically involves the entire processor network. See Figures 364 and 365.

Part C:

Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
U_{11} L_{11}	U_{12}	U_{13}	U_{14}	U_{15}	U_{16}	U_{17}	U_{18}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{21}	U_{22} L_{22}	U_{23}	U_{24}	U_{25}	U_{26}	U_{27}	U_{28}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{31}	L_{32}	U_{33} L_{33}	A_{34}	A_{35}	A_{36}	A_{37}	A_{38}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{41}	L_{42}	L_{43}	A_{44}	A_{45}	A_{46}	A_{47}	A_{48}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{51}	L_{52}	L_{53}	A_{54}	A_{55}	A_{56}	A_{57}	A_{58}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{61}	L_{62}	L_{63}	A_{64}	A_{65}	A_{66}	A_{67}	A_{68}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{71}	L_{72}	L_{73}	A_{74}	A_{75}	A_{76}	A_{77}	A_{78}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{81}	L_{82}	L_{83}	A_{84}	A_{85}	A_{86}	A_{87}	A_{88}

Figure 364

Part D:

Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
U_{11} L_{11}	U_{12}	U_{13}	U_{14}	U_{15}	U_{16}	U_{17}	U_{18}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{21}	U_{22} L_{22}	U_{23}	U_{24}	U_{25}	U_{26}	U_{27}	U_{28}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{31}	L_{32}	U_{33} L_{33}	A_{34}	A_{35}	A_{36}	A_{37}	A_{38}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{41}	L_{42}	L_{43}	A_{44}	A_{45}	A_{46}	A_{47}	A_{48}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{51}	L_{52}	L_{53}	A_{54}	A_{55}	A_{56}	A_{57}	A_{58}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{61}	L_{62}	L_{63}	A_{64}	A_{65}	A_{66}	A_{67}	A_{68}
Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)	Proc(0,0)	Proc(0,1)
L_{71}	L_{72}	L_{73}	A_{74}	A_{75}	A_{76}	A_{77}	A_{78}
Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)	Proc(1,0)	Proc(1,1)
L_{81}	L_{82}	L_{83}	A_{84}	A_{85}	A_{86}	A_{87}	A_{88}

Figure 365

The communication overhead associated with Part D is masked by the matrix multiplications that are performed on each processor.

This completes the $k = 3$ step of parallel block LU with partial pivoting. The process can obviously be repeated on the trailing 5by-5 block matrix. The virtues of the block-cyclic distribution are revealed through the schematics. In particular, the dominating level-3 step (Part D) is load balanced for all but the last few values of k . Subsets of the processor grid are used for the "smaller," level-2 portions of the computation.

We shall not attempt to predict the fraction of time that is devoted to these computations or the propagation of the interchange permutations. Enlightenment in this direction requires benchmarking.

3.6.3 $n \times r \times k$

The decomposition via partial pivoting in Step A requires a lot of communication. An alternative that addresses this issue involves a strategy called tourament pivoting. Here is the main idea. Suppose we want to compute $PW = LU$ where the blocks of

$$W = \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} \in \mathbb{R}^{n \times r}$$

are distributed around some network of processors. Assume that each W_i has many more rows than columns. The goal is to choose r rows from W that can serve as pivot rows. If we compute the "local" factorizations

$$P_1 W_1 = L_1 U_1, \quad P_2 W_2 = L_2 U_2, \quad P_3 W_3 = L_3 U_3, \quad P_4 W_4 = L_4 U_4,$$

via Gaussian elimination with partial pivoting, then the top r rows of the matrices $P_1 W_1$, $P_2 W_2$, $P_3 W_3$, and $P_4 W_4$ are pivot row candidates. Call these square matrices W'_1 , W'_2 , W'_3 , and W'_4 and note that we have reduced the number of possible pivot rows from n to $4r$.

Next we compute the factorizations

$$P_{12} W'_{12} = P_{12} \begin{bmatrix} W'_1 \\ W'_2 \end{bmatrix} = L_{12} U_{12},$$

$$P_{34} W'_{34} = P_{34} \begin{bmatrix} W'_3 \\ W'_4 \end{bmatrix} = L_{34} U_{34},$$

and recognize that the top r rows of $P_{12} W'_{12}$ and the top r rows of $P_{34} W'_{34}$ are even better pivot row candidates. Assemble these $2r$ rows into a matrix W_{1234} and compute

$$P_{1234} W_{1234} = L_{1234} U_{1234}.$$

The top r rows of $P_{1234} W_{1234}$ are then the chosen pivot rows for the LU reduction of W .

Of course, there are communication overheads associated with each round of the "tourament," but the volume of interprocessor data transfers is much reduced. See Demmel, Grigori, and Xiang (2010).

Problems

P3.6.1 . 915881.s..8. e..o .m8.8b o81406. 6 .m.s8...1b ..8..s .8
..8.8> .s.s..m..

P3.6.2 8.....s8. ..8.d r6.m6.m..s ...8m..6 66. om.8.2.800 .s.68.8.
2..8.. . 3.. 1 .8N.8 mf8.8.. .88..... .8.d.8.22.. 8.68.68..6 66.

28.8 ..8 3.. 1 .8..2m..8.. a a 1r=r = x= =x J= xxli= max=xch= x = = + +
<1V Jx= x= = +x x8xx = x

P3.6.3 78..86m.ol..8. . sls.os 6..8.8.. .8... 81P 1818 .8s8.
. 8. .8.. .8..8. 81s.8..2 1008.... m8 ..8.8..8. N. .8.2..8. 8b68 .20..s.8..
a d a = ' lna oEJ0E00 J0a 0.f0a => (Ed => J0 = E0fa = E0 Oe (farmer O.
poOE0a nOE= .o. =. = 0.f0a = 0 Ea = E0000: x0=Of0=0 a0aa nQ1Qa =>
. (Ba f0) = 00 = f = f = a .00I

Notes and References for 3.6

s = *sQ APACK x a = a00oa - = da0= a <.0<0.0xa0 = a(r-x a:+

J. BE = Ny22 Intr duction to Parallel and Vector Solution of Linear Systems, (san.u (paAA28 IIe

K. la < O Laka ne=H0(0a =y03 0= X22pDxx f a= EOE a = H0Ea 0eo= 0x0
" S0= Gay Onlyay = 000ly= Nit. J. Supercomput. Applic. 2, MNy.

J. ® = = y ax.EE a, = 0= = 0=a000> a QOE = =Ny22 Solving Linear Systems on Vector
and Shar d Memory Computers, h+ 4r(, oVM.etM.Memra8AJ.Me3

2capt r ifld The Impact of Vector and Par llel Ar hitetur s on the Gaussian Elimination
Algorithm, sesAtarpaAA28 1 pe

J. (• = 10k, = = ya SEia = = .Ey.XX0= 0.0= La ,Ta = j Q. LJ= .ON 022pD,0o0y= =
(= Wx f < 0x0 = = JAE0Sy.yeS+ = aj = e0 .OoRe N= Obj = 0= Scienc
Pr gr mming 5 Mifly/ .

:dMr Nr dc4n lapMa8n Alx ,8II S4.1 13uAVauan4etMln3 ,AaptalasTACM
T n. Math Sof w. 31, .INEN .

omju.3 1.lneppe3ennP. 9ed.8Min Ild. ck18 pmAnAaSnaelisiPopel s.opSm(,
..asapetaPen.lpa hAtaA: PPar llel Comput. 36 NENf N

ol,pneuant AS.ltM AtpetaiMAontp s,t,p, Int,a lAtSuMT,anMuAAauantensIA,AAam
in:

J. ® .x x 0S. 1= Oy=a = 00.(a = yN20 .pDeys+, - e= x x.= OEaif= 0xs+RE= = = (ba .0= =
. < y= = 0SIAM J. Matr. Anal. Applic. 32, i.MI rME

E. 2= < = x= k00x x ON20 pR e= XXea(.if= x(a Xa = a .00. AaA=<0dEa = (= =
a d S+RE= = E (ba yQ=EQNNOPar 2011 Par llel Pr cessing Lectur Notes in Computer
Science, 2011, Volume 6853/2011, tlf i lty

Chapter 4

Special Linear Systems

4.1 $\mathbf{Yr} = \mathbf{rY}$ $\mathbf{x} \mathbf{r u w m} = \mathbf{x}$

4.2 $\mathbf{k} = \mathbf{Yx} \quad \mathbf{x} \mathbf{m} = \mathbf{x}$

4.3 $\mathbf{Ur} = \mathbf{wxwm} = \mathbf{x}$

D.D $\mathbf{m} = \mathbf{x} \mathbf{f} \mathbf{w x} = \mathbf{x} \mathbf{m} = \mathbf{x}$

D.G $\mathbf{U} = \mathbf{u n w r r m} = \mathbf{x}$

4.6 $\mathbf{qr} = \mathbf{wx} = \mathbf{mw x} = \mathbf{x}$

4.7 $\mathbf{Vr} = \mathbf{r b x} = \mathbf{w n x} = \mathbf{m} = \mathbf{x}$

4.8 $\mathbf{V u} = \mathbf{r r} = \mathbf{w Y} = \mathbf{u} \mathbf{x k x} = \mathbf{m} = \mathbf{x}$

It is a basic tenet of numerical analysis that solution procedures should exploit structure whenever it is present. In numerical linear algebra, this translates into an expectation that algorithms for general linear systems can be streamlined in the presence of such properties as symmetry, definiteness, and handedness. Two themes prevail:

- There are important classes of matrices for which it is safe not to pivot when computing the LU or a related factorization
- There are important classes of matrices with highly structured LU factorizations that can be computed quickly, sometimes very quickly.

Challenges arise when a fast, but unstable, LU factorization is available.

Symmetry and diagonal dominance are prime examples of exploitable matrix structure and we use these properties to introduce some key ideas in §4.1. In §4.2 we examine the case when A is both symmetric and positive definite, deriving the stable Cholesky factorization. Unsymmetric positive definite systems are also investigated in §4.3. Banded versions of the LU and Cholesky factorizations are discussed and this is followed in §4.4 with a treatment of the symmetric indefinite problem. Block matrix idea and sparse matrix idea come together when the matrix of coefficients is block tridiagonal. This important class of systems receives a special treatment in §4.5.

Classical methods for Vandermonde and Toeplitz systems are considered in §4.6 and §4.7. In §4.8 we connect the fast transform discussion in §1.4 to the problem of solving circulant systems and systems that arise when the Poisson problem is discretized using finite differences.

Before we get started, we clarify some terminology associated with structured problems that pertains to this chapter and beyond. Banded matrices and block-banded matrices are examples of sparse matrices, meaning that the vast majority of their entries are zero. Linear equation methods that are appropriate when the zero/nonzero pattern is more arbitrary are discussed in Chapter 11. Toeplitz, Vandermonde, and circulant matrices are data sparse. A matrix $A \in \mathbb{R}^{m \times n}$ is data sparse if it can be parameterized with many fewer than $O(mn)$ numbers. Cauchy-like systems and semiseparable systems are considered in §12.1 and §12.2.

Reading Notes

Knowledge of Chapters 1, 2, and 3 is assumed. Within this chapter there are the following dependencies:

$$\begin{array}{llll} \S 4.1 & \S 4.2 & \S 4.3 & \S 4.4 \\ & & \overset{1}{\S 4.6} & \\ & & \S 4.5 & \S 4.7 & \S 4.8 \end{array}$$

Global references include Stewart (MABD), Higham (ASNA), Watkins (FMC), Trefethen and Bau (NLA), Demmel (ANLA), and Ipsen (NMA).

4.1 Diagonal Dominance and Symmetry

Pivoting is a serious concern in the context of high-performance computing because the cost of moving data around rivals the cost of computation. Equally important, pivoting can destroy exploitable structure. For example, if A is symmetric, then it involves half the data of a general A . Our intuition (correctly) tells us that we should be able to solve a symmetric $Ax = b$ problem with half the arithmetic. However, in the context of Gaussian elimination with pivoting, symmetry can be destroyed at the very start of the reduction, e.g.,

$$\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} a & b & c \\ b & d & e \\ c & e & f \end{bmatrix} = \begin{bmatrix} c & e & f \\ b & d & e \\ a & b & c \end{bmatrix}.$$

Taking advantage of symmetry and other patterns and identifying situations where pivoting is unnecessary are typical activities in the realm of structured $Ax = b$ solving. The goal is to expose computational shortcuts and to justify their use through analysis.

svtghb plt" kttPb pkltt ttask ttnb f esbIRb dtaf lx1 ftf 1kb

If A 's diagonal entries are large compared to its off-diagonal entries, then we anticipate that it is safe to compute $A = LU$ without pivoting. Consider the $n = 2$ case

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ c/a & 1 \end{bmatrix} \begin{bmatrix} a & b \\ 0 & d - (c/a)b \end{bmatrix}.$$

If a_{ii} and d_{ii} "dominate" b_{ij} and c_{ij} in magnitude, then the elements of L and U will be nicely bounded. To quantify this we make a definition. We say that $A \in \mathbb{M}_n$ is row diagonally dominant if

$$\frac{|a_{ii}|}{\#_i} \geq \frac{\sum_{j \neq i} |b_{ij}|}{|d_{ii}|}, \quad i = 1:n. \quad (4.1.1)$$

Similarly, column diagonal dominance means that $|d_{jj}|$ is larger than the sum of all off-diagonal element magnitudes in the same column. If these inequalities are strict, then A is strictly (row or column) diagonally dominant. A diagonally dominant matrix can be singular; e.g., the 2-by-2 matrix of 1's. However, if a nonsingular matrix is diagonally dominant, then it has a "safe" LU factorization.

Theorem 4.1.1. If A is nonsingular and column diagonally dominant, then it has an LU factorization and the entries in $L = [l_{ij}]$ satisfy $|l_{ij}| \leq 1$.

Proof. We proceed by induction. The theorem is obviously true if $n = 1$. Assume that it is true for $(n-1)$ -by- $(n-1)$ nonsingular matrices that are column diagonally dominant. Partition $A \in \mathbb{M}_n$ as follows

$$A = \begin{bmatrix} a & W \\ V & C \end{bmatrix} \quad a \in J, V \in J^{n-1}, C \in J^{(n-1) \times (n-1)}$$

If $a = 0$ then $V = 0$ and A is singular. Thus, $a \neq 0$ and we have the factorization

$$\begin{bmatrix} \alpha & w^T \\ v & C \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ v/\alpha & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} \alpha & w^T \\ 0 & I \end{bmatrix}, \quad (4.1.2)$$

where

$$B = C - \frac{v}{\alpha} W^T$$

Since $\det(A) = a \cdot \det(B)$, it follows that B is nonsingular. It is also column diagonally dominant because

$$\begin{aligned} \left\| \frac{1}{\alpha} \mathbf{j} \right\|_1 &= \left\| \frac{1}{\alpha} \mathbf{f} \mathbf{j} - \frac{v}{\alpha} \mathbf{j} / a \right\|_1 \leq \left\| \frac{1}{\alpha} \mathbf{f} \mathbf{j} \right\|_1 + \frac{\|W\|}{|\alpha|} \left\| \frac{1}{\alpha} \mathbf{f} \mathbf{j} \right\|_1 \\ &< (\|f_j\| - \|W\|) + \frac{\|W\|}{|\alpha|} (\|a\| - \|f_j\|) \leq \|f_j\| - \frac{\|W\|}{|\alpha|} \leq \|f_j\| \end{aligned}$$

By induction, B has an LU factorization $L \mathbf{U}$ and so from (4.1.2) we have

$$A = \begin{bmatrix} 1 & 0 \\ v/a & L \mathbf{U} \end{bmatrix} \begin{bmatrix} a & W \\ 0 & U \end{bmatrix} = L \mathbf{U}$$

The entries in $\frac{V}{a}$ are bounded by 1 because A is column diagonally dominant. By induction, the same can be said about the entries in L . Thus, the entries in L are all bounded by 1 completing the proof. \square

The theorem shows that Gaussian elimination without pivoting is a stable solution procedure for a column diagonally dominant matrix. If the diagonal elements strictly dominate the off-diagonal elements, then we can actually bound $\|A^{-1}\|$.

Theorem 4.1.2. If $A \in \mathbb{R}^{n \times n}$ and

$$8 = \min_{1 \leq j \leq n} \left(|a_{jj}| - \sum_{\substack{i=1 \\ i \neq j}}^n |a_{ij}| \right) > 0 \quad (4.13)$$

then

$$\|A^{-1}\|_1 \leq 1/\delta.$$

Proof Define $D = \text{diag}(a_{11}, \dots, a_{nn})$ and $E = A - D$. If e is the column n -vector of 1's, then

$$e^T |E| \leq e^T |D| - \delta e^T.$$

If $x \in \mathbb{R}^n$, then $Dx = Ax - Ex$ and

$$|D| |x| \leq |Ax| + |E| |x|.$$

Thus

$$e^T |D| |x| \leq e^T |Ax| + e^T |E| |x| \leq \|Ax\|_1 + (e^T |D| - \delta e^T) |x|$$

and so $8\|x\|_1 = \delta e^T |x| \leq \|Ax\|_1$. The bound on $\|A^{-1}\|_1$ follows from the fact that for any $y \in \mathbb{R}^n$,

$$\delta \|A^{-1}y\|_1 \leq \|A(A^{-1}y)\|_1 = \|y\|_1. \quad \square$$

The "dominance" factor 8 defined in (4.13) is important because it has a bearing on the condition of the linear system. Moreover, if it is too small, then diagonal dominance may be lost during the elimination process because of roundoff. That is, the computed version of the B matrix in (4.12) may not be column diagonally dominant.

4.1.2 m x r w g Y g^T e TM c iúßTMqí ÁÁ B Úþ

If A is symmetric and has an LU factorization $A = LU$, then L and U have a connection. For example, if $n = 2$ we have

$$\begin{aligned} \begin{bmatrix} \cdot & \cdot \\ : & \end{bmatrix} &= \begin{bmatrix} \bullet & \\ c, a & \end{bmatrix} \cdot \begin{bmatrix} & d - (\bullet/a)c \\ & \end{bmatrix} \\ &= \begin{bmatrix} \bullet & \\ c, a & \end{bmatrix} \cdot \left(\begin{bmatrix} & c \\ & d - (\bullet/a)c \end{bmatrix} \begin{bmatrix} & a \\ & \end{bmatrix} \right). \end{aligned}$$

It appears that U is a row scaling of L^T . Here is a result that makes this precise.

Theorem 4.1.3. (LDLT Factorization) If $A \in \mathbb{R}^{n \times n}$ is symmetric and the principal submatrix $A(l:k, l:k)$ is nonsingular for $k = 1:n-1$, then there exists a unit lower triangular matrix L and a diagonal matrix

$$D = \text{diag}(d_1, \dots, d_n)$$

such that $A = LDL^T$. The factorization is unique.

Proof By Theorem 3.2.1 we know that A has an LU factorization $A = LU$. Since the matrix

$$L^{-1}AL^{-T} = UL^{-T}$$

is both symmetric and upper triangular, it must be diagonal. The theorem follows by setting $D = UL^{-T}$ and the uniqueness of the LU factorization. \square

Note that once we have the LDL^T factorization, then solving $Ax = b$ is a 3 step process:

$$Lz = b \quad Dy = z, \quad L^T x = y.$$

This works because $Ax = L(D(LTx)) = L(Dy) = Lz = b$

Because there is only one triangular matrix to compute, it is not surprising that the factorization $A = LDL^T$ requires half as many flops to compute as $A = LU$. To see this we derive a Gauß-pivoting procedure that, for $j = 1:n$, computes $L(j+1:n, j)$ and d_j in step j . Note that

$$A(j:n, j) = L(j:n, l:j) \cdot v(l:j)$$

where

$$v(l:j) = \begin{bmatrix} d_1 \ell_{j1} \\ d_2 \ell_{j2} \\ \vdots \\ d_{j-1} \ell_{j,j-1} \\ d_j \end{bmatrix}.$$

From this we conclude that

$$d_j = a_{jj} - \sum_{k=1}^{j-1} d_k \ell_{jk}^2.$$

With d_j available, we can rearrange the equation

$$\begin{aligned} A(j+1:n, j) &= L(j+1:n, l:j) \cdot v(l:j) \\ &= L(j+1:n, l:j-1) \cdot v(l:j-1) + d_j L(j+1:n, j) \end{aligned}$$

to get a recipe for $L(j+1:n, j)$:

$$L(j+1:n, j) = \frac{1}{d_j} (A(j+1:n, j) - L(j+1:n, l:j-1) \cdot v(l:j-1)).$$

Properly sequenced we obtain the following overall procedure

```

for j = 1:n
    for i = 1:j - 1
        v(i) = L(j,i) · d(i)
    end
    dj = A(j,j) - L(j,1:j-1) · v(1:j-1)
    L(j + 1:n,j) = (A(j + 1:n,j) - L(j + 1:n,1:j-1) · v(1:j-1)) / dj
end

```

With overwriting we obtain the following procedure

use `linsolve`. If $A \in \mathbb{R}^{n \times n}$ is symmetric and has an LU factorization, then this algorithm computes a unit lower triangular matrix L and a diagonal matrix $D = \text{diag}(d_1, \dots, d_n)$ so $A = LDL^T$. The entry a_{ij} is overwritten with f_{ij} if $i > j$ and with d_i if $i = j$.

```

for j = 1:n
    for i = 1:j - 1
        v(i) = A(j,i)A(i,i)
    end
    A(j,j) = A(j,j) - A(j,1:j - 1)·v(1:j - 1)
    A(j + 1:n,j) = (A(j + 1:n,j) - A(j + 1:n,1:j - 1)·v(1:j - 1))/A(j,j)
end

```

This algorithm requires $n^3/3f$ ops, about half the number of ops involved in Gaussian elimination.

The computed solution x to $Ax = b$ obtained via Algorithm 4.1.1 and the usual triangular systems solvers of §3.1 can be shown to satisfy a perturbed system $(A+E)x = b$, where

$$|E| \leq n\mathbf{u} \left(2|A| + 4|\hat{L}| |\hat{D}| |\hat{L}^T| \right) + O(\mathbf{u}^2) \quad (414)$$

and \hat{L} and \hat{b} are the computed versions of L and D , respectively.

As in the case of the LU factorization considered in the previous chapter, the upper bound in (4.1.4) is without limit unless A has some special property that guarantees stability. In the next section, we show that if A is symmetric and positive definite, then Algorithm 4.1.1 not only runs to completion, but is extremely stable. If A is symmetric but not positive definite, then, as we discuss in §4.4, it is necessary to consider alternatives to the LDLT factorization.

Problems

P4.1.1 .686 .6.b.s8 .68.2.8..8... fP1d1.6..... .8r2.8..8.f68. A T≡()=T)12k

P4.1.2 ...8 .. 8.8 . .8.2s....s .. .8 m688.8.P1Q.6.. ...8.8. .8 . .868.s8.8... ..
 n... re,...2s..fo.686.6.. || A⁻¹ 10 ≤ μ F^{|n.n} .ⁿ >,ⁿ(.))∞[<]s^{|n} ≤_F, , ≤(0
 ,≤ ,(,-nⁿ,ⁿ (n,| >,) ≤(-jE^mj

$$P4.1.3 \quad 2. .8.8 \quad A T_a(lb) = xT_2T_1 - x(aQ_2nI - lbA_nEQ_2)x = T_1Bx - 2x n^2A = LDL^T.$$

k o p m u v (a1 ors ske (+)er2earid ia L 2x D?) 2 nyA=b2sA,11AE ,()x 1 a2 E
VIII.

Notes and References for §4.1

62...8.c. ..8s. 8≥1..8.c.6. P1.10 .8s..8. .8 .68.8.68.. 8≥3.82. ... 888.c..8188
..80.. (≥€71 II)l), , ,(

$\exists x \in \mathbb{N} \text{ such that } x^2 > 10$

C,At1 eViIpt,uAen oateVII atth af Asl,At1 ,t pa3l.en apl lpones,A,AnnAapt,I Get,lbalp,,

k m9lp emnay, neA(11((7:6 „ B,,))(), 6),0, , n) (), 0,) 1(),r .0,(), ∈I
 Numer. Math. 27, 67v-68lk

JISIAM J. Matr. x Anal. Applic. 19 6t6f 6t6y

SIAM J. Matrix Anal. Appl. 13 MfNMLr

SIAM J. Matrix Anal. Appl. 13(1992) 100-114. © 1992 Society for Industrial and Applied Mathematics

IPKa (344)7: i i ,(-,-) (B)E,,),

5)ent 8 Mla.39 5.epta3enmP. ,poen(344^T) : i ,)(),),)(((),)-9,),]0
 (,) 1(,), (ISIAM J. Matr x Anal. Applic. 30 IIIIf yN.

,ual,,e ,AA,atA Net 1Al,etam8t, t,a BellMTet,BBe meilneVmluMnen tp,f el amA,,AAam
Mnf

1r. (a \oplus 111 17) $_7$: $\quad \text{I}((,-.)),))(-\text{ (},\text{ } \ldots,\text{ })\rightarrow(\text{ } i\text{ },\text{ } -(),\text{ } i((\text{ },\text{ } \ldots),\text{ }),)(\text{ } \ldots,\text{ } 0X8B,$ Numer. Math
 81 NtEf E 16

P: PanmI T 425 emerBennl, IeTeie(1111) :≥(-.9),)-)∈((,),() i , -(), .0i(.),))
 $m_1 \in \text{Numer. Lin. Alg. 5.6uM 6.6}$

$$A. \quad D = E \otimes I = I, \quad W = a(24) \otimes I, \quad \text{where } a(24) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad i = 24(n) \in \{1, 2, \dots, 10\}, \quad i = -10$$

1kPka (844(7..)) Math. Comput. 73, ur 259 I.

SIAM J. Numer. Anal. 45, M₁M₂M₃M₄M₅
 elA, l en-SkMla (341) 7 (:) 9. ,,) () $\in ((0, n),) \in (i, n,) (0, n,) (6, 9,) (\geq (.$

n. i,-(), 0 i(.,))) , (, Numer. Math. 119, 111-111.

4.2 Positive Definite Systems

A matrix $A \in \mathbb{R}^{n \times n}$ is positive definite if $x^T A x > 0$ for all nonzero $x \in \mathbb{R}^n$, positive semidefinite if $x^T A x \geq 0$ for all $x \in \mathbb{R}^n$, and indefinite if we can find $x, y \in \mathbb{R}^n$ so that $(x^T A x)(y^T A y) < 0$. Symmetric positive definite systems constitute one of the most important classes of special problems. Consider the 2-by-2 symmetric case. If

$$A = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

is positive definite then

$$x = [1, 0]^T = x^T A x = a > 0.$$

$$x = [0, 1]^T = x^T A x = \cdot > 0,$$

$$x^T A x = \alpha + \lambda > 0$$

$$x^T A x = \lambda_1 + \dots + \lambda_n > 0$$

These two equations imply $x^T A x \geq (a_{ii})/2$. From these results we see that the largest entry in A is on the diagonal and that it is positive. This turns out to be true in general. (See Theorem 4.2.8 below.) A symmetric positive definite matrix has a diagonal that is sufficiently "heavy" to preclude the need for pivoting. A special factorization called the Cholesky factorization is available for such matrices. It exploits both symmetry and definiteness and its implementation is the main focus of this section. However, before those details are pursued we discuss unsymmetric positive definite matrices. This class of matrices is important in its own right and presents interesting pivot-related issues.

4.2.1 Positive Definiteness

Suppose $A \in J_{n,n}$ is positive definite. It is obvious that a positive definite matrix is nonsingular for otherwise we could find a nonzero x so $x^T A x = 0$. However, much more is implied by the positivity of the quadratic form $x^T A x$. The following results show.

Theorem 4.2.1 If $A \in J_{n,n}$ is positive definite and $X \in J_{n,k}$ has rank k , then $B = X^T A X \in J_{k,k}$ is also positive definite.

Proof. If $z \in J_k$ satisfies $0 = z^T B z = (Xz)^T A (Xz)$, then $Xz = 0$. But since X has full column rank, this implies that $z = 0$. \square

Corollary 4.2.2. If A is positive definite, then all its principal submatrices are positive definite. In particular, all the diagonal entries are positive.

Proof. If v is an integer length k vector with $1 \leq v_1, \dots, v_k \leq n$, then $X = I_n(:,v)$ is a rank- k matrix made up of columns v_1, \dots, v_k of the identity. It follows from Theorem 4.2.1 that $A(v,v) = X^T A X$ is positive definite. \square

Theorem 4.2.3. The matrix $A \in J_{n,n}$ is positive definite if and only if the symmetric matrix

$$T = \frac{A + A^T}{2}$$

has positive eigenvalues.

Proof. Note that $x^T A x = x^T T x$. If $Tx = Ax$ then $x^T A x = x^T x$. Thus, if A is positive definite then A is positive. Conversely, suppose T has positive eigenvalues and $Q^T T Q = \text{diag}(\lambda_i)$ is its Schur decomposition. (See §2.17.) It follows that if $x \in \mathbb{R}^n$ and $y = Q^T x$, then

$$x^T A x = x^T T x = y^T (Q^T T Q) y = \sum_{i=1}^n \lambda_i y_i^2 > 0$$

completing the proof of the theorem. \square

Corollary 4.2.4. If A is positive definite, then it has an LU factorization and the diagonal entries of U are positive.

Pr of. From Corollary 4.2.2 it follows that the submatrices $A(1:k, 1:k)$ are nonsingular for $k = 1:n$ and so from Theorem 3.2.1 the factorization $A = LU$ exists. If we apply Theorem 4.2.1 with $X = (L^{-1})T = L^{-1}r$, then $B = XTAX = L^{-1}(LU)L^{-1} = UL^{-1}T$ is positive definite and therefore has positive diagonal entries. The corollary follows because $L^{-1}T$ is unit upper triangular and this implies $b_{ii} = u_{ii}$, $i = 1:n$. \blacksquare

The mere existence of an LU factorization does not mean that its computation is advisable because the resulting factors may have unacceptably large elements. For example, if $\epsilon > 0$ then the matrix

$$A = \begin{bmatrix} 18 & & \\ & \ddots & \\ & & 1 \end{bmatrix} = \begin{bmatrix} 1 & & \\ -m/\epsilon & \ddots & \\ & & 1 + m^2/\epsilon \end{bmatrix} \begin{bmatrix} m \\ & \ddots \\ & & 1 \end{bmatrix}$$

is positive definite. However, if $m/\epsilon \gg 1$, then it appears that some kind of pivoting is in order. This prompts us to pose an interesting question. Are there conditions that guarantee when it is safe to compute the LU-without-pivoting factorization of a positive definite matrix?

4.2.2 Unsymmetric Positive Definite Systems

The positive definiteness of a general matrix A is inherited if one omits symmetric part

$$T = \frac{A + AT}{2}.$$

Note that for any square matrix we have $A = T + S$ where

$$S = \frac{A - AT}{2}$$

is the skewsymmetric part of A . Recall that a matrix S is skewsymmetric if $s_{ij} = -s_{ji}$. If S is skewsymmetric, then $x^T S x = 0$ for all $x \in \mathbb{R}^n$ and $s_{ii} = 0$ for $i = 1:n$. It follows that A is positive definite if and only if its symmetric part is positive definite.

The derivation and analysis of methods for positive definite systems require an understanding about how the symmetric and skewsymmetric parts interact during the LU process.

Theorem 4.2.5. Suppose

$$A = \begin{bmatrix} \alpha & v^T \\ v & B \end{bmatrix} + \begin{bmatrix} 0 & -w^T \\ w & C \end{bmatrix}$$

α positive definite and that $B \in \mathbb{J}(n-1) \times \mathbb{J}(n-1)$ is symmetric and $C \in \mathbb{J}(n-1) \times \mathbb{J}(n-1)$ is skewsymmetric. Then it follows that

$$A = \begin{bmatrix} 1 & & \\ (v + w)/\alpha & \ddots & \\ & & 0 \end{bmatrix} \begin{bmatrix} \alpha & (v - w)^T \\ & B_1 + C_1 \end{bmatrix} \quad (42.1)$$

where

$$B_1 = B - \frac{1}{\alpha} (vv^T - ww^T) \quad (422)$$

is symmetric positive definite and

$$C_1 = C - \frac{1}{\alpha} (wv^T - vw^T) \quad (423)$$

is skewsymmetric

Proof. Since $A - B$ it follows that (421) holds. It is obvious from their definitions that B_1 is symmetric and that C_1 is skewsymmetric. Thus, all we have to show is that B_1 is positive definite i.e.,

$$0 < z^T B_1 z = z^T B z - \frac{1}{\alpha} (v^T z)^2 + \frac{1}{\alpha} (w^T z)^2 \quad (424)$$

for all nonzero $z \in \mathbb{R}^{n-1}$. For any $\mu \in \mathbb{R}$ and $0 < \alpha \in \mathbb{R}^{n-1}$ we have

$$\begin{aligned} 0 &< \begin{bmatrix} \mu \\ z \end{bmatrix}^T A \begin{bmatrix} \mu \\ z \end{bmatrix} = \begin{bmatrix} \mu \\ z \end{bmatrix}^T \begin{bmatrix} \alpha & v^T \\ v & B \end{bmatrix} \begin{bmatrix} \mu \\ z \end{bmatrix} \\ &= \mu^2 \alpha + 2\mu v^T z + z^T B z. \end{aligned}$$

If $\mu = -(\sqrt{\alpha} z)/\alpha$, then

$$0 < z^T B z - \frac{1}{\alpha} (v^T z)^2,$$

which establishes the inequality (424). \square

From (421) we see that if $B_1 + C_1 = L_1 U_1$ is the LU factorization, then $A = LU$ where

$$L = \begin{bmatrix} 1 & 0 \\ (\mathbf{v} + \mathbf{w})/\alpha & L_1 \end{bmatrix} \begin{bmatrix} \alpha & (\mathbf{v} - \mathbf{w})^T \\ 0 & U_1 \end{bmatrix}.$$

Thus, the theorem shows that triangular factors in $A = LU$ are nicely bounded if S is not too big compared to T^{-1} . Here is a result that makes this precise:

Theorem 4.2.6. Let $A \in \mathbb{R}^{n \times n}$ be positive definite and set $T = (A + AT)/2$ and $S = (A - AT)/2$. If $A = LU$ is the LU factorization, then

$$\| |L||U| \|_F \leq n (\| T \|_2 + \| ST^{-1}S \|_2). \quad (425)$$

Proof. See Golub and Van Loan (1979). \square

The theorem suggests when it is safe not to pivot. Assume that the computed factors \hat{L} and \hat{U} satisfy

$$\| |\hat{L}| |\hat{U}| \|_F \leq c \| |L||U| \|_F, \quad (426)$$

where c is a constant of modest size. It follows from (42.1) and the analysis in §3.3 that if these factors are used to compute a solution to $Ax = b$, then the computed solution x satisfies $(A + E)x = b$ with

$$\|E\|_F \leq \sqrt{2n\|A\|_F} + 4m^2(\|T\|_2 + \|ST^{-1}S\|_2) + O(n^2). \quad (42.7)$$

It is easy to show that $\|T\|_2 \leq \|A\|_2$ and so it follows that if

$$\|H\| = \frac{\|ST^{-1}S\|_2}{\|A\|_2} \quad (42.8)$$

is not too large, then it is safe not to pivot. In other words, the norm of the skew-symmetric part S has to be modest relative to the condition of the symmetric part T . Sometimes it is possible to estimate $\|H\|$ in an application. This is trivially the case when A is symmetric if r then $\|H\| = 0$.

4.2.3 Symmetric Positive Definite Systems

If we apply the above results to a symmetric positive definite matrix we know that the factorization $A = LU$ exists and is stable to compute. The computation of the factorization $A = LDLT$ via Algorithm 4.12 is also stable and exploits symmetry. However, for symmetric positive definite systems it is often handier to work with a variation of $LDLT$.

Theorem 4.2.7 (Cholesky Factorization). If $A \in \mathbb{R}^{n,n}$ is symmetric positive definite, then there exists a unique lower triangular $G \in \mathbb{R}^{n,n}$ with positive diagonal entries such that $A = GG^T$.

From Theorem 4.13 there exists a unit lower triangular L and a diagonal

$$D = \text{diag}(d_1, \dots, d_n)$$

such that $A = LDLT$. Theorem 4.2.1 tells us that $L^{-1}AL^{-T} = D$ is positive definite. Thus, the d_k are positive and the matrix $G = L \text{diag}(\sqrt{d_1}, \dots, \sqrt{d_n})$ is real and lower triangular with positive diagonal entries. It also satisfies $A = GGT$. Uniqueness follows from the uniqueness of the $LDLT$ factorization. \square

The factorization $A = GGT$ is known as the Cholesky factorization and G is the Cholesky factor. Note that if we compute the Cholesky factorization and solve the triangular systems $Gy = b$ and $GTx = y$, then $b = Gy = G(GTx) = (GGT)x = Ax$.

4.2.4 The Cholesky Factor is not a Square Root

A matrix $X \in \mathbb{R}^{n,n}$ that satisfies $A = X^2$ is a square root of A . Note that if A is symmetric, positive definite, and not diagonal, then its Cholesky factor is not a square root. However, if $A = GGT$ and $X = UEVU^T$ where $G = UEV^T$ is the SVD, then

$$X^2 = (UEVU^T)(UEVU^T) = UEV^2U^T = (UEV^T)(UEV^T)U^T = GGT = A$$

Thus, an asymmetric positive definite matrix A has an asymmetric positive definite square root denoted by $A^{1/2}$. We have more to say about matrix square roots in §9.4.2.

4.2.5 T Gaxpy-Rich Cholesky Factorization

Our proof of the Cholesky factorization in Theorem 4.2.7 is constructive. However, we can develop a more effective procedure by comparing columns in $A = GG^T$. If $A \in \mathbb{R}^{n \times n}$ and $1 \leq j \leq n$, then

$$A(:,j) = \sum_{k=1}^j G(j,k) \cdot G(:,k).$$

This says that

$$G(j,j)G(:,j) = A(:,j) - \sum_{k=1}^{j-1} G(j,k) \cdot G(:,k) = v. \quad (429)$$

If the first $j - 1$ columns of G are known, then v is computable. It follows by equating components in (429) that

$$G(j:n, j) = v(j:n) / \sqrt{v(j)}$$

and so we obtain

```

for j = 1:n
    v(j:n) = A(j:n, j)
    for k = 1:j - 1
        v(j:n) = v(j:n) - G(j,k) · G(j:n, k)
    end
    G(j:n, j) = v(j:n) / y
end

```

It is possible to arrange the computations so that G overwrites the lower triangle of A .

Given a symmetric positive definite $A \in \mathbb{R}^{n \times n}$, the following algorithm computes a lower triangular G such that $A = GG^T$. For all $i \neq j$, $G(i,j)$ overwrites $A(i,j)$.

```

for j = 1:n
    if j > 1
        A(j:n, j) = A(j:n, j) - A(j:n, 1:j - 1) · A(j, 1:j - 1)^T
    end
    A(j:n, j) = A(j:n, j) / sqrt(A(j,j))
end

```

This algorithm requires $n^3/3$ fops.

4.2.6 Stability of the Cholesky Process

In exact arithmetic, we know that a symmetric positive definite matrix has a Cholesky factorization. Conversely, if the Cholesky process runs to completion with strictly positive square roots, then A is positive definite. Thus, to find out if a matrix A is

positive definite, we merely try to compute its Cholesky factorization using any of the methods given above.

The situation in the context of roundoff error is more interesting. The numerical stability of the Cholesky algorithm roughly follows from the inequality

$$g_{ij}^2 \leq \sum_{k=1}^i g_{ik}^2 = a_{ii}.$$

This shows that the entries in the Cholesky triangle are nicely bounded. The same conclusion can be reached from the equation $\|G\|_F = \|A\|_F$.

The roundoff errors associated with the Cholesky factorization have been extensively studied in a classical paper by Wilkinson (1968). Using the results in this paper, it can be shown that if \hat{x} is the computed solution to $Ax = b$ obtained via the Cholesky process, then \hat{x} solves the perturbed system

$$(A + E)\hat{x} = b \quad \|E\|_2 \leq c_n u \|A\|_2,$$

where c_n is a small constant that depends upon n . Moreover, Wilkinson shows that if $Q\sqrt{K_2}(A) \leq 1$ where q_1 is another small constant, then the Cholesky process runs to completion, i.e., no square roots of negative numbers arise.

It is important to remember that symmetric positive definite linear systems can be ill-conditioned. Indeed, the eigenvalues and singular values of a symmetric positive definite matrix are the same. This follows from (24.1) and Theorem 4.23. Thus

$$K_2(A) = \frac{\text{Anax}(A)}{\text{Anin}(A)}.$$

The eigenvalue $\text{Anin}(A)$ is the "distance to trouble" in the Cholesky setting. This prompts us to consider a permutation strategy that steers us away from using small diagonal elements that jeopardize the factorization process.

shihBb The LDL^T Factorization with Symmetric Pivoting

With an eye towards handling ill-conditioned symmetric positive definite systems, we return to the LDL^T factorization and develop an outer product implementation with pivoting. We first observe that if A is symmetric and P_1 is a permutation, then $P_1 A P_1^T$ is not symmetric. On the other hand, $P_1 A P_1^T$ is symmetric suggesting that we consider the following factorization:

$$P_1 A P_1^T = \begin{bmatrix} 0 & V^T \\ v & B \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ v/\alpha & \ln 1 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & A \end{bmatrix} \begin{bmatrix} 1 & 0 \\ v/\alpha & \ln 1 \end{bmatrix}^T$$

where

$$\bar{A} = B - \frac{1}{v} vv^T.$$

Note that with this kind of symmetric pivoting the new $(1,1)$ entry α is some diagonal entry a_{ii} . Our plan is to choose P_1 so that α is the largest of A 's diagonal entries. If we apply the same strategy recursively to A and compute

$$P_1 \bar{A} P_1^T = LDL^T,$$

then we emerge with the factorization

$$PAPT = LDLT \quad (42.10)$$

where

$$P = \begin{bmatrix} & \\ & \end{bmatrix} P_1, \quad L = \begin{bmatrix} 1 & 0 \\ v/\alpha & \tilde{L} \end{bmatrix}, \quad D = \begin{bmatrix} \alpha & \\ 0 & \end{bmatrix}.$$

By virtue of this pivot strategy,

$$d_1 \ d_2 \ \dots \ d_n > 0$$

Here is a nonrecursive implementation of the overall algorithm

Algorithm 4.22 (xSLShw5 LDLT Mlin) Given a symmetric positive semidefinite $A \in \mathbb{R}^{n \times n}$, the following algorithm computes a permutation P , a unit lower triangular L , and a diagonal matrix $D = \text{diag}(d_1, \dots, d_n)$ so $PAPT = LDLT$ with $d_1 \ d_2 \ \dots \ d_n > 0$. The matrix element a_{ij} is overwritten by d_i if $i = j$ and by d_j if $i > j$. $P = P_1 \dots P_n$ where P_k is the identity with rows k and $\text{piv}(k)$ interchanged

for $k = 1:n$

```

    piv(k) = j where a(j, :) = max{a(k, :), a(j, :)}
    A(k, :) = A(j, :)
    A(:, k) = A(:, j)
    a = A(k, k)
    v = A(k+1:n, k)
    A(k+1:n, k) = v/a
    A(k+1:n, k+1:n) = A(k+1:n, k+1:n) - v * A(k+1:n, k)
  end

```

If symmetry is exploited in the outer product update, then $n^3/3$ fops are required. To solve $Ax = b$ given $PAPT = LDLT$, we proceed as follows:

$$Lw = Pb \quad Dy = w, \quad L^T z = y, \quad x = PTz$$

We mention that Algorithm 4.22 can be implemented in a way that only references the lower triangular part of A .

It is reasonable to ask why we even bother with the LDLT factorization given that it appears to offer no real advantage over the Cholesky factorization. There are two reasons. First, it is more efficient in narrow band situations because it avoids square roots, see §4.3.6. Second, it is a graceful way to introduce factorizations of the form

$$^T APT = \begin{pmatrix} \text{lower} \\ \text{triangular} \end{pmatrix} \times \begin{pmatrix} \text{simple} \\ \text{matrix} \end{pmatrix} \times \begin{pmatrix} \text{lower} \\ \text{triangular} \end{pmatrix}^T,$$

where P is a permutation arising from a symmetry-exploiting pivot strategy. The symmetric indefinite factorizations that we develop in §4.4 fall under this heading as does the "rank revealing" factorization that we are about to discuss for semidefinite problems.

4.2.8 The Symmetric Semidefinite Case

A symmetric matrix $A \in \mathbb{R}^{n,n}$ is positive semidefinite if

$$x^T A x \geq 0$$

for every $x \in \mathbb{R}^n$. It is easy to show that if $A \in \mathbb{R}^{n,n}$ is symmetric and positive semidefinite, then its eigenvalues satisfy

$$0 = \lambda_1(A) = \dots = \lambda_{r+1}(A) \leq \lambda_r(A) \leq \dots \leq \lambda_n(A) \quad (4211)$$

where r is the rank of A . Our goal is to show that Algorithm 4.2.2 can be used to estimate r and produce a streamlined version of (42.10). But first we establish some useful properties.

Theorem 4.2.8. If $A \in \mathbb{R}^{n,n}$ is symmetric positive semidefinite, then

$$|a_{ij}| \leq (a_{ii} + a_{jj})/2, \quad (4212)$$

$$|a_{ij}| \leq \sqrt{a_{ii}a_{jj}}, \quad \{i; j\}, \quad (4213)$$

$$\max |a_{ij}| = \max a_{ii}, \quad (4214)$$

$$a_{ii} = 0 \quad A(i,:) = Q \quad A(:,i) = Q \quad (4215)$$

Proof. Let e_i denote the i th column of I_n . Since

$$x = e_i + e_j \quad 0 \quad x^T A x = a_{ii} + 2a_{ij} + a_{jj},$$

$$x = e_i - e_j \quad 0 \quad x^T A x = a_{ii} - 2a_{ij} + a_{jj},$$

it follows that

$$-2a_{ij} \leq a_{ii} + a_{jj},$$

$$2a_{ij} \leq a_{ii} + a_{jj}.$$

These two equations confirm (42.12), which in turn implies (42.14).

To prove (42.13), set $x = Te_i + e_j$ where $T \in \mathbb{R}$. It follows that

$$0 \leq x^T A x = a_{ii}T^2 + 2a_{ij}T + a_{jj},$$

must hold for all T . This is a quadratic equation in T and for the inequality to hold, the discriminant $4a_{ij} - 4a_{ii}a_{jj}$ must be negative, i.e., $|a_{ij}| > \sqrt{a_{ii}a_{jj}}$. The implication in (42.15) follows immediately from (42.13). \square

Let us examine what happens when Algorithm 4.2.2 is applied to a rank- r positive semidefinite matrix. If $k = r$, then after k steps we have the factorization

$$\tilde{P} A \tilde{P}^T = \begin{bmatrix} L_{11} & 0 \\ L_{21} & I_{n-k} \end{bmatrix} \begin{bmatrix} D_k & 0 \\ 0 & A_k \end{bmatrix} \begin{bmatrix} L_{11}^T & L_{21}^T \\ 0 & I_{n-k} \end{bmatrix} \quad (4216)$$

where $D_k = \text{diag}(d_1, \dots, d_k) \in \mathbb{R}^{k \times k}$ and $d_1 \geq \dots \geq d_k \geq 0$. By virtue of the pivot strategy, if $d_k = 0$ then A_k has a zero diagonal. Since A_k is positive semidefinite, it follows from (42.15) that $A_k = 0$. This contradicts the assumption that A has rank r unless $k = r$. Thus, if $k = r$, then $d_k > 0$. Moreover, we must have $A_r = 0$ since A has the same rank as $\text{diag}(D_r, A_r)$. It follows from (42.16) that

$$PAP^T = \begin{bmatrix} L_{11} \\ L_{21} \end{bmatrix} D_r \begin{bmatrix} L_{11}^T & L_{21}^T \end{bmatrix} \quad (42.17)$$

where $D_r = \text{diag}(d_1, \dots, d_r)$ has positive diagonal entries, $L_{11} \in \mathbb{R}^{r \times r}$ is unit lower triangular, and $L_{21} \in (\mathbb{R}^{n-r})^{r \times (n-r)}$. If ℓ_j is the j th column of the L -matrix, then we can rewrite (42.17) as a sum of rank 1 matrices:

$$PAP^T = \sum_{j=1}^r d_j \ell_j \ell_j^T.$$

This can be regarded as a relatively cheap alternative to the SVD rank 1 expansion.

It is important to note that our entire semidefinite discussion has been an exact arithmetic discussion. In practice, a threshold tolerance for small diagonal entries has to be built into Algorithm 42.2. If the diagonal of the computed A_k in (42.16) is sufficiently small, then the loop can be terminated and \tilde{r} can be regarded as the numerical rank of A . For more details, see Higham (1989).

11 xdrv Block Cholesky

Just as there are block methods for computing the LU factorization, so are there are block methods for computing the Cholesky factorization. Paralleling the derivation of the block LU algorithm in §3.2.11, we start by blocking $A = GG^T$ as follows:

$$\begin{bmatrix} A_{11} & A_{12}^T \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} G_{11} & 0 \\ G_{21} & G_{22} \end{bmatrix} \begin{bmatrix} G_{11} & 0 \\ G_{21} & G_{22} \end{bmatrix}^T. \quad (42.18)$$

Here $A_{11} \in \mathbb{R}^{r \times r}$, $A_{21} \in (\mathbb{R}^{n-r})^{r \times (n-r)}$, r is a blocking parameter, and G is partitioned accordingly. Comparing blocks in (42.18) we conclude that

$$A_{11} = G_{11}G_{11}^T$$

$$A_{21} = G_{21}G_{11}^T$$

$$A_{22} = G_{21}G_{11}^T + G_{22}G_{22}^T$$

which suggests the following 3-step procedure:

Step 1: Compute the Cholesky factorization of A_{11} to get G_{11} .

Step 2 Solve a lower triangular multiple right-hand-side system for G_{21} .

Step 3 Compute the Cholesky factor G_{22} of A_{22} : $G_{21}G_{11}^T = A_{22} - A_{21}A_{11}^{-1}A_{21}^T$.

In recursive form we obtain the following algorithm:

~~MAE 1100 nshCn e18C - aT8 S2E6 chE28.001 Suppose A ∈ ℝ^{N,N} is symmetric positive definite and r is a positive integer. The following algorithm computes a lower triangular G ∈ ℝ^{N,N} so A = GG^T.~~

```

function G = BlockCholesky(A,n,r)
    if n == r
        Compute the Cholesky factorization A = G GT.
    else
        Compute the Cholesky factorization A(l:r,l:r) = G11G11T.
        Solve G21G11T = A(r+1:n,l:r) for G21.
        A = A(r+1:n,r+1:n) - G21G11T
        G22 = BlockCholesky( ,n-r,r)
        G = [   ]
              :: G22
    end
end

```

If symmetry is exploited in the computation of G_{11} , then this algorithm requires $n^3/3$ fops. A careful accounting of fops reveals that the level-3f action is about $1 - \frac{1}{N^2}$ where $N = n/r$. The "small" Cholesky computation for G_{11} and the "thin" solution process for G_{21} are dominated by the "large" level-3 update for G_{22} .

To develop a nonrecursive implementation, we assume for clarity that $n = Nr$ where N is a positive integer and consider the partitioning

$$\begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & \ddots & \vdots \\ A_{N1} & \cdots & A_{NN} \end{bmatrix} = \begin{bmatrix} G_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ G_{N1} & \cdots & G_{NN} \end{bmatrix} \begin{bmatrix} G_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ G_{N1} & \cdots & G_{NN} \end{bmatrix}^T \quad (42.19)$$

where all blocks are r -by- r . By equating (i,j) blocks with $i \leq j$ it follows that

$$A_{ij} = \sum_k G_{ik}G_{jk}.$$

Define

$$s = A_{ii} - \sum_{k=1}^{j-1} G_{ik}G_{kj}^T = A_{ii} - [G_{i1} | \cdots | G_{i,j-1}] \begin{bmatrix} G_{j1}^T \\ \vdots \\ G_{j,j-1}^T \end{bmatrix}.$$

If $i = j$, then G_{jj} is the Cholesky factor of S . If $i > j$, then $G_{ij}G_{ji}^T = S$ and G_{ij} is the solution to a triangular multiple right hand side problem. Properly sequenced, these equations can be arranged to compute all the G -blocks.

Given a symmetric positive definite $A \in \mathbb{R}^{n \times n}$ with $n = N_r$ with blocking (4.2.19), the following algorithm computes a lower triangular $G \in \mathbb{R}^{n \times n}$ such that $A = G^T G$. The lower triangular part of A is overwritten by the lower triangular part of G .

```

f r j = 1:N
f r i = j:N
Compute S = Aii -  $\sum_{k=1}^{j-1} GikGk$ 
if i = j
    Compute Cholesky factorization S = GiGi.
else
    Solve GiGi = S for Gi.
end
Aii = Gi.
end
end

```

The overall process involves $n^3/3$ operations like the other Cholesky procedures that we have developed. The algorithm is rich in matrix multiplication with a level-3 operation given by $1 - (1/N^3)$. The algorithm can be easily modified to handle the case when r does not divide n .

4.2.10 Recursive Blocking

It is instructive to look a little more deeply into the implementation of a block Cholesky factorization as it is an occasion to stress the importance of designing data structures that are tailored to the problem at hand. High-performance matrix computations are filled with tensions and tradeoffs. For example, a successful pivot strategy might balance concerns about stability and memory traffic. Another tension is between performance and memory constraints. As an example of this, we consider how to achieve level-3 performance in a Cholesky implementation given that the matrix is represented in packed format. This data structure houses the lower (or upper) triangular portion of a matrix $A \in \mathbb{R}^{N \times N}$ in a vector of length $N = n(n + 1)/2$. The symmetric arrangement stacks the lower triangular subcolumns, e.g.,

$$\text{symvec}(A) = [a_{11} \ a_{21} \ a_{31} \ a_{41} \ a_{22} \ a_{32} \ a_{42} \ a_{33} \ a_{43} \ a_{44}]^T. \quad (4.2.20)$$

This layout is not very friendly when it comes to block Cholesky calculations because the assembly of an A -block (say $A(i_1:i_2, j_1:j_2)$) involves irregular memory access patterns. To realize a high-performance matrix multiplication it is usually necessary to have the matrices laid out conventionally as full rectangular arrays that are contiguous in memory, e.g.,

$$\text{vec}(A) = [a_{11} \ a_{21} \ a_{31} \ a_{41} \ a_{12} \ a_{22} \ a_{32} \ a_{42} \ a_{13} \ a_{23} \ a_{33} \ a_{43} \ a_{14} \ a_{24} \ a_{34} \ a_{44}]^T. \quad (4.2.21)$$

(Recall that we introduced the `vec` operation in §1.37.) Thus, the challenge is to develop a high-performance block algorithm that overwrites a symmetric positive definite A in packed format with its Cholesky factor G in packed format. Toward that end, we

present the main idea behind a recursive data structure that supports level-3 computation and is storage efficient. As memory hierarchies get deeper and more complex, recursive data structures are an interesting way to address the problem of blocking for performance.

The starting point is once again a 2by-2 blocking of the equation $A = GGT$:

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} G_{11} & 0 \\ G_{21} & G_{22} \end{bmatrix} \begin{bmatrix} G_{11} & 0 \\ G_{21} & G_{22} \end{bmatrix}^T.$$

However, unlike in (4.2.18) where A_{11} has a chosen block size, we now assume that $A_{11} \in \mathbb{R}^{m \times m}$ where $m = \text{ceil}(n/2)$. In other words, the four blocks are roughly the same size. As before, we equate entries and identify the key subcomputations:

$G_{11}G_{11}^T = A_{11}$	half-sized Cholesky.
$G_{21}G_{11}^T = A_{21}$	multiple right-hand-side triangular solve
$\bar{A}_{22} = A_{22} - G_{21}G_{11}^T$	symmetric matrix multiplication update
$G_{22}G_{22}^T = \bar{A}_{22}$	half-sized Cholesky.

Our goal is to develop a symmetry-exploiting level-3rich procedure that overwrites A with its Cholesky factor G . To do this we introduce the mixed packed format. An $n=9$ example with $A_{11} \in \mathbb{R}^{5 \times 5}$ serves to distinguish this layout from the conventional packed format layout:

1	
2 10	
3 11 18	
4 12 19 25	
5 13 20 26 31	
6 14 21 27 32	36
7 15 22 28 33	37 40
8 16 23 29 34	38 41 43
9 17 24 30 35	39 42 44 45

Packedformat

1	
2 6	
3 4 10	
4 8 11 13	
5 9 12 14 15	
16 20 24 28 32	36
17 21 25 29 33	37 40
18 22 26 30 34	38 41 43
19 23 27 31 35	39 42 44 45

Mixedpackedformat

Notice how the entries from A_{11} and A_{21} are shuffled with the conventional packed format layout. On the other hand, with the mixed packed format layout, the 15 entries that define A_{11} are followed by the 20 numbers that define A_{21} which in turn are followed by the 10 numbers that define A_{22} . The process can be repeated on A_{11} and

$A_{22}:$

1								
2	4							
3	5	6						
7	9	11						
8	10	12						
16	20	24	28	32	36			
17	21	25	29	33	37	38		
18	22	26	30	34	39	41	43	
19	23	27	31	35	40	42	44	45

Thus, the key to this recursively defined data layout is the idea of representing square diagonal blocks in a mixed packed format. To be precise, recall the definition of `vec` and `symvec` in (4220) and (4221). If $C \in \mathbb{Q}^{N \times N}$ is such a block, then

$$\text{mixed}(C) = \begin{bmatrix} \text{sym}(C_{11}) \\ \text{vec}(C_{21}) \\ \text{sym}(C_{22}) \end{bmatrix} \quad (4222)$$

where $m = \text{ceil}(q/2)$, $C_{11} = C(1:m, 1:m)$, $C_{22} = C(m+1:n, m+1:n)$, and $C_{21} = C(m+1:n, 1:m)$. Notice that since C_{21} is conventionally stored, it is ready to be engaged in a high performance matrix multiplication.

We now outline a recursive divide-and-conquer block Cholesky procedure that works with A in packed f mat. To achieve high performance the incoming A is converted to mixed f mat at each level of the recursion. Assuming the existence of a triangular systemsolve procedure $\text{TriSd}(f \ r \ \text{the system } G2|G1 - A2)$ and a symmetric update procedure $\text{SymUpdate}(f \ r \ A22_+ \ A22_- \ G2|G1)$ we have the following framework.

```

function G= PadedBlockCholesky(A)
{A and G in packed f mat}
n= size(A)
if n < nmin
    G is obtained via any level-2 packed f mat Cholesky method.
else
    Set m= ceil(n/2) and overwrite A packed f mat representation
        with its mixed f mat representation
    G11= PadedBlockCholesky(A11)
    G21= TrSd(G11,A21)
    A22= SynUpdate(A22,G21)
    G22= PadedBlockCholesky(A22)
end

```

Here, m_{\min} is a threshold dimension below which it is not possible to achieve level-3 performance. To take full advantage of the mixed format, the procedures `TriSd` and `SyndUpdate` require a recursive design based on blockings that halve problem size. For example, `TriSd` should take the incoming packed format `A11`, convert it to mixed format, and solve a 2-by-2 blocked system of the form

$$\left[\begin{array}{c|c} X_1 & X_2 \end{array} \right] \left[\begin{array}{cc} L_{11} & 0 \\ L_{21} & L_{22} \end{array} \right]^T = \left[\begin{array}{c|c} B_1 & B_2 \end{array} \right].$$

This sets up a recursive solution based on the half-sized problems.

$$X_1 L_{11}^T = \mathbf{Bi},$$

$$X_2 L_{22}^T = B_2 - X_1 L_{21}^T.$$

Likewise, `SynUpdate` should take the incoming packed format A22, convert it to mixed format, and block the required update as follows:

$$\begin{bmatrix} C_{11} & C_{21}^T \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{21}^T \\ C_{21} & C_{22} \end{bmatrix} - \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix}^T.$$

The evaluation is recursive and based on the half-sized updates

$$C_{11} = C_{11} - Y_1 Y_1^T,$$

$$C_{21} = C_{21} - Y_2 Y_1^T,$$

$$C_{22} = C_{22} - Y_2 Y_2^T.$$

Of course, if the incoming matrices are small enough relative to n_{min} then TriSd and SynUpdate carry out their tasks conventionally without any further subdivisions.

Overall, it can be shown that PackedBlockCholesky has a level-3f action approximately equal to $1_f O(nm/n)$.

Problems

P4.2.1 .2. .8.8 .6.H = A + iB QIAEBQG221 1=Q1Q2MAQQAQ< A B EF(c(- yQhA2= Iy21 xHHx > 23J0> O>x=- m2+y(L m2)

$$c = \begin{bmatrix} \bullet \\ ; \end{bmatrix}$$

$$V = \frac{1}{2} \operatorname{Re} \left(\int_{\Gamma} \bar{A}(x) A(x) dx \right) + \frac{1}{2} \operatorname{Im} \left(\int_{\Gamma} \bar{A}(x) B(x) dx \right)$$

P4.2.2 .2.8 8AEF(c) QelbAnEQa2x 1=Q1Q2AavQn1)Q2A 2s 1EQyb E ablnQ1 2
JAEfECA12E BRCXRF + - o E E Rfa = RBT

$$P4.2.3 \quad 58 \quad AE + - nG = O(1) \otimes n \otimes O(1) \quad (A+AT)/2 \otimes I \quad S = (A - AT)/2 \quad t_2 + (L \\ ny) \| A - m \leq \| r - 1 \quad 2I \otimes A^T \quad \| x \leq xTr - \| x \otimes E \otimes x \otimes EF(\quad , \quad ty(L \otimes Q) = LDMT, y) \\ 1/r \quad 1 \otimes G \otimes I_m$$

P4.2.4 fe... 0...10...o...b A LQy nyA E1E2nxTx... 2e= J@ = b@ => OE1=

triaos 6skeaz) (a(s)le tekaL

P4.2.6 .686.6.. .68b2...8. fr ? $\sqrt{x^T(z)}$ || = Q1 v1 4vRnQ2x 011 T,l Q4=ThT2A
xA)QnA

P4.2.7 N8.. 1..8..6. P10₆8.6..b .68..8.88.8b.8..8..8 J. 8. .. 8.82..8.8.6.68.
 .68...8..6.1..8..8.8.8.6.. < 2a =OO= x O =a =Oo«

P4.2.8 88.8s8. 8l.8. -8.2.8 8....8. 8b1..8... f. P101. . ..b.. ...s8.8....8. 8b
1.-8..6. P10101

P4.2.9 1.2.8 .6.. $E_{c \text{ rxn}} = T_A \Delta H_{rxn} - T_B \Delta H_{rxn}$ $= T_A (Q_{rxn}) - T_B (Q_{rxn})$

P4.2.10 .2.8 .8 $\exists_{+uu}^r 1x = \text{ERnnn}2x \parallel_u \exists_{\neg c}^y x \neq y \quad \text{H!x} \quad 1 \text{ Ex H!v} !$

P4.2.11 2.8.8 ERnxnQ=1 bArEta1=TrQMA)Tr22x n2nTr_y(1A=r1 2an(ET=2RQs2)

$$8\mathbf{Ae}_k - \mathbf{I}_N(:,\mathbf{k}), \mathbf{k} \in \mathbf{S} \quad i < j \quad \text{cos}(\alpha_{ij}) \cdot \frac{\partial}{\partial T_i} \mathbf{T}^T \mathbf{T}^{-1} \mathbf{P}_i \mathbf{P}_j + \alpha(e_i e_i^T + e_j e_j^T)$$

P4.2.12 .686 .6.. .b

[
:
]

$$S = C - B^T A^{-1} B.$$

P4.2.13 .2..8 .8.u ER 2x_u ERn, MjxAlLy2ha(xTrQ)-a2 LA) x 22ETx EJ Xn=(ny2h
 X(I + uuu aX U R JT2A 2 AhaQA)28 1EQmbE a(bln Q) X QTfAxQ-n-

P4.2.14 .2..8 .8 D ~~E~~ Ct C 1-~~F~~ 2t= a ~~i~~M.aena.SanteVilpMt.RpluA.tMni
t,a sepiAantp.Mtha uetpM(D + CCT)- 1 LyAEAC EF xr. IQn M=AyA+yAEb23 (FEQ=01
I (x;cEl Ebs2

P4.2.15 .l..8 .8 f 65 .8...28l... .. 8.8... 8...8.6...8.... .. 8...8
 .8...85eb F#5 RE11EW P4 4I 4 4Ik f'(O?)

P4.2.17 m688... 8b.6.. 8 .8... 8 .686.6.. 8.. 68.8... 8b... .8... 8b.866.b.82 6.. 8 ..

bl.b 2.....8. s.8...6.6 68. 8l 6.8 .. 8r2.s.bl.b ..8..8. ...8...6.6 ...

$$C = \begin{bmatrix} I_n & 5 \\ 0 & I_n \\ 0 & - \end{bmatrix}, \quad M_R$$

$$\begin{bmatrix} G_{11} & 0 \\ G_{21} & G_{22} \end{bmatrix}^{-1} = \begin{bmatrix} G_{11}^{-1} & 0 \\ -G_{22}^{-1}G_{21}G_{11}^{-1} & G_{22}^{-1} \end{bmatrix}.$$

kan lGApap Rpiianapes Tl'30 lTr Tl'f. =(-:-; EK ((:-: :| Fx
EN) =W KG

Notes and References for §4.2

fe8.. .1.8.. .88.8... 88.28.. 8b.8...8 8 ..8.8.88or

P... (344) 7 Positive Definite Matrices, (IMatln,nMatlAMtaAA(BMatln3)1

JamMtaMtaA6gemetMJu xTAx.en Pa6anS oaaAteoSMAlJaMmMaMmMmJIAuetMJuJa,nmaAMMallauwill afelA8aBamSA.IatMlletMleMleMle tMe8alet,A iMIMAlAIIeoA AlAMtsn,Miet,MaAAAatAntJa,nAu j t,MAIAMtMmMtaAllo8auela - SA,AMam

4. 9. esa. (11()) 7 : 6 O(()) ,.,(, A = I + H, H + rAld+ IIbArxTaPz. Angew. Math. Mech. 54 MMN N H.

4 9.e8a . f f

On a Gears by H.A. S. Profosia 12. Dat 1 refmu bpsn (IMA J. Numer. Anal. 17, Mf u9)

aaepna(A(d)=j d)(d)c))Adn)n= f d(B)dpcj ()j d=kEd(r)) n1,1=k)(= de2

ul,1 :d2Bnj =.05 k ibj,f)=2 n(n10),ijj ()1dA 8d)dj(,j d=(Ee=)(.np1=i.mLin Alg Applic. 103, Mf MMyw

ka n,uaIM.e.SAA2aASetam8MtAcuSr mapste m3cpuMletSIncmMn,

(1) 1 senAcrenn(111 1-eul, r MId)luAotSni huuatSneM5.ca =ShiAlAStSlnSe r Seni.sept lpSTSI. SIAM J. Matrix Anal. Applic. 23, 66E6r y, P. i, emm PS,pmen N6d,chtphi 5eneM5,cesM,IVaAe,et lpMSIStn3TNA 17, Iuf tNw

ka SAA,at A.IIIl2nisa.asrAapRpuenlnAeeamr Rp)17Ae ep mSA,AAsam

.i1 i,AteAhn r MId c5a.opASlaemAt 4tue tS.I,pSe' sA1,eSnp,aAa,Snaelr 4sia' pe 4VlpSuA3SIBM J. Res. Dev. 41, IEIfri uw

.i1 ioAteAhn3l sanpMeAS3S-An3r T M iAtp 3amS.,Mnir MId c5a,pAMrc skam .et lpusetAnm9,4h.A vp.anA, 'SnaepSia' Ie psVtASApplied Par lld Computing Lar e Scale Scientif c and Industrial Problems, ,at,pa altaS,luA.tap hSan.3BAIMpl I3Vef8 6Mw3Mt N luw

.i1 i,AtAhn enm .lnAAhn Nld, cPMnSlnpeiasSi,M(cpRpuen)lVaAe .ttlpSTetSln MeSlsr Stein53, pAS3SIBM J. Res. Dev. 44 yNEf y6t

r .h4n-cpA31 u1nMa8AeS3nly i,AteAhn r N dy4 5a.pAS.alpusetSln ln),lsaAe e.tlpSTetSln, PetpS8r(ee3mhtpeia ACM T ns. Math Sof w. 27, Ni6N66k RTRVupl3 i,AteAhn3S.lnAAhn8m r 6 M r Nld,08, pAS.a Vlct,lpSt,uAenm s' pSm http,paARpanAPetpS8M' http8epasSIAM Review 46. 6r :

.i. i,SteAhn3 leuFnMc&eSB1 .lnfeppe3enml ,enil, r Nld c5c.teni,V p ssseeam IIuet RI),IVcAe.A4VlpMT e.tlpSTeMlnks,tSl3ennnsncl ACM T ans. Math Sof w. 37, 4ptS.VMtw

It,ap ,Si,MAapvpuer),IVcAe SuA= auantetSIn= ,ma,

.i. i,AteAhn3l, M,pVAhn8r M iAtp r 13dwcsATpS ,tah(),lsiAe HtlpSTetMln 4VlpMt,uASmtacepr 1AtSuleCmosSniACM T ans. Math Softw. 36 ,pts.VMMk i1 9essepm1.auu as o. o= <aa b=0OEJ3E=N20 2pde= x x(Ea = (= - Xaf = 0x.00 00n.0= EJ0a< Z,SIAM /. Sci. Comput. 32, E6trEr N.w

(6) SanSS .i.n tap8nm 564ymciaSln r Nyl c. uS= fiaAVlpMt,u5a= etath t,a - napAShe huuat ssASStSa nS PetpS8 ACM T ans. Math. Sof w. 35, 4ptS.sa EY Pe.1(3tel.M ennSylhtenSSpl.Sr Nld9tianapesS PetpSf n.apAMtsemapt,n, PetIMf Po=tSSASln3S Comput. Appl. Math. 280, NI INyN,

4.3 Banded Systems

In many applications that involve linear systems, the matrix of coefficients is banded. This is the case whenever the equations can be ordered so that each unknown x_i appears in only a few equations in a "neighborhood" of the i th equation. Recall from §1.2.1 that $A = (a_{ij})$ has upper bandwidth q if $a_{ij} = 0$ whenever $j > i + q$ and lower bandwidth p if $a_{ij} = 0$ whenever $i > j + p$. Substantial economies can be realized when solving banded systems because the triangular factors in LU, GGT, and LDLT are also banded.

4.3.1 Band LU Factorization

Our first result shows that if A is banded and $A = LU$, then L inherits the lower bandwidth of A and U inherits the upper bandwidth of A .

Theorem 4.3.1. Suppose $A \in \mathbb{R}^{n \times n}$ has an LU factorization $A = LU$. If A has upper bandwidth q and lower bandwidth p , then U has upper bandwidth q and L has lower bandwidth p .

Pr **b** The proof is by induction on n . Since

$$A = \begin{bmatrix} a & w^T \\ v & B \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ v/a & I_n \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & B - vw^T/a \end{bmatrix} \begin{bmatrix} a & w^T \\ 0 & I_n \end{bmatrix}.$$

It is clear that $B - vw^T/a$ has upper bandwidth q and lower bandwidth p because only the first q components of w and the first p components of v are nonzero. Let $L_1 U_1$ be the LU factorization of this matrix. Using the induction hypothesis and the sparsity of w and v , it follows that

$$L = \begin{bmatrix} 1 & 0 \\ v/\alpha & L_1 \end{bmatrix}, \quad U = \begin{bmatrix} \alpha & w^T \\ 0 & U_1 \end{bmatrix}$$

have the desired bandwidth properties and satisfy $A = LU$. ■

The specialization of Gaussian elimination to banded matrices having an LU factorization is straightforward.

Given $A \in \mathbb{R}^{n \times n}$ with upper bandwidth q and lower bandwidth p , the following algorithm computes the factorization $A = LU$, assuming it exists. $A(i,j)$ is overwritten by $L(i,j)$ if $i > j$ and by $U(i,j)$ otherwise.

```

for k = 1:n-1
    for i = k+1:min{k+p,n}
        A(i,k) = A(i,k)/A(k,k)
    end
    for j = k+1:min{k+q,n}
        A(i,j) = A(i,j) - A(i,k) · A(k,j)
    end
end
end

```

If $n \gg p$ and $n \gg q$ then this algorithm involves about $2pqf$ ops. Effective implementations would involve band matrix data structures; see §12.5. A band version of Algorithm 4.1.1 (LDL^T) is similar and we leave the details to the reader.

4.3.2 Band Triangular System Solving

Banded triangular systemsolving is also fast. Here are the banded analogues of Algorithms 3.1.3 and 3.1.4.

~~4.3.1. Matrix Elimination Methods~~ Let $L \in \mathbb{R}^{n \times n}$ be a unit lower triangular matrix with lower bandwidth p . Given $b \in \mathbb{R}^n$, the following algorithm overwrites b with the solution to $Lx = b$

```

for j = 1:n
    for i = j + 1:min{j + p, n}
        b(i) = b(i) - L(i,j)·bj
    end
end

```

If $n \gg p$, then this algorithm requires about $2npflops$.

~~4.3.2. Matrix Elimination with Pivoting~~ Let $U \in \mathbb{R}^{n \times n}$ be a nonsingular upper triangular matrix with upper bandwidth q . Given $b \in \mathbb{R}^n$, the following algorithm overwrites b with the solution to $Ux = b$

```

for j = n - 1:-1:1
    bj = bj / U(j,j)
    for i = max{1,j - q}:j - 1
        bi = bi - U(i,j)·bj
    end
end

```

If $n \gg q$, then this algorithm requires about $2nqflops$.

4.3.3. Band Gaussian Elimination with Pivoting

Gaussian elimination with partial pivoting can also be specialized to exploit band structure in A . However, if $PA = LU$, then the band properties of L and U are not quite so simple. For example, if A is tridiagonal and the first two rows are interchanged at the very first step of the algorithm, then U_{11} is nonzero. Consequently, row interchanges expand bandwidth. Precisely how the band enlarges is the subject of the following theorem.

Theorem 4.3.2. Suppose $A \in \mathbb{R}^{n \times n}$ is nonsingular and has upper and lower bandwidths q and p , respectively. If Gaussian elimination with partial pivoting is used to compute Gaussian transformations

$$M = I - \alpha \otimes J \quad j = 1:n-1$$

and permutations P_1, \dots, P_{n-1} such that $M_{P_1 \dots P_{n-1}} M P A = U$ is upper triangular, then U has upper bandwidth $p+q$ and $O(1) = 1$ whenever $i < j$ or $i > j + p$.

Proof. Let $PA = LU$ be the factorization computed by Gaussian elimination with partial pivoting and recall that $P = P_{n-1} \dots P_1$. Write $P^T = [e_{\sigma(1)} \dots e_{\sigma(n)}]$, where $\{\sigma(1), \dots, \sigma(n)\}$ is a permutation of $\{1, 2, \dots, n\}$. If $\sigma(i) > i + p$, then it follows that the leading i -by- i principal submatrix of PA is singular, since $[PA]_{ii} = a_{\sigma(i), i}$ if $\sigma(i) = i$; $a_{\sigma(i)-p, i}$ if $\sigma(i) = i + p$, and 0 otherwise. This implies that U and A are singular, a contradiction. Thus

$i + p \leq r \leq i + q$ and therefore P_A has upper bandwidth $p+q$. It follows from Theorem 43.1 that U has upper bandwidth $p+q$. The assertion about the A 's can be verified by observing that M need only zero elements $(j+1, j), \dots, (j+p, j)$ of the partially reduced matrix $P_M P^{-1} U - P_A = 0$.

Thus pivoting destroys band structure in the sense that U becomes "wider" than A 's upper triangle, while nothing at all can be said about the bandwidth of L . However, since the j th column of L is a permutation of the j th Gauss vector α_j , it follows that L has at most $p+1$ nonzero elements per column.

1hrnsb Hessenberg LU

As an example of an unsymmetric band matrix computation, we show how Gaussian elimination with partial pivoting can be applied to factor an upper Hessenberg matrix H . (Recall that if H is upper Hessenberg then $h_{ij} = 0$, $i > j + 1$.) After $k-1$ steps of Gaussian elimination with partial pivoting we are left with an upper Hessenberg matrix of the form

$$\begin{bmatrix} \cdot & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}, \quad k=3, n=5$$

By virtue of the special structure of this matrix, we see that the next permutation P_A is either the identity or the identity with rows 3 and 4 interchanged. Moreover, the next Gauss transformation M_k has a single non-zero multiplier in the $(k+1, k)$ position. This illustrates the k th step of the following algorithm.

Algorithm 43.2 Given an upper Hessenberg matrix $H \in \mathbb{R}^{n \times n}$, the following algorithm computes the upper triangular matrix $M_n P_n \cdots M_1 P_1 H = U$ where each P_k is a permutation and each M_k is a Gauss transformation whose entries are bounded by unity. $H(i, k)$ is overwritten with $U(i, k)$ if $i \neq k$ and by $-M_k H(i, k)$ if $i = k$. An integer vector $piv(1:n-1)$ encodes the permutations. If $P_k = I$, then $piv(k) = 0$. If P_k interchanges rows k and $k+1$, then $piv(k) = 1$.

```

for k=1:n-1
    if |H(k,k)| < |H(k+1,k)|
        piv(k) = 1; H(k,k) = H(k+1,k)
        H(k+1:k+1:n) = H(k+1:k+1:n) - r * H(k,k+1:n)
        H(k+1,k) java
    end
end

```

This algorithm requires n^2 fops.

Algorithm 6A Band Cholesky

The rest of this section is devoted to banded $Ax = b$ problems where the matrix A is also symmetric positive definite. The fact that pivoting is unnecessary for such matrices leads to some very compact, elegant algorithms. In particular, it follows from Theorem 4.3.1 that if $A = GGT$ is the Cholesky factorization of A , then G has the same lower bandwidth as A . This leads to the following banded version of Algorithm 4.2.1.

~~mis. eli - mUni. I el a tn III..ha~~ Given a symmetric positive definite $A \in \mathbb{R}^{n \times n}$ with bandwidth p , the following algorithm computes a lower triangular matrix G with lower bandwidth p such that $A = GGT$. For all $i > j$, $G(i,j)$ overwrites $A(i,j)$.

```

for j = 1:n
    for k = max(1,j-p):j-1
        >= min(k+p,n)
        A(j,:>j) = A(j,:>j) - A(j,k)*A(j,:k)
    end
    >= min(j+p,n)
    A(j,:>j) = A(j,:>j)/sqrt(A(j,j))
end

```

If $n \gg p$, then this algorithm requires about $n(p^2 + 3p)$ fops and n square roots. Of course, in a serious implementation an appropriate data structure for A should be used. For example, if we just store the nonzero lower triangular part, then a $(p+1)$ -by- n array would suffice.

If our band Cholesky procedure is coupled with appropriate band triangular solve routines, then approximately $np^2 + 7p + 2n$ fops and n square roots are required to solve $Ax = b$. For small p it follows that the square roots represent a significant portion of the computation and it is preferable to use the LDLT approach. Indeed, a careful flop count of the steps $A = LDL^T$, $Ly = b$, $Dz = y$, and $L^Tx = z$ reveals that $np^2 + 8p + n$ fops and no square roots are needed.

Algorithm 7B Tridiagonal System Solving

As a sample narrowband LDLT solution procedure, we look at the case of symmetric positive definite tridiagonal systems. Setting

$$L = \begin{bmatrix} 0 & \cdots & 0 \\ \bullet & 1 & \vdots \\ \ddots & \ddots & 0 \\ \cdots & \ell_{n-1} & 1 \end{bmatrix}$$

and $D = \text{diag}(d_1, \dots, d_n)$, we deduce from the equation $A = LDL^T$ that

$$\begin{aligned} au &= di, \\ a_{kk} &= l_{k-1}l_k i, & k = 2:n, \\ a_{kk} &= d_{k+1} - l_{k-1}l_k i = d_{k+1} - l_{k-1}l_k i, & k = 2:n \end{aligned}$$

Thus, the d and i can be resolved as follows:

$$d = au$$

$$f \neq k = 2n$$

$$ik \leftarrow akk / dk$$

$$dk \leftarrow akk - ik *akk$$

end

To obtain the solution to $Ax = b$ we solve $Ly = b$, $Dz = y$, and $LTx = z$. With overwriting we obtain

Given an n -by- n symmetric, tridiagonal, positive definite matrix A and $b \in \mathbb{R}^n$, the following algorithm overwrites b with the solution to $Ax = b$. It is assumed that the diagonal of A is stored in $a(1:n)$ and the superdiagonal in $(1:n-1)$.

$$f \neq k = 2n$$

$$t = .(k-1), .(k-1) = t/a(k-1), a(k) = a(k) - t.(k-1)$$

end

$$for k = 2n$$

$$b(k) = b(k) - (k-1).(k-1)$$

end

$$b(n) = b(n)/a(n)$$

$$f \neq k = n-1-1:1$$

$$b(k) = b(k)/a(k) - .(k).b(k+1)$$

end

This algorithm requires $8n$ flops.

Vectorization Issues

The tridiagonal example brings up a sore point: narrowband problems and vectorization do not mix. However, it is sometimes the case that large, independent sets of such problems must be solved at the same time. Let us examine how such a computation could be arranged in light of the issues raised in §1.5. For simplicity, assume that we must solve n -by- n unit lower bidiagonal systems

$$\begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \quad k = 1:m$$

and that $m = n$. Suppose we have arrays $E(1:n-1, 1:m)$ and $B(1:n, 1:m)$ with the property that $E(1:n-1, k)$ houses the subdiagonal of $A^{(k)}$ and $B(1:n, k)$ houses the k th right-hand side $b^{(k)}$. We can overwrite $b^{(k)}$ with the solution $x^{(k)}$ as follows:

$$for k = 1:m$$

$$f \neq i = 2n$$

$$B(i, k) = B(i, k) - E(i-1, k) \cdot B(i-1, k)$$

end

end

This algorithm sequentially solves each bidiagonal system in turn. Note that the inner loop does not vectorize because of the dependence of $B(i, k)$ on $B(i - 1, k)$. However, if we interchange the order of the two loops, then the calculation does vectorize:

```
for i = 2n
    B(i,:) = B(i,:)
        - E(i-1,:)
        - B(i-1,:)
    end
```

A column-oriented version can be obtained simply by storing the matrix subdiagonals by row in E and the right-hand sides by row in B :

```
for i = 2n
    B(:,i) = B(:,i)
        - E(:,i-1) .*
        B(:,i-1)
    end
```

Upon completion, the transpose of solution $x^{(k)}$ is housed on $B(k,:)$.

4.3.8 The Inverse of a Band Matrix

In general, the inverse of a nonsingular band matrix A is full. However, the off-diagonal blocks of A^{-1} have low rank.

Theorem 4.3.3. Suppose

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

is nonsingular and has lower bandwidth p and upper bandwidth q . Assume that the diagonal blocks are square. If

$$A^{-1} = X = \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix}$$

is partitioned as follows, then

$$\text{rank}(X_{21}) \leq p \quad (431)$$

$$\text{rank}(X_{12}) \leq q \quad (432)$$

Proof. Assume that A_{11} and A_{22} are nonsingular. From the equation $AX = I$ we conclude that

$$\begin{aligned} A_{21}X_{11} + A_{22}X_{21} &= 0 \\ A_{11}X_{12} + A_{12}X_{22} &= 0 \end{aligned}$$

and so

$$\begin{aligned} \text{rank}(X_{21}) &= \text{rank}(A_{22}^{-1}A_{21}X_{11}) \leq \text{rank}(A_{22}^{-1}) \\ \text{rank}(X_{12}) &= \text{rank}(A_{11}^{-1}A_{12}X_{22}) \leq \text{rank}(A_{11}^{-1}) \end{aligned}$$

From the handedness assumptions it follows that A_{11} has at most p nonzero rows and A_{22} has at most q nonzero rows. Thus, $\text{rank}(A_{11}) \leq p$ and $\text{rank}(A_{22}) \leq q$, which proves the theorem for the case when both A_{11} and A_{22} are nonsingular. A simple limit argument can be used to handle the situation when A_{11} and/or A_{22} are singular. See P4311. 7

It can actually be shown that $\text{rank}(A_{11}) = \text{rank}(X_{11})$ and $\text{rank}(A_{22}) = \text{rank}(X_{22})$. See Strang and Nguyen (2004). As we will see in §11.59 and §12.2, the lowrank off-diagonal structure identified by the theorem has important algorithmic ramifications.

11.59 Band Matrices with Banded Inverse

If $A \in \mathbb{R}^{N \times N}$ is a product

$$A = F_1 \cdots F_N \quad (433)$$

and each $F_i \in \mathbb{R}^{N \times N}$ is block diagonal with 1-by-1 and 2-by-2 diagonal blocks, then it follows that both A and

$$A^{-1} = F_N^{-1} \cdots F_1^{-1}$$

are banded, assuming that N is not too big. For example, if

$$A = \begin{bmatrix} x & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & x & x & 0 & 0 & 0 & 0 & 0 \\ 0 & x & x & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & x & 0 & 0 & 0 \\ 0 & 0 & 0 & x & x & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & x & x & 0 \\ 0 & 0 & 0 & 0 & 0 & x & x & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & x \end{bmatrix} \quad \begin{bmatrix} x & x & 0 & 0 & 0 & 0 & 0 & 0 \\ x & x & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & x & x & 0 & 0 & 0 & 0 \\ 0 & 0 & x & x & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & x & x & 0 & 0 \\ 0 & 0 & 0 & 0 & x & x & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & x \end{bmatrix}$$

then

$$A = \begin{bmatrix} x & x & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ x & x & x & x & 0 & 0 & 0 & 0 & 0 \\ x & x & x & x & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & x & x & x & x & 0 & 0 & 0 \\ 0 & 0 & x & x & x & x & 0 & 0 & 0 \\ 0 & 0 & 0 & x & x & x & x & 0 & 0 \\ 0 & 0 & 0 & x & x & x & x & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & 0 & 0 & x & x & x & x \end{bmatrix}, \quad A^{-1} = \begin{bmatrix} x & x & x & 0 & 0 & 0 & 0 & 0 & 0 \\ x & x & x & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & x & x & x & x & 0 & 0 & 0 & 0 \\ 0 & x & x & x & x & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & x & x & x & 0 & 0 \\ 0 & 0 & 0 & x & x & x & x & 0 & 0 \\ 0 & 0 & 0 & 0 & x & x & x & x & x \\ 0 & 0 & 0 & 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & x & x \end{bmatrix}.$$

Strang (2010a, 2010b) has pointed out a very important "reverse" fact. If A and A^{-1} are banded, then there is a factorization of the form (433) with relatively small N . Indeed, he shows that N is very small for certain types of matrices that arise in signal processing. An important consequence of this is that both the forward transform Ax and the inverse transform $A^{-1}x$ can be computed very fast.

Problems

P4.3.1 88. 8e.8...8. 8bl..8.. m.. P1966c..I.2.8. m.em68.cb A Q=I(FAI)wH)EHn =nIAZAA5 254

P4.3.2 .386 686m.882..2ngb1..8..m6. P196P 2.8 m88..8 m68 2.8.8. 8....8 n Hx=b

P4.3.3 ..86 .86 1..8..6. P191P2.. 8.8 .8..o e.868.68 ro8...8. Hx=b

P4.3.4 Nc.8 ...8..6. N. 8s... e. 2....8m... m.c.8.. ...m8. Ax= bly2,)A=)2==T2 AIIb1)A() LTIChETHII2(Q1 fn=y(k FA ,QEA)s 1 1Ef x2Al(E(x2Q1 T)n =l(EHAE ICAAl(EIx2H)4

P4.3.5 f.5 8C ERnxnxA)A nyApfle index m(C,i) = bQ { # Q, 8Japa = f. +CL IC2 - ,A = GGT T=ICA,C(I=A=r1,2anE x2Q0), A ,ICA)m(A,i) = m(G,i) E i = l..n Z2 =H1 IC2(G Ck nCA=2A pr f lek A) ZX, 1f=A A ERnxnQ= bAlEQal=QIT2AA)T=LTrFE(1A TxTaAm = m(A,i) LyAEA = f. l=,BA ny nA T=n(FAT)H (A x1lf)=Q221v. e<< = 3or

$$v = a_1, a_2, \dots, a_2, a_3, \dots, a_3, \dots, a_n, \dots, a_m.$$

J2A 21EQrCbnv2 (2AEfTA=30= • O= = = OofO= .0000yJAJ= < Obae=E G H)x1CA) ,=A= I(254(EIx2H)4(s2Ax= b I(L b21 x1e HFA ,TEA1zAxE C ERnxn2A p(C,i) = b2x { E# 1 k h,AAIAA E,4nyk 2 8M,Ha(EIx2H)A = LU 2x ly2

$$\begin{array}{llll} m(A, 1) & m(A, 2) & \leq \dots & m(A, n), \\ p(A, 1) & p(A, 2) & \leq \dots & p(A, f). \end{array}$$

+y(L ny2 m(A,i) = m(L,i) 21 p(A,i) = p(U,i) E i = fB

P4.3.6 88.8.8.. eb.. 8...8. 8bl..8.. m.. 431

P4.3.7 88.8.8.. 2..86.8.8.. e e88..m6.N. .8.... m68.0.... .8..m..8 8 .cm8 ..8.e. ...8.. A(l)xl(k) = b(k), l=,ba nCA nyA1 20H1=P, fEIIH)1 12x EQyH)1=TAHEA =l(EAx1 E(L T) 2E2D, E, ("") B 2x IC2 b(k) - (AHE mA)Tnx(k).

P4.3.8 N.8 e. 8be...8 8be9 18m... .8.cm... .m8.b 668.8....e.8.es .e.m. .8. .8..m..88 .m81

P4.3.9 2. .8.88.... .8 .m.8 ..8...b A ERnxnCk nyA@HFIHEan fzA.PP

$$A = \begin{bmatrix} x & x & x & x & x \\ x & x & = & = & 0 \\ x & = & x & = & 0 \\ x & = & = & x & 0 \\ x & = & = & = & x \end{bmatrix}.$$

2.a $\frac{1}{k} \frac{1}{x} \frac{1}{x} + x \frac{1}{x} - x \frac{1}{x} = Ax + bx + H$, A = (2AxLTh) ZfX) = Q1nyATCAEH) d (EE)9 I (1w,E1 1b)H ZwiXrAEB)AHAfBrHnQ) bnEQxP = (IC1H nyA=r1 2al(EQxH)Q)

$$PAP^{-1} = GGT$$

aH)w A(b)lnAx L- IC ZfX)l=

P4.3.10 .2..8 .8 A ERnxn T=nEtx2)2P1(- IT2AA)- nf Rv,n)(l = 1bAnETa) J22 HMa A H1(EThyB E a(b)T)1 ICA 1HEA=A)E1 (|ST^-1 S|_11^P -S = (A - AT)/2H1T = (A + AT)/2

P4.3.11 .686 .6.. .bAER'xn2x e > 2a • O=• O=O B ERnxn=,ay ICHAv B WW B yk rCA E(AfBr2s Th=EI)a21 =,bAEtaA= ZE)l=Q12E M=AyT= EA(sH2sK a(b)ArAIC A E(-yAFAb -u+u

P4.3.12 Nc.8e. 2..8. .82.. 8. m.. e.6..6 8bm6gn.b A T)Z - u+uX-

P4.3.13 ..86 m6gnT2x A^-1 . $\begin{pmatrix} T & T \\ j & i \end{pmatrix}, \begin{pmatrix} T & T \\ i & i \end{pmatrix}, \begin{pmatrix} T & T \\ j & i \end{pmatrix} + \begin{pmatrix} T & T \\ i & i \end{pmatrix}_{11} + \begin{pmatrix} -1 & -1 \\ 1 & 1 \end{pmatrix}^P$, e V V1

P4.3.14 8.68A = F1F2 tRAcJ0t 1373,(s(M2)uA) A(23,), A(45,), A(67,)P- 4Hya0=TA (, rL(=T)1HE 3lw,14r=4

Notes and References for §4.3

Notes and References for §4.3

N 8.8.m.m.88.. 8. m6@8..8b..8. ...mo. ..s2.8

1.1N..m. e. k1Pf1sd..8. f.8P52....8m... 88.8.8..m.8.8b38..m..88 ..m8. N.m..
 .8.67 Numer. Math 7, Er rMu

2. h. rep 4Senn1.s. uSVe8Alir Mtd chIV, 4Slnmh.uua4pSennnA.uua4pS9ennR6, 4SlnA
 emn1.a)eV, Ve4SlnRSian.eV, bA enme4pS.aANumer. Math 9, NI tf EM.
 x.l. 4Vil8ap itEd. crf t4Sn.apAhAap4eS9ennre4pS.aANumer. Math 21, NI tf Ny6
 z.) = J.O.20 pD) = . = 0 e j = . = O (= y = (= JOD= .0o(a = (x ∈ 1(e=a = @eo.0x oF N
 Inst. Math. Applic. 16 MET M6N.

1. Menünr. 11, cc, a 5a4pe4Sln 814, up e.4pSn9,nmarhuua4pSr, 4pS.aATNumer.
Lin. Alg. Applic. 14. NEIf Nr. 6

}. Il uAv enhVaulnANt.. cc8SA4t4l pSTe4Slme 9enmar4Sf 3IT 49 6EE 66I.

Sailor W. Maury Alcove/S. V. N. Alcove/Maury

3 SA 54 A 6 M1 61A 4S A

e. 1.5Aar. Mtd c4n4VlpS4wphIV.Sni hAS.WVTAnr SrlsSVhA4auAnxSnaeR6e

P4 revIVuepmj (evuarr Mt c4 14 ra4lmvp hA Spi)s1A lpr SrlSeVhA4auA

P.4 rev. vu en l'evuap M₁ C4.4 Ra₄ Impfia. S11)SIA m₁ S11 S11 S11 S11 S11
ISna R6 e18 Commun ACM 17, M6f MI.
2.1 S11 S11 M₁ C8 S11 Vb S11 Vb

a.1 SS_1, eu_1 r. $\text{Mfd cr. San 4V1pS4wmp}$ $\|_{\text{uA}}, \text{Sni 4,a} \rangle \text{InmS4Shu capine r Smhev$
 r, 4pSf STAM J. Sci. Stat. Comput. 7, ir. fl Mur.

a.1 sSi,eu rTM . cr l,nmSriaRpp!Spn,AASeRVSusShlvp r SmSeVhA4auASIA
 J. Matr. Anal. Applic. 11, r NMF1.
 S. G. HUANG, L. M. LEE, C. Y. LIN, C. Y. LIN, S. G. HUANG, C. Y. LIN, S. G. HUANG

S.h, SVW r thy., cSaVSeMA, 4e4Slnh4.a)lnms4Sh, u capn, r SniSeSre4pSS10(n)
 cSu&SIAM J. Matr. Anal. Applic. 19 IIu5tu.

S.9epQ1enmr. xalnSns Nid c5aVSeMw,4Slmr SrlsSeVhA4auAnxSnaeP6,e4SlmA
 SIAM J. Numer. Anal. 98 MEM6 Mr E.

P.S9.anl,nmr. IAS.1 r N6. ch4eoSVStrhanASS.94nr SmSeikve4 lpST e4854,4
 (S14Sn)3H 44 Ar NIE

15. 9.n. enmб.. rep.Ser Nid. c4 hSuAVS(SltSni h4pe4ai.vp huua4pSr SmSeil
rc4pS 2A8mн Lin Aks 19 var. LVI

reaps a consumer. Lin. Alg is your 1yu
homogeneous system of linear equations. A solution is a vector

sh b4lnar Jtr d(opeVr SuiSeVR6e4SlrhIV anASTM T ps. Math. Soc. w. 1 Nytf. EJ

hA4auAnUm. Lin. Alg. & NtlfMu.
Q.R. a - 0a = N2ZpDXa - 0a <- 0=&a 0g0x0.' (E (0aa& (= 0eo-0o=Na
lled

*Aap₁A₄,pa,ln,apnaraSS4,4,aA4p,4,pn4,aSn,hpAa, oenpm4pSSn,V,ma,
4AAAAnMtdSn,anAaP,4pS,Aa::} u. S, he4SAa =*

{ 4 rS-aAVS Mnd c9enman4pS aSS49enmasn anA3A Comput Appl Math 41

NyMt E
s h^lppenig ai an r N^l6cc a S^ln4prΛV^l5eneΛph q u t o S^l AFIAM Raizay 46 uEL 6w

S. naperin*et al.*,¹ an r. Nod.c,a Sh4apAVeeneAnh,oue4pS.aS4M. Review 46 UET by
S. o4penir INAd c.14 r enAvpuQenmane4pS.aS49enmasm.apA3P r c. National Acad.
Sci. 197 MNCS 1974

s. o4peni Nibd c9enm3n4ps.a&S 49enmaSmnapAAn = LPU," Pr ceedings International Conference on Science and Technology, 2012.

¹See also Qian Lin, Gao Jiong, and Guo Yihong, "A New Proof of the Geometric Ergodicity of the Metropolis-Hastings Algorithm," *Statistica Sinica*, 1999, 9, 113-120.

R = a00nA ==> a0Ra = 10 A. ==> N220Matr Computations and Semiseparable

Matrices, Volume I Linear Systems, II, II ASIAESTHETICASAPASPAASV4Sulpass ETI of

4.4 Symmetric Indefinite Systems

Recall that a matrix whose quadratic form $\mathbf{x}^T \mathbf{A} \mathbf{x}$ takes on both positive and negative values is indefinite. In this section we are concerned with symmetric indefinite linear systems. The LDL^T factorization is not always advisable and the following 2by2 example illustrates:

$$\begin{bmatrix} \epsilon & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1/\epsilon & 1 \end{bmatrix} \begin{bmatrix} \epsilon & 0 \\ 0 & -1/\epsilon \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1/\epsilon & 1 \end{bmatrix}^T.$$

Of course, any of the pivot strategies in §3.4 could be invoked. However, they destroy symmetry and, with it, the chance for a "Cholesky speed" symmetric indefinite system solver. Symmetric pivoting i.e., data reshufflings of the form $A \rightarrow PAPT$, must be used as we discussed in §4.2.8. Unfortunately, symmetric pivoting does not always stabilize the LDL^T computation. If ϵ_1 and ϵ_2 are small, then regardless of P , the matrix

$$\tilde{A} = P \begin{bmatrix} \epsilon_1 & 1 \\ 1 & \epsilon_2 \end{bmatrix} P^T$$

has small diagonal entries and large numbers suffice in the factorization. With symmetric pivoting the pivots are always selected from the diagonal and trouble results if these numbers are small relative to what must be zeroed off the diagonal. Thus, LDL^T with symmetric pivoting cannot be recommended as a reliable approach to symmetric indefinite system solving. It seems that the challenge is to involve the off-diagonal entries in the pivoting process while at the same time maintaining symmetry.

In this section we discuss two ways to do this. The first method is due to Amdahl (1971) and it computes the factorization

$$PAP^T = LTL^T, \quad (441)$$

where $L = (l_{ij})$ is unit lower triangular and T is tridiagonal. P is a permutation chosen such that $|l_{ij}| \leq 1$. In contrast, the diagonal pivoting method due to Bunch and Parlett (1971) computes a permutation P such that

$$PAP^T = LDL^T, \quad (442)$$

where D is a direct sum of 1by1 and 2by2 pivot blocks. Again, P is chosen so that the entries in the unit lower triangular L satisfy $|l_{ij}| \leq 1$. Both factorizations involve $n^3/3$ flops and once computed, can be used to solve $\mathbf{Ax} = \mathbf{b}$ with $O(n^2)$ work.

$$PAP^T = LTL^T, \quad Lz = Pb, \quad Tw = z, \quad LTy = w, \quad x = PTy \quad \Rightarrow \quad Ax = b$$

$$PAP^T = LDL^T, \quad Lz = Pb, \quad Dw = z, \quad LTy = w, \quad x = PTy \quad \Rightarrow \quad Ax = b$$

A few comments need to be made about the $Tw = z$ and $Dw = z$ systems that arise when these methods are invoked.

In Amdahl's method, the symmetric indefinite tridiagonal system $Tw = z$ is solved in $O(n)$ time using band Gaussian elimination with pivoting. Note that there is no serious price to pay for the disregard of symmetry at this level since the overall process is $O(n^3)$.

In the diagonal pivoting approach, the $Dw = z$ system amounts to a set of 1-by-1 and 2by-2 symmetric indefinite systems. The 2by-2 problems can be handled via Gaussian elimination with pivoting. Again, there is no harm in disregarding symmetry during this $O(n)$ phase of the calculation. Thus, the central issue in this section is the efficient computation of the factorizations (44.1) and (44.2).

4.4.1 The Parlett-Reid Algorithm

Parlett and Reid (1970) show how to compute (44.1) using Gauss transforms. Their algorithm is sufficiently illustrated by displaying the $k=2$ step for the case $n=5$. At the beginning of this step the matrix A has been transformed to

$$A^{(1)} = M_1 P_1 A P_1^T M_1^T = \begin{bmatrix} 0 & /1 & 0 & 0 & 0 \\ \beta_1 & 02 & v_3 & v_4 & v_5 \\ 0 & v_3 & \times & \times & \times \\ 0 & v_4 & \times & \times & \times \\ 0 & v_5 & \times & \times & \times \end{bmatrix},$$

where P_1 is a permutation chosen so that the entries in the Gauss transformation M_1 are bounded by unity in modulus. Scanning the vector $[v_3 \ v_4 \ v_5]^T$ for its largest entry, we now determine a 3by-3 permutation P_2 such that

$$\tilde{P}_2 \begin{bmatrix} v_3 \\ v_4 \\ v_5 \end{bmatrix} = \begin{bmatrix} \tilde{v}_3 \\ \tilde{v}_4 \\ \tilde{v}_5 \end{bmatrix} \Rightarrow |\tilde{v}_3| = \max\{|\tilde{v}_3|, |\tilde{v}_4|, |\tilde{v}_5|\}.$$

If this maximal element is zero, we set $M_2 = P_2 = I$ and proceed to the next step. Otherwise, we set $P_2 = \text{diag}(1/\tilde{v}_3)$ and $M_2 = I - a \tilde{P}_2$ with

$$a \tilde{P} = (0 \ 0 \ 0 \ \tilde{v}_4/\tilde{v}_3 \ \tilde{v}_5/\tilde{v}_3)^T.$$

Observe that

$$A^{(1)} = M_2 P_2 A^{(1)} P_2^T M_2^T = \begin{bmatrix} \beta_1 & 0 & 0 & 0 & 0 \\ \beta_1 & 02 & \tilde{v}_3 & 0 & 0 \\ 0 & \cancel{0} & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \end{bmatrix}.$$

In general, the process continues for $n-2$ steps leaving us with a tridiagonal matrix

$$T \text{ ja } A^{(n-2)} = (M_{n-2} P_{n-2} \cdots M_1 P_1) A (M_{n-2} P_{n-2} \cdots M_1 P_1)^T.$$

It can be shown that (44.1) holds with $P = P_{n-2} \cdots P_1$ and

$$L = (M_{n-2} P_{n-2} \cdots M_1 P_1 P^T)^{-1}.$$

Analysis of L reveals that its first column is e_1 and that its subdiagonal entries in column k with $k > 1$ are "made up" of the multipliers in M_{k-1} .

The efficient implementation of the Parlett-Reid method requires care when computing the update

$$A^{(k)} = M_k (P_k A^{(k-1)} P_k^T) M_k^T. \quad (443)$$

To see what is involved with a minimum of notation, suppose $B = B^T \in \mathbb{J}^{(n-k) \times (n-k)}$ has and that we wish to form

$$B_+ = (I - we_1^T)B(I - we_1^T)^T,$$

where $w \in \mathbb{J}^{n-k}$ and e_1 is the first column of \mathbb{J}_{n-k}^k . Such a calculation is at the heart of (443). If we set

$$u = Be_1 - \frac{b_{11}}{2}w,$$

then $B_+ = B - uu^T - uw^T$ and its lower triangular portion can be formed in $2(n-k)^2$ fops. Summing this quantity as k ranges from 1 to $n-2$ indicates that the Parlett-Reid procedure requires $2n^3/3$ fops—twice the volume of work associated with Cholesky.

shsmib The Method of Aasen

An $n^3/3$ approach to computing (441) due to Aasen (1971) can be derived by re-considering some of the computations in the Parlett-Reid approach. We examine the no pivoting case first where the goal is to compute a unit lower triangular matrix L with $L(:, 1) = e_1$ and a tridiagonal matrix

$$T = \begin{bmatrix} \alpha_1 & \beta_1 & & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ \ddots & \ddots & \ddots & \ddots & \\ \vdots & \ddots & \ddots & \ddots & \\ 0 & \cdots & & & f_{n-1} / n \end{bmatrix}.$$

such that $A = LTl^T$. The Aasen method is structured as follows:

```

for j = 1:n
    {a(lj - 1), /(lj - 1) and L(:,lj) are known}
    Compute ai.
    ifj : n- 1
        Compute/i.
    end
    ifj : n- 2
        ComputeL(j + 2n,j + 1).
    end
end

```

(444)

To develop recipes for α_j , β_j , and $L(j + 2n, j + 1)$, we compare the j th columns in the equation $A = LH$ where $H = TL^T$. Noting that H is an upper Hessenberg matrix we obtain

$$A(:, j) = LH(:, j) = \sum_{k=1}^{j+1} L(:, k) \cdot h(k), \quad (445)$$

where $h(1:j + 1) = H(1:j + 1, j)$ and we assume that $j = n - 1$. It follows that

$$h_{j+1} \cdot L(j + 1:n, j + 1) = v(j + 1:n), \quad (446)$$

where

$$v(j + 1:n) = A(j + 1:n, j) - L(j + 1:n, 1:j) \cdot h(1:j). \quad (447)$$

Since L is unit lower triangular and $L(:, 1:j)$ is known, this gives us a working recipe for $L(j + 2n, j + 1)$ provided we know $h(1:j)$. Indeed, from (446) and (447) it is easy to show that

$$L(j + 2n, j + 1) = v(j + 2n)/v(j + 1). \quad (448)$$

To compute $h(1:j)$ we turn to the equation $H = TL^T$ and examine its j th column. The case $j = 5$ amply displays what is going on:

$$\begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \end{bmatrix} = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & 0 & 0 \\ \beta_1 & \alpha_2 & \beta_2 & 0 & 0 \\ 0 & \beta_2 & \alpha_3 & \beta_3 & 0 \\ 0 & 0 & \beta_3 & \alpha_4 & \beta_4 \\ 0 & 0 & 0 & \beta_4 & \alpha_5 \\ 0 & 0 & 0 & 0 & \beta_5 \end{bmatrix} \begin{bmatrix} 0 \\ \ell_{52} \\ \ell_{53} \\ \ell_{54} \\ 1 \end{bmatrix} = \begin{bmatrix} \beta_1 \ell_{52} \\ \alpha_2 \ell_{52} + \beta_2 \ell_{53} \\ \beta_2 \ell_{52} + \alpha_3 \ell_{53} + \beta_3 \ell_{54} \\ \beta_3 \ell_{53} + \alpha_4 \ell_{54} + \beta_4 \\ \beta_4 \ell_{54} + \alpha_5 \\ \beta_5 \end{bmatrix} \quad (449)$$

At the start of step j , we know $a(1:j - 1)$, $/ (1:j - 1)$ and $L(:, 1:j)$. Thus, we can determine $h(1:j - 1)$ as follows:

$$h_1 = \beta_1 \ell_{j2}$$

for $k = 1:j - 1$

$$h_k = \beta_{k-1} \ell_{jk} + \alpha_k \ell_{jk} + \beta_k \ell_{j,k+1} \quad (4410)$$

end

Equation (445) gives us a formula for h_j :

$$h_j = A(j, j) - \sum_{k=1}^{j-1} L(j, k) h_k. \quad (4411)$$

From (449) we infer that

$$\alpha_j = h_j - \beta_{j-1} \ell_{j,j-1}, \quad (4412)$$

$$\beta_j = h_{j+1}. \quad (4413)$$

Combining these equations with (444), (447), (448), (4410), and (4411) we obtain the Aaren method without pivoting.

```

L = In
f r j = l:n
    if j = 1
        α1 = a11
        v(2n) = A(2n, 1)
    else
        h1 = /1\12
        f r k= 2j - 1
            hk = /k lfj,k 1+ Okjk+ fkfj,k+1
        end
        h1 = a11 - L(j, lj - 1) · h(lj - 1)
        G = lj - /j lfj,j-1
        v(j + 1:n) = A(j + 1:n, j) - L(j + 1:n, lj) · h(lj)
    end
    if j <= n - 1
        (1 = v(j + 1))
    end
    if j <= n - 2
        L(j + 2:n, j + 1) = v(j + 2:n)/v(j + 1)
    end
end

```

(44.14)

The dominant operation each pass through the j -loop is an $(n-j)$ -by- j gappy operation. Accounting for the associated flops we see that the overall Aa encomputation involves $n^3/3$ flops, the same as for the Cholesky factorization.

As it now stands, the columns of L are scalings of the v -vectors in (44.14). If any of these scalings are large, i.e., if any $v(j + 1)$ is small, then we are in trouble. To circumvent this problem it is only necessary to permute the largest component of $v(j + 1:n)$ to the top position. Of course, this permutation must be suitably applied to the unreduced portion of A and the previously computed portion of L . With pivoting Aa en's method is stable in the same sense that Gaussian elimination with partial pivoting is stable.

In a practical implementation of the Aa en algorithm, the lower triangular portion of A would be overwritten with L and T , e.g.,

$$A_f \left[\begin{array}{ccccc} a_1 & & & & \\ /1 & a_2 & & & \\ f_3 & f_2 & a_3 & & \\ f_4 & \ell_{43} & \beta_3 & \alpha_4 & \\ f_5 & \ell_{53} & \ell_{54} & \beta_4 & \alpha_5 \end{array} \right].$$

Notice that the columns of L are shifted left in this arrangement.

+0..r Diagonal Pivoting Methods

We next describe the computation of the block LDLT factorization (44.2). We follow the discussion in Bunch and Parlett (1971). Suppose

$$P_1 A P_1^T = \begin{bmatrix} E & \mathbf{0} \\ \mathbf{0} & C \end{bmatrix}_s^s$$

where P_1 is a permutation matrix and $s = \text{rank}(A)$. If A is nonzero, then it is always possible to choose these quantities so that E is nonsingular, thereby enabling us to write

$$P_1 A P_1^T = \begin{bmatrix} C^{-1} E \\ \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} & B \\ B & C^{-1} E \end{bmatrix} \begin{bmatrix} \mathbf{0} & E \\ E & \mathbf{0} \end{bmatrix}.$$

For the sake of stability, the s -by- s "pivot" E should be chosen so that the entries in

$$\tilde{A} = (\tilde{a}_{ij}) \equiv B - CE^{-1}C^T \quad (44.15)$$

are suitably bounded. To this end, let $\alpha = E(0, 1)$ be given and define the size measures

$$\mu_0 = \max_{ij} |a_{ij}|,$$

$$\mu_i = \max_i |a_{ii}|.$$

The Bunch-Parlett pivot strategy is as follows:

```

if  $\mu_i > \alpha\mu_0$ 
     $s = 1$ 
    Choose  $P_1$  so  $|a_{11}| = \mu_1$ .
else
     $s = 2$ 
    Choose  $P_1$  so  $|a_{22}| = \mu_0$ .
end

```

It is easy to verify from (44.15) that if $s = 1$, then

$$|a_{11}| \leq (1 + \alpha^{-1})\mu_0, \quad (44.16)$$

while $s = 2$ implies

$$|\tilde{a}_{11}| \leq \frac{3 - \alpha}{1 - \alpha}\mu_0. \quad (44.17)$$

By equating $(1 + \alpha^{-1})^2$, the growth factor that is associated with two $s = 1$ steps and $(3 - \alpha)/(1 - \alpha)$, the corresponding $s = 2$ factor, Bunch and Parlett conclude that $\alpha = (1 + \sqrt{2})/8$ is optimum from the standpoint of minimizing the bound on element growth.

The reductions outlined above can be repeated on the order $(n-s)$ symmetric matrix A . A simple induction argument establishes that the factorization (44.2) exists and that $n^3/3$ flops are required if the work associated with pivot determination is ignored.

JOA Stability and Efficiency

Diagonal pivoting with the above strategy is shown by Bunch (1971) to be a stable a Gaussian elimination with complete pivoting. Unfortunately, the overall process requires between $n^2/12$ and $n^2/6$ comparisons, since μ_0 involves a two-dimensional search at each stage of the reduction. The actual number of comparisons depends on the total number of 2by-2 pivots but in general the Bunch-Pardell method for computing (442) is considerably slower than the technique of Amdahl. See Barwell and George (1970).

This is not the case with the diagonal pivoting method of Bunch and Kaufman (1971). In their scheme, it is only necessary to scan two columns at each stage of the reduction. The strategy is fully illustrated by considering the very first step in the reduction:

$$a = (1_{+})/8$$

$$A = \text{larg} = \max\{\lvert a_{11} \rvert, \dots, \lvert a_{nn} \rvert\}$$

if $a > 0$

if $|a_{11}| = a$

Set $s = 1$ and $P_1 = I$.

else

$a = |a_{11}| = \max\{|a_{11}|, \dots, |a_{rr}|, |a_{r+1,r+1}|, \dots, |a_{nn}|\}$

if $|a_{11}| \neq a$

Set $s = 1$ and $P_1 = I$

elseif $|a_{11}| = a$

Set $s = 1$ and choose P_1 so $(P_1 A P_1 h)_1 = a_{11}$

else

Set $s = 2$ and choose P_1 so $(P_1 A P_1 h)_1 = a_{11}$

end

end

end

Overall, the Bunch-Kaufman algorithm requires $n^3/3$ flops, $O(n^2)$ comparisons, and like all the methods of this section, $n^2/2$ storage.

slshb A Note on Equilibrium Systems

A very important class of symmetric indefinite matrices have the form

$$A = \begin{bmatrix} C & B \\ B^T & O \end{bmatrix} \begin{bmatrix} n \\ p \end{bmatrix} \quad (4418)$$

where C is symmetric positive definite and B has full column rank. These conditions ensure that A is nonsingular.

Of course, the methods of this section apply to A . However, they do not exploit its structure because the pivot strategies "wipe out" the zero (2,2) block. On the other hand, here is a tempting approach that does exploit A 's block structure:

Step 1 Compute the Cholesky factorization $C = GGT$.

Step 2 Solve $GK = B f \text{ for } K$

Step 3 Compute the Cholesky factorization $HHT = KTK = BTc^{-1}B$.

From this it follows that

$$A = \begin{bmatrix} G & 0 \\ K^T & H \end{bmatrix} \begin{bmatrix} G^T & K \\ 0 & -H^T \end{bmatrix}.$$

In principle, this triangular factorization can be used to solve the equilibrium system

$$\begin{bmatrix} C & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}. \quad (44.19)$$

However, it is clear by considering steps (b) and (c) above that the accuracy of the computed solution depends upon $K(C)$ and this quantity may be much greater than $K(A)$. The situation has been carefully analyzed and various structure-exploiting algorithms have been proposed. A brief review of the literature is given at the end of the section.

It is interesting to consider a special case of (44.19) that clarifies what it means for an algorithm to be stable and illustrates how perturbation analysis can structure the search for better methods. In several important applications, $g = Q_C$ is diagonal, and the solution subvector y is of primary importance. A manipulation shows that this vector is specified by

$$y = (B^T C^{-1} B)^{-1} B^T C^{-1} f. \quad (44.20)$$

Looking at this we are again led to believe that $K(C)$ should have a bearing on the accuracy of the computed y . However, it can be shown that

$$\| (B^T C^{-1} B)^{-1} B^T C^{-1} \| \leq \psi_B \quad (44.21)$$

where the upper bound ψ is independent of C , a result that (correctly) suggests that y is not sensitive to perturbations in C . A stable method for computing this vector should respect this, meaning that the accuracy of the computed y should be independent of C . Vava is (1994) has developed a method with this property. It involves the careful assembly of a matrix V in $\mathbb{R}^{n \times n}$ whose columns are a basis for the nullspace of BTc^{-1} . The n -by- n linear system

$$[B \setminus V] \begin{bmatrix} \end{bmatrix} = f$$

is then solved implying $f = By + Vq$. Thus, $BTc^{-1}f = BTc^{-1}By$ and (44.20) holds.

Problems

- P4.4.1 .316 13.1.168 ...1 00 ..0 .2...1.c.8. 1≥ .. .181... ..1.b
~~A.EA=Q.E zkyA A Q\AESF~~
- P4.4.2 .316 16.1.1 0111.c.8.. 138P2..31or.2≥... ...1.c13. .≥A Q.S=Tk2Mn)TkA+

P4.4.3 $1 \dots 0$ (4.4.14) $\text{Ag}_2\text{C}_2\text{gns.2.a}$ $8\text{gap}2\text{pr}$ Cn.i Ag_2SgnA $\text{QFAEA} = \text{Sk/k}$
 $\text{p}6)$ $\text{PnE}(\text{Ex}(\text{GA(j,j)})$ $\text{Ej} = 1:n, \{ \text{j} \}$ S2AEFQkA(j+1-j) $\text{vpj} = 1:n - 1$ Cn.L(i,j) gap8pr 2aA
 A(i,j-1) $\text{vpj} = 2n - 1$ $\text{Cn.i} = j + 1n.$

$$P4.4.4 .2.1.8 AER^{n \times n} = R^{n \times n} [R^{-1}]^T = R^n [I_{\frac{n}{4}}]^{x_4} [R^{n \times n}]^{x_1} = R^n R^{n \times n} I_n R^n$$

8apan (o(f0= x o ((v(G)JO((yC{ nCE.o (= Pf (= 0C30e=((yo < (= 0

P4.4.5 1 ro...81... ..1.cbA Q_{quasidef nite r} Sf2 2,avpu

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & -A_{22} \end{bmatrix}_{\substack{n \\ p}}$$

LQsAu nP A22 S=HQPA)Qk=F))JP μQ =C O +OEA())OPExO(xPJ (=O
(EP)(CEI(JO(

OE G @M“ApploD P D P L'P1 B'€p QPNEP xpyÀ I2 bBÖp
P Ñ L •P @Ñ xOE •D

$$w = (I_n - B(B^T B)^{-1}B^T)e_k.$$

$$\frac{1}{\|} \quad \frac{1}{\|w\|}$$

Ñ § 728 1CTss. 3,O „ë 78OC ,lf,;H62lh41...Ü,Ü „.3. „.)..1 „,Ü BOÄÄner.
Math. 27, trOM lt.

Dk59, nL, Cn,s MC,nuCñMIt6 chgua htCBat,gA vp)C8L,8CtMniaptMChg8Mni
huuatpMMnaCpAthuA3Math Comput. 31, MNOH

Pscs CgrAA (CtpMzAld. c9,nL,rM(BuCrClgp2 CAM16,huuhtpMSn tn2T a
9Cnj. PCtp23AM J. Matrix Anal. Applic. 14, r r IEQt.

9aLC,AeB2,paTg,unAu2S toa ALCnnMt a AM.T ApgLaSM2A e8C7d AgAAMigad
gotCMGinCf Ar pWICignCM.gMg8gt „ts t,a gt,ap,Cn.34lan,Auat,g MACT „p6ML,
r i6f A.A9,gLeapAMgnAg,gB,ApAgAMohsp2g,AapvpuCnMAAGapAMAL,AAa
Mnf

Is 9Cp8a8n. D'sesia 7Mtd c4)guACpMAP48 gpMt,uAp hg.2nihuuhtpMSn.at nMta
hAtauAgBMnaCp,CtMgnAM T ns. Math. 58f w.

M C h A tMML, Mt „,r c M un SuM
p

EP I Uedu (1990). c1n9gnmAI hC8afpgJa.2SgIAAaomgSpAaTin. Alg Applic. 132, 115-117.
 rz1llogmr1990. c4.Cn2T8ig8S3CIr gM2CIuCI3Sh2A Sgr lSGG(pgiCuuSni 48gpS23TOper. Res. 38, 1006-1018.

4n a6S8S0lSAuA2a6CAAar.16.8BCsaddle point system. haas 11.5.10.

4.5 Block Tridiagonal Systems

Block tridiagonal linear systems of the form

$$\begin{bmatrix} D_1 & F_1 & & \cdots & 0 \\ E_1 & D_2 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & & F_{N-1} \\ 0 & \cdots & E_{N-1} & F_N & D_N \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}. \quad (45)$$

frequently arise in practice. We assume for clarity that all blocks are q by q. In this section we discuss both a block LU approach to this problem as well as a pair of divide-and-conquer schemes.

4.5.1 Block Tridiagonal LU Factorization

If

$$A = \begin{bmatrix} D_1 & F_1 & & \cdots & 0 \\ E_1 & D_2 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & & F_{N-1} \\ 0 & \cdots & E_{N-1} & F_N & D_N \end{bmatrix}$$

then by comparing blocks in

$$A = \begin{bmatrix} I & & \cdots & 0 \\ L_1 & I & & \vdots \\ \ddots & \ddots & & \vdots \\ 0 & \cdots & L_{N-1} & I \end{bmatrix} \begin{bmatrix} U_1 & F_1 & & \cdots & 0 \\ 0 & U_2 & \ddots & & \vdots \\ \ddots & \ddots & \ddots & & \vdots \\ 0 & \cdots & 0 & U_N & F_N \end{bmatrix}$$

we finally obtain the following algorithm for computing the L and U:

$$U_1 = D_1$$

for np X end Y

The procedure is defined along as the U_i are nonsingular.

Having computed the factorization (453), the vector x in (451) can be obtained via block forward elimination and block back substitution

$$\begin{aligned}
 & Y_i = b_i \\
 & f \ r \ i = 2N \\
 & \quad Y_i = b_i - L_{i-1} Y_{i-1} \\
 & \text{end} \\
 & \text{Solve } U_i X_i = Y_i \text{ for } X_i \\
 & f \ r \ i = N-1 : 1 \\
 & \quad \text{Solve } U_i X_i = Y_i - F_i X_{i+1} \text{ for } X_i \\
 & \text{end}
 \end{aligned} \tag{455}$$

To carry out both (454) and (455), each U_i must be factored since linear systems involving these submatrices are solved. This could be done using Gaussian elimination with pivoting. However, this does not guarantee the stability of the overall process.

4.5.2 Block Diagonal Dominance

In order to obtain satisfactory bounds on the L_i and U_i , it is necessary to make additional assumptions about the underlying block matrix. For example, if we have

$$\|D_i^{-1}\|_1 (\|F_{i-1}\|_1 + \|E_i\|_1) < 1, \quad E_N \equiv F_0 \equiv 0, \tag{456}$$

$f \ r \ i = 1:N$, then the factorization (453) exists and it is possible to show that the L_i and U_i satisfy the inequalities

$$\|L_i\|_1 \leq 1, \tag{457}$$

$$\|U_i\|_1 \leq \|A_n\|_1. \tag{458}$$

The conditions (456) define a type of block diagonal dominance.

4.5.3 Block-Cyclic Reduction

We next describe the method of block-cyclic reduction that can be used to solve some important special instances of the block tridiagonal system (451). For simplicity, we assume that A has the form

$$A = \begin{bmatrix} D & F & & \cdots & 0 \\ F & D & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ & \vdots & \ddots & \ddots & F \\ 0 & \cdots & & F & D \end{bmatrix} \quad E \in \mathbb{C}^{N \times N} \tag{459}$$

where F and D are q -by- q matrices that satisfy $DF = FD$. We also assume that $N = 2k-1$. These conditions hold in certain important applications such as the discretization of Poisson's equation on a rectangle. (See §4.8.4)

The basic idea behind cyclic reduction is to halve repeatedly the dimension of the problem on hand repeatedly until we are left with a single q by q system for the unknown subvector \mathbf{x}_q . This system is then solved by standard means. The previously eliminated \mathbf{x} are found by a back-substitution process.

The general procedure is adequately illustrated by considering the case $N = 7$.

$$\begin{aligned} b_1 &= D_{11}x_1 + F_{12}x_2 \\ b_2 &= F_{21}x_1 + D_{22}x_2 + F_{23}x_3 \\ b_3 &= F_{32}x_2 + D_{33}x_3 + F_{34}x_4 \\ b_4 &= F_{43}x_3 + D_{44}x_4 + F_{45}x_5 \\ b_5 &= F_{54}x_4 + D_{55}x_5 + F_{56}x_6 \\ b_6 &= F_{65}x_5 + D_{66}x_6 + F_{67}x_7 \\ b_7 &= F_{76}x_6 + D_{77}x_7 \end{aligned}$$

For $i = 2, 4$, and 6 we multiply equations $i - 1$, i , and $i + 1$ by F , $-D$, and F , respectively, and add the resulting equations to obtain

$$\begin{aligned} (F^2 D^2) x_2 + F^2 x_4 &= F(b_1 + b_3 - b_2) \\ F^2 x_2 + (F^2 D^2) x_4 + F^2 x_6 &= F(b_3 + b_5 - b_4) \\ F^2 x_4 + (F^2 D^2) x_6 &= F(b_5 + b_7 - b_6) \end{aligned}$$

Thus, with this tactic we have removed the odd-indexed \mathbf{x} and are left with a reduced block tridiagonal system of the form

$$\begin{aligned} D^{(1)} x_2 + p^{(1)} x_4 &= b_2^{(1)}, \\ p^{(1)} x_2 + D^{(1)} x_4 + p^{(1)} x_6 &= b_4^{(1)}, \\ p^{(1)} x_4 + D^{(1)} x_6 &= b_6^{(1)}, \end{aligned}$$

where $D^{(1)} = F^2 D^2$ and $p^{(1)} = F^2$ commute. Applying the same elimination strategy as above, we multiply these three equations respectively by $p^{(1)}$, $-D^{(1)}$, and $p^{(1)}$. When these transformed equations are added together, we obtain the single equation

$$(2F^{(1)} D^{(1)}) x_4 - p^{(1)} (b_2^{(1)} + b_6^{(1)}) - D^{(1)} b_4^{(1)}$$

which we write as

$$D^{(2)} x_4 = b_4^{(2)}$$

This completes the cyclic reduction. We now solve this (small) q by q system for x_4 . The vectors \mathbf{x}_2 and \mathbf{x}_6 are then found by solving the systems

$$\begin{aligned} D^{(1)} x_2 &= b_2^{(1)} - p^{(1)} x_4 \\ D^{(1)} x_6 &= b_6^{(1)} - p^{(1)} x_4 \end{aligned}$$

Finally, we use the first, third, fifth, and seventh equations in the original system to compute $\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_5$, and \mathbf{x}_7 , respectively.

The amount of work required to perform these recursions for general N depends greatly upon the sparsity of the $F(\theta)$ and $p(\theta)$. In the worst case when these matrices are full, the overall flop count has order $\log(N)q^2$. Care must be exercised in order to ensure stability during the reduction. For further details, see Buneman (1969).

) c4c)A The SPIKE Framework

A bandwidth p matrix $A \in \mathbb{R}^{Nq \times Nq}$ can also be regarded as a block tridiagonal matrix with banded diagonal blocks and lowrank off-diagonal blocks. Here is an example where $N = 4$, $q = 7$, and $p = 2$

$$A = \begin{bmatrix} \begin{array}{ccccccccc} x & x & x & & & & & & \\ x & x & x & x & & & & & \\ x & x & x & x & x & & & & \\ x & x & x & x & x & x & & & \\ x & x & x & x & x & x & x & & \\ x & x & x & x & x & x & x & x & \\ x & x & x & x & x & x & x & x & x \\ x & x & x & x & x & x & x & x & x \end{array} & | & \begin{array}{cccccc} & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{array} & | & \begin{array}{cccccc} & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{array} & | & \begin{array}{cccccc} & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{array} \end{bmatrix} . \quad (45.11)$$

Note that the diagonal blocks have bandwidth p and the blocks along the subdiagonal and superdiagonal have rank p . The low rank of the off-diagonal blocks makes it possible to simulate a divide-and-conquer procedure known as the "SPIKE" algorithm. The method is of interest because it parallelizes nicely. Our brief discussion is based on Pdizzi and Sameh (2007).

Assume for clarity that the diagonal blocks D_1, \dots, D_4 are sufficiently well conditioned. If we premultiply the above matrix by the inverse of $\text{diag}(D_1, D_2, D_3, D_4)$, then we obtain

$$\tilde{A} = \begin{bmatrix} \begin{array}{ccccccccc} 1 & & & & & & & & \\ 1 & & & & & & & & \\ & 1 & & & & & & & \\ & & 1 & & & & & & \\ & & & 1 & & & & & \\ & & & & 1 & & & & \\ & & & & & 1 & & & \\ & & & & & & 1 & & \\ & & & & & & & 1 & \\ & & & & & & & & 1 \end{array} & | & \begin{array}{cccccc} + & + & & & & \\ + & + & & & & \\ + & + & & & & \\ + & + & & & & \\ + & + & & & & \\ + & + & & & & \\ + & + & & & & \\ + & + & & & & \end{array} & | & \begin{array}{cccccc} & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{array} & | & \begin{array}{cccccc} & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{array} & | & \begin{array}{cccccc} & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{array} \end{bmatrix} . \quad (45.12)$$

With this maneuver, the original linear system

$$\begin{bmatrix} D_1 & F_1 & 0 & 0 \\ E_1 & D_2 & F_2 & 0 \\ 0 & E_2 & D_3 & F_3 \\ 0 & 0 & E_3 & D_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}, \quad (45.13)$$

which corresponds to (45.11), transforms to

$$\begin{bmatrix} I_7 & \tilde{F}_1 & 0 & 0 \\ \tilde{E}_1 & I_7 & \tilde{F}_2 & 0 \\ 0 & \tilde{E}_2 & I_7 & \tilde{F}_3 \\ 0 & 0 & \tilde{E}_3 & I_7 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \tilde{b}_1 \\ \tilde{b}_2 \\ \tilde{b}_3 \\ \tilde{b}_4 \end{bmatrix}, \quad (45.14)$$

where $D_i \tilde{b}_i = b_i$, $D_i \tilde{F}_i = F_i$, and $D_{i+1} \tilde{E}_i = E_i$. Next, we refine the blocking (45.14) by turning each submatrix into a 3 by 3 block matrix and each subvector into a 3 by 1 block vector as follows

$$\left[\begin{array}{c|cc|ccc|ccc} l_2 0 & 0 & K_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a_a & H_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & I_2 & G_1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & R_1 & l_2 & 0 & 0 & K_2 & 0 & 0 \\ 0 & 0 & S_1 & 0 & a_a & 0 & H_2 & 0 & 0 \\ 0 & 0 & T_1 & 0 & 0 & l_2 & G_2 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & R_2 & l_2 & 0 & 0 & K_3 & 0 \\ 0 & 0 & 0 & 0 & S_2 & 0 & [30 & 0 & H_3 & 0 \\ 0 & 0 & 0 & 0 & T_2 & 0 & 0 & l_2 & G_3 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & R_3 & l_2 & I_q & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & S_3 & 0 & I_m & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & T_3 & 0 & l_2 & q \\ \hline \end{array} \right] \left[\begin{array}{c} W \\ Y \\ Z \\ \hline \end{array} \right] = \left[\begin{array}{c} G \\ d \\ J \\ \hline Q \\ h \\ B \\ \hline f_3 \\ Q \\ d_4 \\ f_4 \\ \hline \end{array} \right]. \quad (45.15)$$

The block rows and columns in this equation can be reordered to produce the following equivalent system

$$\left[\begin{array}{c|cc|ccc|ccc} l_2 0 & K_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & l_2 & G_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & R_1 & l_2 & 0 & K_2 & 0 & 0 & 0 & 0 \\ 0 & T_1 & 0 & l_2 & G_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & R_2 & l_2 & 0 & K_3 & 0 & 0 \\ 0 & 0 & 0 & T_2 & 0 & l_2 & G_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & R_3 & l_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & T_3 & 0 & l_2 & 0 & 0 \\ \hline 0 & 0 & H & 0 & 0 & 0 & 0 & 0 & [30 \\ 0 & 0 & S & 0 & 0 & H_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & S_2 & 0 & 0 & H_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & S_3 & 0 & 0 & 0 & [3 \\ \hline \end{array} \right] \left[\begin{array}{c} W \\ Z \\ \hline W \\ Y \\ \hline Z \\ \hline T \\ \hline \end{array} \right] = \left[\begin{array}{c} G \\ J \\ Q \\ h \\ B \\ \hline f_3 \\ Q \\ d_4 \\ f_4 \\ \hline T \\ \hline \end{array} \right]. \quad (45.16)$$

If we assume that $N \gg 1$, then the $(1,1)$ block is a relatively small banded matrix that defines the z_i , and \mathbf{W} . Once these quantities are computed, then the remaining unknowns follow from a decoupled set of large matrix-vector multiplications, e.g., $y_1 = d_1 - H_1 w_2$, $y_2 = d_2 - S_1 z_1 - H_2 w_3$, $y_3 = d_3 - S_2 z_2 - H_3 w_4$, and $y_4 = d_4 - S_3 z_3$. Thus, in a full processor execution of this method, there are (short) communications that involves the w_i and z_i , and a lot of large, local copy computations.

Problems

P4.5.1 . . 616 13.1. .01.d ...1..o . 1.c..1 ..1..b ..1...l ..§ . e8.cb.16.1Rpro ro85
c .ocRPro . roR§§ ro75§

P4.5.2 f..18 . . 8.2....8 . .1cl1. = ESD,F, B k/k EAkE=k/A=S"akS) lls= bLyAEQ Q=
!AkAkAFl 6 JyJ.K+, Aklyk = + O Q=oCx(Cw ov(yOxID,F ER xq.), bERNq

P4.5.3 P16 611 .1l .1 .8 ...18. 1b13gN..

$$\begin{bmatrix} D_1 & F_1 \\ E_1 & D_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

638.8D1.)F D2.xAF QS).")F F1.)F E1:((n,, :111): -{u)n((, 1n - I(nn11)

P4.5.4 re.168... . 8.3reorP ..861.d 16.168 ..8.8.18 .. Pro68ib.6 1..
. 1.d1.b 6.13 1.. .01.d.§re1 1.2.8 16.1 E R qNqyk ,)F Lk,k/ p Cn.2,C2p⁺ .
(C= ve(o yG OeH A(EMCE. o(= Oo vJ(ve@=OooP(CMIO OoPMiO pP
(vE(= =CQ yoo x aJ(y5Fn(= (oO= dGJ(vp(Ev(C4Cf d = ylf e.Cfo.

Notes and References for §4.5

38 16.8.. .1.. 8....31 ..11168 ..1l. l..8. 1b01.d.1..b .1..l1. 1fe1.or
§ e§6 R. 05r2fo 138 .1cl1bPo1d1 .fd.o ...18.. 1..... 1. 38.1.. c.c18
8. 8.8.8Pr2.1.1.oh Math Comput 26 y)teuyys
5 Mgpaiz ty/ slch2CSp.1C2pr CaAhA2aA=SIAM Review 26, MfMD
,D AppSCU(M)dscIn2,asC2opr tCgMge OS;r iigC,AS2,h2gpCia)gnA2pSiAM2A=T
J. Sci. Stat. Comput 6 ,yNQ,NL

,aApgAap2mge ;SCignCguSnCn.6n;S24pr g2AIA8r .C2MgnA2pC2uar n,
eL'Da Snig;Cn;5DHICpiC7,tuNc9ge .SCignC,,gur nC62pS.a6n;'anapC,StC2SgnA
2a'aIA,igpSj,r p,aagnau=flacif c J. Math 12, MN /,fIMN
5LHCpiC7,tlAsdcIn.r Cign68S nCn.4pi2uan2Ap9gn;SniggTf 10," Lin. Alg. Applic.
14 N,MN,,s

Cpua2gS2C2Sng8j2,aS;aGn.8r pa;22SgipaA.pSo&n,

5LuBgena. 7.tu)Dc4.12 .Spa.2hg22Sgn(gSAAgR62C2r gAr Mi2pr4rap,Ar "A= ACM 12, t)f MD

9L 92GammaDxDg,o= Cn.)DuDaSa,Agn7MdD cln .Spa.2,a2gg vp hg8r (iSAAg r A R6,C2r gmgAM J. Numer. Anal. 7, AN, u)us

,aC,,u28C2Sgn2,apSi,2 ,An; j 02A2a8r 2paCZpavp g2,ap88Aha 8g2,a
CgSinx t.Cnq n 8gApc4IA2Co8a g8gSn2,SA ;aAr s,

0.).= Ox(= M0 2D DyeCx f (EvC= 1x0=XG(c 00CA > 0ef1N+v 2Dv(= eE= 0> J= o(ve
x= ovOeXAOx(j OoA(= Ee(= EeyO

.veO A OvO= EC= EO3(0JEeEoE0.o Ev(CE.OoT

F.W. ,C= M0,2.DpJQ=OE0Cn(G=CleveQoO E= OvO XG(v6CC=(-
Review 12, N/y GnuE

SIAEvv(= y

J. Numer. Anal. 8 INNF IEuA
 $\|u\|_{L^2(\Omega)} \leq C \left(\|f\|_{L^2(\Omega)} + \left\| \frac{\partial f}{\partial x} \right\|_{L^2(\Omega)} \right)$

(.) In,A CnA s. Als,o 7ME dAAa II B .MpaPat,lmAM t,a R .MantlapS.Cls,tMD
 lBalmAaAcPcRdMAE6GSI nASTAM J. Numer. Anal. 10 M1E f iM1A
 9kl,9,TcaCnmu. III 7M/ dAcJa.Slat hls,tMlBt,a 9MCp1lnSR ,CtMlBfa.tCni..CI
 5aiMlnA tGn(MAAR6CtSlm UpIai,sGpiMln8TAM J. Numer. Anal. 11, 753-763.
 j sassap'MtA ch11AatABtJa)...S. 5am,tSln4AilpMtJ 1p 9Al.e - SmMCiltsnAI
 hAtaIA STAM J. Numer. Anal. 13, 6y6 6tu.

ICpSl,AanapC.MT C MfaAanAMlnAsM.pa.,.Mln,C.a oaanAplAlAa.f

a. hpCITtIC, Caprik4 $\text{hpaa7MIE dckJa.MI ahlst, tMIDB,a .SAIata1SAAIR68C.MD}$
 $\text{C.MAeSTAM J. Numer. Anal. 10, t11f t1A}$
 5.4 $\text{hpaat7MtAc4 AanapC.SIAM.5a.,tSln 4ilpM13T SIAM J. Num Anal. 11, r 1uf.N1}$
 P.4k.SQlnm Cn. kl.I. jppaMp7MudAcln CsM. 5a.,tSln PatJlmp t,a hl.,tSln lB
 $(\text{IMAAIR66Sln, SIAM J. Numer. Anal. 13, r 6, t1A})$
 5.4 $\text{hpaat7MId c4)..sM.5a.,Mln 4ilpStJ lphla.Snole - MmMCiInCs1AlB4poMtpCI.}$
 $.S1anAMTSIAM J. Numer. Anal. 14, I lu IN1$
 (a. $\text{hpCpTtpC, Gal. 5, hpaat7MtdA clP.tlpCnrrhpC.sAPat,lmAvp t,a .Spa.thls,tSlnlB}$
 $(\text{ISAAIR66CMln3TComput. Appl. Math. 27, N8NuE.})$
 h. $\text{9ln.asmCnm. ACn.ap7Mtd. c)..sM.5am,tSlnvp hAaMG.MmSCiInCs1Alb3SIAM J.}$
 $\text{Matr. Anal. Applic. 15 ENNEE 1A}$

4 N' osl.NAAtalpStJap. tJSr~~7~~M¹n. 7N3~~1~~¹dASpar ppd.1 Gor er d linear system.
hAaS~~1~~¹aJnM⁶AA Aploa1ApMtJ AISATpaCpanSA,AA¹Mn,

u. Al. aptAn. 1.(I..a 7M1dAc9A.eR.S S_nM pSt, lna Utj ptMa 5 ,nAlnsA9II,
mal alSnaChAta1A4..pCta.3TBIT 30 6t1 r II A
u. Al.Capt7M1MdA1Cf HaapACnrae RsS 18 CtShp 9lpmmapam alh_nSIAM J. Matr
Anal. Applic. 12 6ut 6yE.
uk Al.TaptACn. lk. (p.a 7M1d cPSfaS1.e RsM1StMlp lSnaCp hAta1M1MmAp,
.apASTMA J. Numer. Anal. 13 Mf1MMy1A

hAta1A,Ct Cl aslef S.MCilmCs3saAAanoapfsl etIMCni,AGAll,p. Aaa,

A. CSppaCt, \exists nrh As pass \forall 16dAc4i lpStJ 1Ap 4A1lAt9A.l.e. MCilnDSh3ChA.a1A3T
SIAM Review 46 6t yn

.. In P CttCnrA. u. htapCp7MtdAc5l,nmSnRpplpArhl.Sni 9.l.e saAAanoapiAta1A3T
Math. Comput. 65, MMMEr A

I. Aa1MinC \exists SIAM. lltM7N EId.cR.M3nCnrhtCoAals,tSlnBP' PXMnacA.a1A1B
vole saAAanoapi 13TSIAM J. Matri. Anal. Applic. 24, yr Nf ylu

P. saisC \exists m. P.5. 1Aolp \exists Myd. cupCA4pl,n. (CpStSlnSvp 9.l.e 9S.M16,s lSnaChA
ta \exists IMA J. Numer. Anal. 18 EIEfy.E.

c. 5IAAWhnh dS.Cnr7vitt dAc4 (CpCAs \exists). Mpa th.ap vp 9A.l.e- S.MCilnGAta1ApMo
haACpCBat1MBMpoStpCManAS \exists SIAM J. Sci. Comput. 20, 1778-1793
.PhAsIA .Cnn 3ini 7MldAc4slAt (apSlmM.tlpSTCtNB)apt,Sn9d.e - MCni,SC
PCtpMfn.tSln \exists Math. Comput. 69, M1 EM II 1A

KJch(UMIPClaplpeA,AAIptACn.mMTaptA_nl_nC..lpSni tl pJat,aptJ3o,n. SAA,pA_n
.anAa4Al_nStaAa,a tl catCearMBa,MCiInC. sl.e/CTa_nQ,lnSia,3Aaf

Rkl.MTCS14 hC1a, 7MII Ach(UMR14,(CpC.sRn.Spln1 vphIA.SnQ.n.amlSnaChAta1gT
 Comput. Fluids 36 MEM MN1
 .)kM.rSeeAa6i nRkrCni,lis, 7Myd.c4nC..AMBAJaO8n.Cta. h(UMR14ilpMT13TSIAM
 J. Matr. Anal. Applic. 30 M11f Mr it.

4.6 Vandermonde Systems

Suppose $\mathbf{x} \in \mathbb{C}^n$, $\mathbf{f} \in \mathbb{C}^n$. A matrix $V \in \mathbb{C}^{n \times n}$ of the form

$$\mathbf{V} = V(\mathbf{x}_0, \dots, \mathbf{x}_n) = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_n \\ \vdots & \vdots & & \vdots \\ x_0^n & x_1^n & \cdots & x_n^n \end{bmatrix}$$

is said to be a **Vandermonde matrix**. Note that the discrete Fourier transform matrix (§14.1) is a very special complex Vandermonde matrix.

In this section, we show how the systems $VA = f = f(\mathbf{Q}_n)$ and $Vs = b = b(\mathbf{Q}_n)$ can be solved in $O(n^2)$ flops. For convenience, vectors and matrices are subscripted from 0 in this section.

4.6.1 Polynomial Interpolation: $V^T a = f$

Vandermonde systems arise in many approximation and interpolation problems. Indeed, the key to obtaining a fast Vandermonde solver is to recognize that solving $V^T a = f$ is equivalent to polynomial interpolation. This follows because if $VA = f$ and

$$p(x) = \sum_{j=0}^n a_j x^j, \quad (461)$$

then $p(x_i) = f_i$ if $i = 0, \dots, n$.

Recall that if the x_i are distinct then there is a unique polynomial of degree n that interpolates $(x_0, f_0), \dots, (x_n, f_n)$. Consequently, V is nonsingular as long as the x_i are distinct. We assume this throughout the section.

The first step in computing the a_j of (461) is to calculate the Newton representation of the interpolating polynomial p .

$$p(x) = \prod_{k=0}^n c_k \frac{(x - x_k)}{(x - x_{k+1})}. \quad (462)$$

The constants c_k are divided differences and may be determined as follows:

```

 $c(Q_n) = f(Q_n)$ 
 $f \text{ for } k = Q_n - 1$ 
 $\text{for } i = n - 1 : k + 1$ 
 $\quad G = (c_i - G_{i-1}) / (x_i - x_{i-1})$ 
 $\quad \text{end}$ 
 $\text{end}$ 

```

See Conte and deBoor (1980).

The next task is to generate the coefficients a_0, \dots, a_n in (461) from the Newton representation coefficients c_0, \dots, c_n . Define the polynomials $P_0(x), \dots, P_n(x)$ by the iteration

$R_k(x) = G_k$
 for $k = n - 1 : - 1 : 0$
 $P_k(x) = G_k + (x - X_k)P_{k+1}(x)$
 end

and observe that $P_0(x) = p(x)$. Writing

$$P_k(x) = a_k^{(k)} + a_{k-1}^{(k)}x + \dots + a_0^{(k)}x^k$$

and equating like powers of x in the equation $P_k = G_k + (x - X_k)P_{k+1}$ gives the following recursion for the coefficients $a_i^{(k)}$:

$a_n^{(n)} = c$
 for $k = n - 1 : - 1 : 0$
 $a_k^{(k)} = G_k - X_k a_{k+1}^{(k+1)}$
 for $i = k+1 : n-1$
 $a_i^{(k)} = a_i^{(k+1)} - X_{i+1} a_{i+1}^{(k+1)}$
 end
 $a_0^{(k)} = a_0^{(k+1)}$
 end

Consequently, the coefficients $a_i = a_i^{(0)}$ can be calculated as follows

$a(Q) = c(Q)$
 for $k = n - 1 : - 1 : 0$
 for $i = kn - 1$
 $a_i = a_i \cdot X_{i+1}$ {464}
 end
 end

Combining this iteration with (463) gives the following algorithm

To (Tq., hn=0) Given $x(O:n)$ ER^{nH} with distinct entries and $f = f(O:n)$ ER^{n+1} , the following algorithm overwrites f with the solution $a = a(O:n)$ to the Vandermonde system $V(x_0, \dots, x_n)Ta = f$.

for $k = O:n - 1$
 for $i = n - 1:k+1$
 $f(i) = (! (i) - f(i-1)) / (x(i) - x(i-k-1))$
 end
 end
 for $k = n - 1 : - 1 : 0$
 for $i = k:n - 1$
 $f(i) = f(i) - f(i+1) \cdot x(k)$
 end
 end

This algorithm requires $5n^2/2f$ ops

4.6.2 The System $z = b$

Now consider the system $Vz = b$. To derive an efficient algorithm for this problem we describe what Algorithm 46.1 does in matrix-vector language. Define the lower bidiagonal matrix $L_k(a)$ $\in \mathbb{R}^{(n^k) \times (n^k)}$ by

$$L_k(\alpha) = \begin{bmatrix} I_k & & & & & & & \\ & & & & & & & 0 \\ & 1 & 0 & \cdots & & & & 0 \\ -\alpha & & 1 & & & & & \\ & 0 & \ddots & \ddots & & & & \\ 0 & & \vdots & & \ddots & & & \vdots \\ & 0 & & \cdots & & & & 1 \\ & & & & & & -\alpha & 1 \end{bmatrix}$$

and the diagonal matrix D_k by

$$D_k = \text{diag}(1, \dots, 1, X_{k+1} - X_0, \dots, X_n - X_0, k).$$

With these definitions it is easy to verify from (463) that, if $f = f(Q_n)$ and $c = Q(Q_n)$ is the vector of divided differences, then

$$c = U^T f$$

where U is the upper triangular matrix defined by

$$UT = D_k L_n(1) \dots D_0 L_0(1).$$

Similarly, from (464) we have

$$a = LTc,$$

where L is the unit lower triangular matrix defined by

$$LT = L_0 X_0 \dots L_n(1) X_n(1) T.$$

It follows that $a = V^{-T}f$ is given by

$$a = LTUTf.$$

Thus

$$a = LTUT$$

which shows that Algorithm 46.1 solves $Vta = f$ by tacitly computing the "UL factorization" of V^{-T} . Consequently, the solution to the system $Vz = b$ is given by

$$\begin{aligned} z = V^{-T}b &= U(Lb) \\ &= (L_0(1)TD_0(1)\dots L_n(1)TD_n(1)(L_n(1)X_n(1)\dots L_0(1)X_0(1))b). \end{aligned}$$

This observation gives rise to the following algorithm.

Given $x(Qn) \in R^{n+1}$ with distinct entries and $b = b(Qn) \in R^{n+1}$, the following algorithm overwrites b with the solution $z = z(Qn)$ to the Vandermonde system $V(x_0, \dots, x_n)z = b$

```

f r k=Qn- 1
  f r i = n - 1:k+ 1
    b(i) = b(i) - x(k)b(i - 1)
  end
end
for k= n- 1: - 1:0
  for i = k+ 1:n
    b(i) = b(i)/(x(i) - x(i - k - 1))
  end
  f r i = k:n- 1
    b(i) = b(i) - b(i + 1)
  end
end

```

This algorithm requires $5n^2/2$ fops.

Algorithms 461 and 462 are discussed and analyzed by Bjork and Pereyra (1970). Their experience is that these algorithms frequently produce surprisingly accurate solutions, even if V is ill-conditioned.

We mention that related techniques have been developed and analyzed for current VLSI design systems, e.g., systems of the form

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ x_0 & x_1 & 1 & x_3 \\ x_0^2 & x_1^2 & 2x_1 & x_3^2 \\ x_0^3 & x_1^3 & 3x_1^2 & x_3^3 \end{bmatrix}^T \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{bmatrix}.$$

See Higham (1990).

Problems

P4.6.1 .38fo.3.. > $\mathbf{R}, \dots, x_n),)|^n($

$$,^n)\mathbf{R}^n = \in \bullet - \omega_i - \omega_j).$$

P4.6.2 R 2..3. .. PP58.. .38No8gfo..8r2.... N. 38 . .18 ..8.8or

$$\|\mathbf{V}^{-1}\|_\infty \leq \sum_{\substack{0 \leq j \leq n \\ i=0 \\ i \neq j}} \ell \frac{|x_i|}{|x_k - x_i|}.$$

$$| >_0 \dots)_{\hat{B}_*} >_0 \dots)_{\hat{B}_*})^n | 0^n \rangle | 0 \rangle | 0 \rangle | n \rangle | 0 \hat{B}^n | 0_2 \dots ()_{\infty} | n \leq \hat{B}_* | n_{\hat{B}_*} | 0 \rangle | 0 \rangle$$

Notes and References for

fo. 8s 81e.. 8.28s.8 ... 8.. ... 82. ... 6s 1.82. 38 No.86. 8..

11P .sd e 38.8... R.PG 12.8o2.881e.s.8.28sf8 ... 8.. 81Pr...8s. Math Comput. 24, yEe1E.

4. 9J5p. CAkARsr.MDifTEdc4AilpMtJVA)ID ,aDlCDmap1lDhAt 1A,Numer. Math 21, MEEI.

kJa mM.MmMfTap81AtCtMD2a.MA.,AA6patCMA Mnf

h...)lta CD). m9lp 7Mtd Elementary Numerical Analysis: An Algorithmic Approach, kMp. RmMtMIApC8sMs8 1 pe,JCA tapN.

Rpp1fDCsA8ACDmap1lnAjaAta1 AlA.apA MD,s,ma,

a. 1 sMiJC 7ityId cRpp1fDCsAAlAa9J5per (apA,1AilpMt,up hls.MnICDmap1lDma hAta1A,Tnumer. Math 50, uMEuEN

a. 1 sMi,C 7ityd c. lt hlsotMIDI CD.apulnma' hAM1AsnJA.Mpt,lilnCA(ls.Dl1MCAA,, IMA J. Numer. Anal. 8 /IEf /yu.

a. 1AsMiJC 7M1d chtCoMA1DcA.AMM1lMt,Ap hIA.MDID caDlCDmap1lDmahaMMea ta1A,TSIM J. Matrix Anal. Applic. 11, NE i.

h.i. 9CptaAAm 1A sMi,C 17Mtd. cc,a htpt,parhaDAMtM.MChmap1ln.a' IMea1A,T Numer. Math 62, MEE/.

1P. ICpCJitEdcRpp1fD(rapt,poCtMIMHC.map1lDnAtauA,IMA J. Numer. Anal. 13 Mf MN.

S.tapaAtMjDipatM.GAA,st1D.apDM1ia1D.MtMfICDmapulDhAtauAC. f avon. Mnf

u. ACotA,MtIr dcalp1 RAtM1CtApAUD.apAaACDmap1lDhAtpM,TAumer. Math 23, E.I E/I.

u. AC,tA.JMfr dc1AtM1esADmMtM16Dmap1lnP6tpM.aAumer. Math 24, if iN.

1'. AD 7Mtd. cr l,DmAp t j htpt,t,pan9C. 8CpplpAICnm pl1rAta1A,,SIAM J. Matrix Anal. Applic. 20, r er t

9.M4AAap 7Mtd c)IDmMtMfhoap GICnmap1lDhAtpMfsSIAM Review 38, EM/16E

9.a.eap1C 7M1d. cka 1DmMtMf lr paCICDmapulDmfp,sl. CD1AMtM1a,DMta sc.eas1CtpM.aAtumer. Math 85, r r iEI.

ca o1MCailpMt,ApabaDtam oa af taDmthml.ap .ID caDlCDmap1ln.hAta1A,oAl.e ICnmap1ln.mA.Ata1A,CDICDmap1lDhAta1A8MtlJapAls.nl IMCs

i. ACsM1co6tMIA(apapC7M1d. caolapM.CMTapadtMC1hls,tMhP, AtMa1AM1DCA ICnmap1ln.hAta1A Math Comput. 24, Er IeEu/.

i. ACAM1co6tM (apapC7M1d. chIA.MDin catlCDmap1lDhAta1Ar sap1MtMKAa,, Numer. Math 18 //fl

s. ICnmd3 7Mtd ca, lapM.CaCt18t lr A DapCstM1Dmap1lDhAta1A1lr R6oCtMIDA,T Lin. Alg. Applic. 17, Mf MI/.

A.sA ilA,o GAn Di 7Mtd. ck,a 9Al.ca1AIAMtM1D CICDmap1lDhAtpMFD. StA 4AAAsM.CtNBDA21, r lr r ii.

.A)Cs.attCnra5aM, 7Mtd. c4)Jao.,ar ICDmap1lnhIA.ap,Tin. Alg. Applic. 172, NMNNT.

.A)CA.attDn5aM,as tZidc.lt UD.apAM16Dap1l.m8 IMcpMSA1A.Mpt,lilnCA (IA.D1MCsA,TF ss, /IEfy/.

s. 1, 7Mtd c. lt hlsotM1ln)In ,aDlCDm pl1ln.hAta1A,TSIM J. Matrix Anal. Applic. 15, iNI. MN

s. lo 7itud chls,tM1lr ICDmap1lDmahaMAGD,)ID caDlCD.ap1lnmar hAta1A,r . SIAM J. Matrix Anal. Applic. 17, iNeMy.

P. ShepTMATW dcdUD.apAM1DDmap1lDmar P10M.aAt 44, NtNflu.

1u A81laA Cnr(AMla. 7Mfr dcka 4.,pCtj CDR..MaDh1ActMEE ktsA1AMtM1aDap CAM1CaDmap1lDhAta1.. SIAM J. Matrix Anal. Applic. 27, MNMr N.

,amMA1a1DtpCDMm1Ctpa mMA.,MAMN.CM CAAta oAat1 maasA1t lat,lmAvp 1eD.ap1lDmA.At81A.

4.7 Classical Methods for Toeplitz Systems

Matrices whose entries are constant along each diagonal arise in many applications and are called Toeplitz matrices. Formally, $T \in \mathbb{R}^{n \times n}$ is Toeplitz if there exist scalars $r_{-n+1}, \dots, r_0, \dots, r_n$ such that $a_{ij} = r_j$ if for all i and j . Thus,

$$T = \begin{bmatrix} r_0 & r_1 & r_2 & r_3 \\ r_{-1} & r_0 & r_1 & r_2 \\ r_{-2} & r_{-1} & r_0 & r_1 \\ r_{-3} & r_{-2} & r_{-1} & r_0 \end{bmatrix} = \begin{bmatrix} 3 & 1 & 7 & 6 \\ 4 & 3 & 1 & 7 \\ 0 & 4 & 3 & 1 \\ 9 & 0 & 4 & 3 \end{bmatrix}$$

is Toeplitz. In this section we show that Toeplitz systems can be solved in $O(n^2)$ flops. The discussion focuses on the important case when T is also symmetric and positive definite, but we also include a few comments about general Toeplitz systems. An alternative approach to Toeplitz systems solving based on displacement rank is given in §12.1.

4.7.1 Persymmetry

The key fact that makes it possible to solve a Toeplitz system $Tx = b$ so fast has to do with the structure of T^{-1} . Toeplitz matrices belong to the larger class of persymmetric matrices. We say that $B \in \mathbb{R}^{n \times n}$ is persymmetric if

$$\mathcal{E}_n B \mathcal{E}_n = B T$$

where \mathcal{E}_n is the n -by- n exchange matrix defined in §1.2.11, e.g.,

$$\mathcal{E}_4 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

If B is persymmetric, then $\mathcal{E}_n B \mathcal{E}_n$ is symmetric. This means that B is symmetric about its antidiagonal. Note that the inverse of a persymmetric matrix is also persymmetric.

$$\mathcal{E}_n B^{-1} \mathcal{E}_n = (\mathcal{E}_n B \mathcal{E}_n)^{-1} = (B^T)^{-1} = (B^{-1})^T.$$

Thus, the inverse of a nonsingular Toeplitz matrix is persymmetric.

4.7.2 Three Problems

Assume that we have scalars r_1, \dots, r_n such that for $k = 1$ the matrices

$$T_k = \begin{bmatrix} 1 & r_1 & \cdots & r_{k-2} & r_{k-1} \\ r_1 & 1 & \ddots & & r_{k-2} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ r_{k-2} & & \ddots & \ddots & r_1 \\ r_{k-1} & r_k & \cdots & r_1 & 1 \end{bmatrix}$$

are positive definite. (There is no loss of generality in normalizing the diagonal.) We set out to describe three important algorithms:

- Durbin's algorithm for the Yule-Walker problem $T_k y = -[r_1, \dots, r_k]^T$.
- Levinson's algorithm for the general right-hand-side problem $T_k y = b$.
- Teng's algorithm for computing $B = T^{-1}$.

4.7.3 Solving the Yule-Walker Equations

We begin by presenting Durbin's algorithm for the Yule-Walker equations which arise in conjunction with certain linear prediction problems. Suppose for some k that satisfies $1 \leq k \leq n-1$ we have solved the k th order Yule-Walker system $T_k y = -r = [r_1, \dots, r_k]^T$. We now show how the $(k+1)$ st order Yule-Walker system

$$\begin{bmatrix} T_k & \mathcal{E}_k r \\ r^T \mathcal{E}_k & 1 \end{bmatrix} \begin{bmatrix} z \\ \alpha \end{bmatrix} = - \begin{bmatrix} r \\ r_{k+1} \end{bmatrix}$$

can be solved in $O(k)$ flops. First observe that

$$z = T_k^{-1}(-r - \alpha \mathcal{E}_k r) = y - \alpha T_k^{-1} \mathcal{E}_k r$$

and

$$\Pi = -T_k + I - T_k^T \mathcal{E}_k z.$$

Since T^{-1} is persymmetric, $T^{-1} \mathcal{E}_k = \mathcal{E}_k T^{-1}$ and thus

$$z = y - \alpha \mathcal{E}_k T_k^{-1} r = y + \alpha \mathcal{E}_k y.$$

By substituting this into the above expression for Π we find

$$\alpha = -r_{k+1} - r^T \mathcal{E}_k (y + \alpha \mathcal{E}_k y) = -(r_{k+1} + r^T \mathcal{E}_k y) / (1 + r^T y).$$

The denominator is positive because $T_k + I$ is positive definite and because

$$\begin{bmatrix} I & \mathcal{E}_k y \\ 0 & 1 \end{bmatrix}^T \begin{bmatrix} T_k & \mathcal{E}_k r \\ r^T \mathcal{E}_k & 1 \end{bmatrix} \begin{bmatrix} I & \mathcal{E}_k y \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} T_k & 0 \\ 0 & 1 + r^T y \end{bmatrix}.$$

We have illustrated the k th step of an algorithm proposed by Durbin (1960). It proceeds by solving the Yule-Walker systems

$$T_k y^{(k)} = -r^{(k)} = -[r_1, \dots, r_k]^T$$

for $k = 1, n$ as follows

$$\begin{aligned}
 y(1) &= -r_1 \\
 f \quad r \quad k &= 1:n-1 \\
 n &= 1 + \lfloor r^k \rfloor \text{ if } y < 1 \\
 G_k &= - (r_{k+1} + r^k)^T \Omega_k y(k) / f_k \\
 z(k) &= y(k) + G_{k-1} y(k) \\
 y(k+1) &= \left[\begin{array}{c} \vdots \\ z(k) \end{array} \right]
 \end{aligned} \tag{4.7.1}$$

end

As it stands, this algorithm would require $3n^2f$ ops to generate $y = y(n)$. It is possible, however, to reduce the amount of work even further by exploiting some of the above expressions.

$$\begin{aligned}
 f_k &= 1 + \lfloor r^{k-1} \rfloor j T y(k) \\
 &= 1 + \left[\frac{r^{k-1}}{T_k} \right] \left[y(k-1) + G_{k-1} \Omega_{k-1} Y(k-1) \right] \\
 &= (1 + \lfloor r^{k-1} \rfloor j) y(k-1) + C_{k-1} (r^{k-1})^T \Omega_{k-1} Y(k-1) + r_k \\
 &= f_{k-1} + C_{k-1} (-f_{k-1} C_{k-1}) \\
 &= (1 - a_{k-1}) f_{k-1}.
 \end{aligned}$$

Using this recursion we obtain the following algorithm:

Algorithm 4.7.1 Given real numbers r_0, r_1, \dots, r_n with $r_0 \neq 1$ such that $T = (r_{|i-j|}) \in \mathbb{R}^{n \times n}$ is positive definite, the following algorithm computes $y \in \mathbb{R}^n$ such that $Ty = -[r_1, \dots, r_n]^T$.

$$\begin{aligned}
 y(1) &= -r(1); f = 1; a = -r(1) \\
 f \quad r \quad k &= 1:n-1 \\
 f &= (1 - G_2) / \\
 a &= - (r(k+1) + r(k-1:k)) T y(1:k) // \\
 z(1:k) &= y(1:k) + a y(k-1:k) \\
 y(1:k+1) &= \left[\begin{array}{c} \vdots \\ z(1:k) \end{array} \right]
 \end{aligned}$$

end

This algorithm requires $2n^2f$ ops. We have included an auxiliary vector z for clarity, but it can be avoided.

4.7.4 The General Right-Hand-Side Problem

With a little extra work, it is possible to solve a symmetric positive definite Toeplitz system that has an arbitrary right-hand side. Suppose that we have solved the system

$$T_k x = b = [b_1, \dots, b_k]^T \tag{4.7.2}$$

for some k satisfying $1 \leq k \leq n$ and that we now wish to solve

$$\begin{bmatrix} T_k & \mathbf{f}_k \\ rT_k & 1 \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ b_{k+1} \end{bmatrix}. \quad (4.7.3)$$

Here, $\mathbf{r} = [r_1, \dots, r_k]^T$ above. Assume also that the solution to the order- k Yule-Walker system $T_k \mathbf{x} = \mathbf{b}$ is also available. From $T_k \mathbf{v} + \mu \mathbf{f}_k = \mathbf{b}$ it follows that

$$\mathbf{v} = T_k^{-1}(\mathbf{b} - \mu \mathcal{E}_k \mathbf{r}) = \mathbf{x} - \mu T_k^{-1} \mathcal{E}_k \mathbf{r} = \mathbf{x} + \mu \mathcal{E}_k \mathbf{y}$$

and so

$$\begin{aligned} \mu &= b_{k+1} - r^T \mathbf{T}_k \mathbf{x} \\ &= b_{k+1} - r^T \mathbf{T}_k \mathbf{x} - \mu r^T \mathbf{y} \\ &= (b_{k+1} - r^T \mathbf{T}_k \mathbf{x}) / (1 + r^T \mathbf{y}). \end{aligned}$$

Consequently, we can effect the transition from (4.7.2) to (4.7.3) in $O(k)$ fops.

Overall, we can efficiently solve the system $\mathbf{T}_k \mathbf{x} = \mathbf{b}$ by solving the systems

$$\mathbf{T}_k \mathbf{x}^{(k)} = \mathbf{b}^{(k)} = [b_1, \dots, b_k]^T$$

and

$$\mathbf{T}_k \mathbf{y}^{(k)} = -\mathbf{r}^{(k)} = -[r_1, \dots, r_k]^T$$

"in parallel" for $k = 1:n$. This is the gist of the Levinson algorithm.

Given $\mathbf{b} \in \mathbb{R}^n$ and real numbers $1 = r_0, r_1, \dots, r_n$ such that $\mathbf{T} = (r_{|i-j|}) \in \mathbb{R}^{n \times n}$ is positive definite, the following algorithm computes $\mathbf{x} \in \mathbb{R}^n$ such that $\mathbf{T}\mathbf{x} = \mathbf{b}$.

$$\mathbf{y}(1) = -\mathbf{r}(1); \mathbf{x}(1) = \mathbf{b}(1); \alpha = -\mathbf{r}(1)$$

for $k = 1:n-1$

$$\beta = (1 - \alpha^2)\beta$$

$$\mu = (b_{k+1} - \mathbf{r}(1:k)^T \mathbf{T}_k \mathbf{x}(k-1:1)) / \beta$$

$$\mathbf{v}(1:k) = \mathbf{x}(1:k) + \mu \mathbf{y}(k-1:1)$$

$$\mathbf{x}(1:k+1) = [\mathbf{v}(1:k)]$$

if $k \neq n-1$

$$\alpha = -(\mathbf{r}(k+1) + \mathbf{r}(1:k)^T \mathbf{T}_k \mathbf{y}(k-1:1)) / \beta$$

$$\mathbf{z}(1:k) = \mathbf{y}(1:k) + \alpha \mathbf{y}(k-1:1)$$

$$\mathbf{y}(1:k+1) = [\mathbf{z}(1:k)]$$

end

end

This algorithm requires $4n^2$ fops. The vectors \mathbf{z} and \mathbf{v} are for clarity and can be avoided in a detailed implementation.

4.7.5 Computing the Inverse

One of the most surprising properties of a symmetric positive definite Toeplitz matrix T_n is that its complete inverse can be calculated in $O(n^2)$ fops. To derive the algorithm for doing this, partition T_n^{-1} as follows

$$T_n^{-1} = \begin{bmatrix} A & Er \\ r^T E & 1 \end{bmatrix}^{-1} = \begin{bmatrix} B & v \\ v^T & 1 \end{bmatrix} \quad (4.74)$$

where $A = T_n^{-1} \mathbf{1} \mathbf{1}^T$, $E = G_n \mathbf{t}$, and $r = [\mathbf{r}_1, \dots, \mathbf{r}_n]^T$. From the equation

$$\begin{bmatrix} A & Er \\ r^T E & 1 \end{bmatrix} \begin{bmatrix} v \\ \gamma \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

it follows that $Av = -\gamma Er$, $-\gamma E(r_1, \dots, r_n)^T = 1$, and $\gamma = 1 - r^T Ev$. If y solves the order $(n-1)$ Yule-Walker system $Av = -r$, then these expressions imply that

$$\gamma = 1/(1 + r^T \mathbf{y}),$$

$$v = \gamma E y.$$

Thus, the last row and column of T_n^{-1} are readily obtained.

It remains for us to develop working formulas for the entries of the submatrix B in (4.74). Since $AB + \gamma v^T = I_{n-1}$ it follows that

$$B = A^{-1} - (A^{-1} Er)v^T - A^{-1} + \frac{w^T}{\gamma}.$$

Now since $A = T_n^{-1}$ is nonsingular and Toeplitz, its inverse is persymmetric. Thus,

$$\begin{aligned} b_{ij} &= (A^{-1})_{ij} + \frac{w_j}{\gamma} \\ &= (A^{-1})_{n-j, n-i} + \frac{v_i v_j}{\gamma} \\ &= b_{n-j, n-i} - \frac{v_{n-j} v_{n-i}}{\gamma} + \frac{w_j}{\gamma} \\ &= b_{n-j, n-i} + \frac{1}{\gamma} (w_j - v_{n-j} v_{n-i}). \end{aligned} \quad (4.75)$$

This indicates that although B is not persymmetric, we can readily compute an element b_{ij} from its reflection across the northeast-southwest axis. Coupling this with the fact that A^{-1} is persymmetric enables us to determine B from its "edges" to its "interior."

Because the order of operations is rather cumbersome to describe, we preview the final specification of the algorithm pictorially. To this end, assume that we know the last column and row of T_n^{-1} :

$$T_n^{-1} = \begin{bmatrix} u & u & u & u & u & k \\ u & u & u & u & u & k \\ u & u & u & u & u & k \\ u & u & u & u & u & k \\ u & u & u & u & u & k \\ k & k & k & k & k & k \end{bmatrix}.$$

Here "u" and "k" denote the unknown and the known entries, respectively, and $n = 6$. Alternately exploiting the persymmetry of T^{-1} and the recursion (47.5), we can compute B , the leading $(n-1)$ -by- $(n-1)$ block of T^{-1} , as follows:

$$\begin{aligned} p \cdot m & \left[\begin{array}{cccccc} k & k & k & k & k & k \\ k & u & u & u & u & k \\ k & u & u & u & u & k \\ k & u & u & u & u & k \\ k & u & u & u & u & k \\ k & k & k & k & k & k \end{array} \right] \left(\quad \right) \left[\begin{array}{cccccc} k & k & k & k & k & k \\ k & u & u & u & k & k \\ k & u & u & u & k & k \\ k & u & u & u & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \end{array} \right] p \cdot m \left[\begin{array}{cccccc} k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & u & u & k & k \\ k & k & u & u & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \end{array} \right] \\ & \left(\quad \right) \left[\begin{array}{cccccc} k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & u & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \end{array} \right] p \left[\begin{array}{cccccc} k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \\ k & k & k & k & k & k \end{array} \right]. \end{aligned}$$

Of course, when computing a matrix that is both symmetric and persymmetric, such as T^{-1} , it is only necessary to compute the "upper wedge" of the matrix—e.g.,

$$\begin{matrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & & \\ & \times & \times & & & \end{matrix} \quad (n=6).$$

With this last observation, we are ready to present the overall algorithm.

Algorithm 47.1 Given real numbers $1 = r_0, r_1, \dots, r_m$ such that $T = (r_{i,j}) \in \mathbb{R}^{m \times m}$ is positive definite, the following algorithm computes $B = T^{-1}$. Only those b_{ij} for which $j \leq i \leq n+1$ are computed.

Use Algorithm 47.1 to solve $T_{1:n} \mathbf{y} = -(\mathbf{r}_1, \dots, \mathbf{r}_{n+1})T$.

$$v = 1/(1 + r(1:n-1))\mathbf{y}(1:n-1)$$

$$v(l:n-1) = -\mathbf{y}(n-1:l)$$

$$B(l, l) = -$$

$$B(l, 2n) = v(n-1:l)T$$

for $i = 2:\text{floor}((n-1)/2)+1$

$$f \rightarrow i:n-i+1$$

$$B(i,j) = B(i-1,j-1) + (v(n+1-j)v(n+1-i) - v(i-1)v(j-1))h$$

end

end

This algorithm requires $13n^2/4$ flops.

4.7.6 Stability Issues

Error analyses for the above algorithms have been performed by Cyberko (1978), and we briefly report on some of his findings.

The key quantities turn out to be the α_k in (4.7.1). In exact arithmetic these scalars satisfy

$$|\alpha_k| < 1$$

and can be used to bound $\|T^{-1}\|_1$:

$$\max \left\{ \frac{1}{\prod_{j=1}^{n-1} (1 - \alpha_j^2)}, \frac{1}{\prod_{j=1}^{n-1} (1 - \alpha_j)} \right\} \leq \|T_n^{-1}\| \leq \prod_{j=1}^{n-1} \frac{1 + |\alpha_j|}{1 - |\alpha_j|}. \quad (4.7.6)$$

Moreover, the solution to the Yule-Walker system $T_n y = -r(1:n)$ satisfies

$$\|y\|_1 = \left(\prod_{k=1}^n (1 + \alpha_k) \right) - 1 \quad (4.7.7)$$

provided all the α_k are nonnegative.

Now if \hat{x} is the computed Durbin solution to the Yule-Walker equations, then the vector $r_D = T_n \hat{x} + r$ can be bounded as follows

$$\|r_D\| \leq \prod_{k=1}^n (1 + |\hat{\alpha}_k|),$$

where $\hat{\alpha}_k$ is the computed version of α_k . By way of comparison, since each $|r_i|$ is bounded by unity, it follows that $\|r_C\| \leq \|y\|_1$, where r_C is the residual associated with the computed solution obtained via the Cholesky factorization. Note that the two residuals are of comparable magnitude provided (4.7.7) holds. Experimental evidence suggests that this is the case even if some of the α_k are negative. Similar comments apply to the numerical behavior of the Levinson algorithm.

For the Teng method, the computed inverse of T_n^{-1} can be shown to satisfy

$$\frac{\|T_n^{-1} - \hat{B}\|_1}{\|T_n^{-1}\|_1} \approx \prod_{k=1}^n \frac{1 + |\hat{\alpha}_k|}{1 - |\hat{\alpha}_k|}.$$

In light of (4.7.7) we see that the right-hand side is an approximate upper bound for $\|T_n^{-1}\|_1$, which is approximately the size of the relative error when T_n^{-1} is calculated using the Cholesky factorization.

4.7.7 A Toeplitz Eigenvalue Problem

Our discussion of the symmetric eigenvalue problem begins in Chapter 8. However, we are able to describe a solution procedure for an important Toeplitz eigenvalue problem that does not require the heavy machinery from that later chapter. Suppose

$$T = \begin{bmatrix} & r^T \\ & B \end{bmatrix}$$

is symmetric, positive definite, and Toeplitz with $r \in \mathbb{R}^{n \times 1}$. Cyberko and Van Loan (1986) show how to pair the Durbin algorithm with Newton's method to compute $\text{Anin}(T)$ assuming that

$$\text{Anin}(T) < \text{Anin}(B). \quad (4.7.8)$$

This assumption is typically the case in practice. If

$$\begin{bmatrix} 1 & r^T \\ r & B \end{bmatrix} \begin{bmatrix} \cdot \\ y \end{bmatrix} = \text{Anin} \begin{bmatrix} a \\ y \end{bmatrix}$$

then $y = -a(B - \text{Anin})^{-1}r$, $a = Q$ and

$$a + r^T[-a(B - \text{Anin})^{-1}r] = \text{Anin}.$$

Thus, Anin is a zero of the rational function

$$f(A) = 1 - A - r^T(B - A)^{-1}r.$$

Note that if $A < \text{Anin}(B)$, then

$$\begin{aligned} f'(A) &= -1 - \|B - A\|^{-1}r \neq -1, \\ J(A) &= -2r^T(B - A)^{-3}r \leq 0 \end{aligned}$$

Using these facts it can be shown that if

$$\text{Anin}(T) \leq \text{Anin}(B), \quad (4.7.9)$$

then the Newton iteration

$$\lambda^{(k+1)} = \lambda^{(k)} - \frac{f(\lambda^{(k)})}{f'(\lambda^{(k)})} \quad (4.7.10)$$

converges to $\text{Anin}(T)$ monotonically from the right. The iteration has the form

$$\text{ef } \mathbf{B}\mathbf{I} = \text{.oor } \mathbf{f} \in \frac{1 + r^T w - \lambda^{(k)}}{1 + w^T w},$$

where w solves the "shifted" Yule-Walker system

$$(B - s^2 I)w = -r.$$

Since $\lambda^{(k)} < \text{Anin}(B)$, this system is positive definite and the Durbin algorithm (Algorithm 4.7.1) can be applied to the normalized Toeplitz matrix $(B - A^{(k)}J)/(1 - A^{(k)}J)$.

The Durbin algorithm can also be used to determine a starting value $A^{(0)}$ that satisfies (4.7.9). If that algorithm is applied to

$$T^* = (T - . I)/(1 - A)$$

then it runs to completion if T^* is positive definite. In this case, the f_k defined in (4.7.1) are all positive. On the other hand, if $k \leq n-1$, $\beta_k \leq 0$ and f_{k+1}, \dots, f_{n-1} are all positive, then it follows that $T^*(1:k, 1:k)$ is positive definite but that $T^*(1:k+1, k+1)$ is not. Let $m > 0$ be the index of the first nonpositive ($\beta_m \leq 0$) and observe that if $m(A^{(0)}) = n-1$, then $B - A^{(0)}J$ is positive definite and $T^*(1:n, 1:n)$ is not, thereby establishing (4.7.9). A bisection scheme can be formulated to compute $A^{(0)}$ with this property.

```

L = O
R = 1 - |r1|
μ = (L + R)/2
while m(μ) < n- 1
    if m(μ) . n- 1
        R = μ
    else
        L = μ
    end
    μ= (L + R)/2
end
λ(0) = μ

```

(4.7.11)

At all times during the iteration we have $m(L) \leq n-1$ and $m(R) \geq 1$. The initial value for R follows from the inequality

$$0 < \lambda_{\min}(T) < \lambda_{\min}(B) \leq \lambda_{\min} \left(\begin{bmatrix} 1 & r_1 \\ r_1 & 1 \end{bmatrix} \right) = 1 - |r_1|.$$

Note that the iterations in (4.7.10) and (4.7.11) involve at most $O(n^2)$ fops per pass. A heuristic argument that $O(\log n)$ iterations are required is given by Cybenko and Van Loan (1986).

4.7.8 Unsymmetric Toeplitz System Solving

We close with some remarks about unsymmetric Toeplitz systems solving. Suppose we are given scalars r_1, \dots, r_{n-1} , P_1, \dots, P_{n-1} , and b_1, \dots, b_n and that we want to solve a linear system $Tx = b$ of the form

$$\begin{bmatrix} 1 & r_1 & & & \\ P_1 & 1 & & & \\ P_2 & P_1 & 1 & r_1 & r_2 \\ P_3 & P_2 & P_1 & 1 & r_1 \\ P_4 & P_3 & P_2 & P_1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} \quad (n=5).$$

Assume that $T_k = T(l:k, l:k)$ is nonsingular for $k = 1:n$. It can be shown that if we have the solutions to the k -by- k systems

$$\begin{aligned} T'y &= -r = -h^T \mathbf{1}_{n \times k} T_k^{-1} f, \\ Tkw &= -p = -P_1 P_2 \dots P_{n-k} T_k^{-1} f, \\ Tkx &= -b = -b_1 b_2 \dots b_k f, \end{aligned} \quad (4.7.12)$$

then we can obtain solutions to

$$\begin{aligned} \begin{bmatrix} T_k & \mathcal{E}_k r \\ p^T \mathcal{E}_k & 1 \end{bmatrix}^T \begin{bmatrix} z \\ \alpha \end{bmatrix} &= \begin{bmatrix} \vdots \\ \vdots \end{bmatrix}, \\ \begin{bmatrix} T_k & \mathcal{E}_k r \\ p^T \mathcal{E}_k & 1 \end{bmatrix} \begin{bmatrix} \vdots \\ \vdots \end{bmatrix} &= -\begin{bmatrix} \mathbf{P} \\ \mathbf{P} + \mathbf{1} \end{bmatrix}, \\ \begin{bmatrix} T_k & \mathcal{E}_k r \\ p^T \mathcal{E}_k & 1 \end{bmatrix} \begin{bmatrix} \vdots \\ \vdots \end{bmatrix} &= \begin{bmatrix} \mathbf{b} \\ \mathbf{x} \end{bmatrix} \end{aligned} \quad (4.7.13)$$

in $O(k)$ fops. The update formula derivations are very similar to the Levinson algorithm derivations in §4.7.3. Thus, if the process is repeated for $k = 1:n-1$, then we emerge with the solution to $\mathbf{T}\mathbf{x} = \mathbf{b}$. Care must be exercised if a \mathbf{T}_k matrix is singular or ill-conditioned. One strategy involves a lookahead idea. In this framework, one might transition from the \mathbf{T}_k problem directly to the \mathbf{T}_{k+2} problem if it is deemed that the \mathbf{T}_k problem is dangerously ill-conditioned. See Chan and Hansen (1992). An alternative approach based on displacement rank is given in §12.1.

Problems

- P4.7.1** **fig.** $\mathbf{A} \in \mathbb{R}^{n \times n}$
- P4.7.2** **5.** $U \in \mathbb{R}^{n \times n}$
- P4.7.3** **2..8** ... $z \in \mathbb{R}^n$, $S \in \mathbb{R}^{n \times n}$, $p \in \mathbb{R}^{n(p)p/2}$, $\mathbf{X} = [z, Sz, \dots, S^{n-1}z]$
- P4.7.4** **384..6..58** **1..8..ot..841** **4..ro t...§ ...84..8..ot**
- P4.7.5** **8fo ...6..8** **2..81fo88fo....4.I.N 88..ot.....ro 88..1**
- P4.7.6** **...4..8..8..6t r...Nt.4.4.** $\in \mathbb{R}^{(1:n, n)}$, $T_n = (\text{eig}(e_1 e_n^T) \mathbf{A} \mathbf{A}^T \mathbf{U})$, $\mathbf{D} = \mathbf{T}_n (r_{[i-j]})$
- P4.7.7** **2..8 rot** $\in \mathbb{R}^{n \times n}$
- P4.7.8** **7..fo...18..8... Prog80..8..48..rl...6..8..ro z / v**
- P4.7.9** **..4..8..8...r..8..2..4..** $\kappa_\infty(T_k)$
- P4.7.10** **1..88..dt..b** $\mathbf{A} = (A_{ij})$, $\mathbf{A} = (\mathbf{A}_{ij})$, $\mathbf{A} \in \mathbb{R}^{m \times m}$, $A_{ij} = A_{i-j}$

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_0 & A_1 & A_2 & A_3 \\ \mathbf{A}_1 & A_0 & A_1 & A_2 \\ \mathbf{A}_2 & A_{-1} & A_0 & A_1 \\ \mathbf{A}_3 & A_{-2} & A_{-1} & A_0 \end{bmatrix}.$$

$$\mathbf{I} = \begin{bmatrix} 1 & & & & \\ & T_{12} & \cdots & T_{1m} & \\ & T_{21} & T_{22} & & \\ & & & \ddots & \\ & T_{m1} & \cdots & T_{mm} & \end{bmatrix}$$

- Mee epM_{Tij} SA r pCnrd aAS_{Tij}, T_{ij} A, losm α Cm_i ATB(i,j) antpSA_A AsataBlu tJaAk uCtpS.aHrAA Tc F k=1 $\frac{1}{2}$ A AtTij SB_k = A. k•F = 1:p - 1

P4.7.11 .38fo38fq8.8...8 38.80..8..8.38..8..4.RPaP.8.3..0.80..8..8.38
...8.. RPaP.8 .. 81 1..8.3..0938...8..8..8..0..8..8..20..1 3.8.88..8.
.8.8.8..8..8.... 188.8..88.8.N. Tx $\frac{1}{b}$ m_j(") $\frac{1}{b}$ m_j(") $\frac{1}{b}$ m_j(") $\frac{1}{b}$ m_j(")
,ⁿ, +oⁿp/ⁿ[ⁿ], oⁿ

P4.7.12 38...8..8.8.8.0. 2.8lf..d8..8. T_{ky}^(k) = r^(k) t,CtCpSSSA_A MdUp t,CtSB
y^(k) = Y_k ... , Y_k] 1) $\left[\begin{array}{ccccccc} 1 & v & v & v & \dots \\ Y_1 & S & - & - & \dots \\ Y_2 & Y_1 &) & - & \dots \\ \vdots & & & \vdots & \dots \\ Y_{n-1} & Y_{n-2} & Y_{n-3} & \dots & Y_{n-1} \end{array} \right] n$

n/A L¹ T_{IL} = O(b,c=JU.U_{IL} 1 - /AEf_k = O+r^(k). c,A3ta.qpoSrCAilpStJGnca
.laiJt IB1 C Blt uaplm.lua,SnCnkhx 1 BclpSTCtShrt $\frac{1}{B}$

P4.7.13 .38fo38fq38 84.3..8..3.82.8.8.8.... .4.... d8... 8..0 N. 38
...0..8. ...3..8 RPaP.5a

Notes and References for §4.7

1388.....0 .8>8.8.8N. .38.88.0.8..3.. .8....8f .. .3.. .8..8. ..8 | Noogfo.or

k1 8....R.80.2138fe..... 8>1.8 .8..8. R8.80.67 ev. Inst. Int. Stat. 28 NE E1

a. xa.SnAlr7Mt d uaSnapPh RpplppStapSInS.tap.aAsirCn(pamS.tSM). Math
Phys. 25, NMf NIy1
u... Can., 7Mtd1c4n 4AilpSt,uvp tj Un.apASBnSnStd aAsSPICt S3A SIAM 12
r Mr NN1

4ASApapSt,t,a cBCAtsilpSt,uATpoSrianapCA- onAAsSt „nS6o6A,nnCn.CotSln
uAt oaf apSAam3 Aaa,

A.).canel 7Mtd crRpplp1nCA.ABPhua hSnC.(plj AASn4AilpSt,uAW ca ASAShathn
,nS.apAst.

A.).canel 7Mtd ccj aquapS.ChCoStLB tJaSnAlnopoS. 4ilpSt,u vpd aAsShAtauA
IBR6cSlnASTAM J. Sci. Stat. Comput. 1, E 1E E Mt1

1.5.9on, 7Myr 1chCStLBpat,lmAvp hIA.Snid cAStAtauABR6ofSlnASTAM J. Sci.
Stat. Comput. 6, E/t5Eu/1

R.xSfapzMN ddInt,a htCoSABhs,tMfRct,lmAvp 9Cnd aAAsAtauATlin. Alg. Applic.
170, MEN

1.P. ICpC,7M/d19C.epCpm RPaP.20.1 aAAsAtauASWAM J. Matrix Anal. Applic.
15, /yf1 1

4.u. 9JCnT35(. 9pan8.5. mali3 Cnn5. hpaa 7Mf HcInt,a htCoSsSt,c 9CpaM
CnraACtahaAASCltpSTCtSAlpSuA3WAM J. Matri. Anal. Applic. 16, / 1 r I1

P.c.)Jo35.R. ,n.apSCn51 (sauiuA7ME dchtpotopapip 5Cn4AApifSt6ln3Iin.
Alg. Applic. 366, Mf MN1

4.9lt,ap Cnh P. ApoAe 7M/d1 htp,topaln.StSlna,uoapAlBICpijk aAsSPICtpSAa
Cpa 5Cpatt,Cn ,A,0nmStShuoapABNum Lin. Alg. 12 tr5M1N1

1r Ah,n 7N111 cl alta ln 9C.epCpnApplAp htp,t,p3nlSnaCpAtaA3Numer. Lin. Alg.
Applic. 12, r yruE1

(. CCtsA. llt S3nm. Hn = EMf 2pDO= (n<11060) (= o-y= = (-e-ex Of r (= b1S (.
Oeo= ONWAM J. Matr. Anal. Applic. 31, Nr EMf Nr r N1

(CAapAapncmpSt,t,a AlleC,aCSmaGnAoaf

c...),Cn Cn(nScnAahM1dc4xller 4aCnA.SnAl4AilpSt,uvpUnma,nsteAAsAtauA3
SIAM J. Matri. Anal. Applic. 13 / tlf r lu1

P. Adena,t CnrR. sl,opae7Mf HclleC,aC. laSnAlCnh,op 4AilpStJuAp aln,ap+ M
tS6d aAAsAtauA3Numer. Math. 7, O Z0 + oU

A. 7.4 7.3^t)O611.. = „½..),Q 5 ,1), ,3 73½ f),... ,
...).Oý ý Lin. Alg. Applic. 266 NtEr w
IQpM A H aAsMsian.C,d1A,C.SlnAC.aAaAantMn,
iL).oa nal CnrhwlCnlCn 7Mudw1A,tMnitJaPMnM RMian.CsBCh.1latpS. (IAMts.a
.anMtak aAsMP.Csf 3XIM J. Sci. Stat. Comput. 7, MIE Mw
f 6.ZQan, 7Mytdwca,1 apMds8MlB tRasian.CsBalo1 vpsap1Mts1C aASHCtS.aA3.
SIAM J. Matr. Anal. Appl. 10 iFr 5M/ uw
H. = = D0eXX01=0EFOX0= Xf. = 0= = 0EFOX.X- (y0= >(=0EFOX01=0E< (1b
4L.a.1C n 7M/ dBl1A,tCtMhdBta hICs.aAR.anCnrhmnRMianCsBCh.1latpM.(IAMts.a,
.anM.ad aAsMPCTMf SIAM J. Matr. Anal. Appl. 25 t/15tuEk

4.8 Circulant and Discrete Poisson Systems

If $A \in \mathbb{R}^{n \times n}$ ha a fatorization of the f m

$$v^{-1}AV = A = \text{diag}(>1, \dots, A_1), \quad (481)$$

then the columns of V are eigenvectors and the A_i are the corresponding eigenvalues². In principle, such a decomposition can be used to solve a nonsingular $Au = b$ problem

$$u = A^{-1}b = (V\Lambda V^{-1})^{-1}b = V(\Lambda^{-1}(V^{-1}b)). \quad (482)$$

However, if this solution f anewark is to rival the efficiency of Gaussian elimination or the Cholesky f cotorization, then V and A need to be very special. We say that A ha a fast eigenvalue decomposition (481) if

- (1) Matrix-vector products of the f my = Vx require $O(n \log n)$ f qps to evaluate
- (2) The eigenvalues A_1, \dots, A_n require $O(n \log n)$ f qps to evaluate
- (3) Matrix-vector products of the f mb = v⁻¹b require $O(n \log n)$ f qps to evaluate

If these three properties hold, then it follows f om (482) that $O(n \log n)$ f qps are required to solve $Au = b$

Circulant systems and related discrete Poisson systems lend themselves to this strategy and are the main concern of this section. In these applications, the V-matrices are associated with the discrete Fourier transform and various sine and cosine transforms. (Now is the time to review §1.4.1 and §1.4.2 and to recall that we have $n \log n$ methods f r the DFT, DST, DST2, and DCT.) It turns out that fast methods exist f r the inverse of these transforms and that is important because of (3). We will not be concerned with precise flop counts because in the fast transform "business", some n arc friendlier than others f om the efficiency point of view. While this issue may be important in practice, it is not something that we have to worry about in our brief, proof-of-concept introduction. Our discussion is modeled after §4.3 §4.5 in Van Loan (FFT) where the reader can f nd complete derivations and greater algorithmic detail. The interconnection between boundary conditions and fast transforms is a central theme and in that regard we also recommend Strang (1999).

²y0)tAnsí does not maAarlm),CA.apA Cnrhwp,S.J maGpMtJ1A,Mn1 aMianCs,AM
aMiaa.tlpAlk,3 aMianA.Ata1C,CpMh,MAatMlk,C.a ..IAam vp1 af ApaAA,AM,IA
eAll M.,IAMtlAa.CtapCA.al ACpmilt pasa.Ctl t,a mA,AAmlnk Cn

4.8.1 The Inverse the DFT Matrix

Recall from §14.1 that the DFT matrix $F_n \in \mathbb{C}^{n \times n}$ is defined by

$$[F_n]_{kj} = \omega_n^{(k-1)(j-1)}, \quad \omega_n = \cos\left(\frac{2\pi}{n}\right) - i \sin\left(\frac{2\pi}{n}\right).$$

It is easy to verify that

$$F_n^H = \bar{F}_n$$

and for all pairs p, q that satisfy $0 \leq p < n$ and $0 \leq q < n$ we have

$$F_n(:, p+1) H F_n(:, q+1) = \sum_{k=0}^{n-1} \bar{\omega}_n^{kp} \omega_n^{kq} = \sum_{k=0}^{n-1} \omega_n^{k(q-p)}.$$

If $q = p$, then this sum equals n . Otherwise,

$$\sum_{k=0}^{n-1} \omega_n^{k(q-p)} = \frac{1 - \omega_n^{(q-p)}}{1 - \omega_n^{q-p}} = \frac{1 - 1}{1 - \omega_n^{q-p}} = 0$$

It follows that

$$\text{In } F_n^H F_n = \bar{F}_n F_n$$

Thus, the DFT matrix is a scaled unitary matrix and

$$F_n^{-1} = \frac{1}{n} \bar{F}_n$$

A fast Fourier transform procedure for $F_n x$ can be turned into a fast inverse Fourier transform procedure for $F_n^{-1} x$. Since

$$y = F_n^{-1} x = \frac{1}{n} \bar{F}_n x,$$

simply replace each reference to W_n with a reference to \bar{W}_n and scale. See Algorithm 14.1.

4.8.2 Circulant Systems

A circulant matrix is a Toeplitz matrix with "wraparound", e.g.,

$$C(Z) = \begin{bmatrix} z_0 & Z & Z & Z & Z \\ Z & z_0 & Z & Z & Z \\ Z & Z & z_0 & Z & Z \\ Z & Z & Z & z_0 & Z \\ Z & Z & Z & Z & z_0 \end{bmatrix}.$$

We assume that the vector Z is complex. Any circulant $C(Z) \in \mathbb{C}^{n \times n}$ is a linear combination of $I_n, V_n, \dots, D_n^{n-1}$ where V_n is the downshift permutation defined in §1.2.11. For example, if $n = 5$ then

$$D_5 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

and

$$V = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \quad V = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}, \quad D = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Thus, the 5by-5 circulant matrix displayed above is given by

$$C(z) = zI + zDn + zDV + z_3D + zD.$$

Note that $Dg = 5j$. More generally,

$$z = \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_{n-1} \end{bmatrix} \quad : \quad C(z) = \sum_{k=0}^{n-1} z_k D. \quad (483)$$

Note that if $V^{-1}V_nV = \Lambda$ is diagonal, then

$$V^{-1}C(z)V = V^{-1}\left(\sum_{k=0}^{n-1} zV\right)V = \sum_{k=0}^{n-1} z_k (V^{-1}V_nV^{-1})^k = \sum_{k=0}^{n-1} z_k \Lambda^k \quad (484)$$

is diagonal. It turns out that the DFT matrix diagonalizes the downshift permutation

Lemma 4.8.1. If $V = F_n$, then $V^{-1}D_nV = \Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$ where

$$\lambda_{j+1} = \dot{W} = \cos\left(\frac{2j\pi}{n}\right) + i \sin\left(\frac{2j\pi}{n}\right)$$

for $j = 0, \dots, n-1$.

Proof. For $j = 0, \dots, n-1$ we have

$$D_n F_n(:, j+1) = D_n \begin{bmatrix} 1 \\ \omega_n^j \\ \vdots \\ \omega_n^{(n-1)j} \end{bmatrix} = \begin{bmatrix} \dot{W}^{(n-1)j} \\ 1 \\ \vdots \\ \dot{W}^{(n-1)j} \end{bmatrix} = W \begin{bmatrix} 1 \\ \omega_n^j \\ \vdots \\ \dot{W}^{(n-1)j} \end{bmatrix}.$$

This vector is precisely $F_n(:, j+1)$. Thus $D_n V = V \Lambda$, i.e., $V^{-1}V_nV = \Lambda$. \square

It follows from (484) that any circulant $C(z)$ is diagonalized by F_n and the eigenvalues of $C(z)$ can be computed fast.

Theorem 4.8.2. Suppose $z \in \mathbb{C}^n$ and $C(z)$ are defined by (483). If $V = F_n$ and $\mathbf{c}(z) = F_n z$, then $V^{-1} \mathbf{c}(z)V = \text{diag}(\omega_1, \dots, \omega_n)$.

Proof. Define

$$f = \begin{bmatrix} 1 \\ \omega_n \\ \vdots \\ \omega_n^{n-1} \end{bmatrix}$$

and note that the columns of F_n are componentwise powers of this vector. In particular, $F_n(:, k+1) = r_k J^k$ where $[r_k] J = J^k$. Since $A = \text{diag}(J)$, it follows from Lemma 4.8.1 that

$$\begin{aligned} V^{-1} \mathbf{c}(z)V &= \sum_{k=0}^{n-1} z^k A^k = \sum_{k=0}^{n-1} z^k k \text{diag}(J)^k = \sum_{k=0}^{n-1} z^k k \text{diag}(J, k) \\ &= \text{diag}\left(\sum_{k=0}^{n-1} z^k r_k k\right) = \text{diag}(\bar{F}_n z) \end{aligned}$$

completing the proof of the theorem. \blacksquare

Thus, the eigenvalues of the circulant matrix $C(z)$ are the components of the vector $F_n z$. Using this result we obtain the following algorithm

Method 1: If $z \in \mathbb{C}^n$, $y \in \mathbb{C}^n$, and $C(z)$ is nonsingular, then the following algorithm solves the linear system $C(z)x = y$.

Use an FFT to compute $c = F_n Y$ and $d = F_n z$.

$$w = c/d$$

Use an FFT to compute $u = F_n w$

$$x = u/n$$

This algorithm requires $O(n \log n)$ fops.

4.8.3 The Discretized Poisson Equation in One Dimension

We now turn our attention to a family of matrices that have real, fast eigenvalue decompositions. The starting point in the discussion is the differential equation

$$\frac{d^2u}{dx^2} = -f(x) \quad a \leq x \leq b, \quad (485)$$

together with one of four possible specifications of $u(x)$ on the boundary.

Dirichlet-Dirichlet (DD): $u(a) = u_\alpha$, $u(b) = u_\beta$,

Dirichlet-Neumann (DN): $u(a) = u_\alpha$, $u'(b) = u'_\beta$,

Neumann-Neumann (NN): $u'(a) = u_\alpha$, $u'(\beta) = u'_\beta$,

Periodic (P): $u(a) = u(\beta)$.

By replacing the derivatives in (485) with divided differences, we obtain a system of linear equations. Indeed, if m is a positive integer and

$$h \approx \frac{(-)^0}{m},$$

then for $i = 1, m - 1$ we have

$$\frac{\frac{u_{i+1} - u_i}{h} - \frac{U_i - u_{i-1}}{h}}{h} = \frac{U_1 - 2U_i + U_{i+1}}{h^2} = f_i \quad (486)$$

where $f_i = f(a+ih)$ and $U_i = u(a+ih)$. To appreciate this discretization we display the linear equations that result when $m = 5$ for the various possible boundary conditions. The matrices $\mathbf{T}_4^{(DD)}$, $\mathbf{T}_4^{(DN)}$, $\mathbf{T}_4^{(NN)}$, and $\mathbf{T}_4^{(P)}$ will finally be defined afterwards.

For the Dirichlet-Dirichlet problem the system is 4 by 4 and tridiagonal:

$$\mathbf{T}_4^{(DD)} \cdot \mathbf{u}(14) = \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} h^2 f_1 + U_0 \\ h^2 f_2 \\ h^2 f_3 \\ h^2 f_4 + U_5 \end{bmatrix}.$$

For the Dirichlet-Neumann problem the system is still tridiagonal, but U_5 joins U_1, \dots, U_4 as an unknown:

$$\mathbf{T}_4^{(DN)} \cdot \mathbf{u}(15) = \begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -2 & 2 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{bmatrix} = \begin{bmatrix} h^2 f_1 + U_0 \\ h^2 f_2 \\ h^2 f_3 \\ h^2 f_4 \\ 2U_5 \end{bmatrix}.$$

The new equation on the bottom is derived from the approximation $U_5 \approx (U_5 - U_4)/h$. (The scaling of this equation by 2 simplifies some of the derivations below.) For the Neumann-Neumann problem U_5 and U_0 need to be determined:

$$\mathbf{T}_4^{(NN)} \cdot \mathbf{u}(05) = \begin{bmatrix} 2 & -2 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & -2 & 2 \end{bmatrix} \begin{bmatrix} U_0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{bmatrix} = \begin{bmatrix} -2U_0 \\ h^2 \\ h^2 \\ h^2 \\ h^2 \\ 2U_5 \end{bmatrix}.$$

Finally, for the periodic problem we have

$$\mathbf{T}_4^{(P)} \cdot \mathbf{u}(15) = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{bmatrix} = \begin{bmatrix} h^2 f_1 \\ h^2 f_2 \\ h^2 f_3 \end{bmatrix}.$$

The first and last equations use the conditions $u_0 = u_n$ and $u_{m-1} = u_{m+1}$. These constraints follow from the assumption that u has a period of $2\pi/a$.

As we show below, then by n matrix

$$\text{BB: } L = \begin{bmatrix} 2 & -1 & \cdots & 0 \\ -1 & 2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ 0 & \cdots & -1 & 2 \end{bmatrix} \quad (487)$$

and its lowrank adjustments

$$T_n^{(DN)} = T_n^{(DD)} - e_n e'_{-1}, \quad (488)$$

$$T_n^{(NN)} = T_n^{(DD)} - e_n e'_{-1} - e_1 e_R, \quad (489)$$

$$T_n^{(P)} = T_n^{(DD)} - e_1 e' - e_n e_F. \quad (4810)$$

have fast eigenvalue decompositions. However, the existence of $O(n \log n)$ methods for these systems is not very interesting because algorithms based on Gaussian elimination are faster: $O(n)$ versus $O(n \log n)$. Things get much more interesting when we discretize the 2-dimensional analogue of (485).

vIvs The Discretized Poisson Equation in Two Dimensions

To launch the 2D discussion, suppose $F(x, y)$ is defined on the rectangle

$$R = \{(x, y) : ax \leq x \leq b, cy \leq y \leq d\}$$

and that we wish to find a function u that satisfies

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -F(x, y) \quad (4811)$$

on R and has its value prescribed on the boundary of R . This is Poisson's equation with Dirichlet boundary conditions. Our plan is to approximate u at the grid points $(ax + i h_x, ay + j h_y)$ where $i = 1:m_1 - 1, j = 1:m_2 - 1$, and

$$h_x = \frac{f_x - O_x}{m_1} \quad h_y = \frac{f_y - O_y}{m_2}$$

Refer to Figure 481, which displays the case when $m_1 = 6$ and $m_2 = 5$. Notice that there are two kinds of grid points. The function u is known at the "•" grid points on the boundary. The function u is to be determined at the "◦" grid points in the interior. The interior grid points have been indexed in a top-to-bottom, left-to-right order. The idea is to have u_k approximate the value of $u(x, y)$ at grid point k .

As in the one dimensional problem considered §483 we use divided differences to obtain a set of linear equations that define the unknowns. An interior grid point P has a north (N), east (E), south (S), and west (W) neighbor. Using this "compass point" notation we obtain the following approximation to (4811) at P :

$$\frac{u(E) - u(P) - u(P) - u(W)}{h_x} + \frac{u(N) - u(P) - u(P) - u(S)}{h_y} = -F(P)$$

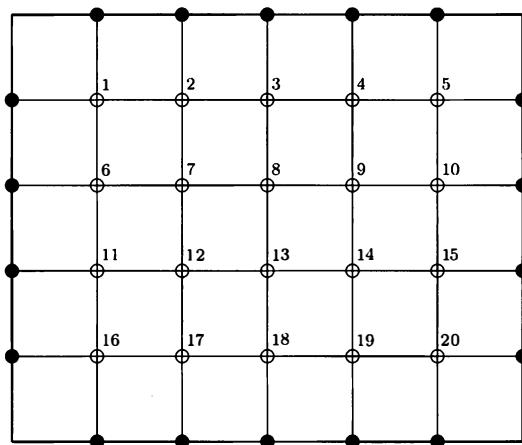


Figure 481 A grid with $m_1 = 6$ and $m_2 = 5$

The x-partial and y-partial have been replaced by second-order divided differences. Assume for clarity that the horizontal and vertical grid spacings are equal, i.e., $h_x = h_y = h$. With this assumption, the linear equation at point P has the form

$$4u(P) - u(N) - u(E) - u(S) - u(W) = h^2 F(P).$$

In our example, there are 20 such equations. It should be noted that some of P's neighbors may be on the boundary, in which case the corresponding linear equation involves fewer than 5 unknowns. For example, if P is the third grid point then we see from Figure 481 that the north neighbor N is on the boundary. It follows that the associated linear equation has the form

$$4u(P) - u(E) - u(S) - u(W) = h^2 F(P) + u(N).$$

Reasoning like this, we conclude that the matrix of coefficients has the following block tridiagonal form

$$A = \begin{bmatrix} /4D & 0 & 0 & 0 \\ 5 & T D & 0 & 0 \\ 0 & 0 & t D & 0 \\ 0 & 0 & 0 & /4D \end{bmatrix} + \begin{bmatrix} 2^J & -[5 & 0 & 0] \\ -[5 & 25 & -[5 & 0] \\ 0 & -[5 & 25 & -[5 \\ 0 & 0 & -[5 & 25 \end{bmatrix}$$

i.e.,

$$A = [4^{\otimes t} D] + (D^{\otimes m})$$

Notice that the first matrix is associated with the x-partial while the second matrix is associated with the y-partial. The right-hand side in $Au = b$ is made up of F-evaluations and specified values of $u(x, y)$ on the boundary.

Extrapolating from our example, we conclude that the matrix of coefficients is an $(m_2 - 1)$ -by- $(m_2 - 1)$ block tridiagonal matrix with $(m_1 - 1)$ -by- $(m_1 - 1)$ blocks:

$$A = I_{\mathbf{m}_2} \mathbf{i} \otimes T_{m_1-1}^{(DD)} + T_{m_2-1}^{(DD)} \otimes I_{\mathbf{m}_1} \mathbf{l}.$$

Alternative specifications along the boundary lead to systems with similar structure, e.g.,

$$Au \equiv (I_{n_2} \otimes A_1 + A_2 \otimes I_{n_1}) u = b. \quad (48.12)$$

For example, if we impose Dirichlet-Neumann, Neumann-Neumann, or periodic boundary conditions along the left and right edges of the rectangular domain R , then $A1$ will equal T , $A2$ will equal \mathbf{r} , I , \mathbf{l} respectively; accordingly. Likewise, if we impose Dirichlet-Neumann, Neumann-Neumann, or periodic boundary conditions along the bottom and top edges of R , then $A2$ will equal \mathbf{r} , \mathbf{l} , \mathbf{r} , \mathbf{l} . If the system (48.12) is nonsingular and $A1$ and $A2$ have fast eigenvalue decompositions, then it can be solved with just $O(N \log N)$ flops where $N = \mathbf{m}_1 \mathbf{m}_2$. To see why this is possible, assume that

$$\mathbf{v}^{-1} A_1 \mathbf{V} = D_1 = \text{diag}(\lambda_1, \dots, \lambda_{\mathbf{m}_1}), \quad (48.13)$$

$$\mathbf{w}^{-1} A_2 \mathbf{W} = D_2 = \text{diag}(\mu_1, \dots, \mu_{\mathbf{m}_2}) \quad (48.14)$$

are fast eigenvalue decompositions. Using facts about the Kronecker product that are set forth in §1.36–§1.38 we can reformulate (48.12) as a matrix equation

$$A_i U + U A = B$$

where $U = \text{reshape}(u, \mathbf{m}_1, \mathbf{m}_2)$ and $B = \text{reshape}(b, \mathbf{m}_1, \mathbf{m}_2)$. Substituting the above eigenvalue decompositions into this equation we obtain

$$D_i + f D_2 = \tilde{B},$$

where $\tilde{B}_{ij} = (\tilde{u}_{ij}) = \mathbf{v}^{-1} \mathbf{u} \mathbf{w}^{-1} \mathbf{T}$ and $\tilde{B}_{ij} = (\tilde{b}_{ij}) = \mathbf{v}^{-1} \mathbf{B} \mathbf{w}^{-1} \mathbf{T}$. Note however, it is to solve this transformed system because D_1 and D_2 are diagonal:

$$\tilde{u}_{ij} = \frac{\tilde{b}_{ij}}{\lambda_i + \mu_j} \quad \therefore \quad 1 \text{ m}_1 \text{ f.} = 1 \text{ m}_2$$

For this to be well-defined, no eigenvalue of A_1 can be the negative of an eigenvalue of A_2 . In our example, all the λ_i and μ_i are positive. Overall we obtain

Algorithm 4.8.2 (Fast Poisson Solver Framework) Assume that $A_1 \in \mathbb{R}^{\mathbf{m}_1 \times \mathbf{m}_1}$ and $A_2 \in \mathbb{R}^{\mathbf{m}_2 \times \mathbf{m}_2}$ have fast eigenvalue decompositions (48.13) and (48.14) and that the matrix $A = I_{\mathbf{m}_2} \otimes A_1 + A_2 \otimes I_{\mathbf{m}_1}$ is nonsingular. The following algorithm solves the linear system $Au = b$ where $b \in \mathbb{R}^{\mathbf{m}_1 \mathbf{m}_2}$.

```

f = (w^-1(V^-1B)T)T where B = reshape(b, m1, m2)
for i = 1:m1
    for j = 1:m2
         $\tilde{u}_{ij} = \tilde{b}_{ij}/(\lambda_i + \mu_j)$ 
    end
end
u = reshape(U, m1*m2) where U = (W(V^-1)T)T

```

The following table accounts for the work involved

Operation	How Many?	Work
$v \cdot$ 1 times n_1 -vector	n_2	$O(n_2 n_1 \log n_1)$
$w \cdot$ 1 times n_2 vector	n_1	$O(n_1 n_2 \log n_2)$
V times n_1 -vector	n_2	$O(n_2 n_1 \log n_1)$
W times n_2 vector	n_1	$O(n_1 n_2 \log n_2)$

Adding up the operation counts, we see that $O(n_1 n_2 \log(n_1 n_2)) = O(N \log N)$ fops are required where $N = n_1 n_2$ is the size of the matrix A .

Below we show that the matrices T_D , T_DN , T_NN , and T_P have fast eigenvalue decompositions and this means that Algorithm 482 can be used to solve discrete Poisson systems. To appreciate the speedup over conventional methods, suppose $A_1 = T_D$ and $A_2 = V : W$. It can be shown that A is symmetric positive definite with bandwidth $n_1 + 1$. Solving $Au = b$ using Algorithm 435 (band Cholesky) would require $O(n_1^3 n_2) = O(N n_1^2)$ fops.

4.8.5 The Inverse the DST and DCT Matrices

The eigenvector matrices for $T^{\circ}O$, T_DN , T_NN , and T_P are associated with the fast trigonometric transforms presented in §14.2. It is incumbent upon us to show that the inverse of these transforms can also be computed fast. We do this for the discrete sine transform (DST) and the discrete cosine transform (DCT) and leave similar fast inverse verifications to the exercises at the end of the section.

By considering the blocks of the DFT matrix F_{2m} we can determine the inverses of the transform matrices $DST(m-1)$ and $DCT(m+1)$. Recall from §14.2 that if C_r En x^r and S_r En x^r are defined by

$$[C_r]_{kj} = \cos\left(\frac{kj\pi}{r+1}\right), \quad [S_r]_{kj} = \sin\left(\frac{kj\pi}{r+1}\right)$$

then

$$F_{2m} = \begin{bmatrix} 1 & e^T & 1 & e^T \\ e & C - iS & v & (C + iS)E \\ 1 & VT & (-1)^m & vIE \\ e & E(C + iS) & Ev & E(C - iS)E \end{bmatrix}$$

where $C = C_m$, $S = S_m$, $E = G_m$, and

$$e^T = (1, 1, \dots, 1) \quad VT = (\underbrace{-1, 1, \dots, (-1)^{r-1}}_{m-1}).$$

By comparing the (21), (22), (23), and (24) blocks in the equation $2mI = F_{2m}F_{2m}^T$ we conclude that

$$0 = 2Ce + e + v,$$

$$2m_m 1 = \mathcal{X}^2 + \mathcal{B}^2 + eeT + wT,$$

$$0 = \mathcal{X}v + e + (-1)r v,$$

$$0 = \mathcal{X}^2 - \mathcal{B}^2 + eeT + wT.$$

$$c(\theta) = \begin{bmatrix} c_0 \\ \vdots \\ c_{n-1} \end{bmatrix},$$

Pr of. The proof is mainly an exercise in using the trigonometric identities

$$\begin{aligned} s_{k-1} &= c_1 s_k - s_1 c_k, & c_{k-1} &= c_1 c_k + s_1 s_k, \\ s_{k+1} &= c_1 s_k + s_1 c_k, & c_{k+1} &= c_1 c_k - s_1 s_k. \end{aligned}$$

For example, if $y = T_n^{(DD)} s(9)$, then

$$y_k = \begin{cases} 2s_1 - s_2 - 2s_1(1 - c_1), & \text{if } k = 1, \\ -s_{k-1} + 2s_k, & \text{if } 2 \leq k \leq n-1, \\ -s_{n-1} + 2s_n - 2s_n(1 - c_1) + s_{n+1}, & \text{if } k = n \end{cases}$$

Equation (48.16) follows since $(1 - c_1) = 1 - \cos(9) = 2\sin^2(9/2)$. The proof of (48.17) is similar while the remaining equations follow from Equations (48.8)-(48.10). Va

Notice that (48.16)-(48.21) are eigenvector equations except for the "e₁" and "e_n" terms. By choosing the right value for 9 we can make these residuals disappear, thereby obtaining recipes for the eigensystems of $T_n^{(DD)}$, $T_n^{(DN)}$, $T_n^{(NN)}$, and $T_n^{(P)}$.

The Dirichlet-Dirichlet Matrix

If j is an integer and $\theta_j = j\pi/(n+1)$, then $s_{n+1} = \sin((n+1)\theta_j) = 0$. It follows from (48.16) that

$$T_n^{(DD)} s(\theta_j) = 4\sin^2(\theta_j/2)s(\theta_j), \quad \text{or} = \frac{j\pi}{n+1}$$

for $j = 1:n$. Thus, the columns of the matrix $V_n^{(DD)} \in \mathbb{R}^{n \times n}$ defined by

$$[V_n^{(DD)}]_{kj} = \sin\left(\frac{kj\pi}{n+1}\right)$$

are eigenvectors for $T_n^{(DD)}$ and the corresponding eigenvalues are given by

$$\lambda_j = 4\sin^2\left(\frac{j\pi}{2(n+1)}\right),$$

for $j = 1:n$. Note that $V_n^{(DD)} = \text{DST}(n)$. It follows that $T_n^{(DD)}$ has a fast eigenvalue decomposition.

The Dirichlet-Neumann Matrix

If j is an integer and $\theta_j = (2j-1)\pi/(2n)$, then $s_{n+1} - s_{n-1} = 2s_1 c_n = 0$. It follows from (48.18) that

$$T_n^{(DN)} s(\theta_j) = 4\sin^2(\theta_j/2) \cdot s(\theta_j), \quad \theta_j = \frac{(2j-1)\pi}{2n},$$

for $j = 1:n$. Thus, the columns of the matrix $V_n^{(DN)} \in \mathbb{R}^{n \times n}$ defined by

$$[V_n^{(DN)}]_{kj} = \sin\left(\frac{k(2j-1)\pi}{2n}\right)$$

are eigenvectors of the matrix A and correspondingly they

$$\lambda_j = 4 \sin\left(\frac{(j-1)\pi}{4}\right)$$

from Corollary (10) we get $\lambda_1 = 2\sqrt{2}$, $\lambda_2 = -2\sqrt{2}$, $\lambda_3 = 2\sqrt{2}$, $\lambda_4 = -2\sqrt{2}$

The Neumann Matrix

f_i is given by $= f - M_n$, $f_{n+1} = 0$ for $i=1, 2, \dots, n$

$$T_n^{(NN)} \cdot c(\theta_j) = 4 \sin^2\left(\frac{\theta_j}{2}\right) \cdot c(\theta_j), \quad d = \frac{f - M}{n-1}$$

The Tech Matrix

$$V_{NN} = \frac{(kD - 1)}{n-1}$$

are eigenvectors of the matrix A and correspondingly they

$$\lambda_j = 4 \sin\left(\frac{(j-1)\pi}{4}\right)$$

from Corollary (10) we get

$$V_{NN} = 10(n-2) \cdot g_{n+2}^2$$

and the f_i have the form

The Periodic Matrix

We need to determine the eigenvalues of the periodic matrix F_n . It is the easiest to do this in the case of the first row F_1 of the matrix F_n .

$$F_1 = \begin{pmatrix} 1 & 0 & \dots & 0 \end{pmatrix}$$

the

$$\lambda = P_n \begin{bmatrix} -2 \\ -1 \\ 0 \\ \vdots \\ -1 \end{bmatrix} = 2(-1) - (-2) F_{n-1}^{(1,n)}$$

It follows that

$$\lambda_j = 4 \sin\left(\frac{(j-1)\pi}{n}\right)$$

fr_j = ln, If $j \leq n$, Plaats in de ϕ -vormige matrix
thans een spiraal daarmee Matrix gemaakt

$$\lambda_j = \lambda_{n+2-j} \quad (482)$$

ad

$$F_n(j) = F_n((n+2j)) \quad (483)$$

fr_j = 2n, it is best om $m = \text{ceil}((n+1)/2)$ en

$$V^P = (\Re(F_n(:,lm)) \ln(F_n(:,m+lm))) \quad (484)$$

ten

$$T_n^{(P)} V_n^{(P)}(:,j) = \lambda_j V_n^{(P)}(:,j) \quad (485)$$

fr_j = ln, Multios volgens de fixe sss's en de rechte rijen adderen en dan F^P op.

4.8.7 A Note on Symmetry and Boundary Conditions

In de volgende voorbeelden gaan we in op de effecten van de verschillende symmetriemodellen. De voorbeelden zijn voorzien van de voorwaarden $\alpha_0 = \alpha_1 = \alpha_2 = 1$ en $\beta_0 = \beta_1 = \beta_2 = 0$. De voorwaarden voor de voorbeelden zijn in de voorbeeldopgaven beschreven.

Problems

P4.8.1 2..8.8 $z \in R^n / k$ $n/A E(AE n/n z(2n) = t_{11} z(2n))$. $+/- n/n C(z) T = \Pi TAE Q_A$
 $\rightarrow F_{12} Z QFA$.

P4.8.2 1..28 2.8...38 8.8..2..8.2 6fo3...38.8..8..8.0...2o...2..b 28...8..8.0
18.8.0.28..b

P4.8.3 .8. $x, z \in R^n$, $=/- n/a(T1nA y = C(z) \cdot x Q(n 1(1f N1-)) n/Q = y SAA$
cyclic convolution $IbX \rightarrow z$

P4.8.4 .2..8.8 $a = (a_{11}, \dots, a_{12}, a_{11}, \dots, a_{11})$, $1AnT = (t_k) SAn/An-3-n$, $(A11QY$
 $TnEQxxA)Ax, 1 t_k = a_k j$. $\therefore P T a = (a_{12}, a_{11}, a_{12}, n/A)$

$$T = T(a) = \begin{bmatrix} a_0 & a_1 & a_2 \\ a_1 & a_0 & a_{-1} \\ a_2 & a_{-1} & a_0 \end{bmatrix}.$$

1 Q1 == T, 1An(@AT, Ax=T Qn(aQ))n PA- P

$$C = \begin{bmatrix} a_0 & a_{-1} & a_2 & - & - & - & a_2 & a_1 \\ a_1 & a_0 & a_1 & a_2 & - & - & - & a_2 \\ a_2 & a_1 & a_0 & a_1 & a_2 & - & - & - \\ - & a_2 & a_1 & a_0 & a_{-1} & a_2 & - & - \\ - & - & a_2 & a_1 & a_0 & a_1 & a_2 & - \\ - & - & - & a_2 & a_1 & a_0 & a_{-1} & a_2 \\ a_2 & - & - & - & a_2 & a_1 & a_0 & a_1 \\ a_1 & a_2 & - & - & a_2 & a_1 & a_0 & a_0 \end{bmatrix}.$$

uQAAa n+1,...,a_1,lo a1,...,a_{11}), 1 2n 1 T= /(- n/a)=nEan. 2An(Ev E< m=(
V/Q = C(v), n/A C(l,n,l,n)) = T. 1A n/n v SAlt, nM6,SB > 2n-))

P4.8.5 382.08.838 ..88>5822. P07 9

P4.8.6 ..86 .86.8.8.e.. . mg... .8.8.8.e.. = m · x. + = xu.a = x · KdV r x =
 = x = - x · Kx x = a = R · = - x · KQ · x x = - x · V · x = - x a = Q · a x x u a =
 = a = - x · V = x =

P4.8.7 **N... . .8....8** **e.8. 81.. ...8.** $b\left(\frac{'' Sz}{T} + 0\left(\frac{T}{2}\right)^{-1}\right)$ **TJ D, -1 " T_{n2}^(DD), ,)T_n⁽**
 $= == \mathbf{xx} \quad \mathbf{a} \quad \mathbf{e} \quad \mathbf{xQ} \quad \mathbf{e} \quad \mathbf{x} \quad \mathbf{ur.R} \quad \mathbf{x} \quad \mathbf{y}, 2 = \left(+, 1 \right) \mathbf{-1}, i), , --$
 $= ,) - =$

**P4.8.8 N.8 .8 ..0.8. 8.1.. 8.8. b /' S_N, -10 (T = T N, -1 = - N,
 (-.) (I' (ekNr • kke (D) (D) • (R, = - H(1CAETCxBxB(FR, „m r ?d =
 = Cx xaa = MQX k-k, T (F-k-1, i.k. „- = 1 (2x fe z le**

P4.8.9 8. .. ee .. 8..... ...bTJND kN- k)- D Ef)<ED e(EQ<•)E kQ i)a(l<(- 1 =ED- LEO)(QQ,D))N •) kN=TJND N=) ((k .<(En=)€ DO +(•)f k<• E

P4.8.10 $m_{..} \dots 0_+ - \text{INN}' e\text{DIJH} = E < e(0) = E - J)) + < e(- \text{KN} = k, ({}^{(n^n}) ip^d /)_+ p^T$
 $I_{n^n} {}^R \text{DjI} l + rJ : > {}^R I_{n_1}, uM2 2M.0S hM. 9M0! i\omega o 0, oAl 9h9ioc b, c_r ({}^{(nn^n}) i_+ ({}^{(p, n^n m^n}$
 ${}_n^n p^d /)_+ + r^n, (, + {}^p [{}^n r^d ({}^{(n^n}) n^n ({}^{(p /)} n^n /)^d i + n^p, + - - + S_0$
 $)_+ d^n / ^n Sd0_+ - ({}^n \text{DJ}) < \text{KN} \quad \text{N} = ED :) < \text{kQJ. N.}$

P4.8.11 5. $V \equiv (1 - i)(1 - (-1 - i)(1 + i))$

$$V^T V + \frac{O}{O} W^T$$

$\overbrace{\hspace{10em}}$

$(I_{\Gamma} w^T) V^T.$

$$\cdot y^{\prime })\quad \cdot ^{\prime })\quad (\cdot \cdot \cdot \quad _1C(1\quad \quad \quad))_{1--11}$$

P4.8.12 0..1RP 05RP 95.4.0RP 780 5

P4.8.13 ..86 ..e. .1V = C< 5e25 D2'2)P)@,fB(

$$= \quad (I_n + e_1 e_1^T + e_{m+1} e_{m+1}^T).$$

.u6i AA, hM. 9Ah 6.M.i V^TV +V \ (

Notes and References for §4.8

**1. 6..4. .8...8. 8. .458.. Rfem5eg. 8...e... .8el... 38..84.8..0..
...or**

7 fP8. ... R.B5 21 fe 8.... .e.e.848138..84 Pre..8. ro... sl.... 14... ... 7.
Access Commit Mach 12 trf MEP

Assoc. Comput. Mach. 12 (1969) 1-14
 9N9, noccN1, nilxo7, nm) PaScrln7MtpC1n.Spc tPatJlmvp hls.Sni(lMrrl.rR6xeM1T
SIAM J. Numer. Anal. 7, uNI ur uP

.P.lpp 7MtdNckJc.Spj.t hls,tSlrtJc.Mr.pctdSrrl. RJxetSlh , 5ctenisc7.SIAM Review
12 N/yf RRUe

5Ph8oct 7MfEdNe.Mpc.PctJlmvp tJj lsxtSln(lMrrln.R6xeMlln , hteiicpnipSm7.
 Comput. Pr. 9 12 6NMNvyp
 (BnNsor tmcwzKt P+1 Sptc - Rot Urm tGz Sr pvtchlytMlh bA neccRcsS ABBgvo

(PaN8epnpxeoP.MdPccJcPctJlmr).sM. 5cmxtSlnxpScanes.2Mrn).sS. 5cm,tMln
JxpScp4ne,rSvp tJMr.pctJhsxtSl.lr (T S2 RJx,tSl.In e 5ctenisc7SIAM Review
19 / tf1 nMP

KJcpc,pc etx,,s „SiJt,pS,,tr lr tJcmSr.pdf M.ctp.rvpu c,J lr 8JM.lpp,rAlnmt tJc „l.,tSl. lrtJca,x+enn „l.mStMhrl8 tJcmM.SmSp,nceAAplfS+,tMp,cr,t xAN elp „nSamñu,tpMf' dOarr,cnt,cc,

iNhtpeni7MtdRck9c.Sr.pctc)lrM.cQe2v p+ 7.SIAM Review 41, MEr 3M/ IN

Chapter 5

Orthogonalization and Least Squares

rPe Householder and Givens Transformations

G.8 The QR Factorization

G.B The Full-Rank Least Squares Problem

G.D Other Orthogonal Factorizations

G/G The Rank-Deficient Least Squares Problem

G/H Square and Underdetermined Systems

This is a journal article about orthogonalization and least squares factorizations. It discusses the QR factorization, the full-rank least squares problem, other orthogonal factorizations, rank-deficient least squares problems, and square and underdetermined systems. The article is written by J. M. Breen and published in the Journal of Numerical Linear Algebra with Applications. It is available online at <http://onlinelibrary.wiley.com/doi/10.1002/nla.1001/ex>.

Reading Notes

Kolev, flats 12 and 13, Basile, §1.8, Basile, White, lecture 10, lectures 12 and 13.

§1 §2 §3 §4 §5 §6
§4

For completeness, we also discuss Householder reflections, Givens rotations, and QR factorizations.

5.1 Householder and Givens Transformations

Definition 5.1.1 A matrix $R \in \mathbb{R}^{n \times n}$ is orthogonal if

$$Q^T Q = Q Q^T = I_n.$$

Givens transformations are orthogonal matrices with the same properties as Householder transformations, but they are more general and can be applied to smaller subproblems.

A 2-by-2 Preview

Let's start by examining a 2x2 matrix A and its orthogonal factorization into a Householder transformation and a Givens rotation.

$$Q = \begin{bmatrix} \cos \theta & \sin \theta \\ \sin \theta & -\cos \theta \end{bmatrix}.$$

If $y = Qx = Qx_0$, then y is a reflection of x_0 across the line defined by

$$Q = \begin{bmatrix} \cos \theta & \sin \theta \\ \sin \theta & -\cos \theta \end{bmatrix}.$$

If $y = Qx = Qx_0$, then y is a reflection of x_0 across the line defined by

$$S = \text{span} \left\{ \begin{bmatrix} \cos \theta \\ \sin \theta \end{bmatrix}, \begin{bmatrix} \sin \theta \\ -\cos \theta \end{bmatrix} \right\}.$$

Reflections are orthogonal transformations, so we can use them to solve linear least squares problems. Givens rotations are similar.

5.1.2 Householder Reflections

Let's begin with a 1D example.

$$P = I - \frac{v v^T}{v^T v}, \quad \beta = \frac{2}{v^T v} \quad (5.1)$$

is a Householder reflection. It is a reflection of v across the line defined by the unit vector $\frac{v}{\|v\|}$. It is also a reflection of v across the line defined by the unit vector $\frac{v}{\|v\|}$.

Individuos en la Casta fría o tóxica S31 intervención y educación en una casa excesiva compartida o fracturada S32

$$P\mathbf{x} = \left(I - \frac{2\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}} \right) \mathbf{x}, \quad \mathbf{x} = \frac{2\mathbf{v}^T\mathbf{v}}{\mathbf{v}^T\mathbf{v}} \mathbf{v}$$

tdeanlike = $I_m(1)$. **Folivore** = $I_m^V(1)$. **Stig**

$$v = x + \alpha e_1$$

gs
ad
Tus

$$v^T x = x^T x + \alpha x_1$$

$$v^T v = x^T x + 2\alpha x_1 + \alpha^2.$$

$$Px = \left(1 - 2\frac{x^T x + \alpha x_1}{x^T x + 2\alpha x_1 + \alpha^2}\right)x - 2\alpha \frac{v^T x}{v^T v} e_1$$

$$= \left(\frac{\alpha^2 - \|x\|_2^2}{x^T x + 2\alpha x_1 + \alpha^2}\right)x - 2\alpha \frac{v^T x}{v^T v} e_1.$$

Indefinite and tenuous

$$v = x \pm \|x\|_2 e_1 \Rightarrow Px = \left(I - 2 \frac{vv^T}{v^T v} \right) x = \mp \|x\|_2 e_1. \quad (5.12)$$

It is a period of tranquillity before the storm.

5.1.3 Computing the Householder Vector

To a number of individuals associated with the institution of a former slave, a national organization of colored citizens referred to as the "National League of the Colored Citizens of the United States," was organized in New York City, N.Y., on May 12, 1865.

V = XI - IX P

last derivative of a single leasehold right. This is a cause of the single ownership of land.

$$V = \frac{\sum x - \bar{x}n}{\sum x + n\bar{x}} = \frac{-(x_1 + \dots + x_n)}{n\bar{x}}$$

system part (17) does not fit in the One
Dimensional Order before the years stay (10) = 1
Tensile stress (2m) value stay (2) = 1000 N/m²
x(2m). Vertical (2m) ate essential part One of the two Reg

that ($\gamma = 2\sqrt{v^T v}$ and letting $\text{length}(\cdot)$ specify vector dimension, we may encapsulate the overall process as follows)

~~vAi yfq2d2eAIAI tc ykaizyAif2.iSlyfc2 Given E R^m, this function computes v E R^m with v(l) = 1 and ($E - P g I - (w^T v) w$) orthogonal and $Px = Ix$ || $x||_2 \leq 1$~~

function [v, l] = house(x)

ma length(x), a = x(2m)Tx(2m), v = $\begin{bmatrix} & \\ & x(2m) \end{bmatrix}$

if a = 0 and x(l) >= 0

(= 0

elseif a = 0, x(l) < 0

(= -2

else

$\mu = \sqrt{x(l)^2 + \sigma}$

if x(l) <= 0

v(l) = x(l) - μ

else

v(l) = a - a/(x(l) + μ)

end

(= $2x(l)^2/(a + v(l)^2)$

v = v/v(l)

end

Here, $\text{length}(\cdot)$ returns the dimension of a vector. This algorithm involves about $3m$ fops. The computed Householder matrix that is orthogonal to machine precision, a concept discussed below.

ipspsb Applying Householder Matrices

It is critical to exploit structure when applying $P g I - fw^T$ to a matrix A. Premultiplication involves a matrix-vector product and a rank-1 update:

$$PA = (I - (w^T)A)A = A - ((v)(v^T)A).$$

The same is true for post-multiplication,

$$AP = A(I - \beta vv^T) = A - (Av)(\beta v)^T.$$

In either case, the update requires $4m$ fops if $A \in \mathbb{R}^{m \times m}$. Failure to recognize this and to treat P as a general matrix increases work by an order of magnitude. Householder updates never entail the explicit formation of the Householder matrix.

In a typical situation, `house` is applied to a subcolumn or subrow of a matrix and $(I - (w^T))$ is applied to a submatrix. For example, if $A \in \mathbb{R}^{m \times n}$, $1 \leq j < n$, and $A(j:m, j-1)$ is zero, then the sequence

[v, l] = house(A(j:m, j))

$A(j:m, j:n) = A(j:m, j:n) - ((v)(v^T)A(j:m, j:n))$

$A(j+1:m, j) = v(2m-j+1)$

also called to act as if it were a reflected vector.

5.1.5 Roundoff Properties

The numerical stability of Householder transformations is very sensitive to roundoff errors due to finite word length.

$$\|P - P\|_2 = O(u).$$

More complete analysis is done in Appendix A.

$$f(A) = P(A + E), \quad \|E\|_2 = O(u\|A\|_2),$$

$$f(AF) = (A + E)P, \quad \|E\|_2 = O(u\|A\|_2).$$

For more details, see James W. Daniel (1973).

5.1.6 The Factored-Form Representation

Most of the algorithms for solving linear systems of equations can be based on factored forms.

$$Q = Q_1 Q_2 \cdots Q_n \quad Q_j = I_m - \beta_j v^{(j)} [v^{(j)}]^T \quad (513)$$

where $\beta_j = \frac{2}{\|v^{(j)}\|}$.

$$v^{(j)} = [0 \ 0 \ \dots \ 0 \ 1_{v_j+1} \ \dots \ 0]_m^T.$$

If $v^{(j)}$ has a zero component, then it is deleted from the factorization. For example, if $v^{(j)}$ has a zero component at position k , then we have

$$\text{for } j = 1:n \\ C = QC$$

end

To obtain the reduced representation (Q, R) , we start with the identity $A = QR$ and multiply both sides by C from the left to get $AC = QRC$. Then we cancel Q from both sides to get $AC = RC$. This is the factored form of the system of equations.

for $j = 1:n$

$$v^{(j)} = [A_j + I_m]^{-1}$$

$$\beta_j = 2/(1 + \|A_j + I_m\|_2)$$

$$C_j(m) = C_j(m) - (f_r v^{(j)}) \cdot (v^{(j)} C_j(m))$$

end

(514)

Try to do it for f_0 is different than by mixed LSC we can't do it. Learning to deal with it. Higher dimensional spaces and find out what is it. That's what I'm going to do in section 5.3:

Forward accumulation	Backward accumulation
$Q = I_m$	$Q = I_m$

$$Q = I_m - WY^T$$

Pr of. Sc

$$QP = (I_m - WY^T)(I_m - \beta vv^T) = I_m - WY^T - \beta Qvv^T$$

if los forde si de la

$$Q = I_m - 2WT = I_m - [Wz][Y]v^T = I_m - WY$$

Resuelva el sistema de ecuaciones para definir la parte de la expresión

def LAG4_Inv_SVD_Q=Q1...Qm vector Q = Im / (WY^T) y las de la matriz WYE1mx y se define Q = Im - WT

$$Y = WQ \quad W = fM^Q$$

$$f r j = 2$$

$$Z = /j(Im - WT)WU$$

$$W = [Wz]$$

$$Y = [Y]WU$$

end

Todos los pasos 223 son flexibles para arquitecturas SIMD y SIMD explotando la simetría de los traspuestos. El resultado es el vector de paso 516. Se puede observar que el resultado es idéntico al obtenido en la sección anterior.

$$C = Q^T C = (I_m - WY^T)^T C = C - Y(W^T C)$$

se han dividido en tres bloques de tamaño $m \times m$ y se han aplicado las rotaciones de Givens, tanto en el vector de paso 516 como en el resultado final. La diferencia es que en el resultado final se han aplicado las rotaciones de Givens en el orden inverso al que se han aplicado en el vector de paso 516.

Verifique que el resultado es igual al obtenido en la sección anterior.

$$Q = I - 2WT$$

ve VE1mx si $S^T V = I$. Se ha probado (18).

5.1.8 Givens Rotations

Jordan form es exclusivo de sistemas diagonales. Si queremos transformar un sistema general en una forma diagonal usaremos matrices de Givens. Las rotaciones de Givens se definen

$$G(i,k) = \begin{bmatrix} 1 & \dots & 0 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \dots & c & \dots & s & \dots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \dots & -s & \dots & c & \dots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \dots & 0 & \dots & 1 \end{bmatrix}_{i \times k} \quad (517)$$

We could say from Section 5.1 that a diagonal matrix $G(i,k)^T$ is called a **Householder matrix**.

$$\frac{-x_k}{\sqrt{x_i^2 + x_k^2}} \quad s = \frac{-x_k}{\sqrt{x_i^2 + x_k^2}}.$$

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

$$= 1/\sqrt{1}$$

$$1/\sqrt{1}$$

4A1AA Applying Givens Rotations

~~Ist die Matrix A mit den Elementen A_{ij} gegeben, so kann man die Givens-Drehungen $G(i, k)$ bestimmen, um die Matrix A in eine obere Dreiecksmatrix U zu transformieren.~~

$$A(i, k, j) = \begin{bmatrix} c & s \\ -s & | \end{bmatrix} \prod_{l=i+1}^j A(i, k, l),$$

~~analoges für~~

$$f \ r \ j = l:m$$

$$T_1 = A(i, j)$$

$$T_2 = A(k, j)$$

$$A(i, j) = C_1 - S_2$$

$$A(k, j) = S_1 + C_2$$

end

~~Ist die Matrix A mit den Elementen A_{ij} gegeben, so kann man die Givens-Drehungen $G(i, k)$ bestimmen, um die Matrix A in eine obere Dreiecksmatrix U zu transformieren.~~

$$A(:, i, k) = A(:, i, k) \begin{bmatrix} c & s \\ -s & | \end{bmatrix},$$

~~analoges für~~

$$f \ r \ j = l:m$$

$$T_1 = A(j, i)$$

$$T_2 = A(j, k)$$

$$A(j, i) = C_1 - S_2$$

$$A(j, k) = S_1 + C_2$$

end

1j:j.s Roundoff Properties

~~Um die Rundungsfehler bei der Givens-Drehung zu verhindern, kann man die Givens-Drehungen so wählen, dass sie die Rundenfehler minimieren.~~

$$a = c(l + E), \quad \epsilon_c = O(u),$$

$$= s(l + E), \quad \epsilon_s = O(u).$$

~~Für das Ausarbeiten der Givens-Drehungen ist es vorteilhaft, die entsprechenden Matrix~~

$$fl[G(i, k)] = G(i, k) \Omega(A + E), \quad \|E\|_F \ll \|A\|_F$$

$$fl[AG(i, k)] = (A + E)G(i, k), \quad \|E\|_F \ll \|A\|_F$$

~~Die Rundungsfehler sind aufgrund der Vorschrift AP 13-3, Fig. 13.8, pass. 33, nicht berücksichtigt.~~

5.1.11 Representing Products of Givens Rotations

Since $\hat{Q} = G_1 \dots G_n$ is a product of Givens rotations, we have $\hat{Q}^T \hat{Q} = I_m$. This implies that $\hat{Q}^T \hat{Q} = I_m = \hat{Q} \hat{Q}^T$, so $\hat{Q} = \hat{Q}^T$. Thus $\hat{Q}^T \hat{Q} = I_m = \hat{Q} \hat{Q}^T$ implies $\hat{Q} = \hat{Q}^T$.

$$\hat{Q} = \begin{bmatrix} \cdot & \cdot & \cdot \\ \vdots & \ddots & \vdots \\ \cdot & \cdot & \cdot \end{bmatrix} \quad c^2 + s^2 = 1$$

level of technology

```

if C O
  p 1
elseif |s| < |c|
  p sign s2
else
  p 2 sign c
end
```

(59)

End of analysis of the first column of A . The first column of A has been reduced to zero.

```

if p 1
  C O S = 1
elseif |p| < 1
  s = 2C = sqrt(1 - c^2)
else
  C = 2p, s = sqrt(1 - p^2)
end
```

(60)

Next, the condition of \hat{Q} is checked. If it fails, a static address of \hat{Q} is set to zero. This is done if the residual term $\| \hat{Q}^T \hat{Q} - I_m \|$ is greater than a given limit (1E-10). Otherwise, the next column is processed.

$$\| \hat{Q}^T \hat{Q} - I_m \| = O(\mathbf{u}).$$

Tenemos de establecer un criterio adicional para evaluar la generatividad de los sistemas. Si el sistema es de tipo G , se considera que es generativo si existe una función ϕ que transforma los sistemas de tipo G en sistemas de tipo G . La función ϕ se define como sigue:

Si G es de tipo G_1 , entonces $\phi(G) = G_1$.

Si G es de tipo G_2 , entonces $\phi(G) = G_2$.

Si G es de tipo G_3 , entonces $\phi(G) = G_3$.

Si G es de tipo G_4 , entonces $\phi(G) = G_4$.

Si G es de tipo G_5 , entonces $\phi(G) = G_5$.

Si G es de tipo G_6 , entonces $\phi(G) = G_6$.

Si G es de tipo G_7 , entonces $\phi(G) = G_7$.

Si G es de tipo G_8 , entonces $\phi(G) = G_8$.

Si G es de tipo G_9 , entonces $\phi(G) = G_9$.

Si G es de tipo G_{10} , entonces $\phi(G) = G_{10}$.

Si G es de tipo G_{11} , entonces $\phi(G) = G_{11}$.

Si G es de tipo G_{12} , entonces $\phi(G) = G_{12}$.

Si G es de tipo G_{13} , entonces $\phi(G) = G_{13}$.

Si G es de tipo G_{14} , entonces $\phi(G) = G_{14}$.

Si G es de tipo G_{15} , entonces $\phi(G) = G_{15}$.

Si G es de tipo G_{16} , entonces $\phi(G) = G_{16}$.

Si G es de tipo G_{17} , entonces $\phi(G) = G_{17}$.

Si G es de tipo G_{18} , entonces $\phi(G) = G_{18}$.

Si G es de tipo G_{19} , entonces $\phi(G) = G_{19}$.

Si G es de tipo G_{20} , entonces $\phi(G) = G_{20}$.

Si G es de tipo G_{21} , entonces $\phi(G) = G_{21}$.

Si G es de tipo G_{22} , entonces $\phi(G) = G_{22}$.

Si G es de tipo G_{23} , entonces $\phi(G) = G_{23}$.

Si G es de tipo G_{24} , entonces $\phi(G) = G_{24}$.

Si G es de tipo G_{25} , entonces $\phi(G) = G_{25}$.

Si G es de tipo G_{26} , entonces $\phi(G) = G_{26}$.

Si G es de tipo G_{27} , entonces $\phi(G) = G_{27}$.

Si G es de tipo G_{28} , entonces $\phi(G) = G_{28}$.

Si G es de tipo G_{29} , entonces $\phi(G) = G_{29}$.

Si G es de tipo G_{30} , entonces $\phi(G) = G_{30}$.

Si G es de tipo G_{31} , entonces $\phi(G) = G_{31}$.

Si G es de tipo G_{32} , entonces $\phi(G) = G_{32}$.

Si G es de tipo G_{33} , entonces $\phi(G) = G_{33}$.

Si G es de tipo G_{34} , entonces $\phi(G) = G_{34}$.

Si G es de tipo G_{35} , entonces $\phi(G) = G_{35}$.

Si G es de tipo G_{36} , entonces $\phi(G) = G_{36}$.

Si G es de tipo G_{37} , entonces $\phi(G) = G_{37}$.

Si G es de tipo G_{38} , entonces $\phi(G) = G_{38}$.

Si G es de tipo G_{39} , entonces $\phi(G) = G_{39}$.

Si G es de tipo G_{40} , entonces $\phi(G) = G_{40}$.

Si G es de tipo G_{41} , entonces $\phi(G) = G_{41}$.

Si G es de tipo G_{42} , entonces $\phi(G) = G_{42}$.

Si G es de tipo G_{43} , entonces $\phi(G) = G_{43}$.

Si G es de tipo G_{44} , entonces $\phi(G) = G_{44}$.

Si G es de tipo G_{45} , entonces $\phi(G) = G_{45}$.

Si G es de tipo G_{46} , entonces $\phi(G) = G_{46}$.

Si G es de tipo G_{47} , entonces $\phi(G) = G_{47}$.

Si G es de tipo G_{48} , entonces $\phi(G) = G_{48}$.

Si G es de tipo G_{49} , entonces $\phi(G) = G_{49}$.

Si G es de tipo G_{50} , entonces $\phi(G) = G_{50}$.

Si G es de tipo G_{51} , entonces $\phi(G) = G_{51}$.

Si G es de tipo G_{52} , entonces $\phi(G) = G_{52}$.

Si G es de tipo G_{53} , entonces $\phi(G) = G_{53}$.

Si G es de tipo G_{54} , entonces $\phi(G) = G_{54}$.

Si G es de tipo G_{55} , entonces $\phi(G) = G_{55}$.

Si G es de tipo G_{56} , entonces $\phi(G) = G_{56}$.

Si G es de tipo G_{57} , entonces $\phi(G) = G_{57}$.

Si G es de tipo G_{58} , entonces $\phi(G) = G_{58}$.

Si G es de tipo G_{59} , entonces $\phi(G) = G_{59}$.

Si G es de tipo G_{60} , entonces $\phi(G) = G_{60}$.

Si G es de tipo G_{61} , entonces $\phi(G) = G_{61}$.

Si G es de tipo G_{62} , entonces $\phi(G) = G_{62}$.

Si G es de tipo G_{63} , entonces $\phi(G) = G_{63}$.

Si G es de tipo G_{64} , entonces $\phi(G) = G_{64}$.

Si G es de tipo G_{65} , entonces $\phi(G) = G_{65}$.

Si G es de tipo G_{66} , entonces $\phi(G) = G_{66}$.

Si G es de tipo G_{67} , entonces $\phi(G) = G_{67}$.

Si G es de tipo G_{68} , entonces $\phi(G) = G_{68}$.

Si G es de tipo G_{69} , entonces $\phi(G) = G_{69}$.

Si G es de tipo G_{70} , entonces $\phi(G) = G_{70}$.

Si G es de tipo G_{71} , entonces $\phi(G) = G_{71}$.

Si G es de tipo G_{72} , entonces $\phi(G) = G_{72}$.

Si G es de tipo G_{73} , entonces $\phi(G) = G_{73}$.

Si G es de tipo G_{74} , entonces $\phi(G) = G_{74}$.

Si G es de tipo G_{75} , entonces $\phi(G) = G_{75}$.

Si G es de tipo G_{76} , entonces $\phi(G) = G_{76}$.

Si G es de tipo G_{77} , entonces $\phi(G) = G_{77}$.

Si G es de tipo G_{78} , entonces $\phi(G) = G_{78}$.

Si G es de tipo G_{79} , entonces $\phi(G) = G_{79}$.

Si G es de tipo G_{80} , entonces $\phi(G) = G_{80}$.

Si G es de tipo G_{81} , entonces $\phi(G) = G_{81}$.

Si G es de tipo G_{82} , entonces $\phi(G) = G_{82}$.

Si G es de tipo G_{83} , entonces $\phi(G) = G_{83}$.

Si G es de tipo G_{84} , entonces $\phi(G) = G_{84}$.

Si G es de tipo G_{85} , entonces $\phi(G) = G_{85}$.

Si G es de tipo G_{86} , entonces $\phi(G) = G_{86}$.

Si G es de tipo G_{87} , entonces $\phi(G) = G_{87}$.

Si G es de tipo G_{88} , entonces $\phi(G) = G_{88}$.

Si G es de tipo G_{89} , entonces $\phi(G) = G_{89}$.

Si G es de tipo G_{90} , entonces $\phi(G) = G_{90}$.

Si G es de tipo G_{91} , entonces $\phi(G) = G_{91}$.

Si G es de tipo G_{92} , entonces $\phi(G) = G_{92}$.

Si G es de tipo G_{93} , entonces $\phi(G) = G_{93}$.

Si G es de tipo G_{94} , entonces $\phi(G) = G_{94}$.

Si G es de tipo G_{95} , entonces $\phi(G) = G_{95}$.

Si G es de tipo G_{96} , entonces $\phi(G) = G_{96}$.

Si G es de tipo G_{97} , entonces $\phi(G) = G_{97}$.

Si G es de tipo G_{98} , entonces $\phi(G) = G_{98}$.

Si G es de tipo G_{99} , entonces $\phi(G) = G_{99}$.

Si G es de tipo G_{100} , entonces $\phi(G) = G_{100}$.

$$A_k = \text{fl}(\hat{Q}_k A_{k-1} \hat{Z}_k), \quad k=1:p.$$

As related by Judd and Gehrlein in *Practical Techniques for Capital Budgeting*, an article in *Journal of Financial Research*, it was found that firms from Taiwan have set up 10 factories in California.

$$B = (Q_p \cdots Q_1)(A + E)(Z_1 \cdots Z_p), \quad (511)$$

Ver E P cula pad is ostituted by ad
p. In the v/s B Sanz, the plaintiff has to prove
the secondary cause of the accident. It is
(Ans 193 § 196).

5.1.13 The Complex Case

Metode analisis statistik dan eksperimen
dilakukan dengan teknik:
1. Klasifikasi (Klasifikasi dan korelasi)
2. Analisis varians
3. Analisis faktor
4. Analisis regresi dan korelasi
5. Analisis diskriminasi
6. Analisis multivari
7. Analisis faktor dan korelasi
8. Analisis faktor dan korelasi
9. Analisis faktor dan korelasi
10. Analisis faktor dan korelasi

$$\|x\|_2^2 = x^H x = |x_1|^2 + \cdots + |x_n|^2$$

Aopexofetastrosauayaxoleim

$$P = I_m - \beta vv^H, \quad \quad 0 \neq v \in \mathbb{C}^m,$$

The $\hat{f}_y = \hat{P}_x$ function is called the **generating function** of the random variable X .

$$x_1 = r e^{i\theta}$$

We lead

$$v = x \pm e^{i\theta} \|x\|_2 e_1, \quad e_1 = I_m(:, 1),$$

Telegatedriebahn **Teleskop** **Gesicht** **Stil** **data** **fix**

$$Q = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

the derivative Voltoptope = $c\theta$ ad $s\sin\theta$

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^H \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix} \quad (512)$$

the $v = u_1 + u_2$ and $v_1 = v_1 + iv_2$ gives $\{c_\alpha, s_\alpha\}, \{c_\beta, s_\beta\}, \{c_\theta, s_\theta\}$ is called coplanar series angles first is called

$$\begin{bmatrix} - & \cdot \\ \cdot & \cdot \end{bmatrix}^T \begin{bmatrix} \cdot \\ \cdot \end{bmatrix} = \begin{bmatrix}) \\ * \end{bmatrix}.$$

$$\begin{bmatrix} \cdot & \cdot \\ \cdot & \cdot \end{bmatrix}^T \begin{bmatrix} \cdot \\ \cdot \end{bmatrix} = \begin{bmatrix} \cdot \\ \cdot \end{bmatrix}.$$

$$\begin{bmatrix} \cdot & \cdot \\ \cdot & \cdot \end{bmatrix}^T \begin{bmatrix} \cdot \\ \cdot \end{bmatrix} = \begin{bmatrix} \cdot \\ \cdot \end{bmatrix}.$$

ad

Net $= \text{ad}_{GSI.Y+.c}$ If vector

$$-G = qx^b D28 = (E_2 c_+ + s_2 s_+) + i(c_\alpha s_\beta - c_\beta s_\alpha),$$

$c = c_\theta$, and $s = s_\theta e^{i\phi}$, then

$$\bar{s}u + cv = s_\theta e^{-i\phi} r_u e^{-i\alpha} + c_\theta r_v e^{-i\beta} = e^{-i\beta}(s_\theta r_u + c_\theta r_v) = 0$$

which is (512).

Problems

P5.1.1 18. x, ny can install $ctrl$ $SrRm$ $T2B$ H $H1ERn$ lx $x2n$ EQ Q $1H$ $(4BC(lkAx$ $BHnQ$ aC $nCHnT$. $b4nQ$ B Fy .

P5.1.2 ro.8Pge.8.8.8.e...8. .8..86 .. .det(I - xy^T) =) Rx^Ty \(\Rightarrow\) AEJa = Q = E00>0= 1- > OE= = E + n

P5.1.3 (d) $-x, c)^n$ x, y ER $CH24$ $Tn3$ $(xH$ $)R2AH$ $H1EQ$ $yurCHa(b4B$. n $Q2A$). $x(nT)$ Q . $(nCH_y = QT_x$, $S a$ ax lx \bullet x \bullet $Q2A$. s), $\|_p^p$ x, ny , $ITxNSt$ at $SrRm$ $T2A$ $H1(xHc4Q1)$ $T2A$. mH . $nHnQ$. LC $ga(b1B$. H) $(n$ $(0.1$ $Q.4C$ nQF $x = y$.

P5.1.4 P. .8...0.8 .. .8P .8.8.8. e.8.e.. .8..0.. .80. 8N. .8t.8b
...8.. .8.e..8.N

P5.1.5 e..8.8... Q = I - YTYTQ(E)($1H$ LCB EY Enm HxT EJ xj $Q41Ax$ nQ) $1H$ 2 $+C$ I Cn $QF = QP$ $LCAxP = I - 2W^T/vTvQH$ ($= AC(lkAx)b^TnQen$ AQ aH). A $Bxk2Ax$ Q nGA $Q^+ = I - Y_T + Y^T$ 1^n $+$ Y^T ERm (j^+I) HxT E $R(j^+I)x(j^+I)$ $Q41Ax$ nQ) $1Ix$ $-CTQ$ $nCATQx$ $ACQx$ CA compact WY r pr sentation. hcj hJ I $cSohm$, n , len $7MfdP$

P5.1.6 .e..8.8 Q = I_m - $Y1T1Y1HxQ = I_m - Y2T2Y^T$ $^{(m)}(p)p^nd + 1_{n,n}^{n,n}$ $Y1ER$ x_i , $Y2ERm$ x^2T_1 $EWtxr1HxT$ $Ew2x2$ 1_b $nChT1HxP$ xj $41A$ nQ) $1H$ bm $(ICL$ $n(aT1nA$ YER $xrHx41Ax$ nQH) $1xT$ Ew $xrLThQ = 9pM9$. $(nCHQ2Q1 = I_m - YTYT$.

P5.1.7 N..8 .8.e..8.8.8...8. .811.8... .9 5.. . .roe..8e. valj vi
 $^{n(n}n^{n,n}n^{n,n}n^{n,n})$ 1_d $^{n,p}_T$ $^{n,n}_T$ $^{n,n}_T$ $^{(R^p,+)(p,+)}_T$ $^{-(+p)}_T$ $^{(n,p)}_T$ $^{(n,p)}_T$ $^{(n,p)}_T$ $($ $1t$ $J/ -$ $)$ Y QB $AanT2AK$ $xAkAA)nAxT) I 14 $k(AAxxxA)AAx)11$ $Bn41$ $rCAW$$

P5.1.8 ..86 ot.eot ..> .6ot.fdr T= 1 Xi aCQ= (I + S)(I - 'df1Q(FHC(12 126 CB
 T2HQ" Qa2=BaCEayle tr nsf or LB'd)lrf p,feprn 'QINf JeMBQH2Baa(EaGn)Qx
 T>BE[ka-h Q ECA(b\B)a 2

P5.1.9 .2.8 .o PER^{m × m}, R^m, # pTP 1 I_m P = E < O O = 3 ad + Ad O = y > aa < r P + 2EQ aCB QaE2H + e) 2x aCH p - uvT P = E r P = U E V T QaCB + (P.

P5.1.10 . .8.. A ER x² MxMnEChA0xQaQ0.T-aCB a1(BaHaQ6(A slrqpfJ, fJc slAcAf phOcfSf.lA? ulpr 8Sf.fJclohnS Alpu,

P5.1.11 868 1-8 ..ot.. P .m98.. m.otg.2..

P5.1.12 (Fast Givens Transformations) hAAAI

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \mathbf{z}\mathbf{x} \quad \mathbf{D} = \begin{bmatrix} \mathbf{d}_1 & \mathbf{I} \\ \cdot & \mathbf{d}_2 \end{bmatrix}$$

LTaCl₁Hx d2 1.QaQW+CL ,L a(a(blam

$$M_1 = \begin{bmatrix} 1 & \cdot & \cdot \\ \vdots & \ddots & \vdots \end{bmatrix}$$

(aC2Q_y = MxH_y x € = M D! i, aCA_y 2a = g0+0aa y=j fn=2d! 1EB1=2Bx ,1

$$M_2 = \begin{bmatrix} 0 & 1 \\ -2 & 0 \end{bmatrix}$$

ZX+C(LaCH₂QaCIM D! 1½ -3 e(E 1M D! 2½ -3 eF 6X1=BxER^m m

R^m(**I**)ⁿ**E****R**ⁿ_xⁿ(,¹(d)^p_nd-₁(**R**^m(^p,^j**R**^m(-^l,^j)^p)[/]+₊/**R**^m_n+₊**G**₋ⁿ₋^m_j(+₊)_ienm 8Sf**M** i < . . . m ,

= == rr = -V= E Ra^{n-p} Rⁿ(dy) = ! x 2x € = ! TDMi aGm) == = * x = u = r .. 1€ 12: 21D 12 6X (b HFaGxXa(valxB aCHQ = D12Mb = . - .

r4 Rd8=dt : { GAtet 5,77 abnSp ejptgO CAetu Pt 1rdd ira65 SIAM J. Numer. Anal. 25, Mf Ne
 5khkJIaSoalnrk.Cn IT ,fMty. c4 hI Ciar R.San1 5aAIaran,fSh (lma.r IB sloraJIAmrlnrvl 1, fSlrSIAM J. Sci. Stat. Comput. 10, eNel.
)k(oIASnMN. cPlnM,,fShB.JaslorajIAmrh.Jlm 91dmf JAl1AC.. u1 5aAIaran.C Sln3SIAM J. Sci. Stat. Comput. 13, INEF INu.
 k,n Cn)ksk9Sr.JlEMtey. c4 91Srr Ma5aAIaranf eTl1h JlilnCs P,fISIAM J. Matr. Anal. Appl. 16, Mf MfM
 kk1 ICShB k ll83hpoSnf,ne II fS18n maiaSjnGnm .ek.aa TN11u c4.,1A,Sni sloraJIAmCnryl uC.Mrf53rM.ar3M T ns. Math. Sof w. 323Mf6MI rkjhCm3Gra1khCACIN1ly. c4 ala ln huAsa.fS9sl.a 5aOaf llr3TNA 33, 6e n.
 iMANIIC.Mlnra nClar6 aICssC.iS.anrA J ael arl1a rooAa.Mh1.Sef an8Mf.JaMI .1IAoC.SlnCnhaAIaran8Cif
 ikuk hf a8CIMty. ckjAr.lnl IS.Cshf II,ialB(sCna5l.Cf Mlnr3Mer. Math. 25, MBME .s 9Smak11 as8kM,J,nCnm H(E2.0+ 22Di=E= Xf p q(=)E*= = E0< (= n< e (= 0a E(O= AGME Ns. Math. Sof w. 28, N luffey.
 S.SrAlrrSofh,iIai,f a IlfC.Mfhi,nrv1 1,fSlnfl,JSaaJSiAaIv1 1,n.a3raa,
 9klCn TMty crSnlaas E914h Mf, fSln 91m4s1ISf J lrSIAM J. Sci. Comput. 19 uNuf E6
 .1 iS.anr.ICnrv1 1,SlnrTraa(e.:..MfCIA- H,r IIahm1 squar -r of e Givens tr nsfor-
 mations. T5.CsAJ., Cr6oCIhlf Ixrf II mSnCks1A .a. mISnJh v1 1,fSlnIBiS.anr
 ICnrv1u,SlnykJalaCraaICs 8C.b B If ..siS;fSlnr,n ca,II,niambaa,
 Pkian sauCimfE da1 hJCIa1Aof.fSlnro iS.hnr Qnrv1 1,fSlnr 8SfJl,fhJoCIa
 5l,r3TJ. Inst. Math. Appl. 12, ENt EEU.
)k.wCnllCn TMty cianal CsStaInsel.,soar uST 4sll Mf J Gm14AM.,Sln3TJ. kJaSr3
 ,nS IrSBPM.JSiGn841 Is
 hksC11CIAM1M6. c4 alfa ln PlmMSlnr fl fJa iM.anr 5AGnln3T Inst. Math
 Applic. 13, NMfNMf.
 1kskuSs3SlnTMtH chlu5a.anf 4m,nar Sna1 aIS.CSraCIAiaod eMf the State of the
 Art in Numerical Analysis. k4wsVtlor TanweCma1S(larr3a8 IIIa3 Mf eE.
 4k4k4nmCnusk(CI3Mf6y. c f(sCna 5T f&fSlnrM. hCSni3SIAM J. Matr. Anal.
 Applic. 15, r· r ·

$$\begin{matrix} v_{,c}, v_{,} \\ w^{*,c} v_{,c} \end{matrix} = \begin{matrix} v_{,c} \\ v_{,} \end{matrix} \cdot \begin{matrix} \cdot \\ L \end{matrix} = \begin{matrix} = \\ = \end{matrix} \begin{matrix} 4 \\ 4 \end{matrix} \begin{matrix} * \\ 4 \end{matrix} \begin{matrix} c \\ \overline{c} \end{matrix} \begin{matrix} c \\ 4 \end{matrix} \begin{matrix} \overline{c} \\ \overline{c} \end{matrix} \begin{matrix} \therefore \\ \therefore \end{matrix} \begin{matrix} * \\ t6 \end{matrix}$$

5.2 The QR Factorization

A rectangular matrix $E_{m,n}$ can be factored into an orthogonal matrix Q and a rectangular matrix R .

This factorization is called the QR factorization, and it is similar to the LU factorization. It is used for solving linear systems, least squares problems, and eigenvalue problems.

5.2.1 Existence and Properties

What is the profile of the factorization?

Theorem 5.2.1 (QR Factorization). If $A \in \mathbb{R}^{m,n}$, then there exists an orthogonal matrix $Q \in \mathbb{R}^{m,m}$ and an upper triangular matrix $R \in \mathbb{R}^{m,n}$ so that $A = QR$.

Pr 6 We use induction. Suppose $n = 1$ and that Q is a Householder matrix so that if $R = QTA$, then $R(2m) = 0$. It follows that $A = QR$ is a QR factorization of A . For general n we partition A

$$A = [A_1 \ A_2 \ \dots \ A_n]$$

where $V = A(:, n)$. By induction, there exists an orthogonal $Q \in \mathbb{R}^{m \times m}$ so that $R_1 = Q A_1$ is upper triangular. Set $w = QV$ and let $w(nm) = QR_2$ be the QR factorization of $w(nm)$. If

$$Q = Q_1 \begin{bmatrix} I_{n-1} & 0 \\ 0 & Q_2 \end{bmatrix},$$

then

$$A = Q \left[R_1 \mid \begin{array}{c|c} w(1:n-1) \\ \hline R_2 \end{array} \right]$$

is a QR factorization of A .

The columns of Q have an important connection to the range of A and its orthogonal complement.

Theorem 5.2.2. If $A = QR$ is a QR factorization of all column rank $A \in \mathbb{R}^{m \times n}$ and

$$A = [a_1 \ a_2 \ \dots \ a_n],$$

$$Q = [q_1 \ q_2 \ \dots \ q_m]$$

or column partitions, then for $k = 1:n$

$$\text{span}\{a_1, \dots, a_k\} = \text{span}\{q_1, \dots, q_k\} \quad (521)$$

and $\text{rk}_k = 0$. Moreover, if $Q_1 = Q(1:m, 1:n)$, $Q_2 = Q(1:m, n+1:m)$, and $R_1 = R(1:n, 1:n)$, then

$$\text{ran}(A) = \text{ran}(Q_1),$$

$$\text{ran}(A) = \text{ran}(Q_2),$$

and

$$A = Q_1 R_1. \quad (522)$$

Pr 6 Comparing the k th columns in $A = QR$ we conclude that

$$a_k = \sum_{i=1}^k \text{rk}_i Q \in \text{span}(Q_1, \dots, Q_k), \quad (523)$$

and so

$$\text{span}\{a_1, \dots, a_k\} \subseteq \text{span}\{q_1, \dots, q_k\}.$$

If $\text{rk}_k = 0$ then a_1, \dots, a_k are dependent. Thus, R cannot have a zero on its diagonal and so $\text{span}\{a_1, \dots, a_k\}$ has dimension k . Coupled with (523) this establishes (521). To prove (522) we note that

$$A = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = Q_1 R_1. \quad \square$$

Tentatives de QR et de la méthode de Gram-Schmidt pour la résolution des systèmes linéaires

Theorem 5.2.3 (Thin QR Factorization). Suppose $A \in \mathbb{R}^{m \times n}$ has full column rank. Then the thin QR factorization

$$A = QR_1$$

is unique where $Q_1 \in \mathbb{R}^{m \times n}$ has orthonormal columns and R_1 is upper triangular with positive diagonal entries. Moreover, $R_1 = G^T$ where G is the lower triangular Cholesky factor of $A^T A$.

See also § 5.2.2 for a supply chain example.

Lower and upper triangular matrices are used in the solution of linear systems for solving triangular systems. See a detailed discussion of iterative refinement methods in § 5.2.3.

$$K_2(A) = \frac{\sigma_{\min}(A)}{\sigma_{\max}(A)}. \quad (5.2.1)$$

Methods of analysis of the singular value decomposition are discussed in § 5.2.3.

5.2.2 Householder QR

We will focus on the Householder QR factorization. This section follows the treatment by Trefethen and Bau in § 5.2.2.

$$H_2 H_1 A = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & x & x & x \end{bmatrix}.$$

Contracting this with the identity matrix I_4

$$\tilde{H}_3 \begin{bmatrix} x \\ x \\ x \\ x \end{bmatrix} = \begin{bmatrix} x \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

If $B = \text{diag}(F)$, then

$$H_3 H_2 H_1 A = \begin{bmatrix} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & 0 & * \\ 0 & 0 & 0 & 0 & * \end{bmatrix}.$$

After shifting to the right, $H_3 H_2 H_1 \dots H_1 A = R$ (Racky).
 Using $Q = H_1 \dots H_n$ with $A = Q^T R$, we have

$Q = R^{-1} S^{-1} U^{-1} \dots U^{-1} S^{-1} T^{-1} H_1 \dots H_n$.
 Since $U = \text{diag}(F)$, we have

$$f \ r \ j = n$$

$[v_j] = \text{base}(A(j:mj))$

$$A(j:mj:n) = (I - wT)A(j:mj:n)$$

if $j < m$

$$A(j + 1:mj) = v(2m - j + 1)$$

end

end

This algorithm has $\frac{2}{3}m \cdot n^3$ flops
 for $m > n$.

$$V^j = [Q, \dots, Q^1, V_1^j, \dots, V_{n-j}^j]^T$$

is left to develop the option

$$A = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} \\ V_1^1 & r_{22} & r_{23} & r_{24} & r_{25} \\ V_1^2 & V_2^2 & r_{33} & T_{34} & r_{35} \\ V_1^3 & V_2^3 & V_3^3 & T_{44} & r_{45} \\ V_1^4 & V_2^4 & V_3^4 & V_4^4 & r_{55} \\ V_1^5 & V_2^5 & V_3^5 & V_4^5 & V_5^5 \end{bmatrix}.$$

If matrix $Q = H_1 \dots H_n$ is selected according to the T in $T = Q^T R$, then $R = Q^T A$ and $R = A^T Q$. The QR factorization is called the Gram-Schmidt process.

$$r_{j,j} = \frac{\sqrt{2}}{1 + \|A(j+1:mj)\|_F^2}$$

*Venio late optet peti gantix istex fraeb
A Nieseta*

5.2.3 _ Block Householder QR Factorization

Археология и архитектура в контексте инновационных технологий. Материалы международной научно-практической конференции (Москва, 15–16 марта 2018 г.)

$$A(:,412) = (J - W T) A(:,412)$$

Net vegetatian in Aqina 21 July testasfr
tobacco 72 under hothouse in B-WX
is full of staled grain.

$$Q \vdash I_m A = l; K \vdash o$$

while A < n

$$T \in \mathbb{R}^{(n+r-1)n}; k = k_1$$

Sezione 21, *l'eteregrado* (AmAT),
sezione di *ordenes* H,...,H.

Using a tag toolbox

$$A(Ar, T+1p) = (1 - W_{K\bar{K}})TA(Ar, T+1p)$$

$$\bar{Q}_{Am} = Q_{Am}(J - W_{KJ})$$

$$A = T_{+1}$$

end

**Text procedure field developed for H. ins-
taevisA-1 lysowka and HCV deepen
parasitism** **Hepatitis C virus infection**

$$A = [A_1 | \cdots | A_N], \quad N = \dim A$$

Technique 5.2 extends the idea of Gram-Schmidt orthogonalization to handle rectangular matrices. It is based on the same idea of Gram-Schmidt orthogonalization, but it uses Householder reflections to reduce the matrix to an upper triangular form.

5.2.4 Block Recursive QR

An efficient algorithm for computing the QR factorization of a matrix $A \in \mathbb{R}^{m \times n}$ is the block recursive QR factorization.

$$\begin{bmatrix} A_1 & A_2 \end{bmatrix} = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}.$$

where A_1 for n_1 , A_2 for n_2 , $n_1 + n_2 = n$, $A_1 Q_1 R_{11}^{-1}$ and $A_2 Q_2 R_{22}^{-1}$. For $n_1 < n$, $Q_1 = A_1 - R_{12}^T Q_2 A_2$ and $R_{22} = A_2 - Q_2 R_{12}^T Q_1$.

and recursive step: $Q_1 = A_1 - R_{12}^T Q_2 A_2$ and $R_{22} = A_2 - Q_2 R_{12}^T Q_1$

function $[Q, R] = \text{BlockQR}(A, n, n)$

if $n = n$

else

$n_1 = \text{floor}(n/2)$

$[Q_1, R_1] = \text{BlockQR}(A(:, 1:n_1), n_1, n)$

$R_2 = Q_1 A(:, n_1 + 1:n)$

$A(:, n_1 + 1:n) = A(:, n_1 + 1:n) - Q_1 R_2$

$[Q_2, R_2] = \text{BlockQR}(A(:, n_1 + 1:n), n - n_1, n)$

$Q = [Q_1 | Q_2], R = \begin{bmatrix} R_1 & R_{12} \\ 0 & R_{22} \end{bmatrix}$

end

end

This block recursive QR factorization has a time complexity of $O(n^2 m)$ and is very efficient for rectangular matrices with $m \gg n$.

5.2.5 Givens QR Methods

Givens rank disclosed the Orthogonalization by 3x3
Householder

$$\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \xrightarrow{(3,4)} \begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{(2,3)} \begin{bmatrix} x & x & x \\ x & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{(1,2)} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix}$$

$$\begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{(3,4)} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & 0 & x \end{bmatrix} \xrightarrow{(2,3)} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & x \end{bmatrix} \xrightarrow{(3,4)} R.$$

With the 2nd standardizing fractions f, dets
tect P's example to the Q_tA_tR supermatrix
Q_tG₁...G_t adjusted need to follow an veae
det f_t f_{t+1} 5.2.4.181R9I.p25rp1 A_tE_n xⁿM_n n_n
converting Q_tA_tR supermatrix to Q_tA_tR

$$f_{rj} = \ln \frac{f_r}{f_{ri=m:-l_j+1}}$$

[cs] = givens(A(i-1,j), A(i,j))

$$A(i-l_i, j:n) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T A(i-l_i, j:n)$$

end

end

This algorithm is 3^m - 12³ for N=4 odd, it is even
to prove those of the condition for a single
catch matrix A(i,j) can work in the case of even N
With this algorithm, it can be seen that
decomposition is done in a row wise manner
it will be a leap forward to implement this in a column
wise loop, we will then follow in the next section

[cs] = givens(A(j,j), A(i,j))

$$A([j:i], j:n) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T A([j:i], j:n)$$

and here with the Orthogonalization is done by
lower rightmost index column,

$$\begin{bmatrix} x & x \\ x & x \\ 25 & x \\ 146 & x \end{bmatrix},$$

the problem

```

f r i = 2m
f r j = 1:i - 1
[c, S] J givens(A(j, j), A(i, j))
A([j i], j:n) n [I]: R_A([j i], j:n)
end
end

```

it takes by eye

$$\begin{bmatrix} \times & \times & \times \\ 1 & \times & \times \\ 2 & 3 & \times \\ 4 & 5 & 6 \end{bmatrix}.$$

5.2.6 Hessenberg QR via Givens

A sample of Givens rotations that reduce a matrix to upper triangular form. The first two rows are zeroed out. The third row has non-zero entries at indices 3 and 4. An example of a sequence of 6 rotations to reduce a general matrix to upper triangular form.

$$G(23) f G(1,2) f T_A = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}.$$

Next up: \$G(34) f G(23) f G(1,2) f A\$

$$G(34) f G(23) f G(1,2) f A = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}.$$

Concluding with the final algorithm

If \$T_A = E_1 \dots E_n\$ is the left factor, then \$PQ = G_1 \dots G_n\$ is a product of Givens rotations, latter in \$G = G(j, j+1, \theta_j)\$.

for $j = 1:n - 1$

$$[\mathbf{C}, \mathbf{S}] = \text{givens}(A(j, j), A(j + 1, j))$$

$$A(j:j + 1, j:n) = \begin{bmatrix} \mathbf{C} & \mathbf{S} \\ -\mathbf{S} & \mathbf{C} \end{bmatrix}^T A(j:j + 1, j:n)$$

end

Trigonometrische QR-faktorisierung

5.2.7 Classical Gram-Schmidt Algorithm

Werkstckskripten und Latacadas beschreibt in R-faktorisierung $= Q_1 R_1$ die Vektoren \mathbf{b} und \mathbf{a}_k der QR-faktorisierung

$$q_k = \left(a_k - \sum_{i=1}^{k-1} r_{ik} q_i \right) / r_{kk}.$$

Toward a linear algebra introduction

$$z_k = \mathbf{b} \cdot \frac{\mathbf{a}_k}{\|\mathbf{a}_k\|}$$

Vektoren $z_k \in \text{span}\{\mathbf{a}_1, \dots, \mathbf{a}_{k-1}\}$

$$r_{ik} = q_i^T a_k, \quad i = 1:k-1.$$

Trägt die Classical Gram-Schmidt (CS) mit rückwärtsprojektion $= Q_1 R_1$.

$$R(1, 1) = \|A(:, 1)\|_2$$

$$Q(:, 1) = A(:, 1)/R(1, 1)$$

for $k = 2:n$

$$R(1:k - 1, k) = Q(1:m, 1:k - 1)^T A(1:m, k)$$

$$z = A(1:m, k) - Q(1:m, 1:k - 1) \cdot R(1:k - 1, k)$$

$$R(k, k) = \|z\|_2$$

$$Q(1:m, k) = z/R(k, k)$$

end

Hilfestellung durch abgeleitete

5.2.8 Modified Gram-Schmidt Algorithm

Umstyling CS durch von Gram-Schmidt her leis
tigere Vektoren \mathbf{b} und \mathbf{a}_k zu erhalten. Es ist eine
einfache Modifikation des Gram-Schmidt-Algorithmus.
Von der ursprünglichen Form des Gram-Schmidt-Algorithmus
unterscheidet sie sich darin, dass die Vektoren \mathbf{b} und \mathbf{a}_k nicht
nur im gleichen Schritt aktualisiert werden, sondern in einem

and the QR factorization is given by

$$[0 \mid A^{(k)}] = A - \sum_{i=k}^{k-1} q_i r_i^T = \sum_{i=k}^n q_i r_i^T.$$

If $\|q_k\|_2 = 0$

$$A^{(k)} = [v \ I \ B \ I]$$

then $R^{(k)} = B$ and we would append v to R . We could copy selected rows of A to R .

for $k = 1:n$
 $R(k) = \|A(1:m, k)\|_2$
 $Q(1:m, k) = A(1:m, R(k))$
 $\text{for } j = k+1:n$
 $R(j) = Q(1:m, R(k))$
 $A(1:m, j) = A(1:m, j) - Q(1:m, R(j))$

end
 end

This requires $O(mn^2)$ for its n iterations which is about $1/2$ of the cost of Gaussian elimination.

5.2.9 Work and Accuracy

For standard orthogonal basis functions the cost of applying S^{-1} is $O(n^3)$ instead of $O(n^2)$. This is largely due to the first column of S^{-1} being the inverse of a lower triangular matrix. This is a general result for orthogonal basis functions (Sorensen 1967).

Various methods have been proposed to reduce the cost of applying S^{-1} .

To understand why the QR method is more accurate than the LU method we need to look at the error analysis.

Violated orthogonal factor model matrix
 $\hat{A} = \hat{Q}\hat{R}$, $\hat{Q}^T\hat{Q} = I$ and $\hat{R}^T\hat{R} = R^T R$
 singular value decomposition (SVD) and related
 generalized division

5.2.10 A Note on Complex Householder QR

Complex Householder QR (§5.1.13) based on the QR
 decomposition $A \in \mathbb{C}^{m \times n}$. Algorithm 5.1 works

for j :
 Complex Householder matrix Q_j is updated
 through iterations

$$A = Q_j A$$

end

Iteration later decomposes matrix $\in \mathbb{C}^{m \times n}$ and
 via $A = QR$, $V = Q = [v_1 \dots v_n]$ fully reduces A to
 the required form

Problems

P5.2.1 1... P8e..8 t7 ...8... 8... ..4 ...4.e ...m. .6..4 AE Rmn
 $CP = (LBx1 \text{ } mQx1)$ a n Gf. Ma n 030 = .

P5.2.2 .e..8... AE Rmn2x Bn, B nB=xCmV BBEmnQV (nQxBx1 xB2BEQrCExBx
 $(F_rCE) L Qn, 62X + GC2QR E RmnQ41mkrEQ, V411FCBL = 4'4 (+A= 3nE (a n y: A
 MmQ, = 3= 3E x f: a= 10 = • ⇒ E Rmn2A" = (LBx1Q2/4-21 Enmn < (nC2A = QL
 P, bQ1 rCB2/2Q = QH2 E lamx4Bx a(b1Q1 rCB+ FHxQHQ) v>$

P5.2.3 1... N...4. t7 1 .8.. ..848... 8... .. .8... 4..8.e...8...
 m... .. m4..... ..8...4... 8.... (m, 1), (m - 1, 1), (m, 2), (m - 2, 1), (m - 1, 2), (m, e=e
), A

P5.2.4 1... N...4. t7 1..8.. ..84..8... 8... .. .8... 4.. fe4... .06..4 AQ=
 $f d f 2x nEQxQ2 . l, B rCmC4xQn/2 PxQ2/2E2x 4BEQ2V FA2B rxBx Q$
 $e(1:n-1), a(1:n), f(1:n-1), .f(-), 1.0A i, , -0(), -(), f(), f(-1, 0) .. (1, B)),$
 $20), (..(-0), (T.$

P5.2.5 .e..8... LE Rmn LQrGm ≥ n, f (B)), .)7.(B , (B 2n (f. (.,)),),,,
 $H_i.mEBx2v, B , Bx r(x=rBEQ2B=(LBx1Q2/4="xL1ERmn.($

$$H_1 \dots H_L = \begin{bmatrix} : & 1 \end{bmatrix}.$$

(- I t) [H S F] V[H]] [M] W[H2. (rCm

$$H_2 \begin{bmatrix} x & 0 & 0 \\ x & x & 0 \\ x & x & x \\ x & x & 0 \\ x & x & 0 \\ x & x & 0 \end{bmatrix} = \begin{bmatrix} x & 0 & 0 \\ x & x & 0 \\ x & x & x \\ x & 0 & 0 \\ x & 0 & 0 \\ x & 0 & 0 \end{bmatrix}$$

LQrGB 1(BEn1 nmrxL) 2x u2B1nFn=(VB[

P5.2.6 .e..8... AE Rmn2x D = xQH6, ..., d) E Rmn". +C(LC(L n(a(vnE4an2v (x(C(I)2-
 " .4C1C2n

$$Q^T A - D Q^T = R$$

Q41Bx mQ2/4-21 En(vnL(FE1 2/4n BhaQval rQn4n2/ mBEQ2t+ bHT14-21V

P5.2.7 ..8 6 .86 .8 .8 .e... t7 ..8.. ..881.. ..8.e..

Az Av...A2Ai

1..H B..... C. c - Ai,...,Apn(BrCBEn=42 nC2n BjC Q=-,2BiH)nE
1) rCB 7 EynE (= n

$$Qf A = Qf AaQ2Qf A2QiQf Ai$$

2x xBnB2B (BrC(1)21 Q=(nC2nQf (A;Q, 1} Q41B EnQ2=2E. (Qo= 1.)

P5.2.8 RN.8 A ERmnQ)4BEQa21B-4Q2B)n(rCB x=nBII I(4=BC(1xBx+
211Qbx(

$$\begin{bmatrix} & \\ & \end{bmatrix}$$

¶ On QnCf?B xB(22 nEQnBxTf1C2nCQ=n2nB)n QnEB2 BECBx=n=ma1 E2C
2BrC(xQa(21BrBxm

P5.2.9 7..... .. 88.8.... .. 1.8... P 088N.58... R SA+Axtcm,x+n o lsx+n:

P5.2.10 P86 .. 8.... .. e.... .. 8 ..bt7 ..8.. ..8.8...2.. 8e.... .. R

P5.2.11 8....8. .8.b. ...8.81.. N.... t7 1.8.. ..8.. 6....8.. 81R SA
nlhncietS.ccc-r MEm

P5.2.12 ..861A ERnxn2x l6 xXc2

$$|\det(A)| = \| ai \|_2 \cdots \| a_n \|_2$$

IQnEM=B tCF2n(EQ22)m

P5.2.13 .e..8 .. A ERmnLQnC z 1 e= =o= E.E. = = • =QER(n+1)x(n+1) LQnGCB
E[Bx1 nC26E OEXQ2 =a21nQIB(FA IQn.Fa ER QaC(-B)k(BE1rCB) - α²ATA
CA2,C(B=11 F2n(EF2nQ)m

P5.2.14 .e..8 .. A ERmn)2-(1= n(l11 EQnC2nB+eC(LC(L FAn)T2B)=nE2=1k2hT)=
6+mh)3Xa2,B 4-k n(a24n 11 ERmxm2x 2xQ2)2 DER "xmlQnQ=QnQ23T2)21
B)nkB<(nC2nMIA=r Q41Bx nEQ241E2x MJ/Tz D. +B12BM 2x r n' l=t+ han(E-m

P5.2.15 (Par lled Givens QR) hxAAcAcA ER932x nC2nLB (EQ2B 2)QB=t+= (nC2nCB
=,,xQ2)21 B)rEQ2x xBx(Bx (2EB(4E-BFnrB) ®nQBnB1=A l=(L-E

	+nB1	0nEQB=BEfBx
v 0	6eX	
v z 3	6yX	
v z u	6vX 6e3X	
v z +	6zX 6Ye3X	
v z +	6+jX 6Me3X6euX	
v 5	6+jX 65e3X 6Yeux	
v z M	6eX 6E e3X6euX	
v z Y	6dX 6+e3X 65euX	
v .	6ue3X 6+euX	
v z -)	6-euX	

l=4B nC2n2E(r2nT) Q 1=2B (i - M 7MnAAct tcl e +etISfcntI. (i,j). Utv,l8A tJ et tJc
IltetSlnOA1Metj86tJen. iM.crtS+cAtcASnls.cmSAJ1Asf1A pl8Aenm+e tJcl cvl coc
l+AxtcmMAel escsII cf-e-AscfnxpStf+AtcA u7 tJSle7E 761r 7edm7I 7dteSln:
cJlccAcAeIetA lcAlAlpA lclAcctJctJIccxAmetcRf tleAsetPl+ tJMAe+A,cl tJc
-Qar =Oc enm AJ18 tJd85 r .tlIMtetMlm oc.l+AxtanSrnD RfxnQB=rB1I(L
21 (FrC(=B nQB L41B1Q21Bn cnlnl cpseAASnpfetMlnAA

Notes and References for §5.2

1. 812... P82...8....N. ..8.. .8 8...re.... ..8... 6 ..8.8... ..or

QC trVgAAtln 9,f 7 bDrison VodVAo)Fosia rao4ngulshp uersis5 J. ACM 5, EETf E6NN
 kJcAp,tS.esmcf.S8pc8lpaclnxf M.,
 (. 9xrMniepm i.s. ils,0MtNlNlSc,p lcIt h6xepohl,,tMlmo. slxrcJl,mcp penrvpue,
 tSln7INumer. Math 7, NuttfN,
 i.2Nil,xo 7Mnd.caxucpM.P,ctJlmvp hls.M.ilM.cep lc116epcr(ploscur7INumer. Math
 7, N lufMuN
 kJc o1S. pcrpcpc.arp iM.cr p5 M.,xmh-
 u iS.j.r 7 tnydluAxtetMl.lr S,enc., Mf e51tetSl.r QervpuS.i e ic.cpe, PetpMif
 Qsenix,epi pu7SIAM J. Appl. Math. 6 NuflN
 PNj ntscue 7MENcRpp1ne,rSlr p5 .cluAlrMf MlariS.cr QervpuetMl.r 7Lin. Alg
 Applic. 10, MfMtl,
 kJcpcepculmM.,etMhp tJp5 retlpMnetSlhetue3cMtlpc ettpf M8Jc. mcesS8MtJena
 mc,Mc..khcc-nN 6NcptJcsrn8Jc. luoSncn8MtJ tJhStSlertSuetMmMl.E, nN
 tpmsL,s p5 retlpSnetSL. oh x2jmemmpijar mj „Mhn. Mrrx3r,
 l... lrcp 7MudN5ena,nmaxshA.h)es.xsetSlnrMi.PetpSfc.luAlrStMl8M Jllxun
 - ntcp.Jeicr 7In. Alg Applic. 74, 6f fM,
 kJc ccJ,SlplrfJ , „.N fIp8JcnT SIAhptxpoefMl M.thp3rf NeMp3rx,Mrl ettJc
 pc,tSniJ,fc Sf emn epcolxnmcm tJc.lnmStMl T tSicr JcpcsetS.c Jeisit 7cc,
 iN ulhtc8ept7MEdNI (cptxpoetMl. 9lxpmfJq5.e.fIpSn,fSlr ePef pS7SIAM J. Numer.
 Anal. 14, nlt,nMyN
 s. Je 7MEdNc4)luAlncnf 8SrsqctxpoetS4ne,rMrJtJq5 .cluAl rStSlrs7AM J. Matr
 Anal. Applic. 4 MfMNE MN
 isuj htc8epf7MEd dc1 tJc(cptxpoefSlnl,)Jl,cra.7emp5 e.tlpSntSlr 7SIAM J. Matr
 Anal. Applic. 14, MfMNE6nN
 4.9eppe, n7 tcdN(cpt,poetMl, mr vptJcic.cpesMnp5 e tlpMfSln7Lin. Alg. Applic.
 207, NnMMN
 1N i.hx. 7MndN c1 Scpf xpoetS9l, mr vp tJcp5 e.fIpSntSlr 7Lin. Alg. Applic. 215
 trf MMN,
 O.r u)J,i emn)N)S,Sic7MldNluAlncnt8Sr8cpt,poetMl, rcr vp fJq5 .tIMn
 tSlr 7Numer. Math 88 EMf6nN
 lpi,nM-estSlr tJ3luAxt,tSl. rl tJettJ3cntpsSsir mcAhmtS.xlrx,. l. tJj htpSjS.T Sr
 mMr,rrfri
 kN.Nscul. em..)N hpcnrc.7(y6Nc4 altc l. tJc)luAxtetSl. lr e. lpfJl.pue, 9ISrvp
 tJca,s, hA,c lr e PetpS7I Mathematical Pr gr mmimg 2. + ' L+ 1f J
 j n. = n = En + 1ma x - Olf = x (En + nO = (2a..+e = > n = E = x n + •(n EAd OnT+
 k,j,j (EnMop2p- df n = (xma = a+x - Olf = x•(O=y = 19. Maia Comput. 20, ENENy.
 4 9J5p.37Mld.chls.Sni lSncep h6epcr(posh a ipeurhJ Sfpt lile,SntSl.7BIT
 7, (fNM= aN a4omsues7MfN5lx.ml Rppl4es.rMvpip,ur hJuSmtJl,Amhl,xtSl.lr lSnce
 lcIt h6epcr(plo,cur 7BIT 11, E6nuN
 4N5,J3 7MEdNcaxu pM4A3trlr ipeur hJuSmptJlil.,,SnetSl. lr .ctlp 7Lin. Alg
 Applic. 52/53 ntMlii
 uw1so emn (JS,MA7MMinht.oSSt.4es.rMint - uAlcuct lr tJc9.la ipeu' hJuSmt
 4ilpMtu 7SIAM J. Sci. Stat. Comput. 12, Mf. I (M1
 6x9J5p.annji). (eSic7MNdclrr em5ceAtxpdrlptJlileeSt. S.f JdPlnS,cripeur hJuSmt
 4ilpStJut 7SIAM J. Matrix Anal. Applic. 13, MfMtnN
 4 9J5p.37MfN ca,u cpS.lr ipeur hJuMintJlile,MntSl 7Lin. Alg. Applic. 197/198
 Nt! EMU.
 1. iMpmem1j leil 7N Edp45oxrt)pStcpSlp tRdmSnppe hJuSmptJl8MtJ
 hcsc7S5lptJlileSntSl.7SIAM J. Sci. Comput. 25, 6Mf6MN
 iNu htc8ept7MndN Rpp1ne,rSlr tJcp,1SQipei hJuSmptJl8MtJ 7SIAM J. Matr x Anal.
 Applic. 27, 6EOn1uN

= $\tilde{Q}_1 \cdot H_1 \cdot R_1 \cdot Q_2 \cdots \tilde{Q}_{k-1} \cdot H_{k-1} \cdot R_{k-1} \cdot Q_k \cdots Q_m$, $H_i = U_i D_i V_i^T$, $D_i = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}$, $U_i = \begin{pmatrix} u_{i1} & \dots & u_{in} \end{pmatrix}$, $V_i = \begin{pmatrix} v_{i1} & \dots & v_{in} \end{pmatrix}$. Numer. Math. 101, y. MII

41 hulatxnl8ut3II II 9Cpsl8CnmllCnixl 7Mludlc4 alta ln tJ3R1IIlp4nCsrsr)AOruC, ilCuQhJ MII Numer. Math. 105, Nttf MEI

IChlxrJuiJr Aalvl uCmaxAal t6nMhtJap5 .tluT.tSGpanMr.xrram

9IPCtuniABPai.p3Cnrl II taiC7MtdcyltJlilnCs5mx.tMlf IatII)luAxtapr SIAM J. Sci. Stat. Comput. 19, E1NFNAY

(I41 MuuJt7Mlt H cOtaCnixsCtSpxstMCnrlJap5 .aluAl rStSln3, Alg. Applic. 221, utf MII

II 1)CIpMIII Cnrl II Pa.al 7Mtd cR.SantMlxra lAp5apC.t II utCtMlh hxAaIr.CsCI (I.larrllr3ACM . Math. Sof w. 23 EuN E1yl

I ICnmaprtC7ahld c4n4.xlCta (CICAsa ilCur hJuS4tMlJ8MtJlxalptJliI InCsMII3, Numer. Lin. Alg. 7, NMtNEu

RIRsuIltJc.mli iixrtC.rln 7Mldlc4As.urniaxprSltl hapSCAn(CICAps5 C.tlpMT CtMln laCml 9at apvluC.a3, IBM J. Res. Dev. 44, ulr uN I

PCn.MuAlptCIMiJr AapvpuC AsauantCtSina6AAa6xCstl 1,3)Jlsara.Cnnp53raa,

41 9xHII II Cnixl3II Mx1Ca8nnm, = y (E22p Dyex ζ (Z+ = (ApasSθ " (FAYOnE (y< y = = Q= H:xΩE=yEΩ• O= OHar: llE@amp;F35, Efy EI

II Mx1CaMCSaCnrl .lniCIC7NM 1dth.JaxsMrana IMnaCIAiod CAAapCtShm Pxstlpa (I.larrllr3Cncur ncy Comput. Pr ct. Ex er. 22, Mr66

II .auuaA3I ipMiB 3Mlauar8Cnrl ICilx7N MNPatJlmEnaAilpMtJup hMaSt)luAxtMnluuxnM.Ctulnr IAt(CICAAsnra6x3ntM5,Cnrl C. tluutMln SIAM J. Sci. Comput. 34, 4N In 4NETl

MSrtlSpc panlapnansStJACICsasap5 Mn.sxma,

f IP liantsauCfnnMclMxni tyWdpCpMS CnixsCpS.GtShrltsMIIIC r8IE Pr c. 298 MII

.IRIMasA6hMI)I.I AraMElchrltsSaat8Iarvp II tJlilnCAa.luAlrMtSlnrSIAM J. Sci. Stat. Comput. 4 NuM N.tl

PI)lrltnCpm8PI PxsAal 8 5Cnph7Mtdl c(CICAps5A.aluArlutMlhr C5atCnixsCI PCISf3umer. Math 48 NEt5Nnll

II RsmGm5l hJ1aMcMtytd c4n4AAs.Ctulnr hrtsMIIIC.rtl ls.aCpMr.Iata ssrlam (ploAiu8,SIAM J. Sci. Stat. Comput. 7, ytNf t1EI

I C lxa 7Mtdl c4 5ltCtSlrPatJlmp)luAxtMnJap5 .t tlIMtCtSII SIAM J. Sci. Stat. Comput. 7, 6r 1601

II II PlmSPIC6hCiaa7Mtdlc4n4staInCtSiaanr II maK63Numer. Math. 43 yEf tII

cJap5 rGIMtCtSlnCrtlx.txIamCtpMfrxCAstlxtxpastrara,

41f I 9IJC...a3 5I (I 9ian8Cn. I5I maMli 7Mtdl cp5 .tllIMtCtuln k aAAuPCtpMar7, Numer. Math. 49, yM61

hlpMC7MtdlcM.opMsil putJwl .Ot k aAsSfttJlil.CsM1Cn3Numer. Math. 53, Er Mful)I II .auax7Mtdl cCrtp5 C.tlpMtCtulnCnmapulnRtCtI uas Lin. Alg. Applic. 122/ 123/ 124 Mur f it/I

l. 5aS.Jis7MMdt.Ot p5 .i.luAlrMtulnr I Cnmapulnrm RltafaCnrfA.nluMDaOt h6xCpa4Aplf StClnSIAM J. Matrix Anal. Applic. 12, r r NOr u6I

J51h8cat7MtdlcOt 9sl.a klaAsSttJlilnCsMSK63Numer. Math. u85ENtI

pxCnxluAxtCtMHOCrSntapartlunia.tulrtl .luAsafiS.anrIltCtSl.Cnrl aMIAAsS.Ctuln tl .atlrl3raa,

iI).ca.al 7Mtd5amx.unpxCnxu)luAxtCtMnl RsauintCpn StCIOCArvI uCtMlnr3T Comput. Sci. Eng. 3, NIENI

- II(1 laCp.Cnm EMAAla7Mtdl cp5 .C.tllIMtCtSlMni 56rtpM.tatlr 5ltCtMlnr3T ETNA 21, N15N

aI.I PaIuMIIId Quantum Computer Science)CudSmjauaIrut(larr3a8 1 IaI

5.3 The Full-Rank Least Squares Problem

Consider the full-rank system $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$. We can write $Ax = b$ as $Ax_1 = b_1, Ax_2 = b_2, \dots, Ax_p = b_p$. The system has a unique solution if A is full rank. If A is not full rank, we can find a least squares solution instead.

$$\begin{array}{lll} p & = 1 & : x_{\text{opt}} = b_2, \\ p & = 2 & : x_{\text{opt}} = (b_1 + b_2 + b_3)/3 \\ p & = 0 & : x_{\text{opt}} = (b_1 + b_3)/2 \end{array}$$

Notice that the first two solutions are flat (lie on the same line), while the third is a point. This illustrates that the least squares solution is unique if the system is full rank, and it is not unique if it is not full rank.

Rotating a point into the least squares (SVD)

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2 \quad (53)$$

is similar to this:

$Q = A^{-1}A$ is a full-rank system so it is consistent. The solution is a vector in the column space of A . If A is not full rank, the solution is not unique. The 2-norm is equivalent to the Frobenius norm. This is a weighted least squares problem.

In this case, the two points are free to lie on any line through the origin. For a general approach,

5.3.1 Implications of Full Rank

Since $x \in \mathbb{R}^n$, $Z \in \mathbb{R}^{n \times n}$, $a_i \in \mathbb{R}^n$, and the entries

$$Ax = b \iff b = Ax_1 + a_2^\top A x_2 + \dots + a_n^\top A x_n$$

we have $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. Excluding SVD (53), the rest of the text is based on this assumption. $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ are assumed to be full rank. A is full rank if $\det(A) \neq 0$. A is full rank if $\text{rank}(A) = n$. A is full rank if $\text{rank}(A) = m$. A is full rank if $\text{rank}(A) = \text{rank}(A^\top A)$. A is full rank if $\text{rank}(A) = \text{rank}(A^\top A)$.

$$A^\top A = A^\top b$$

Terrengdeleksjonsløsning

$$c(x) = \frac{1}{2} \|Ax - b\|^2,$$

ten

$$\nabla c(x) = A^T(Ax - b),$$

single iteration

$$r_{LS} = b - Ax_{LS}$$

the minimum residual

$$\rho_{LS} = \|Ax_{LS} - b\|_2$$

total least squares (TLS) is the standard approach by singular value decomposition (SVD). It is a generalization of the least squares method for linear systems of equations. For every matrix $A \in \mathbb{R}^{m \times n}$, there exists a unique SVD factorization $A = U \Sigma V^T$, where $U \in \mathbb{R}^{m \times m}$, $\Sigma \in \mathbb{R}^{m \times n}$, and $V \in \mathbb{R}^{n \times n}$.

$$A = U \Sigma V^T = \sum_{i=1}^n \sigma_i u_i v_i^T$$

is the full rank matrix $A \in \mathbb{R}^{m \times n}$, then

$$\|Ax - b\|_2^2 = \|U\Sigma V^T x - b\|_2^2 = \sum_{i=1}^n (\sigma_i u_i^T x - b_i)^2 = \sum_{i=1}^n (\sigma_i u_i^T x - \sigma_i b_i)^2 = \sum_{i=1}^n \sigma_i^2 (u_i^T x - b_i)^2$$

where $x = V^T y$. If y is the solution to the system $Ax = b$, then $y = U^T b$.

$$x_{LS} = \sum_{i=1}^n \frac{u_i^T b}{\sigma_i} v_i \quad (532)$$

and

$$\rho_{LS} = \sqrt{\sum_{i=1}^n \frac{(u_i^T b)^2}{\sigma_i^2}} \quad (533)$$

It is also possible to find the least squares solution to the overdetermined system $Ax = b$ by using the SVD. The solution is given by $x_{LS} = V^T \Sigma^{-1} U^T b$.

- How close is \hat{x}_{LS} to x_{LS} ?

Is $\|x_{LS} - \hat{x}_{LS}\|_2 = \|b - Ax_{LS}\|_2$?

Influence of temperature on the topographic distribution of *Acacia* species in the semi-arid region of Brazil

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 10^{-6} \\ 0 & 0 \end{bmatrix}, \quad \delta A = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 10^{-8} \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad \delta b = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$x_{LS} = \begin{bmatrix} \end{bmatrix}, \hat{x}_{LS} = \begin{bmatrix} 999 \\ 999 \cdot 10^4 \end{bmatrix}, r_{LS} = \begin{bmatrix} \end{bmatrix}, \hat{r}_{LS} \sqrt{n} \begin{bmatrix} -999 \\ 999 \cdot 10^0 \end{bmatrix}.$$

Relative error of factored matrix of lag to singular value $\kappa_2(A) = 10^6$ varies

$$\frac{\|\hat{x}_{\text{LS}} - x_{\text{LS}}\|_2}{\|x_{\text{LS}}\|_2} = \frac{9999 \cdot 10^4}{\kappa_2(A)^2 \|A\|_2} \frac{\|\delta A\|_2}{\|A\|_2} = 10^{12} \cdot 10^{-8}$$

ad

$$\frac{\|\hat{r}_{\text{LS}} - r_{\text{LS}}\|_2}{\|b\|_2} \approx .7070 \cdot 10^{-2} \leq \kappa_2(A) \frac{\|\delta A\|_2}{\|A\|_2} = 10^6 \cdot 10^{-8}.$$

Teachers' states of mind and personal development^(A). Pewe dean's sustainable tacit knowledge

5.3.2 _ The Method of Normal Equations

A direct method for single-track sphere

Goal: Implement a function `rank(A)` that takes a matrix A as input and returns its rank. The function should use the singular value decomposition (SVD) to find the rank.

Controllability of $\mathbf{C}_{=A^TA}$

Introduction of soft data $A^T b$

Constitutor - GET

W. H. D. Green

Teach her to speak more
natural language, and she will speak
more easily and clearly. Teach her to speak
more easily and clearly, and she will speak
more natural language.

If $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $\text{rank}(A) = n$, if $\hat{x}_{\text{LS}} = A^T b / \|A^T b\|_2$

$$(A^T A + E)\hat{x}_{\text{LS}} = A^T b$$

we

To see exact

$$\frac{\|\hat{x}_{\text{LS}} - x_{\text{LS}}\|_2}{\|x_{\text{LS}}\|_2} \approx \text{u}\kappa_2(A^T A) = \text{u}\kappa_2(A)^2. \quad (53)$$

In the steady-state optimal problem, it is often the case that the term $\text{rank}(A) < n$ (e.g., $n=3$, $m=2$). In such cases, the solution to the LS problem is unique, but the matrix $A^T A$ is singular, so that

$$A = \begin{bmatrix} J & 1 \\ 0 & J \end{bmatrix},$$

then $\text{rank}(A) = 1$.

$$\text{fl}(A^T A) = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

which is singular. This leads to numerical breakdowns.

5.3.3 LS Solution Via QR Factorization

Let $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $\text{rank}(A) = n$. Then there exists a unique $Q \in \mathbb{R}^{m \times m}$ and $R \in \mathbb{R}^{m \times n}$ such that

$$Q^T A = R = \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \in \mathbb{R}^{m \times n} \quad (55)$$

is upper triangular

$$Q^T b = \begin{bmatrix} \cdot \\ \vdots \\ \cdot \end{bmatrix} \in \mathbb{R}^m$$

then

$$\|Ax - b\|_2^2 = \|Q^T Ax - Q^T b\|_2^2 = \|R_1 x - c\|_2^2 + \|d\|_2^2$$

from $x \in \mathbb{R}^n$, since $\text{rank}(A) = \text{rank}(R_1) = n$, if $\hat{x}_{\text{LS}} = R_1^{-1}c$ is the least squares solution

then

$$R_1 \hat{x}_{\text{LS}} = c.$$

$$\rho_{\text{LS}} = \|d\|_2.$$

Another advantage of the LS method is that it can be easily extended to solve the Tikhonov regularized least squares problem. If $A \in \mathbb{R}^{m \times n}$, $\beta \in \mathbb{R}$, $b \in \mathbb{R}^m$, then the Tikhonov regularized least squares problem is given by

$$\|Ax_{LS} - b\|^2 + \beta \|x\|^2$$

for $j = 1:n$

$$v = \begin{bmatrix} A(j+1:m, j) \\ A(j+1:m, j) \end{bmatrix}$$

$$f = 2/v^T v$$

$$b(j:m) = b(j:m) - \beta(v^T b(j:m))v$$

end

So

$$S_{LR}(1:n, 1:n) \cdot x_{LS} = b(1:n).$$

The total cost of solving the LS problem is $O(mn^2(m-n/3))$ for the full rank case and $O(mn^2)$ for the near-rank-deficient case.

If a system of equations is

$$(A + \delta A)x = b \quad (5.3.6)$$

where

$$\|\delta A\|_F \leq (6m - 3n + 41)n \mathbf{u} \|A\|_F + O(\mathbf{u}^2) \quad (5.3.7)$$

and

$$\|\delta A\|_F \leq (6m - 3n + 40)n \mathbf{u} \|A\|_F + O(\mathbf{u}^2). \quad (5.3.8)$$

Then the LS solution is given by $x_{LS} = S_{LR}(1:n, 1:n)^{-1} b$. It is not clear if a general MGS algorithm can be used to solve such a system of equations. This is a serious open problem.

5.3.4 Breakdown in Near-Rank-Deficient Case

Another advantage of the LS method is that it can be easily extended to solve the Tikhonov regularized least squares problem. If $A \in \mathbb{R}^{m \times n}$, $\beta \in \mathbb{R}$, $b \in \mathbb{R}^m$, then the Tikhonov regularized least squares problem is given by

$$\|Ax_{LS} - b\|^2 + \beta \|x\|^2$$

5.3.5 A Note on the MGS Approach

In high MGS cost due to the fact that QR Triangularization is slow for large matrices. This is a serious problem.

$(A^T A)_x = A^T b$ to determine the system of equations $R_1 x = Q_1 b$. If R_1 has full rank, this approach is called the QR factorization method. In the case where $A^T A$ is not invertible, one can use the GSLS (Givens-Schur) algorithm, which is based on the QR factorization of A .

The GSLS algorithm is a modified version of the QR factorization method. It uses Givens rotations to transform the matrix A into an upper triangular matrix R , while maintaining the property that $A^T R = Q$. This transformation is used to solve the system $A^T R x = A^T b$ for x . The GSLS algorithm is particularly useful for solving large-scale least squares problems because it requires fewer operations than the standard QR factorization.

5.3.6 The Sensitivity of the LS Problem

We have studied the sensitivity of the LS problem, that is, the sensitivity of the solution of the LS problem to changes in the data. We have shown that the condition number of the LS problem is related to the condition number of the matrix A . We have also shown that the condition number of the LS problem is bounded by the condition number of the matrix A .

$$\begin{aligned} 1 &= \|A(A^T A)^{-1} A^T\|_2 & \frac{1}{\sigma_n(A)} &= \|(A^T A)^{-1} A^T\|_2, \\ 1 &= \|I - A(A^T A)^{-1} A^T\|_2 & \frac{1}{\sigma_n(A)^2} &= \|(A^T A)^{-1}\|_2. \end{aligned} \quad (539)$$

These results are very useful for the study of the sensitivity of the LS problem.

Theorem 5.3.1. Suppose that x_{LS} , r_{LS} , \hat{x}_{LS} , and \hat{r}_{LS} satisfy

$$\begin{aligned} \|Ax_{LS} - b\|_2 &= \min_{x \in \mathbb{R}^n} \|Ax - b\|_2 & r_{LS} &= b - Ax_{LS}, \\ \| (A + 8A)^{-1} b - (b + 8) \|_2 &= \min_{x \in \mathbb{R}^n} \| (A + 8A)^{-1} b - x \|_2 & \hat{x}_{LS} &= (b + 8) - (A + 8A)x_{LS}, \end{aligned}$$

where A has rank n and $\|8A\|_2 < \alpha_1(A)$. Assume that b , r_{LS} , and x_{LS} are not zero. Let $\theta_{LS} \in (0, \pi/2)$ be defined by

$$\sin(\theta_{LS}) = \frac{\|r_{LS}\|_2}{\|b\|_2}.$$

If

$$\epsilon = \max \left\{ \frac{\|8A\|_2 \|8\|_2}{\|A\|_2 \|1\|_2} \right\}$$

and

$$\nu_{LS} = \frac{\|Ax_{LS}\|_2}{\sigma_n(A)\|x_{LS}\|_2} \quad (5310)$$

then

$$\frac{\|\hat{x}_{LS} - x_{LS}\|_2}{\|x\|_2} \leq \epsilon \left\{ \frac{v_{LS}}{\cos(\theta_{LS})} + [1 + v_{LS} \tan(\theta_{LS})] \kappa_2(A) \right\} + O(\epsilon^2) \quad (53)$$

and

$$\frac{\|\hat{r}_{LS} - r_{LS}\|_2}{\|r_{LS}\|_2} \leq \epsilon \left\{ \frac{1}{\sin(\theta_{LS})} + \left[\frac{1}{v_{LS} \tan(\theta_{LS})} + 1 \right] \kappa_2(A) \right\} + O(\epsilon^2). \quad (53)$$

From the condition $\text{rank}(A + tE) = n$ for $t \in [0, \epsilon]$, it follows that $\theta_{LS} = A/\epsilon$ and $v_{LS} = \delta b/\epsilon$. By Theorem 5.3, we have

$$(A + tE)^T (A + tE)x(t) = (A + tE)^T (b + tf) \quad (53)$$

is only feasible for $t \in [0, \epsilon]$. Since $x_{LS} = x(0)$ and $r_{LS} = x(\epsilon)$, we have

$$\hat{x}_{LS} = x_{LS} + \epsilon \dot{x}(0) + O(\epsilon^2).$$

By (53), we have $\|\hat{x}_{LS} - x_{LS}\|_2 \leq \epsilon \|\dot{x}(0)\|_2$.

$$\frac{\|\hat{x}_{LS} - x_{LS}\|_2}{\|x_{LS}\|_2} = \epsilon \frac{\|\dot{x}(0)\|_2}{\|x_{LS}\|_2} + O(\epsilon^2). \quad (53)$$

In addition, we have (53) because $\|x_{LS}\|_2 = \|\dot{x}(0)\|_2$. This is of interest. This

is



$$(53)$$

Using (53), we can express $\|\hat{x}(t)\|_2$ for $t \in [0, \epsilon]$ and $\|r(t)\|_2$ for $t \in [0, \epsilon]$ if we start

$$\begin{aligned} \|\dot{x}(0)\| &\leq \|(A^T A)^{-1} A^T f\|_2 + \|(A^T A)^{-1} A^T E x_{LS}\|_2 + \|(A^T A)^{-1} E^T r_{LS}\|_2 \\ &\leq \frac{\|b\|_2}{\sigma_n(A)} + \frac{\|A\|_2 \|x_{LS}\|_2}{\sigma_n(A)} + \frac{\|A\|_2 \|r_{LS}\|_2}{\sigma_n(A)^2}. \end{aligned}$$

Replacing (53) in (53), we have

$$\frac{\|\hat{x}_{LS} - x_{LS}\|_2}{\|x_{LS}\|_2} \leq \epsilon \left(\frac{\|b\|_2}{\sigma_n(A) \|x_{LS}\|_2} + \frac{\|A\|_2}{\sigma_n(A)} + \frac{\|A\|_2 \|r_{LS}\|_2}{\sigma_n(A)^2 \|x_{LS}\|_2} \right) + O(\epsilon^2).$$

Replacing (53) in (53), we have

$$\cos(\theta_{LS}) = \frac{\|Ax_{LS}\|_2}{\|b\|_2}, \quad \tan(\theta_{LS}) = \frac{\|r_{LS}\|_2}{\|Ax_{LS}\|_2}. \quad (53)$$

Therefore, from (53), we have

$$r(t) = (b + tf) - (A + tE)x(t)$$

and relate $r_{LS} = r(0)$ and $\hat{r}_{LS} = r(\epsilon)$. Thus

$$\frac{\|\hat{r}_{LS} - r_{LS}\|_2}{\|r_{LS}\|_2} = \epsilon \frac{\|\dot{r}(0)\|_2}{\|r_{LS}\|_2} + O(\epsilon^2). \quad (537)$$

For (537) we have

$(Q = (I - A(A^T A)^{-1}A^T)(I - E x_{LS}) - A(A^T A)^{-1}E^T r_{LS})$.
By ignoring (539) and equating $\|f\|_2 = \|b\|_2 = \|E\|_2 = \|A\|_2$, we get

$$\|Q\|_2 = \|b\|_2 + \|A\|_2 \stackrel{\text{P.S.P}}{\approx} \frac{\|A\|_2 \|r_{LS}\|_2}{\sigma_n(A)}.$$

and for (537) we have

$$\frac{\|\hat{r}_{LS} - r_{LS}\|_2}{\|r_{LS}\|_2} \leq \frac{\|b\|_2}{\|r_{LS}\|_2} + \frac{\|A\|_2 \|x_{LS}\|_2}{\|r_{LS}\|_2} + \frac{\|A\|_2}{\sigma_n(A)}.$$

To apply (537) to the following $\kappa_2(A)$ and $\kappa_1(A)$

It is stated in (537) that $\kappa_2(A)$ is the ratio of the largest singular value to the smallest singular value of A . It is also stated in (537) that $\kappa_1(A)$ is the ratio of the largest singular value to the smallest singular value of $A^T A$.

$$\nu_{LS} = \frac{\|Ax_{LS}\|_2}{\sigma_n(A)\|x_{LS}\|_2} \stackrel{\text{P.S.P}}{\approx} \frac{\|A\|_2}{\sigma_n(A)} = \kappa_2(A).$$

To S.D. of (537) we have $\|b\|_2 = \|A\|_2$ and $\|x_{LS}\|_2 = \nu_{LS}$ given by (537), we get

Plugging (537) into (539) we get the inequality for (539) to be true if and only if $\nu_{LS} \approx \kappa_2(A)$.

$$\frac{\|\hat{r}_{LS} - r_{LS}\|_2}{\|r_{LS}\|_2} \leq (\kappa_2(A))^2. \quad (538)$$

Along this line, it is shown in Section 5.3 that $\kappa_1(A)$ is the ratio of the largest singular value to the smallest singular value of $A^T A$ and $\kappa_2(A)$ is the ratio of the largest singular value to the smallest singular value of A . It is also shown in Section 5.3 that $\kappa_1(A) \approx \kappa_2(A)$ if and only if A is well-conditioned. This is true for (538) as well.

$$\frac{\|\hat{r}_{LS} - r_{LS}\|_2}{\|r_{LS}\|_2} \leq \kappa_1(A). \quad (539)$$

From this it follows from (537) and (539) that $\kappa_1(A) \approx \kappa_2(A)$ if and only if $\nu_{LS} \approx \kappa_2(A)$.

5.3.7 Normal Equations Versus QR

Historical note: the first journal paper to define the QR algorithm was by J. H. Wilkinson in 1961.

Ten years earlier, in 1951, Wilkinson had developed Q, A, a dot product algorithm for the solution of the Ax = b system as part of his I-Solver.

The algorithm, like Givens' LS solver, Subr, has a series of orthogonal transformations to reduce the system to upper triangular form.

Today, a practical implementation of LS is part of ASVNS.

Having a good understanding of the LS method is important for the development of a more efficient algorithm. In addition, it is useful to understand the numerical behavior of the QR algorithm. Since the QR algorithm is based on the Gram-Schmidt process, it is important to understand the effects of rounding errors in the computation of the orthogonal basis.

5.3.8 Iterative Improvement

A brief history of iterative methods can be found in Golub and Van Loan (1967, 1968).

In this section, we will focus on the Jacobi and Gauss-Seidel methods. These methods are iterative, and they are used to solve linear systems of equations. They are based on the idea of approximating the solution of a system of linear equations by successive approximations. The Jacobi method is a direct method, while the Gauss-Seidel method is an iterative method. Both methods are based on the same principle: at each iteration, the solution is updated by using the previous iteration's solution as the starting point for the next iteration.

$$r^Q = 0, x^Q = 0$$

$$f \quad r \quad k = 0, 1, \dots$$

$$\begin{bmatrix} \vdots \\ \vdots \\ A \end{bmatrix} = \begin{bmatrix} \vdots \\ \vdots \\ A \end{bmatrix} \cdot \begin{bmatrix} A \\ \vdots \\ A \end{bmatrix} \begin{bmatrix} r^{(k)} \\ x^{(k)} \end{bmatrix}$$

$$\begin{bmatrix} A \\ \vdots \\ A \end{bmatrix} \begin{bmatrix} p^{(k)} \\ z^{(k)} \end{bmatrix} = \begin{bmatrix} f^{(k)} \\ g^{(k)} \end{bmatrix}$$

$$\begin{bmatrix} r^{(k+1)} \\ x^{(k+1)} \end{bmatrix} = \begin{bmatrix} r^{(k)} \\ x^{(k)} \end{bmatrix} + \begin{bmatrix} p^{(k)} \\ z^{(k)} \end{bmatrix}$$

end

$$\begin{bmatrix} I \\ A^T \end{bmatrix} \begin{bmatrix} p \\ z \end{bmatrix} = \begin{bmatrix} \end{bmatrix}$$

tasfirs

$$\begin{bmatrix} I_n & 0 & R_1 \\ 0 & I_{m-n} & 0 \\ R_1^T & 0 & 0 \end{bmatrix} \begin{bmatrix} h \\ f_2 \\ z \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ g \end{bmatrix}$$

Vice

$$Q^T f = \left[\begin{smallmatrix} & & \\ \vdots & \ddots & \\ & & \end{smallmatrix} \right]_{mn}^n, \quad Q^T p = \left[\begin{smallmatrix} & & \\ & \ddots & \\ & & \end{smallmatrix} \right]_{mn}^n.$$

**Tos _p ad calcederiebysigletiagh yes h _g ad
R_yz = J - hacking**

$$p = Q \begin{bmatrix} h \\ f_2 \end{bmatrix}.$$

5.3.9 Some Point/Line/Plane Nearness Problems in 3-Space

Tefekl son lega salone yioaach teih amiteng
naxxides leewi or tsejedek leies hoperallie
ios esadee of xel fessalim studie al sruem
yland se the hoj fejend am oder & a relatly
tfejedus spols wa vira preevien an oach
Tefekl son lega salone yioaach teih amiteng
naxxides leewi or tsejedek leies hoperallie
ios esadee of xel fessalim studie al sruem
yland se the hoj fejend am oder & a relatly
tfejedus spols wa vira preevien an oach

$$p \times q = \begin{bmatrix} p_2q_3 - p_3q_2 \\ p_3q_1 - p_1q_3 \\ p_1q_2 - p_2q_1 \end{bmatrix}.$$

Treatment of a malignant palpable Foay^{VER³}, *Elephantiasis* by

$$v^c = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}.$$

Itflostat

$$p \times q = p^c \cdot q = -q^c \cdot p = -(q \times p).$$

Using symmetry, it is easy to show that

$$p \times q \in \text{span}\{p, q\}. \quad (5.3.21)$$

Geppteside

$$(p \times q) \times r = (p^c \cdot q)^c r = (qp^T - pq^T)r = (p^T r) \cdot q - (q^T r) \cdot p, \quad (5.3.22)$$

$$(p \times q)T(r \times s) = (pq)T.(rs) = dt((qf)[rs]), \quad (5.3.23)$$

$$P_{PC} = PPT - \mathbb{E}[P]I \cdot l, \quad (5.3.24)$$

$$\| \mathbf{p}^T \mathbf{q} \|_2^2 = \| \mathbf{P} \|_2^2 \cdot \| \mathbf{q} \|_2^2 \cdot \left(1 - \left(\frac{\| \mathbf{p}^T \mathbf{q} \|_2}{\| \mathbf{p} \|_2 \cdot \| \mathbf{q} \|_2} \right)^2 \right). \quad (5.3.25)$$

Wij voegden de theorieën van de verschillende theoreti^{sch}
Folksalociaal leedresistente^s toe.
P.5.3.13-P5.3.15.
Gedlapot, fidapot ollaissto,
Pr Henk Jelle
jesse

$$\min_{\mathbf{z}} \|\mathbf{z} - \mathbf{y}\|_2.$$

Hastings dictots and teicarcosxlat

$$z^{\text{opt}} = y + \frac{1}{\eta T_N} v^c v^c (y - p_1), \quad \mathbf{v} = \mathbf{P} \mathbf{Z}^- \mathbf{P} \mathbf{l} \quad (5.3.26)$$

Fr. Henr. 2 Gedächtnisfelder mit opt. 11taisstd 2nd teopt 2 opt 11taisstd 2nd teopt 2 opt 11taisstd 2nd

$$\min_{z_1 \in L_1, z_2 \in L_2} \|z_1 - z_2\|_2.$$

Flaxseed participants and ad flaxseed participants and telomerase

$$z_1^{\text{opt}} = \mathbf{P} \mathbf{I} + \frac{1}{\|\mathbf{r}\|} \mathbf{w} \mathbf{w}^T \cdot \mathbf{r} \mathbf{q} (\mathbf{q} - \mathbf{P} \mathbf{I}), \quad (5.3.27)$$

$$z_2^{\text{opt}} = \dot{\mathbf{q}} + \frac{1}{r^T r} \cdot \mathbf{W}^T \cdot {}_T C(\mathbf{q} - \mathbf{p}_l), \quad (5.3.28)$$

Value = $P_2 \cdot \pi$, **w** = $\varphi \cdot q$, **ad** = $v \omega$

Pr. h. m. 3 Geapie adopt, fideopt or taissto
y, ieſſe

$$\min_{z \in P} \|z - y\|_2.$$

If P passing the δ -criterion, ad_{δ} the table below.

$$\mathbf{Z}^{\text{pt}} \cdot \mathbf{J} \cdot p_1 = \frac{1}{\sqrt{\mathbf{V}^\top \mathbf{V}}} \cdot \mathbf{V}^\top (\mathbf{y} - p_1) \quad (5.3.29)$$

$$\text{We have } (p_2 - p_1)^c (p_3 - p_1).$$

Teniedse Finshtos (5.3.26) (5.3.29) ardevis in bantue
instas en vele opghalees in Afrika en Europa
het seker (2011).

Problems

$$\text{P5.3.1 1.22. } \text{ATAx} = \text{Arb} (\text{ATA} + \text{F})\text{x} = \text{Arb } 2\text{x } 2\text{IF } 12 \quad a_n(A)^2 + C(LnC2Q) = b - Ax \\ 2x = b - Ax, r(CB) - r = A(AT + F) - 1Fx 2x$$

$$\| \hat{r} - r \|_2 \leq 2 \kappa_2(A) \frac{\| F \|_2}{\| A \|_2} \| x \|_2.$$

P5.3.2 1.22. ATAx = ATb₂x rCaATAx = Arb+ LCBxH l₂ qAT l₂b l₂2x A
Ck l₂a(14Fv x2)- +C(LrCa2)

$$\frac{\|x - \hat{x}\|_2}{\|x\|_2} \leq c u \kappa_2(A)^2 \frac{\|A^T\|_2 \|b\|_2}{\|A^T b\|}.$$

P5.3.3 5. AERmknm x n), w ERn, 2x xB.vB

$$B = \begin{bmatrix} A \\ w^T \end{bmatrix}.$$

$$+ \frac{C}{A} \frac{(\mathbf{I}_n \mathbf{C}_n \mathbf{C}_n^T \mathbf{B})}{G_1^T \beta_{lk}^T G_{1,k}} \quad A_{\frac{1}{k}} \frac{(A)}{\beta_{lk}^T} \frac{2x}{A_{\frac{1}{k}l k l}} \frac{a_1(B)}{a_1(B)} = \sqrt{\|\mathbf{A} \mathbf{I} + \mathbf{I}_w\|_2^2}. \quad | \quad \Psi_k: \tilde{G}\beta = \langle A, A\beta \rangle \quad /,^TA \quad /.$$

**P5.3.4 R3... 1973 +,1(B rC2AERmnCk x& n)+ () (7) 0---,() -+1)((' 1+(,1)-().
 1,)'(-,-0-+(',1R(-,-5),'-,)'(1 PA = LU, LCBxERnxnQ,)Qn=(LAnxQ)112xP
 UERnxnQ,1Bx nQ112xi 2P PERmxmQ2 BxT,r2nQ(vtx12Q) C(L rCBBa(T1QnQ) Qv
 ,+23+a2 .2 .Ax n(vx2Baxn ERn.aC rC2H2 - Pb1pQTQTRBx +G(C2AQBx= z.
 x 1Ax-xb\2QTQQT4T2C(LrC2nCQTBrC(x(o.(12Q1nB8+ k.=AT QT(xABhaQA)nC2
 I(4BC=xBx+)b rCBn1 1Qn(o 2QBLCBvB2Bx < 3/n/3.**

P5.3.5 m.. 2...b $C = (ATA)^{-1} LCB$, rank(A) = n, $\lambda_1(\gamma - Q(-\beta))^{-1}$, $(\lambda_1 - \alpha_1^2)^{-1}$, $\lambda_1^2 + \alpha_1^2$, $A = QR$, $Q^T C Q = R^T B R^{-1}$, $R^T B R^{-1} = 2I_n$, $Q^T C Q = 2I_n$, $C = R^{-1} B^T R$, $C = 2I_n$.

$$R = \begin{bmatrix} & & & \\ & \mathbb{R} & & \\ & & & \end{bmatrix} \quad C = (R^T R)^{-1} = \begin{bmatrix} (1 + \frac{1}{c^2}) & . & . & . \\ . & c^2 & -v_1 & v_2 \\ . & . & c & 0 \\ . & . & 0 & c \end{bmatrix}$$

C1 = (srs)- 1. 6xMQ16xQ1B2 21xQ1CT1C2(BxLxQ.B. 11Bx m.Q)1-2x 1xQ1
 (oR LQxCCB1Bx m.Q2 11x 1xT1) (oC. n.x 2-1xQ1CTC(4x xBj.QxP2.3/3N1-

P5.3.6 .2..8.. A ERnxnQ. II TBn x Q 2x nC2r = b- Ax LCBxR, bx ERn 2x x Q)(wAx-
+CL C(L n(a(T14B 2. II TBn x Q E RnxnLQ(Q)QTr ! (B)Q4) (TnC2(A + E)x = b
IQnEMBrCB+ hanxQw2Q(o x Ir) 2x)(nBLC2Ex = r (QTEQ)(QTx) = QTTr.

P5.3.7 38... $P_1 \dots P_{n-1} rCBr_2 QC2Bx2a((xQ)2B.x1 \dots X_n B)(L rC2x1 = -2x LTC$
 $r(a(T1rB x2 \dots X_n QBvrC2a) K22-AnQT2Adj (orCBA22T)A$

$$x_i - x_j \approx d_{ij}, \quad 1 \leq i < j \leq n.$$

$\vdash (\exists x \cdot X) \rightarrow \neg Q \quad \text{EX} \vdash \neg Q$

$$c(x_1, \dots, x_n) = \sum_{i=1}^{n-1} \sum_{j=i+1}^n (x_i - x_j - d_j)^2$$

gicp= r Ae prng=nd a x1= 0

P5.3.8.e..8 .. A \in R $m \times n$ Ck 411 Ext aCab \in Rm \times x c \in Rn EA T2AvnC(LC(L a(a(b14 A a = CT . LTaC(4a(b14aTv1 AxLTaTa1)Tvaf+411 A Z n., 1.AmlgeAM.DMnb D AND zTcT. b14-aTf=0 I $_n$ (,n,. Wed < ==3 da = (ZTcT . LCAEA . blVIdlxAIIAy - b)2 LTaC y= zTx.vx A= AZ

P5.3.9.e..8 .. A \in R xn, x b \in RmLTaG 2 n. 3= 3B : e0+= d>0 << = (+n8: (A00 f= nyO) An O (OA (n0<<0 d=0 Rm m.4aCaCaMTz r T ,1AEaETv1E .vx MTM=D T.xT.fv=:

P5.3.10 582A \in R $m \times n$ C2AEvl n (n 0=a 2 [0xA)A

$$M(a) = \begin{bmatrix} ai_m & A \\ AT & 0 \end{bmatrix}.$$

+C(LaCa

$$C_{m+n}(M(a)) = bTv \left\{ a, - \sqrt{cn(A)^2 + (\)^2} \right\}$$

[a aCabTv bTv A 2(M(a)).

P5.3.11 14got.8ot8..ot..88.8.84ot .8ot.8. N. 5..8..8.. .. ot.8.86.4.or

$$\begin{aligned} x(k) &= 0 \\ \text{for } k &= 0, 1, \dots \\ r(k) &= b - Ax(k) \quad 0x_0 = A \text{EAaTTvX} \\ \|Az(k) - r(k)\|_2 &= bTv \\ x(k+1) &= x(k) + z(k) \end{aligned}$$

end

6X1.4Tv1 aCaaAt+ oaa(ETxaT)(FA T.2.T=,=AP(L dv1 N1 1AETaAEaT)EA EA 4TEAx 6X+C(LC.a aCA(2A TIAEA(T)EA,1 AaaTv(g)=0Tv aCAta2EaT2AbE(2AbAva aCAdA T2AvT) \$538

P5.3.12 8..>(532)- (532).

P5.3.13 8..>(532) v(aTv) aCaL = {P r(p2- B) Lr \in R}.

P5.3.14 8..1.(532) v(aTv) aCaC2Tv bTv AE^{opt} \in R² (o V_{W-q1}) - [p2- P] Q Ω 12 T.FA1A2vaJ

P5.3.15 8..1.(532) v(aT) 1aCaP = { x: xT((p2- p) x (p3- P)) = 0 }

Notes and References for § 3

.8 8.1.... .8 .04.8N. ot.881otr2.8. ..8..8. .4..e.8or

fe55Ne8. (1965). ®(ITdI).II) LTaA1CaAx+(L ,(d)Tv aT(v. 1E +(12Tv18TvAE(-,II). .v. 8AkA +- 4FA,E,1Ab P-Nu er. Math 7, 338 352

J12)(1, .vx U4J11Tv.(v (1966). ®1aA(v aCAAEaT2AAvBbAva(o8Bka+- 4EB.+(-4IT)PR R Numer. Math. 9, 139 148

12 2v xAE IT. (1975). ®+ITTa1oCA+(14IT(v. (o8TvABAkA +- 4FA,E,1Ad P-Numer. Math 29, 241 254

-CA4A(F).4. aEv.1EdaTl(v. a. (=2aCa8+ E,1Ab Ck aaEaaAx(bA .aAvat(v,Aa4A CA1 FAaCA1Fa,(,2 aG I,AC(lxAE (E)T2Av. baETaPAAF

). AaAE .vx 2. II=ITv(v (1970). ®-CA8AkA +- 4FA,E,12d .vx ,A4k(dv2AEAP. Comput. J. 19, 309 316

lnE4=TVA(1973). ®lv (=TdTv aT(v AaC(x E aCA (=4AT(o 8TvAE8aka +- 4E2,E,1AdPR R SIAM J. Numer. Anal. 10, 283 289

+m2Ad(v. (1974). ®8TvAE 8AkA4FA,1 (ITbTv aT(vvx -)+z= J. ACM 21, 581 585

S. seminar al equations epo S.c. o. $\mathbf{R}^T \mathbf{R}_X = \mathbf{A} \mathbf{T} \mathbf{b}$ $\mathbf{S} \mathbf{J} \mathbf{C} \mathbf{P} \mathbf{Q} = \mathbf{Q} \mathbf{R}$. Ste. ocr JT \mathbf{S} eto rls.Mi tJcrc M.lpu e xef Sler e.cAteosclh rl,xtSl. Sl of eS.crf hc rtcAlnuf cnApc.Mr \$Mtc etM, MuApLc+c \$laCpvpu chocc

6x 9J5pr7MtdP chteS= 8t e .rSlm t.RctJlmhcuS.lpue, RJxtSl.r7Lin. Alg. Applic. 88/89 5/6

hJ.c. tl cetu3tr ln lh A3pt2l otlM U clp. M...mde8rl. em 2erl. 7hhd 7t38eptemh. 7P (kd7e.m9J5pa7aPhd8ccesrk

(6) 46cmM.7MEdc(ptxI oetMlkJdp vp (rcxmlSl rcr 7BIT 13 NMINE6N
6x 9J5pa7MtdP luAlycyt' 86pc xpoetSle= rSlmRppl0lx.mrvp Slop h6el cr

hlsxtMl.r7BIT 31, NEY5 / P
96e m. 756Mepsr1.1ch. 7ttnd61AtSue9er 8d (cpt poef Sl9lx.mrvp tJclM.cellclt
hJ.epcr(plo cu 7Numerical Lin. Alg. Applic. 2 NMf Nyu

16Qib2 7MtdP c1AtMs er 8epo fxiat l. lx.mvI tJclS.cel lcitr hJxepo plo cu
8MtP xtSA 3SiJtr 2.rhSmc SIAM J. Numer. Anal. 16 MfM6

16Qib2 7Mtd6c1 1AtSue er 8 epo poetSl9lx.mrvp tJds.ceplc lt hJxepo plos3u7T
BIT 37, (Itf Myy6

56Mepsrle.m9Pye= m. 7MtdP RrtSue lnSlAf sies 9er 8epo ptxpoetMl9lx.mrvp tJc
IM.celc lt hJxepo pT osZBIT 37, yu@5/6

1P QhR 7Mtd6c1 1AtSue9er 8 ptxpoetMl9lx.mrvp tJcs3el lcit hJxel cploscu7T
BIT 37, r tfMyp

PH, 7Mydt 9.a8 pt (cpt2 6125T vpxS.c. plc t hJxepo ploscu7SIAM J. Matr x
Anal. Applic. 20 ...5 INT

P 641M160eoT x5mh6pettT 7011dPc4 (epSe)T .mMastluopvpplM.cep lc16epcr
(plo cu 7SIAM J. Matrix Anal. Applic. 29 / MEE6

r: 9edlx,S.3161ieppe7 h6pett,7 em 1, leil 7MtdPc)luAxtS.i tJlMmStMl.MintJc
)luAl.ctr InelS.cel lc,rt hJxepo tMl.7Num. Lin. Alg. Applic. 167nMInEE6

P 69edlx,S..m h6 ipet7Mtd6c,bS.i xl skc J.S6xctl .cl M.)luAl.ct8Mrc em PMf cm
)l.MtMh2u0cprvp. IM.3ep ..TSlnex3 thJ,epo 2lsxtMT BIT 49 E MtW

1Pip ep7MtdTcax=c,p lp+1 5er hNetpS.cvp hActpe)l.mMtSlaxuocprln5ea lcit
h6epc2hl= 2tSl.24pS. h1E NIE6/

1Pip ep7N 1rPdActpe)l.mStSl. ax+pd lpf l.l.e (pJcfSl.iem ls 5er IM.celc lt
h6epc3crSmx SIAM J. Matr x Anal. Applic. 31, NtE/f @/t

(ItM.esS.rMiJ5ifltJct.pe. 1 e.luA,tem,certr.JxeIcrls,tSl. e. ocloteM.ca eAA,M.i
tJc 1.MtSbtSuetSTmcIn-E6n6Tc RAX x A = QR T p)HJcr netlp lnATA
rJlx,m e llue o6etMl. eAphc lpmSr.xrrMlh' rAcMnMtMrtMts.7ccc

i6u6htc8ep7ry1tP k9R .SctcpcetST1.3emu lpt9l.e= PetpS.o8StJe 4AA,M.etSl.
tl)l.MtSIRrtSuetlI rSIAM J. Numer. Anal. 17, / 1Ef tP

h6pet 1.7rt tP cl. fJolMmMfSla+cop lnLS.ceplc lt hJxel a(plo cur Me ucMiJtcm
l o3S2alpu7BIT 36h@ f nE 16

6hIMc.c.7 4.1 Sa na rHO jOn-M02 D0= a =e@+n0@ a 0= =n(xe= S00=fa. E
On.a =SIAM J. Matrix Anal. Applic. 19, tlu tNET

lpx pcstpMts1,clt rJxep3eAapl SuetMif e ltc eieS.rtus.MM-etSIS.ltJcp.lpurP
kJcp epct ..)l.r 8Jc St TMMred a.S.S ||Ax - b||_p L p = M ,m a = V au = = x
S = = xx = = a = = = = :: =

• n ri C..=1. a n av O a =,a a S=x-5x =Sx = x= x=a == =aa r x a xia
n . - = SIAM J. Numer. Anal. 13, @tEf lt6

56s6 9epcsr7).l.2P 56w6Jel eseuodlx 7MtdP 1),M.c.r .Mpc t PpJhM.i

1cpm3cpus.dM.cenbtcubS.tJclO On n20SIAM J. Numer. Anal. 15 Nrf@116
ik,)l cue em'T ls 7Mtd6c4 islo=s.empxempetS.es)l. dict 4.c hes.S.iPctJT m
vi LS.cepli X=n" 2Mathematical Pr gr mming 56, Series A, Mf INNw

IM/MfPc4,loes.)l.cptc PctJlmvpLp X= n 2SIAM J. Optim. 3, ultf uNtP
Je! 7Mtd6 4pSutxes .tcpS(lM.t4AApple.Jvp)luA,tSi tJclLi 2x lo O= < : = (= n
= 1> n = OJ.n = \$M(L Qe+) = 2Nk Optim. Theory Applic. 77, ENEfM6

StcpetMcaAluct MtJcsclt rJxepc1tcf f S mSr.xrrM

- Ota GSvi ot 1tE EA)aga 9y,Eia b4se ra 7Atos,le e)eas ira Ues Gvteg GSvsragN5
N mer. Math. 9, MET,M6y.
- 6 9r 51.ann,sl ils,o 7MtdcUtaletS3an, cntr,Snael,aert hJxeIahlsxtMro sl,raQ
JlsmalnrvI uetSlnrBT7, ENN (EI).
- 6 9r 51.5Mtdc tc letM manauantr,Sn qlt hJ,el arhl,xtST nbitr, N I NIy.
- 6x 9r 51.5Mtd. c tale 5ma13dt ln,Snael,aIt hJxH9h,xtMlnB-BT8 y,E1.
- 1 is, .9 8raeennt, hulat,nl8S.n 7Mtd. chs.Mrtja ,Snael,act hJ,elaS(lloa1 8Mt,tl.
SMi,baseS,axl tr T Computing 45, E6r ,EW 6
- 1: auua,r ! sMmemR.15ManzNtdveRf tler (la,Mral etS,5a,nauantVI 1j Imatal uMnam
,alt hJ,el ar(llosaur TCM T ans. Math. Sof w. 35, 41 tM,SN .
- kJa v,sl8Sni taf trtl aet.el Sl,r iauatl S,uetIMAI losa lr t9MlnMlnA,tal il eAJMerm
SrMjn
- 4h. iseSrAaMtd. An I tr duction to Ray Tracing PIIien Mexrueyn 9x sMngtP4:
5. sel tsu, emn4. Sbral uer7N16d, Multiple View Geometr in Computer Vision, ha.lnmRmMtMln
)euolMian Mal rMtar 2a8 II aw
- Pw (JellP. s,uA9yar 7M1dPhysically Based Rendering f om Theory to Implementation,
ha,T RmMtSllien Me,neyrn9xISnlhrP4
- Il en,ual M.eAa9rAatMaa,
- u. Me9en7Mlydwcl1AxMn)9lrrr (lImx tnc 5letMlnMNQnm r mSuandMl,Seen
hAearrJlA W888 r.dasa.amSvJenWPs M IW)9Amr

5.4 Other Orthogonal Factorizations

Spo A ∈ 1nx4 laatin Df orizat f le f logim

$$A = [a_1, a_2, a_3, a_4] = [q_1, q_2, q_3, q_4] \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Nelat ladesion hiderder
spor(A) = 1.0000000000000002
sa(1) = 0.8944271909999158
sa(2) = 0.4472135954999579
sa(3) = 0.2236067977499791
sa(4) = 0.1118033988749896
lens = 0.0000000000000000
cosine = 0.0000000000000000
dotprod = 0.0000000000000000
norm = 1.0000000000000002
sumsq = 1.0000000000000002
sumsq1 = 0.8944271909999158
sumsq2 = 0.4472135954999579
sumsq3 = 0.2236067977499791
sumsq4 = 0.1118033988749896
sumsq5 = 0.0000000000000000
sumsq6 = 0.0000000000000000
sumsq7 = 0.0000000000000000
sumsq8 = 0.0000000000000000
sumsq9 = 0.0000000000000000
sumsq10 = 0.0000000000000000
sumsq11 = 0.0000000000000000
sumsq12 = 0.0000000000000000
sumsq13 = 0.0000000000000000
sumsq14 = 0.0000000000000000
sumsq15 = 0.0000000000000000
sumsq16 = 0.0000000000000000
sumsq17 = 0.0000000000000000
sumsq18 = 0.0000000000000000
sumsq19 = 0.0000000000000000
sumsq20 = 0.0000000000000000
sumsq21 = 0.0000000000000000
sumsq22 = 0.0000000000000000
sumsq23 = 0.0000000000000000
sumsq24 = 0.0000000000000000
sumsq25 = 0.0000000000000000
sumsq26 = 0.0000000000000000
sumsq27 = 0.0000000000000000
sumsq28 = 0.0000000000000000
sumsq29 = 0.0000000000000000
sumsq30 = 0.0000000000000000
sumsq31 = 0.0000000000000000
sumsq32 = 0.0000000000000000
sumsq33 = 0.0000000000000000
sumsq34 = 0.0000000000000000
sumsq35 = 0.0000000000000000
sumsq36 = 0.0000000000000000
sumsq37 = 0.0000000000000000
sumsq38 = 0.0000000000000000
sumsq39 = 0.0000000000000000
sumsq40 = 0.0000000000000000
sumsq41 = 0.0000000000000000
sumsq42 = 0.0000000000000000
sumsq43 = 0.0000000000000000
sumsq44 = 0.0000000000000000
sumsq45 = 0.0000000000000000
sumsq46 = 0.0000000000000000
sumsq47 = 0.0000000000000000
sumsq48 = 0.0000000000000000
sumsq49 = 0.0000000000000000
sumsq50 = 0.0000000000000000
sumsq51 = 0.0000000000000000
sumsq52 = 0.0000000000000000
sumsq53 = 0.0000000000000000
sumsq54 = 0.0000000000000000
sumsq55 = 0.0000000000000000
sumsq56 = 0.0000000000000000
sumsq57 = 0.0000000000000000
sumsq58 = 0.0000000000000000
sumsq59 = 0.0000000000000000
sumsq60 = 0.0000000000000000
sumsq61 = 0.0000000000000000
sumsq62 = 0.0000000000000000
sumsq63 = 0.0000000000000000
sumsq64 = 0.0000000000000000
sumsq65 = 0.0000000000000000
sumsq66 = 0.0000000000000000
sumsq67 = 0.0000000000000000
sumsq68 = 0.0000000000000000
sumsq69 = 0.0000000000000000
sumsq70 = 0.0000000000000000
sumsq71 = 0.0000000000000000
sumsq72 = 0.0000000000000000
sumsq73 = 0.0000000000000000
sumsq74 = 0.0000000000000000
sumsq75 = 0.0000000000000000
sumsq76 = 0.0000000000000000
sumsq77 = 0.0000000000000000
sumsq78 = 0.0000000000000000
sumsq79 = 0.0000000000000000
sumsq80 = 0.0000000000000000
sumsq81 = 0.0000000000000000
sumsq82 = 0.0000000000000000
sumsq83 = 0.0000000000000000
sumsq84 = 0.0000000000000000
sumsq85 = 0.0000000000000000
sumsq86 = 0.0000000000000000
sumsq87 = 0.0000000000000000
sumsq88 = 0.0000000000000000
sumsq89 = 0.0000000000000000
sumsq90 = 0.0000000000000000
sumsq91 = 0.0000000000000000
sumsq92 = 0.0000000000000000
sumsq93 = 0.0000000000000000
sumsq94 = 0.0000000000000000
sumsq95 = 0.0000000000000000
sumsq96 = 0.0000000000000000
sumsq97 = 0.0000000000000000
sumsq98 = 0.0000000000000000
sumsq99 = 0.0000000000000000
sumsq100 = 0.0000000000000000
sumsq101 = 0.0000000000000000
sumsq102 = 0.0000000000000000
sumsq103 = 0.0000000000000000
sumsq104 = 0.0000000000000000
sumsq105 = 0.0000000000000000
sumsq106 = 0.0000000000000000
sumsq107 = 0.0000000000000000
sumsq108 = 0.0000000000000000
sumsq109 = 0.0000000000000000
sumsq110 = 0.0000000000000000
sumsq111 = 0.0000000000000000
sumsq112 = 0.0000000000000000
sumsq113 = 0.0000000000000000
sumsq114 = 0.0000000000000000
sumsq115 = 0.0000000000000000
sumsq116 = 0.0000000000000000
sumsq117 = 0.0000000000000000
sumsq118 = 0.0000000000000000
sumsq119 = 0.0000000000000000
sumsq120 = 0.0000000000000000
sumsq121 = 0.0000000000000000
sumsq122 = 0.0000000000000000
sumsq123 = 0.0000000000000000
sumsq124 = 0.0000000000000000
sumsq125 = 0.0000000000000000
sumsq126 = 0.0000000000000000
sumsq127 = 0.0000000000000000
sumsq128 = 0.0000000000000000
sumsq129 = 0.0000000000000000
sumsq130 = 0.0000000000000000
sumsq131 = 0.0000000000000000
sumsq132 = 0.0000000000000000
sumsq133 = 0.0000000000000000
sumsq134 = 0.0000000000000000
sumsq135 = 0.0000000000000000
sumsq136 = 0.0000000000000000
sumsq137 = 0.0000000000000000
sumsq138 = 0.0000000000000000
sumsq139 = 0.0000000000000000
sumsq140 = 0.0000000000000000
sumsq141 = 0.0000000000000000
sumsq142 = 0.0000000000000000
sumsq143 = 0.0000000000000000
sumsq144 = 0.0000000000000000
sumsq145 = 0.0000000000000000
sumsq146 = 0.0000000000000000
sumsq147 = 0.0000000000000000
sumsq148 = 0.0000000000000000
sumsq149 = 0.0000000000000000
sumsq150 = 0.0000000000000000
sumsq151 = 0.0000000000000000
sumsq152 = 0.0000000000000000
sumsq153 = 0.0000000000000000
sumsq154 = 0.0000000000000000
sumsq155 = 0.0000000000000000
sumsq156 = 0.0000000000000000
sumsq157 = 0.0000000000000000
sumsq158 = 0.0000000000000000
sumsq159 = 0.0000000000000000
sumsq160 = 0.0000000000000000
sumsq161 = 0.0000000000000000
sumsq162 = 0.0000000000000000
sumsq163 = 0.0000000000000000
sumsq164 = 0.0000000000000000
sumsq165 = 0.0000000000000000
sumsq166 = 0.0000000000000000
sumsq167 = 0.0000000000000000
sumsq168 = 0.0000000000000000
sumsq169 = 0.0000000000000000
sumsq170 = 0.0000000000000000
sumsq171 = 0.0000000000000000
sumsq172 = 0.0000000000000000
sumsq173 = 0.0000000000000000
sumsq174 = 0.0000000000000000
sumsq175 = 0.0000000000000000
sumsq176 = 0.0000000000000000
sumsq177 = 0.0000000000000000
sumsq178 = 0.0000000000000000
sumsq179 = 0.0000000000000000
sumsq180 = 0.0000000000000000
sumsq181 = 0.0000000000000000
sumsq182 = 0.0000000000000000
sumsq183 = 0.0000000000000000
sumsq184 = 0.0000000000000000
sumsq185 = 0.0000000000000000
sumsq186 = 0.0000000000000000
sumsq187 = 0.0000000000000000
sumsq188 = 0.0000000000000000
sumsq189 = 0.0000000000000000
sumsq190 = 0.0000000000000000
sumsq191 = 0.0000000000000000
sumsq192 = 0.0000000000000000
sumsq193 = 0.0000000000000000
sumsq194 = 0.0000000000000000
sumsq195 = 0.0000000000000000
sumsq196 = 0.0000000000000000
sumsq197 = 0.0000000000000000
sumsq198 = 0.0000000000000000
sumsq199 = 0.0000000000000000
sumsq200 = 0.0000000000000000
sumsq201 = 0.0000000000000000
sumsq202 = 0.0000000000000000
sumsq203 = 0.0000000000000000
sumsq204 = 0.0000000000000000
sumsq205 = 0.0000000000000000
sumsq206 = 0.0000000000000000
sumsq207 = 0.0000000000000000
sumsq208 = 0.0000000000000000
sumsq209 = 0.0000000000000000
sumsq210 = 0.0000000000000000
sumsq211 = 0.0000000000000000
sumsq212 = 0.0000000000000000
sumsq213 = 0.0000000000000000
sumsq214 = 0.0000000000000000
sumsq215 = 0.0000000000000000
sumsq216 = 0.0000000000000000
sumsq217 = 0.0000000000000000
sumsq218 = 0.0000000000000000
sumsq219 = 0.0000000000000000
sumsq220 = 0.0000000000000000
sumsq221 = 0.0000000000000000
sumsq222 = 0.0000000000000000
sumsq223 = 0.0000000000000000
sumsq224 = 0.0000000000000000
sumsq225 = 0.0000000000000000
sumsq226 = 0.0000000000000000
sumsq227 = 0.0000000000000000
sumsq228 = 0.0000000000000000
sumsq229 = 0.0000000000000000
sumsq230 = 0.0000000000000000
sumsq231 = 0.0000000000000000
sumsq232 = 0.0000000000000000
sumsq233 = 0.0000000000000000
sumsq234 = 0.0000000000000000
sumsq235 = 0.0000000000000000
sumsq236 = 0.0000000000000000
sumsq237 = 0.0000000000000000
sumsq238 = 0.0000000000000000
sumsq239 = 0.0000000000000000
sumsq240 = 0.0000000000000000
sumsq241 = 0.0000000000000000
sumsq242 = 0.0000000000000000
sumsq243 = 0.0000000000000000
sumsq244 = 0.0000000000000000
sumsq245 = 0.0000000000000000
sumsq246 = 0.0000000000000000
sumsq247 = 0.0000000000000000
sumsq248 = 0.0000000000000000
sumsq249 = 0.0000000000000000
sumsq250 = 0.0000000000000000
sumsq251 = 0.0000000000000000
sumsq252 = 0.0000000000000000
sumsq253 = 0.0000000000000000
sumsq254 = 0.0000000000000000
sumsq255 = 0.0000000000000000
sumsq256 = 0.0000000000000000
sumsq257 = 0.0000000000000000
sumsq258 = 0.0000000000000000
sumsq259 = 0.0000000000000000
sumsq260 = 0.0000000000000000
sumsq261 = 0.0000000000000000
sumsq262 = 0.0000000000000000
sumsq263 = 0.0000000000000000
sumsq264 = 0.0000000000000000
sumsq265 = 0.0000000000000000
sumsq266 = 0.0000000000000000
sumsq267 = 0.0000000000000000
sumsq268 = 0.0000000000000000
sumsq269 = 0.0000000000000000
sumsq270 = 0.0000000000000000
sumsq271 = 0.0000000000000000
sumsq272 = 0.0000000000000000
sumsq273 = 0.0000000000000000
sumsq274 = 0.0000000000000000
sumsq275 = 0.0000000000000000
sumsq276 = 0.0000000000000000
sumsq277 = 0.0000000000000000
sumsq278 = 0.0000000000000000
sumsq279 = 0.0000000000000000
sumsq280 = 0.0000000000000000
sumsq281 = 0.0000000000000000
sumsq282 = 0.0000000000000000
sumsq283 = 0.0000000000000000
sumsq284 = 0.0000000000000000
sumsq285 = 0.0000000000000000
sumsq286 = 0.0000000000000000
sumsq287 = 0.0000000000000000
sumsq288 = 0.0000000000000000
sumsq289 = 0.0000000000000000
sumsq290 = 0.0000000000000000
sumsq291 = 0.0000000000000000
sumsq292 = 0.0000000000000000
sumsq293 = 0.0000000000000000
sumsq294 = 0.0000000000000000
sumsq295 = 0.0000000000000000
sumsq296 = 0.0000000000000000
sumsq297 = 0.0000000000000000
sumsq298 = 0.0000000000000000
sumsq299 = 0.0000000000000000
sumsq300 = 0.0000000000000000
sumsq301 = 0.0000000000000000
sumsq302 = 0.0000000000000000
sumsq303 = 0.0000000000000000
sumsq304 = 0.0000000000000000
sumsq305 = 0.0000000000000000
sumsq306 = 0.0000000000000000
sumsq307 = 0.0000000000000000
sumsq308 = 0.0000000000000000
sumsq309 = 0.0000000000000000
sumsq310 = 0.0000000000000000
sumsq311 = 0.0000000000000000
sumsq312 = 0.0000000000000000
sumsq313 = 0.0000000000000000
sumsq314 = 0.0000000000000000
sumsq315 = 0.0000000000000000
sumsq316 = 0.0000000000000000
sumsq317 = 0.0000000000000000
sumsq318 = 0.0000000000000000
sumsq319 = 0.0000000000000000
sumsq320 = 0.0000000000000000
sumsq321 = 0.0000000000000000
sumsq322 = 0.0000000000000000
sumsq323 = 0.0000000000000000
sumsq324 = 0.0000000000000000
sumsq325 = 0.0000000000000000
sumsq326 = 0.0000000000000000
sumsq327 = 0.0000000000000000
sumsq328 = 0.0000000000000000
sumsq329 = 0.0000000000000000
sumsq330 = 0.0000000000000000
sumsq331 = 0.0000000000000000
sumsq332 = 0.0000000000000000
sumsq333 = 0.0000000000000000
sumsq334 = 0.0000000000000000
sumsq335 = 0.0000000000000000
sumsq336 = 0.0000000000000000
sumsq337 = 0.0000000000000000
sumsq338 = 0.0000000000000000
sumsq339 = 0.0000000000000000
sumsq340 = 0.0000000000000000
sumsq341 = 0.0000000000000000
sumsq342 = 0.0000000000000000
sumsq343 = 0.0000000000000000
sumsq344 = 0.0000000000000000
sumsq345 = 0.0000000000000000
sumsq346 = 0.0000000000000000
sumsq347 = 0.0000000000000000
sumsq348 = 0.0000000000000000
sumsq349 = 0.0000000000000000
sumsq350 = 0.0000000000000000
sumsq351 = 0.0000000000000000
sumsq352 = 0.0000000000000000
sumsq353 = 0.0000000000000000
sumsq354 = 0.0000000000000000
sumsq355 = 0.0000000000000000
sumsq356 = 0.0000000000000000
sumsq357 = 0.0000000000000000
sumsq358 = 0.0000000000000000
sumsq359 = 0.0000000000000000
sumsq360 = 0.0000000000000000
sumsq361 = 0.0000000000000000
sumsq362 = 0.0000000000000000
sumsq363 = 0.0000000000000000
sumsq364 = 0.0000000000000000
sumsq365 = 0.0000000000000000
sumsq366 = 0.0000000000000000
sumsq367 = 0.0000000000000000
sumsq368 = 0.0000000000000000
sumsq369 = 0.0000000000000000
sumsq370 = 0.0000000000000000
sumsq371 = 0.0000000000000000
sumsq372 = 0.0000000000000000
sumsq373 = 0.0000000000000000
sumsq374 = 0.0000000000000000
sumsq375 = 0.0000000000000000
sumsq376 = 0.0000000000000000
sumsq377 = 0.0000000000000000
sumsq378 = 0.0000000000000000
sumsq379 = 0.0000000000000000
sumsq380 = 0.0000000000000000
sumsq381 = 0.0000000000000000
sumsq382 = 0.0000000000000000
sumsq383 = 0.0000000000000000
sumsq384 = 0.0000000000000000
sumsq385 = 0.0000000000000000
sumsq386 = 0.0000000000000000
sumsq387 = 0.0000000000000000
sumsq388 = 0.0000000000000000
sumsq389 = 0.0000000000000000
sumsq390 = 0.0000000000000000
sumsq391 = 0.0000000000000000
sumsq392 = 0.0000000000000000
sumsq393 = 0.0000000000000000
sumsq394 = 0.0000000000000000
sumsq395 = 0.0000000000000000
sumsq396 = 0.0000000000000000
sumsq397 = 0.0000000000000000
sumsq398 = 0.0000000000000000
sumsq399 = 0.0000000000000000
sumsq400 = 0.0000000000000000
sumsq401 = 0.0000000000000000
sumsq402 = 0.0000000000000000
sumsq403 = 0.0000000000000000
sumsq404 = 0.0000000000000000
sumsq405 = 0.0000000000000000
sumsq406 = 0.0000000000000000
sumsq407 = 0.0000000000000000
sumsq408 = 0.0000000000000000
sumsq409 = 0.0000000000000000
sumsq410 = 0.0000000000000000
sumsq411 = 0.0000000000000000
sumsq412 = 0.0000000000000000
sumsq413 = 0.0000000000000000
sumsq414 = 0.0

which is often difficult in floating-point arithmetic because the condition number of some vectors can be quite large.

5.4.1 Numerical Rank and the SVD

Since $A \in \mathbb{R}^{m \times n}$ has $\text{SD}(A^T A) = E = \text{diag}(\sigma_1^2, \dots, \sigma_r^2)$, if $\text{rank}(A) = r < n$, then $A^T A$ is not invertible, so A is not invertible.

$$A = L \text{ diag}(\sigma_1, \dots, \sigma_r) V^T \quad (5.4.1)$$

The singular values of A are the square roots of the eigenvalues of $A^T A$. These vectors are obtained from a single SVD if A is full rank, or by singular value iteration if A is not full rank.

$$U^T A V = \text{diag}(\hat{\sigma}_1, \dots, \hat{\sigma}_r), \quad \hat{\sigma}_1 \geq \dots \geq \hat{\sigma}_n \geq 0.$$

The SVD also allows us to uniformly scale and rotate a vector space. This is very useful for tasks such as dimensionality reduction, feature extraction, and classification. It can also be used to calculate the condition number of a matrix.

$$A = \sum_{i=1}^n \hat{\sigma}_i \hat{u}_i \hat{v}_i^T \quad (5.4.2)$$

The numerical rank of A is the number of non-zero singular values of A . This is an approximation of the rank of A obtained from the SVD.

$$A = \sum_{i=1}^{\hat{r}} \hat{\sigma}_i \hat{u}_i \hat{v}_i^T, \quad \hat{r} \leq \hat{n} \quad (5.4.3)$$

where \hat{r} is the numerical rank. This approximation is called the truncated SVD.

$$\hat{U} = W + \Delta U, \quad W^T W = I_m, \quad \|U\|_2 \leq f,$$

$$V = Z + \Delta Z, \quad Z^T Z = I_n, \quad \|V\|_2 \leq f, \quad (5.4.4)$$

$$\hat{\Sigma} = W \hat{U} (\hat{U}^T \hat{U})^{-1} Z^T, \quad \|A\|_2 \leq \epsilon \|A\|_2,$$

This is a simplified diagram illustrating the SVD approximation of a matrix A .

Note that f and V aren't necessarily close to their exact counterparts. However, we can show that J_N is close to σ_k as follows. Using Corollary 2.4.6 we have

$$\sigma_k = \min_{\text{rank}(B)=k-1} \|A - B\|_2 = \min_{\text{rank}(B)=k-1} \|(\hat{\Sigma} - B) - E\|_2$$

where

$$E = W^T(\Delta A)Z$$

and

$$\|E\|_2 \leq \epsilon \|A\|_2 = \epsilon \sigma_1.$$

Since

$$\|\hat{\Sigma} - B\| - \|E\| \leq \|\hat{\Sigma} - B\| \leq \|\hat{\Sigma} - B\| + \|E\|$$

and

$$\min_{\text{rank}(B)=k-1} \|\hat{\Sigma}_k - B\|_2 = \hat{\sigma}_k,$$

it follows that

$$|\sigma_k - \hat{\sigma}_k| \leq \epsilon \sigma_1$$

If $r = 1, n$. Thus, if A has rank r , then we can expect $n - r$ of the computed singular values to be small. Near rank deficiency in A cannot escape detection if the SVD of A is computed.

Of course, all this hinges on having a definition of "small." This amounts to choosing a tolerance $\delta > 0$ and declaring A to have numerical rank r if the computed singular values satisfy

$$\hat{\sigma}_1 \geq \dots \geq \hat{\sigma}_{\hat{r}} > \delta \geq \hat{\sigma}_{\hat{r}+1} \geq \dots \geq \hat{\sigma}_n. \quad (5.4.5)$$

We refer to the integer r as the *rank* of A . The tolerance should be consistent with the machine precision, e.g., $\delta = \epsilon \|A\|_\infty$. However, if the general level of relative error in the data is larger than ϵ , then δ should be correspondingly bigger, e.g., $\delta = 10^{-2} \|A\|_\infty$ if the entries in A are correct to two digits.

For a given δ it is important to stress that, although the SVD provides a great deal of rank-related insight, it does not guarantee that the determination of numerical rank is a sensitive computation. If the gap between $\hat{\sigma}_r$ and $\hat{\sigma}_{r+1}$ is small, then A is also close (in the $\|\cdot\|_\infty$ sense) to a matrix with rank $r - 1$. Thus, the amount of confidence we have in the correctness of A and in how we proceed to use the approximation (5.4.2) depends on the gap between $\hat{\sigma}_r$ and $\hat{\sigma}_{r+1}$.

5.4.2 QR with Column Pivoting

We now examine alternative rank-revealing strategies to the SVD starting with a modification of the Householder QR factorization procedure (Algorithm 5.2.1). In exact arithmetic, the modified algorithm computes the factorization

$$Q^T A \Pi = \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix}_{m-r}^r \quad (5.4.6)$$

where $r = \text{rank}(A)$, Q is orthogonal, R_{11} is upper triangular and nonsingular, and I is a permutation. If we have the column partitions $A = [a_{c_1} \ I \cdots I a_{c_n}]$ and $Q = [Q \ I_1 \cdots I_q]$, then for $k = 1, n$ we have

$$a_{c_k} = \sum_{l=1}^{\min\{k, r\}} T_{lk} q_l \in \text{span}\{q_1, \dots, q_r\}$$

implying

$$\text{rank}(A) = \text{span}\{q_1, \dots, q_r\}.$$

To see how to compute such a factorization, assume for some k that we have computed Householder matrices H_1, \dots, H_{k-1} and permutations I_1, \dots, I_{k-1} such that

$$(H_{k-1} \cdots H_1)A(\Pi_1 \cdots \Pi_{k-1}) = R^{(k-1)} = \begin{bmatrix} R^{-1} & R^{-1} \\ 0 & R^{-1} \end{bmatrix}_{k \times k} \quad (5.4.7)$$

where R^{-1} is a nonsingular and upper triangular matrix. Now suppose that

$$R^{(k-1)} = [I_{k-1} \ I_1 \cdots I_{n-k}]$$

is a column partitioning and let $p = k$ be the smallest index such that

$$I^{(k-1)} = \{I_{k-1}, \dots, I_{n-p+1}\}. \quad (5.4.8)$$

Note that if $\text{rank}(A) < k-1$, then this maximum is zero and we are finished. Otherwise, let I_k be the n -by- n identity with columns p and k interchanged and determine a Householder matrix H_k such that if

$$R^{(k)} = H_k R^{(k-1)} \Pi_k,$$

then $R^{(k)}_{j,k+1:m,k} = 0$. In other words, I_k moves the largest column in R^{-1} to the lead position and H_k zeroes all of its subdiagonal components.

The column norms do not have to be recomputed at each stage if we exploit the property

$$Q^T Z = \begin{bmatrix} Q \\ w \end{bmatrix}_1 \implies I_w \mathbf{I} = I_Z \mathbf{I} - \alpha^2,$$

which holds for any orthogonal matrix $Q \in \mathbb{R}^{n \times n}$. This reduces the overhead associated with column pivoting from $O(nm^2)$ fops to $O(m)$ fops because we can get the new column norms by updating the old column norms, e.g.,

$$I_{k-1}^T I_{k-1}^2 - I_k^T I_k^2 = I_{k-1}^T I_{k-1}^2 \quad j = k+1:n$$

Combining all of the above we obtain the following algorithm first presented by Businger and Golub (1965):

nkE.TiQ 5.4.1 ef 67.916f.91p 25p C4.lp 16a 79Ip f4R6:I.rp Given $A \in \mathbb{R}^{m \times n}$ with $m \geq n$, the following algorithm computes $r = \text{rank}(A)$ and the factorization (5.4.6) with $Q = H_1 \cdots H_r$ and $I = I_1 \cdots I_r$. The upper triangular part of A is overwritten by the upper triangular part of R and components $j+1:m$ of the j th Householder vector are stored in $A(j+1:m, j)$. The permutation I is encoded in an integer vector piv . In particular, I_j is the identity with rows j and $\text{piv}(j)$ interchanged.

```

for r j = 1:n
    c(j) = A(1:mj)TA(1:mj)
end
r = 0
tau = max{c(1), ..., c(n)}
while tau > 0 and r < n
    r = r + 1
    Find smallest k with r : k: n such that c(k) = 0
    piv(r) = k
    A(1:m, r) = A(1:m, k)
    c(r) = c(k)
    [v, ] = house(A(r:m, r))
    A(r:m, r:n) = (I_{m-r} - v*v^T)A(r:m, r:n)
    A(r+1:m, r) = v(2m-r+1)
    for i = r+1:n
        c(i) = c(i) - A(r, i)*2
    end
    tau = max{c(r+1), ..., c(n)}
end

```

This algorithm requires $4mr - 2r^2(m+n) + 4\frac{3}{2}n^2$ fops where $r = \text{rank}(A)$.

5.4.3 Numerical Rank and AI I(+)

In principle, QR with column pivoting reveals rank. But how informative is the method in the context of floating point arithmetic? After k steps we have

$$\text{fl}(H_k \cdots H_1 A \Pi_1 \cdots \Pi_k) = \widehat{R}^{(k)} = \begin{bmatrix} R_1^{(k)} & R_2^{(k)} \\ 0 & R_{22}^{(k)} \end{bmatrix}_{m \times k}. \quad (5.4.9)$$

If W is suitably small in norm then it is reasonable to terminate the reduction and declare A to have rank k . A typical termination criteria might be

$$\| \widehat{R}_{22}^{(k)} \|_2 \leq \epsilon_1 \| A \|_2$$

for some small machine-dependent parameter ϵ . In view of the roundoff properties associated with Householder matrix computation (cf. §5.1.12), we know that $\hat{R}^{(k)}$ is the exact R-factor of a matrix $A + E_k$ where

$$\|E_k\|_2 \leq \epsilon_2 \|A\|_2, \quad \mathcal{Q} = O(\mathbf{u}).$$

Using Corollary 244 we have

$$\mathbf{Q}_{k+1}(A + E_k) = \mathbf{Q}_{k+1}(\hat{R}^{(k)}) \quad \|\hat{R}_{22}^{(k)}\|_2$$

Since $\mathbf{Q}_{k+1}(A) = \mathbf{Q}_{k+1}(A + E_k) + \mathbf{E}_k$ it follows that

$$\sigma_{k+1}(A) \leq (\epsilon_1 + \epsilon_2) \|A\|_2.$$

In other words, a relative perturbation of $O(\epsilon_1 + \epsilon_2)$ in A can yield a rank- k matrix. With this termination criterion, we conclude that QR with column pivoting discovers rank deficiency if $\hat{R}_{22}^{(k)}$ is small for some $k \leq n$. However, it does not follow that the matrix $\hat{R}_{22}^{(k)}$ in (549) is small if $\text{rank}(A) = k$. There are examples of nearly rank deficient matrices whose R-factor look perfectly "normal." A famous example is the Kahan matrix

$$\text{Kahn}(s) = \text{diag}(1, s, \dots, s^n) \begin{bmatrix} 1 & -1 & -1 & \cdots & -1 \\ 0 & 1 & -1 & \cdots & 1,1 \\ & \ddots & & \vdots & \vdots \\ \vdots & & & 1 & -1 \\ 0 & \cdots & & & 1 \end{bmatrix}.$$

Here, $2 + s^2 = 1$ with $s > 0$. (See Lawson and Hanson (SLS, p. 31).) These matrices are unaltered by Algorithm 54.1 and thus $\|\hat{R}_{22}^{(k)}\|_2 \approx s^n$ for $k = 1, \dots, n-1$. This inequality implies (for example) that the matrix Kahn(0.99) has no particularly small trailing principal submatrix since $s^{29} \approx 0.05$. However, a calculation shows that $\hat{R}_{22}^{(29)} = O(10^{-19})$.

Nevertheless, in practice, small trailing R-submatrices almost always emerge that correlate well with the underlying rank. In other words, it is almost always the case that $\hat{R}_{22}^{(k)}$ is small if A has rank k .

5.4.4 Finding a Good Column Ordering

It is important to appreciate that Algorithm 54.1 is just one way to determine the column permutation \mathbf{I} . The following result sets

\$G\mathbf{B}\mathbf{A}, R

Pr 6 Suppose $I \in \mathbb{J}^{n \times n}$ is a permutation such that if $w = I^T v$, then

$$\|w\| = \max \|M_i\|.$$

Since V_i is the largest component of a unit 2normvector, $\|V_i\| = 1/\sqrt{n}$. If $AII = QR$ is a QR factorization, then

$$u = \|Av\|_2 = \|Q(RA)I(I^T v)\|_2 = \|R(I_n I_n)w\|_2 = \|R\|_2 \|I_n\|_2 \|M_i\|_2 \|v\|_2. \quad i \in p$$

Note that if $v = V_i$ is the right singular vector corresponding to $\sigma_{ii} c_i$, then $\|u\| = \sqrt{n}\sigma_{ii}$. This suggests a framework whereby the column permutation matrix I is based on an estimate of V_i .

Step 1 Compute the QR factorization $A = Qd$ and note that R has the same right singular vectors as A .

Step 2 Use condition estimation techniques to obtain a unit vector v with $\|Av\|_2 = 1$.

Step 3 Determine I and the QR factorization $AII = QR$.

See Chan (1987) for details about this approach to rank determination. The permutation I can be generated as a sequence of swap permutations. This supports a very economical Givens rotation method for generating Q and R from Qd and R .

5.4.5 More General Rank-Revealing Decompositions

Additional rank-revealing strategies emerge if we allow general orthogonal recombinations of the A 's columns instead of just permutations. That is, we look for an orthogonal Z so that the QR factorization

$$AZ = QR$$

produces a rank-revealing R . To impart the spirit of this type of matrix reduction we show how the rank-revealing properties of a given $AZ = QR$ factorization can be improved by replacing Z , Q and R with

$$Z_{\text{new}} = ZZ_G, \quad Q_{\text{new}} = QQ_G, \quad R_{\text{new}} = Q_G^T R Z_G,$$

respectively, where Q_G and Z_G are products of Givens rotations and R is upper triangular. The rotations are generated by introducing zeros into a unit 2norm vector V which we assume approximates the n -th right singular vector of AZ . In particular, if $Z = v = I_n(:, n)$ and $\|Rv\|_2 = 1$, then

$$\|Rv\|_2 = \|Q R z_{\text{new}}\|_2 = \|Q R v\|_2 = \|Rv\|_2 = 1$$

This says that the norm of the last column of R is approximately the smallest singular value of A , which is certainly one way to reveal the underlying matrix rank.

We use the case $n = 4$ to illustrate how the Givens rotations arise and why the overall process is economical. Because we are transforming V to e_4 and not e_1 , we need to "flip" the notion of the 2by-2 rotations in the Z_G computations so that top components are zeroed, i.e.,

$$\begin{bmatrix} & \\ & \\ & \ddots \\ & : \end{bmatrix} = \begin{bmatrix} & \\ & \\ \ddots & \\ & : \end{bmatrix} \begin{bmatrix} x \\ \vdots \\ x \\ \vdots \end{bmatrix}.$$

This requires only a slight modification of Algorithm 5.1.3.

In the $n = 4$ case we start with

$$R = \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix} \quad v = \begin{bmatrix} \times \\ \times \\ \times \\ \times \end{bmatrix}$$

and proceed to compute

$$Z_G = G_{12}G_{23}G_{34}$$

and

$$Q_G = H_{12}H_{23}H_{34}$$

a products of Givens rotations. The first step is to zero the top component of v with a "fipped" (1,2) rotation and update R accordingly:

$$R \leftarrow RG_{12} = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix}, \quad v \leftarrow G_{12}^T v = \begin{bmatrix} 0 \\ \times \\ \times \\ \times \end{bmatrix}.$$

To remove the unwanted subdiagonal in R , we apply a conventional (nonfipped) Givens rotation from the left to R (but not v):

$$R \leftarrow H_{12}^T R = \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix}, \quad v = \begin{bmatrix} 0 \\ \times \\ \times \\ \times \end{bmatrix}.$$

The next step is analogous

$$R \leftarrow RG_{23} = \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix}, \quad v \leftarrow G_{23}^T v = \begin{bmatrix} 0 \\ 0 \\ \times \\ \times \end{bmatrix}.$$

$$R \leftarrow H_{23}^T R = \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix}, \quad v = \begin{bmatrix} 0 \\ 0 \\ \times \\ \times \end{bmatrix}.$$

And finally,

$$R \leftarrow RG_{34} = \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}, \quad v = G_{34}^T v = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \times \end{bmatrix},$$

$$R \leftarrow H_{34}^T R = \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \end{bmatrix}, \quad v = \begin{bmatrix} 0 \\ 0 \\ 0 \\ x \end{bmatrix}.$$

The pattern is clear; for $i = 1:n-1$, a $G_{i,i+1}$ is used to zero the current v_i and an $H_{i,i+1}$ is used to zero the current $r_{i+1,i}$. The overall transition from $\{Q, Z, R\}$ to $\{Q_{\text{new}}, Z_{\text{new}}, R_{\text{new}}\}$ involves $O(mn)$ flops. If the Givens rotations are kept in factored form, this flop count is reduced to $O(mn^2)$. We mention that the ideas in this subsection can be iterated to develop matrix reductions that expose the structure of matrices whose rank is less than $n-1$. "Zero chasing" with Givens rotations is at the heart of many important matrix algorithms, see §6.3, §7.5, and §8.3.

5.4.6 The Framework

As mentioned at the start of this section, we are interested in factorizations that are cheaper than the SVD but which provide the same high quality information about rank, range, and nullspace. Factorizations of this type are referred to as UV factorizations where the "T" stands for triangular and the "U" and "V" remind us of the SVD and orthogonal U and V matrices of singular vectors.

The matrix T can be upper triangular (these are the URV factorizations) or lower triangular (these are the ULV factorizations). It turns out that in a particular application one may favor a URV approach over a ULV approach, see §6.3. Moreover, the two partitions have different approximation properties. For example, suppose $\|S - \sigma_n(A)\| > \|A\|$ and S is the subspace spanned by A's right singular vectors $\{v_1, \dots, v_n\}$. Think of S as an approximate nullspace of A . Following Stewart (1993), if

$$UTAV = R = \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & \mathbf{R} & \\ & & & \mathbf{K} \end{bmatrix}$$

and $V = [V_1 | V_2]$ is partitioned conformably, then

$$\text{dist}(r_{11}), S) \leq \frac{\|R_{21}\|}{(1 - p_n^m)\|R_{11}\|} \quad (5.4.10)$$

where

$$p_n^m = \frac{\|R_{21}\|}{\|R_{11}\|}$$

is assumed to be less than 1. On the other hand, in the ULV setting we have

$$UTAV = L = \begin{bmatrix} L_1 & 0 & & \\ & L_2 & & \\ & & \mathbf{L} & \\ & & & \mathbf{K} \end{bmatrix}$$

If $V = [V_1 | V_2]$ is partitioned conformably, then

$$\text{dist}(\text{ran}(V_2), S) \leq \frac{\|V_2\|_2}{(1 - \frac{r}{n}) \min(L_{11})} \quad (54.11)$$

where

$$\nu = \frac{\|V_2\|_2}{\sigma_{\min}(L_{11})}$$

is also assumed to be less than 1. However, in practice the pfactors in both (54.10) and (54.11) are often much less than 1. Observe that when this is the case, the upper bound in (54.11) is much smaller than the upper bound in (54.10).

5.4.7 Complete Orthogonal Decompositions

Related to the UTV framework is the idea of a complete orthogonal factorization. Here we compute orthogonal U and V such that

$$UTAV = \begin{bmatrix} T_{11} & 0 \\ 0 & 0 \end{bmatrix}_{r \times n-r}^r \quad (54.12)$$

where $r = \text{rank}(A)$. The SVD is obviously an example of a decomposition that has this structure. However, a cheaper, two-step QR process is also possible. We first use Algorithm 54.1 to compute

$$U^T A II = \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix}_{r \times n-r}^r$$

and then follow up with a second QR factorization

$$Q^T \begin{bmatrix} R_{11}^T \\ R_{12}^T \end{bmatrix} = \begin{bmatrix} S_1 \\ 0 \end{bmatrix}$$

via Algorithm 52.1. If we set $V = IIQ$, then (54.12) is realized with $Tu = S$. Note that two important subspaces are defined by selected columns of $U = [u_1 | \dots | u_m]$ and $V = [v_1 | \dots | v_n]$:

$$\text{ran}(A) = \text{span}\{u_1, \dots, u_r\},$$

$$\text{null}(A) = \text{span}\{v_{r+1}, \dots, v_n\}.$$

Of course, the computation of a complete orthogonal decomposition in practice would require the careful handling of numerical rank.

5.4.8 Bidiagonalization

There is one other two-sided orthogonal factorization that is important to discuss and that is the bidiagonal factorization. It is not a rank-revealing factorization per se, but it has a useful role to play because it rivals the SVD in terms of data compression.

Suppose $A \in \mathbb{R}^{m \times n}$ and $m \geq n$. The idea is to compute orthogonal $U_n \in \mathbb{R}^{m \times m}$ and $V_2 \in \mathbb{R}^{n \times n}$ such that

$$U_n A V_2 = \begin{bmatrix} d_1 & j_1 & 0 & \cdots & 0 \\ 0 & d_2 & h & & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & d_{n-1} & f_{n-1} & 1 \\ 0 & \cdots & 0 & d_n & \end{bmatrix}. \quad (5.4.13)$$

-S

$U_n = U_1 \cdots U_n$ and $V_2 = V_1 \cdots V_n$ can each be determined as a product of Householder matrices, e.g.,

$$\begin{bmatrix} & x & x & x \\ & x & x & x \end{bmatrix} \xrightarrow{U_1} \begin{bmatrix} x & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{V_1}$$

$$\begin{bmatrix} x & x & 0 & 0 \\ 0 & x & x & x \end{bmatrix} \xrightarrow{U_2} \begin{bmatrix} x & x & 0 & 0 \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & x & x \\ 0 & 0 & x & x \end{bmatrix} \xrightarrow{V_2}$$

$$\begin{bmatrix} x & x & 0 & 0 \\ 0 & x & x & 0 \\ 0 & 0 & x & x \\ 0 & 0 & x & x \\ 0 & 0 & x & x \end{bmatrix} \xrightarrow{U_3} \begin{bmatrix} x & x & 0 & 0 \\ 0 & x & x & 0 \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \\ 0 & 0 & 0 & x \end{bmatrix} \xrightarrow{U_4} \begin{bmatrix} x & x & 0 & 0 \\ 0 & x & x & 0 \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

In general, U_k introduces zeros into the k th column, while V_k zeros the appropriate entries in row k . Overall we have

```

for j = 1:n
    [v, ] = house(A(j:mj))
    A(j:mj:n) = Um;+1- , wT)A(j:mj:n)
    A(j + 1:mj) = v(2m- j + 1)
    if j < n - 2
        [v, β] = house(A(j, j + 1:n)T)
        A(j:mj + 1:n) = A(j:mj + 1:n)(In, j - , wT)
        A(j, j + 2:n) = v(2n- j)T
    end
end

```

This algorithm requires $4m^2 - 4n^3/3$ fops. Such a technique is used by Golub and Kahan (1965), where bidiagonalization is first described. If the matrices U_8 and V_8 are explicitly desired, then they can be accumulated in $4m^2n - 4n^3/3$ and $4n^3/3$ fops, respectively. The bidiagonalization of A is related to the tridiagonalization of ATA . See §8.3.1.

5.4.9 l U wr r r

If $m \gg n$, then a faster method of bidiagonalization method results if we upper triangularize A first before applying Algorithm 54.2. In particular, suppose we compute an orthogonal $Q \in \mathbb{R}^{mn}$ such that

$$Q^T A = \begin{bmatrix} R_1 \\ 0 \end{bmatrix}$$

is upper triangular. We then bidiagonalize the square matrix R_1 ,

$$U_r^T R_1 V_B = B_1,$$

where U_h and V_h are orthogonal. If $U_8 = Q \text{diag}(U_h I_{m,n})$, then

$$U^T A V = \begin{bmatrix} B_1 \\ 0 \end{bmatrix} \equiv B$$

is a bidiagonalization of A .

The idea of computing the bidiagonalization in this manner is mentioned by Lawson and Hanson (SLS, p. 119) and more fully analyzed by Chan (1982). We refer to this method a R-bidiagonalization and it requires $(2m^2 + 2n^3)$ fops. This is less than the flop count for Algorithm 54.2 whenever $m > 5n/3$.

Problems

P5.4.1 5. $x, y \in \mathbb{R}^{m \times n}, m > n$

$$Q^T x = \begin{bmatrix} q \\ u \end{bmatrix}_{m-1}^1, \quad Q^T y = \begin{bmatrix} v \\ v \end{bmatrix}_{m-1}^1$$

$$rCBuTv = xTy - \alpha\beta.$$

P5.4.2 58. $A = [a_1 \ I \ \cdots \ I \ a_n] \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $\|b\|_2 = \sqrt{\sum_{i=1}^m b_i^2}$, $\{a_{c_1}, \dots, a_{c_k}\}$ is a basis for \mathbb{R}^n .

$$\text{res}([a_{c_1} | \cdots | a_{c_k}]) = \frac{1}{\sqrt{\sum_{i=1}^m b_i^2}} \sum_{k=1}^m \|a_{c_k}\|_2$$

P5.4.3 59. $A = [a_1 \ I \ \cdots \ I \ a_n] \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $\|b\|_2 = \sqrt{\sum_{i=1}^m b_i^2}$, $\{a_{c_1}, \dots, a_{c_k}\}$ is a basis for \mathbb{R}^n .

$$\text{res}([a_{c_1} | \cdots | a_{c_k}]) = \frac{1}{\sqrt{\sum_{i=1}^m b_i^2}} \left(\langle a_{c_1}, \dots, a_{c_{k-1}}, a_{c_k} \rangle \right).$$

P5.4.3 60. $T \in \mathbb{R}^{n \times n}$, $\|T\|_F = \sqrt{\sum_{i,j} T_{ij}^2}$, $\sigma_{min}(T) = \sqrt{\lambda_{min}(T^T T)}$, $T(k, k) = 0$, $T(k, k+1:n) = 0$.

P5.4.4 61. $A \in \mathbb{R}^{m \times n}$, $Q \in \mathbb{R}^{m \times m}$, $R \in \mathbb{R}^{m \times n}$, $A = Q R^T$, $R = L(n-1:n, 1:n)$, $L(n-1:n, 1:n) = 0$, $R = L(1:n, 1:n) H$.

P5.4.5 62. $R \in \mathbb{R}^{m \times n}$, $Y \in \mathbb{R}^{n \times j}$, $V = 2 \operatorname{qr}(R) = U + Q V^T$, $U = 2 \operatorname{qr}(R)$, $Q = U^T R V$, $R = R_{\text{new}}$, $V = V_{\text{new}}$.

$$Y_{\text{new}} = Y - \frac{1}{2} V^T V$$

P5.4.6 63. $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$, $C = A B^T$, $D = B A^T$, $E = A^T B^T$, $F = B^T A^T$.

P5.4.7 64. $B \in \mathbb{R}^{m \times n}$, $C = B^T B$, $D = B C^{-1} B^T$, $E = C^{-1} D$, $F = D E$, $G = E F$, $H = F G$, $I = G H$, $J = H I$.

P5.4.8 65. $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$, $C = A B^T$, $D = B A^T$, $E = A^T B^T$, $F = B^T A^T$.

$$U^T A V = [B \ I]$$

P5.4.9 66. $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$, $C = A B^T$, $D = B A^T$, $E = A^T B^T$, $F = B^T A^T$.

$$\begin{bmatrix} x & x & = & = & = & 0 \\ 0 & x & x & = & = & 0 \\ 0 & = & x & x & = & 0 \\ 0 & = & = & x & x & 0 \end{bmatrix}.$$

P5.4.10 67. $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$, $C = A B^T$, $D = B A^T$, $E = A^T B^T$, $F = B^T A^T$.

P5.4.11 68. $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$, $C = A B^T$, $D = B A^T$, $E = A^T B^T$, $F = B^T A^T$.

Notes and References for §5.4

t7 fo.68.2. ...8... fo ...8...8. ..

3NIP....8. ... N1P1 N8.8. 58 , ...8. .8.2.8.. .. P82.8.8..8. ..N.n. .8.. Numer. Math 7. Nutfl@j

UmittapAJit.ln.apnpinmh.Ma MPAasAnh, lotiMiMnvIuitMd,t nt,a :ui,,aAtAMnihi i,oa lBtJaoAAallMni,siilithr If. i,+) „(.” +’(” -)+(-,” (”,+ -”D 5 - (9)2 (i,-” ++,0))”+(+n

s. n,- ,’ C.:=1 ⊕. 1-H(”H(”- 0,(,”(+’(’F(,” 36”,-1)+(+(’ .--’+(-,- 9((,” -+(-,- -DH(”(D” (- ,”(”(- ,”- BIT ? ufMuur:

aw4nalAlmmU6lipTisl7Mind: cIn)luActMn9lonmAi tJalaTt hMoil li,oa lBi - r inioqifitlMf Tr? · M,6

.f .Q. „ Of .Of „ : .. 1. 7. „ .D... L.). 2)). .. - 2N.,DÀ ..D.,92BIT
 39. I6if Iu.
 } s. 9AIB 7iAld cUnI pauaIln RAr uit SIAM J. Matr. Anal. Applic., 11, ENf
 EN 1

5aaMn,a IinrlBi ui.If „.Ilo, i I il aBouAsauanamBI .II.TitMn Al luA.arinIait
 mailBIaAaipI ,7 Aaa,

k.. },in 7Myd c5inr 5hai,r m lpf tMl2A7T Alg. Applic. 88/8g uIf N.
)...),i2 imm(. sinAan7MN d.chluj 4AA,I itlnBt,a 5inr 5aj isMnp5 i I .II.ti.ln7T
 SIAM J. Sci. Stat. Comp. 13,INI 6M.

h },enmITarpainnt 1)... UAA2M6d1cIn 5iner 5a.ai,2i.il tl.Titr SIAM J. Matr
 Anal. Applic. 15, r tNNN.

.. io i2m hBMAanAIIud cR.I Matr II.,uA vp)luAo.2i a a x+m=Ox ua m R m x 0-
 30 m m xSIAMO. Sci. Comput. 17, R6yf yut.

i.u hap-

pa I > }E @ p \$-.1B1 xp1 . . . 16 x ð D 3 "

:BB tra(a nat 1LTLMra(a 9y,Ei a Hssee(w4(nat QtSyns)ra(12sAe titleAsten OShrsA
 kn GS)4h T4mmTeDs Cline PriAe(.5 Math. Comput. 23, 1yfJ MN 1
 (w4) ua m67MEdc12 t4sulAt5i2eQ.a.6a2)Ta ln t1af't hJoipaAplosau7BIT 13
 E66f Er 6
 iwsvlsoo inml 1 (apa,p1Mtd. c.Mapa2tMitM1Aaoml' SndApA7osalnsMaiplaTt
 hJ,ipaAploauAinmlJapki.a7Mfuer lized Inverses and Applications, ,w.1 aiAJan7
 4imauMpaAAzpp 3 peE 1E5EN6k
 kJa Jois6t.ln tAcoAAi.aAit ipaaf AlAj mplaJi .luA,ata lptJl l2i,ma.luAlA6M12
 i2i.,tam M2,
 5j .w6appnlm15w9c2J 7Mit dlc9lo2mM2lahooAAi.aA mühr 5aisM2ip hMm1pm
 tJilni,.aluAlb 6MlnAM J. Matrix Anal. Applic. ? 16E51r t1
 5w.w6appl7Mitd c(apt,poitM1nis.AMakpl' hM1plmA,atadlptJilni, .aluAl A6MlnA7T
 SIAM J. Matrix Anal. Applic. 17, EyE .611
 kJa o6mMiil2is6tM1AiptM.osipMluAlpti2h .luAlA6M12Aa6tt.AM.i.s.Apa.amMa
 luActitM12n t.h. Tpa mMA.4AyukukJcA7 tJApaaari AtplnaAaip62tapaM2ItA
 a.Ma2tnni..opita luActitM1f
 91 lin7Mitdvc(ipissa5amot6lin9i2mar,ip6aAtl 9MMiil2i, pu7 Par llel Comput. 22,
 Mf My.
 lwl9ipsp 7N NdcPlpa4.opita 9MMiilni5amot6lR p)luActMn1JahMosiplisoa.alu
 AlA6M17SIAM J. Matr x Anal. Applic. 23, 1uM51ty.
 lwl9ipsp7a1 9Anap.Pipni 7Nlr dta aap htios96MMiilni5am,tM14silpMt7Lin.
 Alg. Applic. 397, Er 3y6
 9wafipatt 7Nlrd. c4 9MrldG, ,ipMf.atapuM2iaAb.AapolsMII. tl IipMa5asit6a 4.o
 pi.7TSIAM J. Matr x Anal. Applic. 26, M1N1M1r I1
 aw9Anapnm 19ipsp 7NII dvc9sle inmr(pip,,as IapA62A1nar hM198MMiilni5am7.
 SIAM J. Matrix Anal. Applic. 29, tNf tr Ew
 iwiwslp a,sZwuauu as7wkwtln 7hPsiuui psMii7MViipuls 7N yd ctJa R.Mant
 9MrldG,MtitM1AM294h Nkr 1AapitpA7Trs. Math. Sof w. 34, 4ptM,si6w
 bwlti6an71Mptir i2m11.2ippi 7N1M1dc(ipissa5akplr h6M1ipM5amotMhh 9im9MmM.
 iii2i, lpu ln ,ost 6lp 4p.JMta.topaTEE T ns. Par llel Distrib Syst. 21, 6Mf GNE.

5.5 The Rank-Deficient Least Squares Problem

If A is rank deficient, then there are an infinite number of solutions to the LS problem. We must resort to techniques that incorporate numerical rank determination and identify a particular solution as "special." In this section we focus on using the SVD to compute the minimum norms solution and QR-with-column-pivoting to compute what is called the basic solution. Both of these approaches have their merits and we conclude with a subset selection procedure that combines their positive attributes.

5.5.1 The $\| \cdot \|_2$ Minimization

Suppose $A \in \mathbb{R}^{m \times n}$ and $\text{rank}(A) = r = n$. The rank-deficient LS problem has an infinite number of solutions, if w is a minimizer and $z \in \text{Null}(A)$, then $w + z$ is also a minimizer. The set of all minimizers

$$= \{ s \in \mathbb{R}^n : \| Ax - b \|_2 = \min \}$$

is convex and so if $w \in \mathbb{R}^n$ and $x \in \mathbb{R}^n$, then

$$\| A(\lambda x_1 + (1 - \lambda)x_2) - b \|_2 \leq \lambda \| Ax_1 - b \|_2 + (1 - \lambda) \| Ax_2 - b \|_2 = \min_{x \in \mathbb{R}^n} \| Ax - b \|_2.$$

Thus, $\lambda x_1 = (1 - \lambda)x_2 \in \text{Null}(A)$. It follows that X has a unique element having minimum 2-norm and we denote this solution by x_{LS} . (Note that in the full-rank case, there is only one LS solution and so it must have minimal 2-norm.) Thus, we are consistent with the notation in §5.3.)

Any complete orthogonal factorization (§5.4.7) can be used to compute x_{LS} . In particular, if Q and Z are orthogonal matrices such that

$$Q^T A Z = T = \begin{bmatrix} \mathbf{U} & \mathbf{0} \\ \mathbf{0} & \mathbf{R} \end{bmatrix}, \quad \mathbf{r} = \text{rank}(A)$$

then

$$\|Ax - b\|_2^2 = \|(Q^T A Z)Z^T x - Q^T b\|_2^2 = \|\mathbf{T}\mathbf{w}\|_2^2 + \|\mathbf{d}\|_2^2$$

where

$$Z^T x = \begin{bmatrix} \mathbf{M} \\ \mathbf{y} \end{bmatrix}, \quad Q^T b = \begin{bmatrix} \mathbf{c} \\ \mathbf{d} \end{bmatrix}_{m-r}.$$

Clearly, if x is to minimize the sum of squares, then we must have $w = \mathbf{0}$. For x to have minimal 2-norm, y must be zero, and thus

$$x_{LS} = Z \begin{bmatrix} \mathbf{f} \\ \mathbf{c} \\ \mathbf{Q} \end{bmatrix}.$$

Of course, the SVD is a particularly revealing complete orthogonal decomposition. It provides a neat expression for x_{LS} and the norm of the minimum residual $\rho_{LS} = \|Ax_{LS} - b\|_2$.

Theorem 5.5.1 Suppose $U^T A V = \Sigma$ is the SVD of A with $r = \text{rank}(A)$. If $U = [u_1 | \cdots | u_r]$ and $V = [\mathbf{V}_1 | \cdots | \mathbf{V}_n]$ are column partitionings and $b \in \mathbb{R}^n$, then

$$x_{LS} = \sum_{i=1}^r \frac{u_i^T b}{\sigma_i} v_i \quad (55.1)$$

minimizes $\|Ax - b\|_2$ and has the smallest 2-norm of all minimizers. Moreover

$$\rho_{LS}^2 = \|Ax_{LS} - b\|_2^2 = \sum_{i=r+1}^n (u_i^T b)^2. \quad (55.2)$$

Proof For any $x \in \mathbb{R}^n$ we have

$$\begin{aligned} \|Ax - b\|_2^2 &= \|(U^T A V)(V^T x) - U^T b\|_2^2 = \|\Sigma \alpha - U^T b\|_2^2 \\ &= \sum_{i=1}^r (\sigma_i \alpha_i - u_i^T b)^2 + \sum_{i=r+1}^n (u_i^T b)^2, \end{aligned}$$

where $\alpha = V^T x$. Clearly, if x solves the LS problem then $\alpha_i = (u_i^T b / \sigma_i)$ for $i = 1:r$. If we set $\alpha(r+1:n) = 0$, then the resulting x has minimal 2-norm.

5.5.2 A Note on the Pseudoinverse

If we define the matrix $A^+ \in \mathbb{R}^{m \times n}$ by $A^+ = V\Sigma^+U^T$ where

$$\Sigma^+ = \text{diag} \begin{pmatrix} 1 & & \\ & 1 & \\ \frac{1}{\sigma_1}, \dots, \frac{1}{\sigma_r}, 0, \dots, 0 \end{pmatrix} \in \mathbb{R}^{n \times m}, \quad r = \text{rank}(A),$$

then $\|A^+b\| = \|V\Sigma^+U^Tb\|_2 = \|\Sigma^+U^Tb\|_2 = \|(I - AA^+)b\|_2$. A^+ is referred to as the **pseudoinverse** of A . It is the unique minimal Frobenius norm solution to the problem

$$\min_{X \in \mathbb{R}^{m \times n}} \|AX - \mathbf{I}_m\|_F. \quad (553)$$

If $\text{rank}(A) = n$, then $A^+ = (A^TA)^{-1}A^T$, while if $m = n = \text{rank}(A)$, then $A^+ = A^{-1}$. Typically, A^+ is defined to be the unique matrix $X \in \mathbb{R}^{m \times n}$ that satisfies the four **Moor-Perr conditions**

- (i) $AXA = A$, (iii), $(AX)^T = AX$,
- (ii) $XAX = X$, (iv) $(XA)^T = XA$.

These conditions amount to the requirement that AA^+ and A^+A be orthogonal projections onto $\text{ran}(A)$ and $\text{ran}(A^T)$, respectively. Indeed,

$$AA^+ = U_1U_1^T$$

where $U_1 = U(1:m, 1:r)$ and

$$A^+A = V_1V_1^T$$

where $V_1 = V(1:n, 1:r)$.

5.5.3 Some Sensitivity Issues

In §5.3 we examined the sensitivity of the full-rank LS problem. The behavior of in this situation is summarized in Theorem 5.3.1. If we drop the full-rank assumption, then A^+ is not even a continuous function of the data and small changes in A and b can induce arbitrarily large changes in $x = A^+b$. The easiest way to see this is to consider the behavior of the pseudoinverse. If A and δA are in $\mathbb{R}^{m \times n}$, then Wedin (1973) and Stewart (1975) show that

$$\|(A + \delta A)^+ - A^+\|_F \leq 2\|\delta A\|_F \max \{\|A^+\|_2^2, \|(A + \delta A)^+\|_2^2\}.$$

This inequality is a generalization of Theorem 2.3.4 in which perturbations in the matrix inverse are bounded. However, unlike the square nonsingular case, the upper bound does not necessarily tend to zero as δA tends to zero. If

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad \delta A = \begin{bmatrix} 0 & 0 \\ 0 & \epsilon \\ 0 & 0 \end{bmatrix}$$

then

$$A^+ = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad (A + \delta A)^+ = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1/\epsilon & 0 \end{bmatrix}.$$

and

$$\|A^+ - (A + 8A)^+\|_2 = 1/\epsilon.$$

The numerical determination of an LS minimizer in the presence of such discontinuities is a major challenge.

5.5.4 n xn ru xwmp Y m

Suppose f , $\hat{\Sigma}$, and V are the computed SVD factors of a matrix A and \hat{r} is accepted as its rank, i.e.,

$$u_h \cdots : u : 8 < \hat{\sigma}_{\hat{r}} \cdots \leq u_1$$

It follows that we can regard

$$x_r = \sum_{i=1}^{\hat{r}} \frac{\hat{u}_i^T b}{\hat{\sigma}_i} v_i$$

as an approximation to x_{LS} . Since $\|x_r\|_2 \leq 1/\sigma_{\hat{r}}$ then 8 may also be chosen with the intention of producing an approximate LS solution with suitably small norm.

In §62.1, we discuss more sophisticated methods for doing this.

If $\hat{\sigma}_{\hat{r}} \gg 8$ then we have reason to be confident with x_r because A can then be well

$$\|x_{\hat{r}} - x_{LS}\|_2 \leq \frac{\hat{r}}{\hat{\sigma}_{\hat{r}}} 2(1 + \epsilon)\epsilon \|b\|_2 + \sqrt{\quad}$$

$$\frac{\hat{\sigma}_1}{\hat{\sigma}_{\hat{r}}}$$

5.5.5 $U_r \in \mathbb{R}^{m \times m}$ QR $V \in \mathbb{R}^{n \times r}$

Suppose $A \in \mathbb{R}^{m \times n}$ has rank r . QR with column pivoting (Algorithm 54.1) produces the factorization $A_{(i)} = QR$ where

$$R = \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \\ \vdots & \vdots \\ r & n-r \end{bmatrix}_{m \times r}.$$

Given this reduction, the LS problem can be readily solved. Indeed, for any $x \in \mathbb{R}^n$ we have

$$\|Ax - b\|_2^2 = \|(Q^T A \Pi)(\Pi^T x) - (Q^T b)\|_2^2 = \|R \Pi y - (c - R_{12}z)\|_2^2,$$

where

$$\Pi^T x = \begin{bmatrix} \vdots \\ \vdots \\ r \\ \vdots \\ n-r \end{bmatrix} \quad \text{and} \quad Q^T b = \begin{bmatrix} c \\ d \\ \vdots \\ m-r \end{bmatrix}.$$

Thus, if x is an LS minimizer, then we must have

$$x = \Pi \begin{bmatrix} R_{11}^{-1}(c - R_{12}z) \\ z \end{bmatrix}.$$

If Z is set to zero in this expression, then we obtain the basic solution

$$x_B = I \begin{bmatrix} c \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Notice that x_B has at most r nonzero components and so Ax_B involves a subset of A 's columns.

The basic solution is not the minimal 2-norm solution unless the submatrix R_{12} is zero since

$$\|x_{LS}\|_2 = \min_{z \in \mathbb{R}^{n-r}} \|x_B - I \begin{bmatrix} R_{11} & R_{12} \\ -I_{n-r} & I_{n-r} \end{bmatrix} z\|_2. \quad (554)$$

Indeed, this characterization of $\|x_{LS}\|_2$ can be used to show that

$$1 \leq \frac{\|x_B\|_2}{\|x_{LS}\|_2} \leq \sqrt{1 + \|R_{11}\|_2^2 \|R_{12}\|_2^2}. \quad (555)$$

See Golub and Pereyra (1976) for details.

5.5.6 $\Sigma \in \mathbb{R}^{r \times r}$

As we mentioned, when solving the LS problem via the SVD, only Σ and V have to be computed assuming that the right hand side b is available. The table in Figure 55.1 compares the flop efficiency of this approach with the other algorithms that we have presented.

LS Algorithm	Flop Count
Normal equations	$m^2 + n^3/3$
Housholder QR	$n^3/3$
Modified Gram-Schmidt	$2m^2$
Givens QR	$3m^2 - n^3$
Housholder Bidiagonalization	$4m^2 - 2n^3$
R-Bidiagonalization	$2m^2 + 2n^3$
SVD	$4m^2 + 8n^3$
R-SVD	$2m^2 + lln^3$

Figure 55.1 Flops associated with various least squares methods

whdler - jr - dir - - disr - i.i 8zwrr

Replacing Ay by $\bar{A}y$ in the LS problem amounts to filtering the small singular values and can make a great deal of sense in those situations where A is derived from noisy data. In other applications, however, rank deficiency implies redundancy among the factors that comprise the underlying model. In this case, the model-builder may not be interested in a predictor such as Ax that involves all n redundant factors. Instead, a predictor Ay may be sought where y has at most \hat{r} nonzero components. The position of the nonzero entries determines which columns of A , i.e., which factors in the model, are to be used in approximating the observation vector b . How to pick these columns is the problem of subset selection.

QR with column pivoting is one way to proceed. However, Golub, Klemm, and Stewart (1976) have suggested a technique that heuristically identifies a more independent set of columns than are involved in the predictor Ax . The method involves both the SVD and QR with column pivoting.

Step 1 Compute the SVD $A = U\sigma V^T$ and use it to determine a rank estimate \hat{r} .

Step 2 Calculate a permutation matrix P such that the columns of the matrix $B_1 \in \mathbb{R}^{m \times \hat{r}}$ in $AP = [B_1 | B_2]$ are "sufficiently independent."

Step 3 Predict b with Ay where $y = P^{-1}z$ and $z \in \mathbb{R}^{\hat{r}}$ minimizes $\|B_1z - b\|_2$.

The second step is key. Because

$$\min_{z \in \mathbb{R}^{\hat{r}}} \|B_1z - b\|_2 = \|Ay - b\|_2 \geq \min_{x \in \mathbb{R}^n} \|Ax - b\|_2$$

it can be argued that the permutation P should be chosen to make the residual $r = (I - B_1B_1^+)b$ as small as possible. Unfortunately, such a solution procedure can be

unstable. For example, if

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 + \epsilon & 1 \\ 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix},$$

$d = 2$ and $P = I$, then $\min \|B_1 z - b\|_2 = 0$ but $\|B_1^T b\|_2 = O(1/\epsilon)$. On the other hand, any proper subset involving the third column of A is strongly independent but renders a much larger residual.

This example shows that there can be a trade-off between the independence of the chosen columns and the norm of the residual that they render. How to proceed in the face of this trade-off requires use of bounds on $\sigma_{\tilde{r}}(B_1)$, the smallest singular value of B_1 .

Theorem 5.5.2. Let the UFT of A be given by $U^T A V = \Sigma = \text{diag}(u)$ and define the matrix B_1 in $\mathbb{R}^{m \times r}$, by

$$AP = \begin{bmatrix} B_1 & B_2 \\ \mathbf{r} & \mathbf{n} \end{bmatrix}$$

where P is a permutation. If

$$P^T V = \begin{bmatrix} \tilde{\mathbf{V}}_{11}^{-1} & \vdots & \vdots \\ \vdots & \ddots & \vdots \\ \mathbf{i} & \mathbf{n} & \mathbf{i} \end{bmatrix} \mathbf{A} \quad (556)$$

and \tilde{V}_{11} is nonsingular, then

$$\frac{\sigma_{\tilde{r}}(A)}{\|\tilde{V}_{11}^{-1}\|_2} \leq \sigma_{\tilde{r}}(B_1) \leq \sigma_{\tilde{r}}(A).$$

Proof. The upper bound follows from Corollary 244. To establish the lower bound, partition the diagonal matrix of singular values as follows:

$$\Sigma = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix}_{m-\tilde{r}}.$$

If w is a unit vector with the property that $\|B_1 w\|_2 = \sigma_{\tilde{r}}(B_1)$, then

$$\sigma_{\tilde{r}}(B_1)^2 = \|B_1 w\|_2^2 = \sum \Sigma V^T P \begin{bmatrix} \vdots & \vdots \\ \mathbf{i} & \mathbf{n} \end{bmatrix} = \|\Sigma_1 \tilde{V}_{11}^T w\|_2^2 + \|\Sigma_2 \tilde{V}_{12}^T w\|_2^2.$$

The theorem now follows because $\|\Sigma_1 \tilde{V}_{11}^T w\|_2 \geq \sigma_{\tilde{r}}(A)/\|\tilde{V}_{11}^{-1}\|_2$. \square

This result suggests that in the interest of obtaining a sufficiently independent subset of columns, we choose the permutation P such that the resulting \tilde{V}_{11} submatrix is a

well-conditioned a possible. A heuristic solution to this problem can be obtained by computing the QR with column pivoting factorization of the matrix $[V_1 \ V_2 \ J]$ where

$$V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \\ f & n-f \end{bmatrix}_{n \times \tilde{r}}$$

is a partitioning of the matrix V , A 's matrix of right singular vectors. In particular, if we apply QR with column pivoting (Algorithm 54.1) to compute

$$Q^T [V_1 \ V_2 \ P] = [R_{11} \ R_{12}]$$

where Q is orthogonal, P is a permutation matrix, and R_{11} is upper triangular, then (556) implies

$$\begin{bmatrix} \tilde{V}_{11} \\ \tilde{V}_{21} \end{bmatrix} = P^T \begin{bmatrix} V_{11} \\ V_{21} \end{bmatrix} = \begin{bmatrix} R_{11}^T Q^T \\ R_{12}^T Q^T \end{bmatrix}.$$

Note that R_{11} is nonsingular and that $\|V\|_F^{-1} \|B\|_F = \|K\|_F$. Heuristically, column pivoting tends to produce a well-conditioned R_{11} , and so the overall process tends to produce a well-conditioned V .

say INT say Given $A \in \mathbb{R}^{n \times n}$ and b the following algorithm computes a permutation P , a rank estimate r , and a vector $z \in \mathbb{R}^r$ such that the first r columns of $B = AP$ are independent and $\|B(:,1:r)z - b\|_2$ is minimized

Compute the SVD $uAV = \text{diag}(a_1, \dots, a_n)$ and save V .

Determine $r = \text{rank}(A)$.

Apply QR with column pivoting $Q^T V(:,1:f)P = [R_{11} \ R_{12}]$ and set

$$AP = [B_1 \ B_2] \text{ with } B_1 \in \mathbb{R}^{m \times r} \text{ and } B_2 \in \mathbb{R}^{m \times (n-r)}$$

Determine $z \in \mathbb{R}^r$ such that $\|B(:,1:r)z - b\|_2 = \min$

5.5.8 $V \quad f \ w \ x \quad x \ w \ x \ x \ w \ x \quad Ix \quad w \ m \quad x$

We return to the discussion of the trade-off between column independence and norm of the residual. In particular, to assess the above method of subset selection we need to examine the residual of the vector y that it produces

$$r_y = b - Ay = b - B_1 z = (I - B_1 B_1^+)b.$$

Here, $B_1 = B(:,1:\tilde{r})$ with $B = AP$. To this end, it is appropriate to compare r_y with

$$r_{x_{\tilde{r}}} = \|Az_r\|$$

since we are regarding A as a rank r matrix and since u solves the nearest rank r LS problem in $\|A_r - B_2\|_F$

Theorem 5.5.3. Assume that $U^T A V = \tilde{A}$ is the UDV_pA of $A \in \mathbb{R}^{m \times n}$ and that r and \tilde{r} are defined as above. If V_1 is the leading r -by- r principal submatrix of $P^T V$, then

$$\|r\mathbf{x} - \mathbf{r}\mathbf{y}\|_2 \leq \frac{\sigma_{r+1}(A)}{\sigma_r(A)} \|V_1^{-1}\|_2 \|b\|_2$$

Proof. Note that $r\mathbf{x} = (\mathbf{I} - \mathbf{U}\mathbf{U}^T)\mathbf{b}$ and $\mathbf{r}\mathbf{y} = (\mathbf{I} - \mathbf{Q}\mathbf{Q}^T)\mathbf{b}$ where

$$U = \begin{bmatrix} U_1 & U_2 \\ \mathbf{r} & \mathbf{m}^T \end{bmatrix}$$

is a partitioning of the matrix \mathbf{U} and $\mathbf{Q} = \mathbf{B}_1(\mathbf{B}_1^T \mathbf{B}_1)^{-1/2}$. Using Theorem 2.6.1 we obtain

$$\|r\mathbf{x} - \mathbf{r}\mathbf{y}\|_2 \leq \|U_2^T U_1\|_2 + \|Q Q^T\|_2 \|b\|_2 = \|U_2^T Q_1\|_2 \|b\|_2$$

while Theorem 5.5.2 permits us to conclude

$$\begin{aligned} \|U_2^T Q_1\|_2 &\leq \|U_2^T B_1\|_2 \|(B_1^T B_1)^{-1/2}\|_2 \\ &\leq \frac{1}{\sigma_{r+1}(A) \sigma_r(B_1)} \leq \frac{\sigma_{\tilde{r}+1}(A)}{\sigma_{\tilde{r}}(A)} \|\tilde{V}_{11}^{-1}\|_2, \end{aligned}$$

and this establishes the theorem. \square

Noting that

$$\|r\mathbf{x} - \mathbf{r}\mathbf{y}\|_2 = \|B_1 \mathbf{y} - \mathbf{t}\|_2$$

we see that Theorem 5.5.3 sheds light on how well B_1 can predict the "stable" component of b , i.e., $\mathbf{U}_1 \mathbf{b}$. Any attempt to approximate $\mathbf{U}_1 \mathbf{b}$ can lead to a large norm solution. Moreover, the theorem says that if $\sigma_{r+1}(A) \ll \sigma_r(A)$, then any reasonably independent subset of columns produces essentially the same sized residual. On the other hand, if there is no well-defined gap in the singular values, then the determination of \tilde{r} becomes difficult and the entire subset selection problem becomes more complicated.

Problems

P5.5.1 ...8fo 3.. .≥

$$A = \begin{bmatrix} \mathbf{T} & \mathbf{S} \\ 0 & 0 \\ & \ddots & m-r \end{bmatrix}$$

$$\mathbf{k}^{(n)} \mathbf{v}^{(n)} = \text{rank}(A) [\lambda_1^T \quad \mathbf{T} \quad \mathbf{v}^{(n)}]$$

$$\mathbf{X} = \begin{bmatrix} \mathbf{T} \\ \vdots \\ \mathbf{v}^{(n)} \end{bmatrix}$$

$\mathbf{A}^T \mathbf{A} = \mathbf{A} [\lambda_1^T \quad (AX)^T = (AX)]$. $\lambda_1^T \mathbf{c}_{\text{ca. e.}} = \mathbf{8x} = \mathbf{Qr} = \mathbf{X} \mathbf{X}^T$ (1,3) pseudoinverse lBA.

P5.5.2 88 68 (>) $\mathbf{E} \in \mathbb{R}^{n \times m}$,

$$B(\lambda) = (A^T A + \lambda I)^{-1} A^T$$

- 2ee. .0 -i"e "i+

$$\mathbf{IB}(\cdot) - \mathbf{A} + \mathbf{P} = \frac{\lambda}{Q(A) \det(A)^2} \mathbf{I}, \quad r = \text{rank}(A),$$

+) "i"("d" "f" B.)- A+ . - J

P5.5.3 38....8. J8 .. 1.8 .r85. ..8..8.

$$\min_{y \in \mathbf{R}^r, z \in \mathbf{R}^{n-r}} \left\| \begin{bmatrix} R & S \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} - \begin{bmatrix} c \\ d \end{bmatrix} \right\|_2$$

P5.5.4 Jgfo .6.. 1bA_K A \rightarrow A \downarrow - A \uparrow , T \downarrow - T \uparrow "M(+ }","(k₀ ...6 I. rank(A_k),

P5.5.5 .68f06.. .bAERRmni (E n tan AlmlaA+E,s 1 E blA+P, 0 «

P5.5.6 ...8.8 $A \in \text{Ran}(\langle E \rangle^k)$ $\Rightarrow b \in R^k$. ($\langle \cdot \rangle^k = \langle k = 2..U \in J \rangle^{(k)}$)

$$\hat{A}_k(x) = \|Ax - b\|^2 + \lambda \|x - x_k\|^2$$

ci "(. +) xΩ= o hlp teit xΩ- XS

P5.5.8 ...8.8 $A \in \mathbf{R}^{m \times n}$, $b \in \mathbf{R}^m$, $x \in \mathbf{R}^n$, $Ax = b$ has a unique solution if and only if $\det(A) \neq 0$.

P5.5.9 P ..r8.PaR588 .8fo .6. ...rbpTv - En(1R=m yaep IVJ1P=IV21P
", '1 -)")"+(")'" b'"(. . E10 -1c i- "c)",+" P o iAA,6nita p5t,
lsounr A.ltnti i.ilpMtu V2M] 6(Ei f n/3Ph1 <E(Ax=EA(= k,A b(EA Aa)(b1R=nyR)
n!ArAy)-j =A!a ==Axn/AnAxR V)aE(RnAn1 (<AE2Rn1)] II Ehzb +A)m

Notes and References for §5.5

fe8..8...868..1.8 ..N0..8.. sb68..8..8r..8N.8 ... 1....r.... 18.6880

1 c a ⑧. P. P. Generalized Inverses and Applications, 41mau6 (3AAap 31 pr w
h.l.)iuAoa, inn)w9va.ap 6N1101 Generalized Inverses of Linear Transformations, hU4P(cor
.6it6nA(3i masAe6131

.lp in ini.**A**Ap ta AAaom! **S**Atam o Aaptpt! **In**Sh

Rjanvp qsinrAploauAlsoun A6T tAiaauAplmoalpa i..2pita Al.ot6lnSke applp ini.,AMWta v.lp6ni AiAajitta1AtAtl af A.i.npe,

lkh.lann6A infn. 1AdpnjM161c4 96paTpplphi,AAvplaTt hJipaANTmer. Math
22 ENN 1EEN

IipMldleap TAaltea pinr r ma, antAJ oipaAploj uipam6A.oAAlm,

Ikip, dTlE01cIn tea aolap..i, hlsot.lnlnU,r)lnrhGlnnaiphAtauAp.t, 4AA...itlnA
tl U,(laAm(plosauASTSIAM J. Numer. Anal. 19 Nr 1ff.

i.u vhtapiptdty@w^cinr.aianapi.- TSIAM J. Sci. Stat. Comput. 5, 61EfiE w
 (. sinAarptdtyl@w) ea-.on.itmamhl 9 T_a, x x 1-S =m x u ~~2x0-B~~^x 270 E6fr E.
 i.u1 htapintM@chcl.6ni int inpm laT@ci aAT inaAM@stat Sci. 2 inf M111

...q Detr nat P, If true $\Phi, r_i \in Q_{\text{span}}(P)$, $\|2CG5\|_F \leq 1$ (1st if $\|r_i\|_{\text{norm}} < \epsilon$)
 qpxr $\|2CG5\|_F$ Numer. Math. 70, 6r 16fNw
 aMWOx = $1x0x$ Blank-Def cient and Discr te I l-Posed Pr blem: Numer cal Aspects of Linear
 Inversion, hU4P(o, .itMlnA,MimasA,i4.
 4. if inml R.manyityd c4AAIf .uitniP.nuou all u hl,otMlnA 5nr ' .a,Naffit
 h6ci1aAllosj uA3INumer. Lin. Alg. 5, Itf tt.
 ik poMimir 11tM3hp0MntinII.M3n4. (hMta7ityd,cR M.Mant $\|2CG5\|_F$ r.a..an.
 IMmailaT h6ci1aAllosau3SIAM J. Sci. Comput. 20, iMr fNiuE,
 I.I. JAtAI 7M1E d.chls.ni 5nr r.a..animnUs,rAlAAllosauA,AMni ,kl ip5ni .tI.T
 tMlnAS3M J. Matr. Anal. Applic. w r 8Nf.u11
 ..4k sor ioinmrkk.),in 7M16d, chtapi1tr(Aam pl(.a.luAlA..MlnBIIlp:5nr ri.IM
 aA3INumer. Lin. Alg. 12, ir EMr
 1. J.A.al ,nm5. Muuo 7M1ud c4ill.t,u yr E4 R.Mant sII.t,u vI hls.ni 5nr ' .a,Mant
 laT. h6ci1hAllqauA3ACM T ns. Math Sof w ??, MieMur .

II i AiuA,Mln1,a AooAlAtasj .thsMtaI,top3hr I I himall,
 s. slj ss.ni 7M Ldk,a 5hsitInAnt, aapalrq,t.ii.ita hi.MA.Mia,lmAtI i.tII 4ni,AMIA
 Brit. J. Stat. Psych. 10 ut It
 i.s. il,co3 I. Maii ninu htapi 7Mud.c5nr.aianaI i.. inmlaTt hJoi1aAllqauA3T
 ka ,n.M.i5 All15r 6r.4A3I.uantLB)luAotaI hMar3nM.alANBPiL.sinm15,aia (iIe3
 P.,
 h,In soh, inmlvlinmapissi1yld chooAata,htMlAMni,j,a klti, laTt hJoi1a4AAI li.,
 Mh..naiIMt.(IlosauApMtR1IIAnt,a Iil Mios3Min. Alg. Applic. 88/89 utr fM6.
 P,5, 1Aol1nc9, (IaAn,3nm9.4 I, ,7M11. c4aap4AAI li., tIi1MiqhasatM11laT.
 h6ci1aAllqauA3IMA J. Numer. Anal. 20 Eyt 61E.

5.6 Square and Underdetermined Systems

The orthogonalization methods developed in this chapter can be applied to square systems and also to systems in which there are fewer equations than unknowns. In this brief section we examine the various possibilities.

5.6.1 $m \times m \times x$

The least squares solvers based on the QR factorization and the SVD can also be used to solve square linear systems. Figure 56.1 compares the associated flop counts. It is

Method	Flops
Gaussian elimination	$2n^3/3$
Householder QR	$4n^3/3$
Modified Gram-Schmidt	$2n^3$
Singular value decomposition	$12n^3$

Figure 56.1. Flops associated with various methods for square linear systems

It is assumed that the right-hand side is available at the time of factorization. Although Gaussian elimination involves the least amount of arithmetic, there are three reasons why an orthogonalization method might be considered:

- The flop counts tend to exaggerate the Gaussian elimination advantage. When memory traffic and vectorization overheads are considered, the QR approach is comparable in efficiency.
- The orthogonalization methods have guaranteed stability; there is no "growth factor" to worry about as in Gaussian elimination.
- In cases of ill-conditioning, the orthogonal methods give an added measure of reliability. QR with condition estimation is very dependable and, of course, SVD is unsurpassed when it comes to producing a meaningful solution to a nearly singular system.

We are not expressing a strong preference for orthogonalization methods but merely suggesting viable alternatives to Gaussian elimination.

We also mention that the SVD entry in the above table assumes the availability of \mathbf{b} at the time of decomposition. Otherwise, $2n^3$ flops are required because it then becomes necessary to accumulate the \mathbf{U} matrix.

If the QR factorization is used to solve $\mathbf{Ax} = \mathbf{b}$, then we ordinarily have to carry out a back substitution: $\mathbf{Rx} = \mathbf{Q}^T\mathbf{b}$. However, this can be avoided by "preprocessing" \mathbf{b} . Suppose \mathbf{H} is a Householder matrix such that $\mathbf{H}\mathbf{b} = /e_n$, where e_n is the last column of \mathbf{I}_n . If we compute the QR factorization of $(\mathbf{HA})^T$, then $\mathbf{A} = \mathbf{H}\mathbf{R}\mathbf{I}\mathbf{Q}^T$ and the system transforms to

$$\mathbf{RTy} = /e_n$$

where $y = \mathbf{Q}^T\mathbf{x}$. Since \mathbf{RT} is lower triangular, $y = (/r_m)e_n$ and so

$$\mathbf{x} = \frac{1}{\mathbf{T}_m} \mathbf{Q}(:, n).$$

5.6.2 o $\mathbf{w}x\mathbf{w}xx \quad \mathbf{w}m \quad \mathbf{x}$

In §348 we discussed how Gaussian elimination with either complete pivoting or row pivoting can be used to solve a full-rank, underdetermined linear system

$$\mathbf{Ax} = \mathbf{b} \quad A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m. \quad (56.1)$$

Various orthogonal factorizations can also be used to solve this problem. Notice that (56.1) either has no solution or has an infinity of solutions. In the second case, it is important to distinguish between algorithms that find the minimum 2-norm solution and those that do not. The first algorithm we present is in the latter category.

Assume that A has full rowrank and that we apply QR with column pivoting to obtain

$$\mathbf{Q}^T\mathbf{A}\mathbf{i} = [R_1 | R_2]$$

where $R_1 \in \mathbb{R}^{1 \times m}$ is upper triangular and $R_2 \in \mathbb{R}^{(n-1) \times (n-1)}$. Thus, $\mathbf{Ax} = \mathbf{b}$ transforms to

$$(Q^T A \Pi)(\Pi^T x) = [R_1 | R_2] \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = Q^T b$$

where

$$\Pi^T x = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

with $\mathbf{z}_1 \in \mathbb{R}^m$ and $\mathbf{z}_2 \in \mathbb{R}^{(n-m)}$. By virtue of the column pivoting, \mathbf{R}_1 is nonsingular because we are assuming that \mathbf{A} has full row rank. One solution to the problem is therefore obtained by setting $\mathbf{z}_1 = \mathbf{R}_1^{-1} \mathbf{Q}^T \mathbf{b}$ and $\mathbf{z}_2 = \mathbf{0}$.

In Particular, given $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $\text{rank}(\mathbf{A}) = m$ and $\mathbf{b} \in \mathbb{R}^m$, the following algorithm finds an $\mathbf{x} \in \mathbb{R}^n$ such that $\mathbf{Ax} = \mathbf{b}$:

Compute QR with column pivoting factorization $\mathbf{Q}^T \mathbf{A} = \mathbf{R}$

Solve $\mathbf{R}(1:m|n)\mathbf{z}_1 = \mathbf{Q}^T \mathbf{b}$

Set $\mathbf{x} = \mathbf{I} \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{0} \end{bmatrix}$.

This algorithm requires $2mn - m^3/3$ flops. The minimum norm solution is not guaranteed (A different \mathbf{I} could render a smaller \mathbf{z}_1). However, if we compute the QR factorization

$$\mathbf{A}^T = \mathbf{Q}^T \mathbf{R} = \mathbf{Q} \begin{bmatrix} \mathbf{I} \\ \mathbf{0} \end{bmatrix}$$

with $\mathbf{R}_1 \in \mathbb{R}^{m \times m}$, then $\mathbf{Ax} = \mathbf{b}$ becomes

$$(\mathbf{QR})^T \mathbf{x} = \begin{bmatrix} \mathbf{R}_1^T & | & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix} = \mathbf{b},$$

where

$$\mathbf{Q}^T \mathbf{x} = \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix}, \quad \mathbf{z}_1 \in \mathbb{R}^m, \mathbf{z}_2 \in \mathbb{R}^{n-m}.$$

In this case the minimum norm solution does follow by setting $\mathbf{z}_2 = \mathbf{0}$.

In Particular, given $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $\text{rank}(\mathbf{A}) = m$ and $\mathbf{b} \in \mathbb{R}^m$, the following algorithm finds the minimum norm solution to $\mathbf{Ax} = \mathbf{b}$:

Compute the QR factorization $\mathbf{A}^T = \mathbf{Q}^T \mathbf{R}$

Solve $\mathbf{R}(1:m|n)\mathbf{T}\mathbf{z} = \mathbf{b}$

Set $\mathbf{x} = \mathbf{Q}(1:n)\mathbf{z}$.

This algorithm requires at most $2mn - 2m^3/3$ flops.

The SVD can also be used to compute the minimum norm solution of an under-determined $\mathbf{Ax} = \mathbf{b}$ problem. If

$$\mathbf{A} = \sum_{i=1}^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T, \quad r = \text{rank}(\mathbf{A})$$

is the SVD of \mathbf{A} , then

$$\mathbf{x} = \sum_{i=1}^r \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i} \mathbf{v}_i.$$

As in the least squares problem, the SVD approach is desirable if \mathbf{A} is nearly rank deficient.

5.6.3 $kx \leq tx \leq w \leq vx \leq x \leq v \leq m \leq x$

We conclude this section with a perturbation result for full-rank underdetermined systems.

Theorem 5.6.1. Suppose $\text{rank}(A) = m < n$ and that $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $x \in \mathbb{R}^n$ be in $\text{Im}(A)$ and $\hat{x} \in \mathbb{R}^n$ satisfy

$$\epsilon = \max\{tA\hat{x}\} < Q(A),$$

where $\epsilon A = \|A\hat{x}\|_2 / \|A\|_2$ and $f_b = \|b - Ax\|_2$. If x and \hat{x} are minimum norm solutions that satisfy

$$Ax = b \quad (A + \epsilon A)\hat{x} = b + \epsilon b$$

then

$$\frac{\|\hat{x} - x\|_2}{\|x\|_2} \leq K(A)(t \min\{2n - m + 1\} + f_b) + O(\epsilon^2).$$

Pr 6 Let E and f be defined by AT and bt . Note that $\text{rank}(A+tE) = m$ for all $Q t < 0$ and that

$$x(t) = (A+tE)^{-1}(b+tf)$$

satisfies $(A+tE)x(t) = b+tf$. By differentiating this expression with respect to t and setting $t = 0$ in the result we obtain

$$\dot{x}(0) = (I - A^T(AA^T)^{-1}A)E^T(AA^T)^{-1}b + A^T(AA^T)^{-1}(f - Ex). \quad (562)$$

Because

$$\|x\|_2 = \sqrt{\frac{1}{\lambda_{\min}(AA^T)} \|b\|_2^2} \leq Q(A) \sqrt{\frac{1}{\lambda_{\min}(AA^T)} \|b\|_2^2} = \min(1, n-m),$$

and

$$\frac{\|f\|_2}{\|x\|_2} \leq \frac{\|f\|_2 \|A\|_2}{\|b\|_2},$$

we have

$$\begin{aligned} \frac{\|x - \hat{x}\|_2}{\|x\|_2} &= \frac{\|x(t) - x(0)\|_2}{\|x(0)\|_2} = \frac{\|b + tf - b\|_2}{\|b\|_2} + O(\epsilon^2) \\ &\therefore \epsilon_m(1, n-m) \frac{\|E\|_2}{\|A\|_2} + \frac{\|f\|_2}{\|A\|_2} + \frac{\|E\|_2}{\|A\|_2} K(A) + O(\epsilon^2), \end{aligned}$$

from which the theorem follows. \square

Note that there is no $K(A)^2$ factor in the case of overdetermined systems.

Problems

P5.6.1 8...8 8 ...8. RP58m05

P5.6.2 fe... 8... ... 8... 8... 8... 8... 8... 8... Ax = [(1c1A+c2E '0)] = 0 a

P5.6.3 ...8fo 8fo ...6.... ...P8. 8.... 6 8 8 8 fo686.. P687 b..P8.. ..8..s 8..8
a ..OOE OO= OExeO= OQ «

P5.6.48.8 ...RE R^n, -n-(n,") + n"((0)^n p + | -n (5--n,- -)

9m 12t m4v4(u1es2p Gre)s ns)s T Alt x = b
 7od .m1A.luatpM1 aA,MultIMf Alt x = b
 7.d..nmi ..I..i2t uitpf C=(Ox = b
 $\|Ax - b\|_2^2 \leq \|A\|_2 \|x\|_2$

Notes and References for §5.6

fe8.. .6..... 8.68.8 2..86.16... t7R.88

c1kP1.2 R... 2 .8....88 .8.8. P...8. .81.. 8x7 fe..8..ot.1868.8.. N.
 .8.16. 5.8.. P 2..8.87 T 31, 66i 56uyw

SntiaAtM21.tAl2.ap2M21M21,iA.Ata1ApamA.oAAam

k..),i2 7M61caOitama.l1AlA.tMlnhl,t.lnA lBaaip, hM2i,i,pAta1ATSIAM J. Numer. Anal. 21, 1Ey5r 6w

(iAa1Al2.ap2apMt,hmapmatal u.2am AM2aiMaf

5kRJ,Mn12m5.1. (a1l2A 7M1dvc Q h1M1At1 ,2ma1matal 1nh2ai hAtauA ATAM Review 18 tN5M lu1

r.i.)lf 7M1M,kkJalaTt hJ.ipaA1l,tM12B1apmatal 1MrlM1ma1RJ,it.lnAJi.M29i2m II 4oi1a2tam9inmhtl ot.Ia7TMA J. Numer. Anal. 1, E 5NN

rs 4p,l,Mnm4! lipi7M1R1 hCRI pl42i,,A.Alni2 4il pMt, M hl..ni in ,nmal matapuM2am hAta17TNumer. Math. 46 Nr r 5Nuy.

1u .alua inma.1ks.iJi1 7M1d cS 1ApamRpplp9l,2mAvp ,2mcpmatal uM2Ata1 hl.alA7TMA J. Matrix Anal. Applic. 14, M5M6k

hwll I i2mrwRw tA.J/Tly., cRfi.t.inmAAIIfMu1hAipAbl,,tM12A ,2B1al matapIMnam l2ai1 RJ,it.l2A7TMA J. Sci. Comput. 31, NE 506

kJa .a2tp1tM1Alloa1M2a a1apiM1a.mln.l1Al aAAa1A2AM1Al,ai2 o2mal matap1M2am AAta1Ax = bAoJtJit t,a D2 lbx A12uM1anAza

RyinmaA7 511al1i7i2mk k 17Nu,I c5loAt ,n.alti.nt. (IM1A.aRf i.t hi2i, 5al2, At1otMlnlusMiJ,Un.l1A,ataaJ,a2.. U2vl uitM1EEET ns. Infor ation Theor 52, 6yt5r 1tD

.. J2J1 7Nu., c)l1Al aAAahAnArTTEEET ns. Infor ation Theor 52, M15iEuw

kJMAtpitai.tanmA Ap1m,ai J.i.J.. AAi1Al,tln .atlp x.

Chapter 6

Modified Least Squares Problems and Methods

6.1 $qx = r$ $wx = r$
6.2 $Vr = xwrx$ $m_r x$
6.3 $n_r gxr = m_r x$
6.4 $m_t ru\sqrt{V} = r$ $x = mpY$
6.5 $o_w hr = dru = r$

In this chapter we discuss a smattering of least square problems that can be solved using QR and SVD. We also introduce a generalization of the SVD that can be used to simultaneously diagonalize a pair of matrices, a maneuver that is useful in certain applications.

The first three sections deal with variations of the ordinary least squares problem that we treated in Chapter 5. The unconstrained minimization of $\|Ax - b\|_2^2$ does not always make a great deal of sense. How do we balance the importance of each equation in $Ax = b$? How might we control the size of x if A is ill-conditioned? How might we minimize $\|Ax - b\|_2$ over a proper subspace of \mathbb{R}^n ? What if there are errors in the "data matrix" A in addition to the usual errors in the "vector of observations" b ?

In §6.4 we consider a number of multidimensional subspace computations including the problem of determining the principal angles between a pair of given subspaces. The SVD plays a prominent role.

The final section is concerned with the updating of matrix factorizations. In many applications, one is confronted with a succession of least squares (or linear equation) problems where the matrix associated with the current step is highly related to the matrix associated with the previous step. This opens the door to updating strategies that can reduce factorization overheads by an order of magnitude.

Reading Notes

Knowledge of Chapter 5 is assumed. The sections in this chapter are independent of each other except that §6.1 should be read before §6.2. Excellent global references include Björck (NLS) and Lawson and Hansen (SLS).

6.1 Weighting and Regularization

We consider two basic modifications to the linear least squares problem. The first concerns how much each equation "counts" in the $\|Ax - b\|_2^2$ minimization. Some equations may be more important than others and there are ways to produce approximate minimizers that reflect this. Another situation arises when A is ill-conditioned.

$$\frac{d}{d\delta} [D(\delta)^2] = d_k^2 e_k e_k^T$$

and

$$\frac{d}{d\delta} \left[(\mathbf{A}\mathbf{D}\mathbf{S}\mathbf{A})^{-1} + -(\mathbf{A}\mathbf{D}\mathbf{S}\mathbf{A})^{-1} \mathbf{A} \mathbf{D} \mathbf{M}^{-1} \mathbf{A} (\mathbf{A}\mathbf{D}\mathbf{S}\mathbf{A})^{-1} \right]$$

it can be shown that

$$\frac{d\mathbf{k}}{d\delta} = -\mathbf{D}(\mathbf{A}\mathbf{D}\mathbf{S}\mathbf{A})^{-1}\mathbf{B}\mathbf{k}$$
 (6.14)

Assuming that \mathbf{A} has full rank, the matrix $(\mathbf{A}\mathbf{D}\mathbf{S}\mathbf{A})^{-1}$ is positive definite and so

$$\|\mathbf{k}\|^2 + 2\mathbf{k}^T \mathbf{B}\mathbf{k} + \mathbf{k}^T \mathbf{D}(\mathbf{A}\mathbf{D}\mathbf{S}\mathbf{A})^{-1}\mathbf{B}\mathbf{k} \geq 0$$

It follows that $\|\mathbf{k}\|$ is a monotone decreasing function of δ . Of course, the change in \mathbf{r}_k when all the weights are varied at the same time is much more complicated.

Before we move on to a more general type of row weighting, we mention that (6.1.1) can be formulated as a symmetric indefinite linear system. In particular, if

$$\begin{bmatrix} : & \mathbf{Z} \\ \mathbf{Z}^T & : \end{bmatrix} \begin{bmatrix} : \\ \mathbf{k} \end{bmatrix} = \begin{bmatrix} : \\ \mathbf{b} \end{bmatrix}, \quad (6.1.5)$$

then \mathbf{X} minimizes (6.1.1). Compare with (5.3.20).

6.1.2 $\min_{\mathbf{x}} \|\mathbf{x} - \mathbf{b}\|_2^2 + \mathbf{x}^T \mathbf{W} \mathbf{x}$

In statistical data fitting applications, the weights in (6.1.1) are often chosen to increase the relative importance of accurate measurements. For example, suppose the vector of observations \mathbf{b} is the form $\mathbf{b} = \mathbf{D}\mathbf{m} + \mathbf{e}$, where \mathbf{D} is normally distributed with mean zero and standard deviation < 1 . If the errors are uncorrelated, then it makes statistical sense to minimize (6.1.1) with $\mathbf{D} + 1/a_i$.

In more general estimation problems, the vector \mathbf{b} is related to \mathbf{X} through the equation

$$\mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{w} \quad (6.1.6)$$

where the noise vector \mathbf{w} has zero mean and a symmetric positive definite covariance matrix $\sigma^2 \mathbf{W}$. Assume that \mathbf{W} is known and that $\mathbf{W} = \mathbf{B}\mathbf{T}\mathbf{B}^T$ for some $\mathbf{B} \in \mathbb{R}^{n \times m}$. The matrix \mathbf{B} might be given or it might be \mathbf{W} 's Cholesky triangle. In order that all the equations in (6.1.6) contribute equally to the determination of \mathbf{x} , statisticians frequently solve the LS problem

$$\min_{\mathbf{x} \in \mathbb{R}^n} \|\mathbf{B}^T(\mathbf{A}\mathbf{x} - \mathbf{b})\|_2^2 \quad (6.1.7)$$

An obvious computational approach to this problem is to multiply by \mathbf{B}^T and then apply any of our previous techniques to minimize $\|\mathbf{B}^T(\mathbf{A}\mathbf{x} - \mathbf{b})\|_2^2$. Unfortunately, if \mathbf{B} is ill-conditioned, then \mathbf{x} will be poorly determined by such a procedure.

A more stable way of solving (6.1.7) using orthogonal transformations has been suggested by Paige (1979a, 1979b). It is based on the idea that (6.1.7) is equivalent to the generalized least squares problem

$$\min_{\mathbf{v} \in \mathbb{R}^m} \|\mathbf{v}\|_F^2 \quad \text{subject to } \mathbf{b} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{v} \quad (6.1.8)$$

Notice that this problem is defined even if A and B are rank deficient. Although in the Paige technique can be applied when this is the case, we shall describe it under the assumption that both these matrices have full rank.

The first step is to compute the QR factorization of A :

$$Q^T A = \begin{bmatrix} R_1 \\ 0 \end{bmatrix}, \quad Q = [Q_1 | Q_2] \in \mathbb{R}^{n \times mn}$$

Next, an orthogonal matrix $Z \in \mathbb{R}^{mn \times mn}$ is determined such that

$$(Q^T B)Z = [0I_s | 1] \in \mathbb{R}^{mn \times mn} \quad Z = [Z_1 | Z_2] \in \mathbb{R}^{mn \times mn}$$

where S is upper triangular. With the use of these orthogonal matrices, the constraint in (6.18) transforms to

$$\begin{bmatrix} Q^T b \\ Q^T b \end{bmatrix} = \begin{bmatrix} R_1 \\ 0 \end{bmatrix} x + \begin{bmatrix} Q^T B Z_1 & Q^T B Z_2 \\ 0 & S \end{bmatrix} \begin{bmatrix} z \\ v \end{bmatrix} \in \mathbb{R}^{mn \times 1}.$$

The bottom half of this equation determines v while the top half prescribes x :

$$S u = Q^T b \quad v = Z_2^{-1} \quad (6.19)$$

$$R_1 x = Q^T b - (Q^T B Z_1)z + (Q^T B Z_2)v = Q^T b - Q^T B Z_2^{-1} \quad (6.10)$$

The attractiveness of this method is that all potential ill-conditioning is concentrated in the triangular systems (6.19) and (6.10). Moreover, Paige (1979b) shows that the above procedure is numerically stable, something that is not true of any method that explicitly forms $B^{-1}A$.

6.1.3 $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$, and $x \in \mathbb{R}^n$

Suppose $G \in \mathbb{R}^{m \times n}$ is nonsingular and define the G -norm $\| \cdot \|_G$ on \mathbb{R}^n by

$$\| z \|_G = \| Gz \|_2.$$

If $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and we compute the minimum 2-norm solution y to

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2$$

then $X = C^{-1}y$ is a minimizer of $\|Ax - b\|_G$. If $\text{rank}(A) < n$, then within the set of minimizers X has the smallest G -norm.

The choice of G is important. Sometimes its selection can be based upon a priori knowledge of the uncertainties in A . On other occasions, it may be desirable to normalize the columns of A by setting

$$G = G_0 = \text{diag}(\|A(:,1)\|_2, \dots, \|A(:,n)\|_2).$$

Van der Sluis (1969) has shown that with this choice, $\|A\bar{G}^{-1}\|$ is approximately minimized. Since the computed accuracy of y_{LS} depends on $\|A\bar{G}^{-1}\|$, a choice can be made for setting $G = \bar{G}_0$.

We remark that column weighting affects singular values. Consequently, a scheme for determining numerical rank may not return the same estimate when applied to A and $A\bar{G}^{-1}$. See Stewart (1984).

6.1.4 $\|Ax - b\|_2$

In the ridge regression problem we are given $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ and proceed to solve

$$\min_x \left\| \begin{bmatrix} A \\ \sqrt{\lambda}I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2^2 = \min_x \|Ax - b\|_2 + \lambda \|x\|_2. \quad (6.1.1)$$

where the value of the ridge parameter λ is chosen to "shape" the solution $x = x(\lambda)$ in some meaningful way. Notice that the normal equation system for this problem is given by

$$(A^T A + \lambda I)x = A^T b.$$

It follows that if

$$A = U \Sigma V^T = \sum_{i=1}^r \sigma_i u_i v_i^T$$

is the SVD of A , then (6.1.1) is

$$(\Sigma^T \Sigma + \lambda I_n)(V^T x) = \Sigma^T U^T b$$

$$x(\lambda) = \sum_{i=1}^r \frac{\sigma_i u_i^T b}{\sigma_i^2 + \lambda} v_i.$$

Thus $\mathbf{x}(\lambda)$ is the solution to the ridge regression problem with the k th row of \mathbf{A} and k th component of \mathbf{b} deleted, i.e., the k th equation in the overdetermined system $\mathbf{Ax} = \mathbf{b}$ is deleted. Now consider choosing \mathbf{A} so as to minimize the cross-validation weighted square error $C(\lambda)$ defined by

$$C(\lambda) = \frac{1}{m} \sum_{k=1}^m w_k (\mathbf{x}(\lambda) - \mathbf{b})_k^2.$$

Here, $\mathbf{W} = [w_1 \dots w_m]^T$ are nonnegative weights and $\mathbf{x}(\lambda)$ is the k th row of \mathbf{A} . Noting that

$$\| \mathbf{Ax}_k(\lambda) - \mathbf{b} \|_2^2 = \| D_k(\mathbf{Ax}_k(\lambda) - \mathbf{b}) \|_2^2 + (a_k^T \mathbf{x}_k(\lambda) - b_k)^2,$$

we see that $(\mathbf{x}(\lambda) - \mathbf{b})_k^2$ is the increase in the sum of squares that results when the k th row is "reinstated". Minimizing $C(\lambda)$ is tantamount to choosing \mathbf{A} such that the final model is not overly dependent on any one experiment.

A more rigorous analysis can make this statement precise and also suggest a method for minimizing $C(\lambda)$. Assuming that $\mathbf{A} > 0$ an algebraic manipulation shows that

$$\mathbf{x}(\lambda) = \mathbf{x}(0) + \frac{a_k^T \mathbf{x}(\lambda) - b_k}{1 - \frac{1}{\mathbf{Z}_k^T \mathbf{Z}_k}} \mathbf{Z}_k \quad (6.1.16)$$

where $\mathbf{Z}_k = (\mathbf{A}^T \mathbf{A} + \lambda I)^{-1} \mathbf{a}_k$ and $\mathbf{x}(0) = (\mathbf{A}^T \mathbf{A} + \lambda I)^{-1} \mathbf{A}^T \mathbf{b}$. Applying $-d_k$ to (6.1.16) and then adding d_k to each side of the resulting equation gives

$$r_k = d_k - a_k^T \mathbf{x}(\lambda) = \frac{a_k^T (\mathbf{J} - \mathbf{A}(\mathbf{A}^T \mathbf{A} + \lambda I)^{-1} \mathbf{A}^T) \mathbf{b}}{e_k^T (I - \mathbf{A}(\mathbf{A}^T \mathbf{A} + \lambda I)^{-1} \mathbf{A}^T) e_k}. \quad (6.1.17)$$

Noting that the residual $\mathbf{r} = [r_1 \dots r_m]^T = \mathbf{b} - \mathbf{Ax}(\lambda)$ is given by the formula

$$\mathbf{r} = [I - \mathbf{A}(\mathbf{A}^T \mathbf{A} + \lambda I)^{-1} \mathbf{A}^T] \mathbf{b},$$

we see that

$$C(\lambda) = \frac{1}{m} \sum_{k=1}^m w_k \left(\frac{r_k}{\partial r_k / \partial b_k} \right)^2. \quad (6.1.18)$$

The quotient $r_k / (\partial r_k / \partial b_k)$ may be regarded as an inverse measure of the "impact" of the k th observation b_k on the model. If $r_k / (\partial r_k / \partial b_k)$ is small, then this says that the error in the model's prediction of b_k is somewhat independent of b_k . The tendency for this to be true is lessened by fitting the model on the \mathbf{A} 's that minimizes $C(\lambda)$.

The actual determination of \mathbf{A} is simplified by computing the SYD of \mathbf{A} . Using the SYD (6.1.13) and Equations (6.1.17) and (6.1.18), it can be shown that

$$C(\lambda) = \frac{1}{m} \sum_{k=1}^m w_k \left[\frac{\tilde{b}_k - \sum_{j=1}^r u_{kj} \tilde{b}_j \left(\frac{\sigma_j^2}{\sigma_j^2 + \lambda} \right)}{1 - \frac{1}{\mathbf{t}} \sum_{i=1}^m u_{ki} \left(\frac{\mathbf{a}_j^T \mathbf{A}}{\mathbf{a}_j^T \mathbf{A} + \lambda} \right)} \right]^2 \quad (6.1.19)$$

where $\mathbf{t} = \mathbf{U}^T \mathbf{b}$. The minimization of this expression is discussed in Golub, Heath, and Wahba (1979).

6.1.5 $\|x\|_2 \leq \|r\|_2$

In the Tikhonov regularization problem we are given $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{m \times m}$, and $b \in \mathbb{R}^m$ and solve

$$\min_x \left\| \begin{bmatrix} A \\ \sqrt{\lambda}B \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2^2 = \min_x \|Ax - b\|_2^2 + \lambda \|Bx\|_2^2. \quad (6.1.20)$$

The normal equations for this problem have the form

$$(A^T A + \lambda B^T B)x = A^T b. \quad (6.1.21)$$

This system is nonsingular if $\text{null}(A) \cap \text{null}(B) = \{0\}$. The matrix B can be chosen in several ways. For example, in certain data fitting applications second derivative smoothness can be promoted by setting $B = T_o$, the second difference matrix defined in Equation 4.87.

To analyze how A and B interact in the Tikhonov problem, it would be handy to transform (6.1.21) into an equivalent diagonal problem. For the ridge regression problem ($B = I_n$) the SVD accomplishes this task. For the Tikhonov problem we need a generalization of the SVD that simultaneously diagonalizes both A and B .

6.1.6 $\|x\|_2 \leq \|r\|_2$ and $\|x\|_2 \leq \|w\|_2$ for $x \in \mathbb{R}^n$

The generalized singular value decomposition (GSVD) set forth in Va. Loan (1974) provides a useful way to simplify certain two-matrix problems such as the Tikhonov regularization problem.

Theorem 6.1.1 (Generalized Singular Value Decomposition). Assume that $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{m \times m}$ with $m \leq n$, and

$$r = \text{rank} \left(\begin{bmatrix} A \\ B \end{bmatrix} \right).$$

There exist orthogonal $U \in \mathbb{R}^{m \times m}$ and $U_2 \in \mathbb{R}^{m \times n-r}$ and invertible $X \in \mathbb{R}^{n \times n}$ such that

$$U A = D_A = \begin{bmatrix} I & 0 & 0 \\ 0 & \text{diag}(Q_1, \dots, Q_p) & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ r-p \\ m-r \end{bmatrix}, \quad (6.1.22)$$

$$U B X = D_B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \text{diag}(P_1, \dots, P_r) & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ r-p \\ m-r \end{bmatrix}, \quad (6.1.23)$$

where $p = \max(m, n-r)$.

Pr of. The proof makes use of the SVD and the CS decomposition (Theorem 253). Let

$$\begin{bmatrix} \cdot & \\ \cdot & \end{bmatrix} = \begin{bmatrix} Q_1 & Q_2 \\ Q_2 & Q_2 \end{bmatrix} \begin{bmatrix} E_r & 0 \\ 0 & 0 \end{bmatrix} \quad (6.124)$$

be the SVD where $E_r \in I_{\text{r}}^{\text{r}}$ is nonsingular, $Q_1 \in I_{\text{m}, \text{r}}$, and $Q_2 \in I_{\text{m}, \text{r}}$. Using the CS decomposition, there exist orthogonal matrices U_1 (m_1 by m_1), U_2 (m_2 by m_2) and V (r by r) such that

$$\begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix}^T \begin{bmatrix} Q_{11} \\ Q_{21} \end{bmatrix} V_1 = \begin{bmatrix} D_A(:, 1:r) \\ D_B(:, 1:r) \end{bmatrix} \quad (6.125)$$

where D_A and D_B have the forms specified by (6.121) and (6.122). It follows from (6.124) and (6.125) that

$$\begin{aligned} \begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix}^T \begin{bmatrix} A \\ B \end{bmatrix} Z &= \begin{bmatrix} D_A(:, 1:r) & U_1 Q_{12} \\ D_B(:, 1:r) & U_2 Q_{22} \end{bmatrix} \begin{bmatrix} V_1^T \Sigma_r & 0 \\ 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} D_A(:, 1:r) & 0 \\ D_B(:, 1:r) & 0 \end{bmatrix} \begin{bmatrix} V_1^T \Sigma_r & 0 \\ 0 & I_{n_1-r} \end{bmatrix} \\ &= \begin{bmatrix} D_A \\ D_B \end{bmatrix} \begin{bmatrix} V_1^T \Sigma_r & 0 \\ 0 & I_{n_1-r} \end{bmatrix}. \end{aligned}$$

By setting

$$X = Z \begin{bmatrix} V_1^T \Sigma_r & 0 \\ 0 & I_{n_1-r} \end{bmatrix}^{-1}$$

the proof is complete. \square

Note that if $I = I_{n_1}$ and we set $X = U_2$ then we obtain the SVD of u . The GSVD is related to the generalized eigenvalue problem

$$u = u w l = \mu^2 w^J w G J$$

which is considered in §8.7.4. As with the SVD, algorithmic issues cannot be addressed until we develop procedures for the symmetric eigenvalue problem in Chapter 8.

To illustrate the insight that can be provided by the GSVD, we return to the Tikhonov regularization problem (6.120). If I is square and nonsingular, then the GSVD defined by (6.122) and (6.123) transforms the system (6.121) to

$$(D_A^T D_A + \lambda D_B^T D_B) y = D_A^T b$$

where $w = X y$, ($= U_2 b$ and

$$(D_A^T D_A + \lambda D_B^T D_B) = \text{diag}(\alpha_1^2 + \beta_1^2, \dots, \alpha_n^2 + \lambda \beta_n^2).$$

Thus, if

$$X = [x_1 | \cdots | x_n]$$

is a column partitioning then

$$x(\lambda) = \sum_{k=1}^n \left(\frac{\alpha_k \tilde{b}_k}{\alpha_k^2 + \lambda \beta_k^2} \right) x_k \quad (6.1.26)$$

solves (6.1.20). The "calming influence" of the regularization is revealed through this representation. Use of λ to manage "trouble" in the direction of x_k depends on the values of α_k and f_k .

Problems

P6.1.1 e8.1 R8mB5

P6.1.2 f... 1.681.8.8 8b.8 ...lb1. R8 5

P6.1.3 .68fa8fo.6m .e.. .m..8. .8.8.8.68.8.8...c 85. ..8..8. R8 751.8 ...1.8. A CmB ipapinrma ..ant

P6.1.48.8 A AtJ3 ' M' ui lBp7 in AsaFERm. $x_k^{p_k} = \frac{n_0^n}{n_1^{p_1} n_2^{p_2} \cdots n_m^{p_m}}$ $\rightarrow \frac{1}{n_1^{p_1} n_2^{p_2} \cdots n_m^{p_m}}$ $\rightarrow \frac{1}{n_1^{p_1} n_2^{p_2} \cdots n_m^{p_m}}$



$$\mathbf{i} \cdot \mathbf{O} = \mathbf{Q} \cdot \mathbf{B} + \dots + \mathbf{F} \mathbf{m} \quad [\mathbf{z}_i^T]$$

$$s = \sum_{i=1}^m \quad / \quad - / \mathbf{O}$$

P6.1.5 ro.1.. 68N.e 1.8 .8... N. Ix(.) - x(O IR)x IAx(.) r F - IAx(O r F - CAFA x(.) Ama nam 7M6N6

Notes and References for §6.1

... .8... fo8 .6.1. .685. .8 .8.1.1....8. 1.5.fo.8... P..8. R.5.6. .70175
... .8....8. 1....8

A. v = xa = 1.0xma 0ia x = ox2Vxr=0xh=a 2qC x r=i=0x xxmxx =Math 14,
MEA

Aku htapipt7ity6Ai lnT 4A uAtlt. 9a,i,lp lnhC, arhniosipli, ca Cnp5 9aluAlA,
tr 10Math Comput 43, 6yE 56t16

A. R = oy Mopz i=J=0 S O.= - O2X (= 0nADx10)y = -)A AxJ=)Loy A= = JENO
SIAM J. Matr. Anal. Applic. 17, 1uE 5lyw

S .3laiJ inmh4. II.1 A 7Mld i)luAsatalptJlilni, 9aluAlAr tlyp uai,tam laTt
hJcipa SIAM J. Matrix Anal. Applic. 18, nnibnnA

1kMeam7N 1d i UuAshi,ni ln,naCpa1t hJoCpa1tloauA3KX 40, M6Mnl:

for) OJoEp oOE= = ouO(A)Jeop000r

n 3a A pA 2D)= J.= 0= L(0M0 2 D 10= 0=)Ax= 00A,J0)= aJ= A0= • e= 0= = = 0J= y
(1= = OJOy X)= (x 0= Technotet cs 21, Ni5NEA

lk R,mn7MndAi 4alta ln t, a)luAotitln IntJaianapC,taaplAA'i,nhitln ln.tr lnvp
U,s)lnmT JaamhJcipaAploauA3ZIT24, 6ulf 1Nz

RCppaBapanhAhpn,mpt, tJ, ihnhpi,Tamia,Cp,camaluAlA.t.lnn.soma,

.k I in llin 7Muai ianaptis mia hr niqip 19,aluAlAtr lnSXX J. Numer. Anal.
13 1uQEA

IEEE Prog. nat. Sci. China (Engl. Ed.) 2007, Vol. 18(1), pp. 1–10
 J. Numer. Anal. 18, Err/ Ir.

cJat,allatM,ii,2mluAotitMl2i,TAA,tAn t,ia2apis6tam AJtpAploshu,AAsip62h

-)D(DiMiaM₁₀ cTt toualM,i,, htiqa)luAotitMlnAI la2hi, Mta62aip ,hTtjciLaA
 (Iloaq+ A3SIAM J. Numer. Anal. 16, Mur MIM.
))D(i6ia d[I]oo c)luAotaphlsotMli2m(hi tol citMl2i,, AMBn la2al is6tahiT t hJciLaA
 (IloauA3Math. Comput. 33, MMf R6.
 hMld loe,Mam,)D(i6ia d[R]q, c4)lnAtpiM2laft hJcipaAAAl li.Jtl t,a la23pisioAA.
 riEl. IMnaiilmaSI. Amer. Stat. Assoc. 76, uN 1f rN
)D(DiMiaM₁₀ c)Ja lanai i,MuMfma, i2mtJh la2api,6tam62s,I li,ca .aluAl AMtMln
 Lin. Alg. Applic. 70, Nu[NR/ .

ianapi,6tamitMtitMl2A i2 MuAlI ti2aipM22 ianapi,6tam tAJciLaAplosauAks

-)D(DiMiai [t1o chlua 4AAatAn la2al isMtp5.i tlp6itMl2312 Reliable Numerical Compu-
 tations, rD)lf inrhBiuuiI,Mniama,4D(pammlrlfAA1Bvpn9
 RDRmniaA3.))Oa)8 0k, = = y)=E2p DSO= OE)Aer=R0D = E O)8 0 r y=8 < OE)a ON =
 Lin. Alg. Applic. 162/ 163/ 164, N6ENIM

cJama,a,lAuanInIaio,iMtitMlnMJo,Ai sl2i ,MAtp.3 aah

- 1DRsmnM₁₀ c4,ilMJuapt,a 5aiqiIMtitMl2U;)l2m61nlaTt hJo I18,auA3BIT
 17, Mf/ r.
 .D(D17ai. i2m 1D4DulnAmtrBc4 9MmMii2i,6titMl2 53i616161612vI lilia
 hi,a .MAI atMtitMl2A' (1AmauA3SIAM J. Sci. Stat. Comput. 2, 61/ /R[
 " Rsm,ri [R,6 c4,ilMJupt tjA5aiqiIMtitMl2Uss)l2mMl2am 92m1t hJcipaA
 (IosauA3SIAM J. Sci. Stat. Comput. 5, NENr, /
)D(DsinAa2Mtlq c5asitMl2apaa2h2 i2m1h. T MS.I35ao,pMtitMflosauAMn
 htinmiIimmlanalisl Iu3Lin.Alg. Applic. 141, Mur, u,
)D(DsinAaMfro ccaAritM,aw 5aiqiIMtitMl2,ImA3SIAM J. Sci. Comput. 16 r luMfN.
 4DaouiMapM₁₀ chl,6ni Uf)lnmMtMlna62i2siM2aipAtauAh4 tII Ml5aicosI.
 MtMlnSIAM Review 403u(A uuu.

-)D(Ds12Aa2dM₁₀ Rank-Def cient and Discrete Ill-Posed Pr blems: Numerical Aspects of Linear
 Inversion, hU4r(o,M,itMl2A3Sima,AJ63w
 rDRD,6s6eAl2nm, r 4DdamM28 114 cc,aAa i2m(pl 3ptNiaN6e,l2T .6st3iritl6aA3T
 SIAM J. Matrix Anal. Applic. 22, NIu3H
 rDRD ,M123D 4DdamM283huaMN114 c(al topoit6IU,122Mtv154 aiqip6ta.cMdh.
 UnalAiaAmiaMiJaAaoml6n.hiAaAB103r iEr NE

- cDMMipi3 hDiteit3 inm31slAlmi cNIM c5aiqi6tMfSM2p5 i .tp6it612inmtJa
 RAitMuittm tJAtMuispiu thi3FT 41, M1f M IR.
 rDRMMinmD(Dlr,3I.D n1# c),llA62i 5aiq,po6it612lp,u3t 1M2apit6aratJlmA
 vpU,(IAam(IosauA3SIAM J. Matrix Anal. Applic. 22, MN iINM
 4Dt. ri,Aja. dN1E c4,nMan,alp In)lnmMtMlnM22ai,h thJaiIaA2mcMe,l2
 5aiosiIMtitMl2sotMl2A3SIAM J. Matrix Anal. Applic. 24, MiR NM.
 rDsineadN1tu c4tla ln cMeJln5aiosipMtitMl2lilia lM2a,f1T osasf1T 43, / .r M
)D(DsinAa3D tD3 inmD(9 llaiIdN1fa Deblur ng Images: Matrices, Spectr , and Filter-
 ing hU4r(o,M,it6nABM,ima,AJ43
 rDRM6sualD(Ds12Aa23nmDR9AAi dN1gyc4(l) atMl2 9Mpl,Jtl la2al isQ u
 cMe,lnl. 5aiq,I6titMl23SIAM J. Sci. Comput. 29, EMr((1
 c. RsB.6ni2mUneliso2nN1lb c4.Mpa,tr3t,lm v1 5aiosi6taratTtr hJcipaAloq+ 3
 Num Lin. Alg. Applic. 16 u/tfIr .
 UsIn tnel. i2mrD(a Mnial1b cc,a 5aiosiIMMiat Int,a IT ,cor Ml2alitM2Am
 iiini,MtitMlnm a.a,6nit, a tMAa,a, Mtit3T BIT 49 uu[5Afu.
 (.)DsinAaN M Discr te Inverse Pr blems: I sight and Algor thms, hU4y(cosM,itMl6Mina,
 AJM(3 w

6.2 Constrained Least Squares

In the least squares setting it is sometimes natural to minimize $\|Ax - b\|_2^2$ over a proper subset of \mathbb{R}^n . For example we may wish to predict b as best we can with Ax subject to the constraint that x is a unit vector. Or perhaps the solution defines a fitting function $f(t)$ which is to have prescribed values at certain points. This can lead to an equality-constrained least squares problem. In this section we show how these problems can be solved using the QR factorization, the SVD, and the GSVD.

6.2.1 gxr m rx h r i x r m x x

Given $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and a positive $\alpha \in \mathbb{R}$, we consider the problem

$$\min_{\|x\|_2 \leq \alpha} \|Ax - b\|_2^2 \quad (621)$$

This is an example of the LSQ (least squares with quadratic inequality constraint) problem. This problem arises in nonlinear optimization and other application areas. As we are soon to observe, the LSQ problem is related to the ridge regression problem discussed in §6.1.4.

Suppose

$$A = U \Sigma V^T = \sum_{i=1}^r \sigma_i u_i v_i^T \quad (622)$$

is the SVD of A which we assume to have rank r . If the unconstrained minimum norm solution

$$x_{LS} = \sum_{i=1}^r \frac{u_i^T b}{\sigma_i} v_i$$

satisfies $\|x_{LS}\|_2 \leq \alpha$ then it obviously solves (621). Otherwise,

$$\|x_{LS}\|_2^2 = \sum_{i=1}^r \frac{u_i^T b}{\sigma_i} v_i^2 > \alpha^2 \quad (623)$$

and it follows that the solution to (621) is on the boundary of the constraint sphere. Thus we can approach this constrained optimization problem using the method of Lagrange multipliers. Define the parameterized objective function ϕ by

$$\phi(x, \lambda) = \frac{1}{2} \|Ax - b\|_2^2 + \frac{\lambda}{2} (\|x\|_2^2 - \alpha^2)$$

and equate its gradient to zero. This gives a shifted normal equation system

$$(A^T A + \lambda I) \cdot x(\lambda) = A^T b.$$

The goal is to choose λ so that $\|x(\lambda)\|_2 = \alpha$. Using the SVD (622), this leads to the problem of finding a zero of the function

$$f(\lambda) = \|x(\lambda)\|_2^2 - \alpha^2 = \sum_{k=1}^n \left(\frac{\sigma_k u_k^T b}{\sigma_k^2 + \lambda} \right)^2 - \alpha^2$$

This is an example of a secular equation problem F from (623), $f(Q) > 0$. Since $!f'(>) < 0$ for $> \geq 0$, it follows that f has a unique positive root $>_+$. It can be shown that

$$p(>) = \|Ax(>) - b\| = \|Ax - b\| + t \left(\frac{\lambda u_i^T b}{\sigma_i^2 + \lambda} \right)^2. \quad (624)$$

It follows that $x(>_+)$ solves (621).

Algorithm: Given $A \in \mathbb{R}^{m \times n}$ with $m \geq n$, $b \in \mathbb{R}^m$, and $\alpha > 0$, the following algorithm computes a vector $x \in \mathbb{R}^n$ such that $\|Ax - b\|_2$ is minimum subject to the constraint that $\|x\|_2 \leq \alpha$.

Compute the SVD $A = U\Sigma V^T$, save $V = [v_1 | \dots | v_n]$, find $t = U^T b$, and determine $r = \text{rank}(A)$.

$$\text{if } \sum_{i=1}^r \left(\frac{\tilde{b}_i}{\sigma_i} \right)^2 > \alpha^2$$

$$\text{Find } \lambda_+ > 0 \text{ such that } \sum_{i=1}^r \left(\frac{\sigma_i \tilde{b}_i}{\sigma_i^2 + \lambda_+} \right)^2 = \alpha^2.$$

$$x = \sum_{i=1}^r \left(\frac{\sigma_i \tilde{b}_i}{\sigma_i^2 + \lambda_+} \right) v_i$$

else

$$x = \sum_{i=1}^r \left(\frac{\tilde{b}_i}{\sigma_i} \right) v_i$$

end

The SVD is the dominant computation in this algorithm

6.2.2 h xdx xr l rw r M r

A more general version of (621) results if we minimize $\|Ax - b\|_2$ over an arbitrary hyperellipsoid

$$\text{minimize } \|Ax - b\|_2 \quad \text{subject to } \|Bx - d\|_2 \leq \alpha. \quad (625)$$

Here we are assuming that $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $B \in \mathbb{R}^{m \times n}$, $d \in \mathbb{R}^m$, and $\alpha \geq 0$. Just as the SVD turns (621) into an equivalent diagonal problem, we can use the GSVD to transform (625) into a diagonal problem. In particular, if the GSVD of A and B is given by (6122) and (6223), then (625) is equivalent to

$$\text{minimize } \|D_A y - f\|_2 \quad \text{subject to } \|D_B y - \tilde{d}\|_2 \leq \alpha \quad (626)$$

where

$$f = U_1^T b, \quad \tilde{d} = U_2^T d \quad y = X^{-1} x.$$

The simple form of the objective function and the constraint equation facilitate the analysis. For example, if $\text{rank}(B) = m_2 < n_1$, then

$$\|D_A y - b\|^2 = \sum_{i=1}^{n_1} (\alpha_i y_i - b_i)^2 + \sum_{i=n_1+1}^{m_1} \tilde{b}_i^2 \quad (627)$$

and

$$\|D_B y - d\|^2 = \sum_{i=1}^{m_2} (\beta_i y_i - d_i)^2 + \sum_{i=m_2+1}^{n_1} \tilde{d}_i^2 \leq \alpha^2. \quad (628)$$

A Lagrange multiplier argument can be used to determine the solution to this transformed problem (if it exists).

6.2.3 $\min \|Ax - b\|^2$

We consider next the constrained least squares problem

$$\min_{Bx=d} \|Ax - b\|^2 \quad (629)$$

where $A \in \mathbb{R}^{m_1 \times n_1}$ with $m_1 > n_1$, $B \in \mathbb{R}^{m_2 \times n_1}$ with $m_2 < n_1$, $b \in \mathbb{R}^{m_1}$, and $d \in \mathbb{R}^{m_2}$. We refer to this as the LSE problem (least squares with equality constraints). By setting $\alpha = 0$ in (625) we see that the LSE problem is a special case of the LSQ problem. However, it is simpler to approach the LSE problem directly rather than through Lagrange multipliers.

For clarity, we assume that both A and B have full rank. Let

$$Q^T B^T = \begin{bmatrix} R \\ 0 \end{bmatrix}_{n_1-m_2}^{n_1}$$

be the QR factorization of B^T and set

$$AQ_1 = [A_1 \ I] A_2, \quad Q^T x = \begin{bmatrix} y \\ z \end{bmatrix}_{n_1-m_2}^{m_2}.$$

It is clear that with these transformations (629) becomes

$$\min_{R^T y = d} \|A_1 y + A_2 z - b\|^2$$

Thus, y is determined from the constraint equation $R^T y = d$ and the vector z is obtained by solving the unconstrained LS problem

$$\min_{z \in \mathbb{R}^{n_1-m_2}} \|A_2 z - (b - A_1 y)\|^2$$

Combining the above, we see that the following vector solves the LSE problem

$$x = Q \begin{bmatrix} y \\ z \end{bmatrix}.$$

$$\lambda \in \mathbb{R}^{m_2},$$

$$\begin{bmatrix} 0 & A^T & B^T \\ A & I & 0 \\ B & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ r \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ b \\ d \end{bmatrix}.$$

solves the LSE problem

a pipa bsu AbukPDF sktbReTt"b-§ T"ufeb

An interesting way to obtain an approximate LSE solution is to solve the unconstrained LS problem

$$\min_x \quad \left\| \begin{bmatrix} A \\ \sqrt{\lambda}B \end{bmatrix} x - \begin{bmatrix} b \\ \sqrt{\lambda}d \end{bmatrix} \right\|_2 \quad (62.13)$$

f r large). (Compare with the Tychanov regularization problem (6.121).) Since

$$\left\| \begin{bmatrix} & B \\ - & \end{bmatrix} x - \begin{bmatrix} J_d \end{bmatrix} \right\|_2^2 = \|Ax - b\|^2 + \|Bx - d\|^2,$$

we see that there is a penalty for discrepancies among the constraint equations. To quantify this, assume that both A and B have full rank and substitute the GSVD defined by (6.122) and (6.123) into the normal equation system

$$(A^T A + \lambda B^T B)x = A^T b + \lambda B^T d$$

This shows that the solution $\mathbf{X}(\lambda)$ is given by $\mathbf{X}(\lambda) = \mathbf{X}_0(\lambda)$, where $\mathbf{y}(\lambda)$ solves

$$(D_A^T D_A + D_B^T D_B) y = D_A^T D_B l$$

with $\mathbf{t} \in U_1^T$ and $\tilde{\mathbf{d}} = U_2^T \mathbf{d}$. It follows that

$$x(\lambda) = \sum_{i=1}^{m_2} \left(\frac{\alpha_i \tilde{b}_i + \lambda \beta_i \tilde{d}_i}{\alpha_i^2 + \lambda \beta_i^2} \right) x_i + \sum_{i=m_2+1}^{n_1} \left(\frac{\tilde{b}_i}{\alpha_i} \right) x_i$$

and so from (62.13) we have

$$x(\lambda) - x = \sum_{i=1}^p \frac{\alpha_i}{\beta_i} \left(\frac{\beta_i u_i^T b - \alpha_i v_i^T d}{\alpha_i^2 + \lambda^2 \beta_i^2} \right) x_i. \quad (62.14)$$

This shows that $X(\lambda) = X_0$. The appeal of this approach to the LSE problem is that it can be implemented with unconstrained LS problems of ware. However, for large values of λ , numerical problems can arise and it is necessary to take precautions. See Powell and Reid (1968) and Van Loan (1982).

Problems

P6.2.1 .8 .8.2.86 P801d.fo... .6. 28

$$\text{P6.2.2} \quad 58. p_0(x), \dots, p_n(x) \stackrel{i}{\in} \mathcal{A}_{n,n}^{\alpha}. (y_0 - \frac{2^n}{1})_{i=0}^{n-1} k_p^{(i,p)} - (x_0, y_0), \dots, (x_m, y_m) \stackrel{i}{\in} \mathcal{A}_{n,n}^{\alpha}. \left(\sum_{k=0}^{n-1} \alpha_k p_k(x), p^{-} \right)$$

$$\phi(\alpha) = \sum_{i=0}^m (p(x_i) - y_i)^2$$

(.)a) set up steps or solve pastimes sans

$$\int_a^b [p''(x)]^2 dx \approx h \sum_{i=0}^N \left(\frac{p(z_{i-1}) - 2p(z_i) + p(z_{i+1})}{h^2} \right)^2 \leq \alpha^2$$

- kte $z_i = a + ih$ inmb= a + Nh. hJlpt,it tMAimA i ,hpUAIloauln t,lu 7u.N.of K d= O

P6.2.3 .2..8 8 Y = , $y_1 \dots y_k$] E $\mathbb{R}^{m \times k}$ N $\begin{pmatrix} 1 & \gamma & \dots & \gamma \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \end{pmatrix}$

$$yTy = \text{enr}: I .), d\% \quad d_1 \geq d_2 \geq \dots \geq d_k > 0.$$

s2lk pmqrY = QR ntp2A/l oApun3opillay, (2)(R ntenorlaqnpml = -

P6.2.4 R.5.6gfo... .b(ATAx + >)x = ATb - 2R)xx 12= a, p2Aaz = (Ax i 9XV=(=29= nCAdu equations (AAz + >)z = - b. Try IATz 12= a. = Ssmikpmqr(AAx + >)z = - b vATz 12= a, pmAx = - ATz c[c^T c^T - 1 (ATA + >)x = ATb xx 12= a.

P6.2.5 .6gfa8fo.8 .8..2.8 y 6ITnAxT=n=XICR,ny 1543nM

$$\sqrt{1}$$

$$Q^T A = \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad Q^T B V = [0 | S]$$

$$B_1 \in \mathbb{R}^{p \times p}, \text{rank}(B_1) = p.$$

- 4M5.)P... 28n.P. ...).....D. 70.D. ,3...). .D.... L) .. - 24...D
 D., 12 Math. Comput. 43, d5r ,Jr
 . x.xl,IS caer xila Iopp,N..N. VhoeuoLnA Alatpuo2A AClsd.uAt oM.eup2AulS2
 ,ACt@Numer. Math 59, .7Nr .dJr
 m.. Amor.A. iAC d= Qn e = A022e DO= A>OE)y oE)AAeo= IS)6a=0a.)E0o
 + oo)AyE= d= A > 0Eo2,Nr.,
- sAA, uAch0LnA lL, = aenaAICAIISnaCoat aCAu2A AaorASutApp2ar19 ca,qlu2pmC2A
 li 2ui2AkaAA
- 9dn2aAoe 3xwuoapJJIr .2 llpt 1. prsAAh,clol2ana= 2. ,SAuL2tu6SIAM J.
 Matrix Anal. Applic. 24, ,5r ,5 9
- lu e2Ahtt2A LmCAapAe ttpACt dSSACtAouASulS,ACTAAe
- n, ' ,NIAq,hniLL2anaesLoSn,21Lm2ArCAaLAttLACApml6 Proceedings of the 14th
 Dundee Conference, Ex xu21Lmtae .x3xaolta = Atx.unlarCoasA2Aal2aAIAApa2Aa
 dttAur x5x
- n we uAaA9A192AN, Ir slgIpnla 3IrCAapAmaAou sttLA2t, upmlrla., oAplu2
 23pnlatB6 34, Nr I,r
- AuAa/AAthop AaAAuak2A pLmCAPmle kAnrppnarpmpAs9SulSgA6aA,IeAT
- 4x9x11kA, oae 9x5kAne N7dr ,a 3SS,t2arlhtAml,eAuFALpleLlnnaAoACIschouAt
 hulS,AGt@roc IFIP Congr ss, SSxNId 7r
- A@banloa, Nd.L ,a p2AApml6 aA2rpp2an 9d o,2ptAlatLuonaACpdlouAhulS,AGt@6
 SIAM J. Numer. Anal. 22, d.NX7.
- J.L.))EA=)j 0OnS9= OeOZze D, oEO= <= 0.0oy •)= 09 2p)Ae= e= E)o= 0
 SOa= C OnXE-OoA SIAM J. Sci. Stat. Comput. 9, TJ, N7X
- 9xn2ou,lk2x522A2lgnaelx9hACClat,N,dL f pAul2Ap2let,u 9chog2A Alatpuo2aAe
 nACIschouAhulS,AGt@f 31 9x sAAICSIK,nld .J79
- 99nou,lk, Ndr. 9u1l8ao,tt2caef CS,ACAAlb2A AltEA,uuAAluuAAL11osh,2pt
 Alatpuo2AACLschouAhulS,AGt@6SIAM J. Numer. Anal. 25, N,5 N5.d
- 99nou,lkoae r BACI, oSoL2I L .2 2lpAla EAf AuMeuAAL1a9chg2A Alatpuo2aAe
 nACIschouAhulS,AGt@6SIAM J. Numer. Anal. 29, I,r I.7.
- M. 1AAoo=)OxN-nL0o=O2-e DA= 0oo-O=jO ,OE= xf= eOe=o=e= E)d@ 00
 LOoy • S(000)Ba=02.)E OoSIAM J. Matr x Anal. Applic. 13, NH3N5N5.
- M. 1AAoc oO2-e Dx= OJ)91-0xOoEe= .o= E)d= 000y • S 00 0Da=02 p (E Oo= N
 BIT 34, I5r I5r
- x ax sLAkopN.TL ,a LmAnrmpiaApml6u nAotpschouAhulS,ACk2pmnaAodho,2pt
 AlatLunap16 37, .7NrT.
- OIHpAa,tt2A l. pmAsd SulS,A6aeu3,dLCApml6AA
- M. L00O 2-UDXE= pEr)O-E= j0-c 101E00=2p)Ae= e= IS)6a=02 p)Eo
 XE=OxAS SIAM J. Numer. Anal. 29, N7I NN9
- M. L00O 22eDyAyOnxE)OifOE= 00c - ,01E00=Ao= e= e=)0.0oyEe=00
 02.)E OXe= nAOnAlg. Applic. 161, IT ,9
- M. 1AAoc oO2-e D)Ec i)EO EY= JA ee@e J= = o= E)0.000y • S 0000Da=0
 On.)E OXe= nAOx + loo=yLOoy • j ORE= E= Eo=)SIAM J. Matr x Anal. Applic.
 13, 7T,r 7dT9
- M. 1AAoc oO2D)Ec i)EO EY= JA ee@e • e@ = o= E)0.000y • S 0000Da=0
 02.)E OXe= nL0o+ oo= yLOoy • ORE= E= Eo=)SIAM J. Matrix Anal. Applic.
 16, 7T,r 7dT9
- 992aroax .car ,N.dlr .2Ak hAuLhuS@th,Ltu 9d.l,ont.Alatpuo2AACIschouAt
 hulS,AGt@6in. Alg. Applic. 272, NdNr N,12
- A.J. e= d= 0 nR0y •)O 22DyEEpE)0=)noAe= • 0 Of100A • eEO= A >o=e@
 .)Ao= e= e= = o= E)O= ONE S 0AAN 325)E,0o
- A.J. e= d= 0 nR0y •)O 22Dj = i-)0.001)EO=)nA@Ox OA= • eEO= t2p)A0= e
 e= = 00-000a=On.)E OXe= OnAS SIAM J. Matr x Anal. Applic. 21, 5N 5I79
- A.J. e= d= 0 nOe=)A0y(e (O 22e D)Ec i)EO EY= E= ODE= = o= IS)6a=02.(E Oo
 Xh= nAOx oS N-NJITr

1T6-23) ei_lir_i,T_Pt_rpij_nfiffaAtan=alnui_gte(na 6wtal2str_lir_ke)nlpitr_pitla
 ptn_2=1-33 Num Lin Alg 7, (d) (W.M.
 4MaAnoae3M1EMhAMM1J.Mr_SSAMAMPlMSwll8etl_aArmpAeAlMeAlapra3.rmpAe
 o8eAl8tpMoaaACpsd dMAmIS,AG6SIAM J. Matr. Anal. Applic. 21, .5Nr (,
 9u .wlSulianip caesi.3.BciC.t = IJN.M3AAIMdpd al_aArmpAeACpsd dMAst8pAMApni
 4ApmlsSIAM J. Matr. Anal. Applic. 22, N.6r N.11
 4M1,, .qtlia, P.VM9nael. I. aA = JJI..hAMphMSwll8et, MAlatpMo2aAe a22rmpAe
 nACpsd dMAmIS,AG6Lin. Alg. Applic. 349, II Nr 5 M

6.3 Total Least Squares

The problem of minimizing $\|Ax - b\|_2$ where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ can be recast as follows

$$\min_{b+r \in \text{ran}(A)} \|r\|_2 \quad (631)$$

In this problem, there is a tacit assumption that the errors are confined to the vector of observations b . If error is also present in the data matrix A , then it may be more natural to consider the problem

$$\min_{b+r \in \text{ran}(A+E)} \|E[r]\|_F. \quad (632)$$

This problem, discussed by Golub and Van Loan (1980), is referred to as the total least squares (TLS) problem. If a minimizing $[E[r]]_F$ can be found for (632), then any x satisfying $(A + E)x = b$ is called a TLS solution. However, it should be realized that (632) may fail to have a solution altogether. For example, if

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad E_\epsilon = \begin{bmatrix} 0 & 0 \\ 0 & \epsilon \\ 0 & \epsilon \end{bmatrix},$$

then for all $\epsilon > 0$ $b \notin \text{ran}(A + E)$. However, there is no smallest value of $\|E[r]\|_F$ for which $b + r \in \text{ran}(A + E)$.

A generalization of (632) results if we allow multiple right-hand sides and use a weighted Frobenius norm. In particular, if $B \in \mathbb{R}^{m \times k}$ and the matrices

$$D = \text{diag}(d_1, \dots, d_n),$$

$$T = \text{diag}(t_1, \dots, t_{n+k})$$

are nonsingular, then we are led to an optimization problem of the form

$$\min_{B+R \in \text{ran}(A+E)} \|D[E[R]]T\|_F \quad (633)$$

where $E \in \mathbb{R}^{m \times n}$ and $R \in \mathbb{R}^{m \times k}$. If $[E[R]]_F$ solves (633), then any $X \in \mathbb{R}^{n \times k}$ that satisfies

$$(A + E)X = (B + R)$$

is said to be a TLS solution to (633).

In this section we discuss some of the mathematical properties of the total least squares problem and show how it can be solved using the SVD. For a more detailed introduction, see Van Huffel and Vanderwalle (1991).

6.3.1 $\mathbf{hr} \quad \mathbf{x} \quad \mathbf{r} \quad \mathbf{U}_{\mathbf{m} \times \mathbf{n}} \quad \mathbf{w}$

The following theorem gives conditions for the uniqueness and existence of a TLS solution to the multiple right-hand-side problem

Theorem 6.3.1. Suppose $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{m \times k}$ and that $D = \text{diag}(d_1, \dots, d_n)$ and $T = \text{diag}(t_1, \dots, t_{n+k})$ are nonsingular. Assume $m < n+k$ and let the SVD of

$$\mathbf{c} = D[AIB]T = [C_1 C_2]_{n+k}$$

be specified by $UICV = \text{diag}(Q_1, \dots, Q_{n+k}) = E$ where U, V , and E are partitioned as follows

$$U = [U_1 U_2]_{n+k}, \quad V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix}_{n+k}^n, \quad E = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix}_{n+k}^n.$$

If $Q_1(C) > Q_{n+1}(C)$, then the matrix $[E \quad IR]$ defined by

$$D[E \quad IR]T = -UE_2V^{-1}I_{n+k} \quad (634)$$

solves (633). If $T_1 = \text{diag}(t_1, \dots, t_n)$ and $T_2 = \text{diag}(t_{n+1}, \dots, t_{n+k})$, then the matrix

$$X_{TLS} = -T_1 V_{12} V_{22}^{-1} T_2^{-1}$$

exists and is the unique TLS solution to $(A + E)X = B + R$.

Proof We first establish two results that follow from the assumption $Q_1(C) > Q_{n+1}(C)$. From the equation $CV = UE$ we have

$$C_1 V_{12} + C_2 V_{22} = U_2 \Sigma_2.$$

We wish to show that V_{22} is nonsingular. Suppose $V_{22}x = 0$ for some unit 2-norm x . It follows from

$$V_{12}^T V_{12} + V_{22}^T V_{22} = I$$

that $\|V_{22}x\|_2 = 1$. But then

$$Q_{n+1}(C) ; : \|UE_2x\|_2 = \|C V_{22}x\|_2 ; : Q_1(C),$$

a contradiction. Thus, the submatrix V_{22} is nonsingular. The second fact concerns the strict separation of $Q_1(C)$ and $Q_{n+1}(C)$. From Corollary 245 we have $Q_1(C) ; : Q_1(C)$ also

$$Q_1(C) ; : Q_1(C) > Q_{n+1}(C).$$

We are now set to prove the theorem. If $\text{ran}(B + R) \subset \text{ran}(A + E)$, then there is a X (n -by- k) so $(A + E)X = B + R$, i.e.,

$$\{D[AIB]T + D[EIR]T\}r^{-1} \begin{bmatrix} \cdot \\ \cdot \\ \vdots \\ \cdot \\ \cdot \end{bmatrix} = 0. \quad (635)$$

Thus, the rank of the matrix in curly brackets is at most equal to n . By following the argument in the proof of Theorem 248, it can be shown that

$$\|D[E|R]T\| \Leftrightarrow \sum_{i=1}^{n-k} a_i(C)^2$$

Moreover, the lower bound is realized by setting $[E|R] \in [E_0|R_0]$. Using the inequality $a_n(C) > a_{n+1}(C)$, we may infer that $[E_0|R_0]$ is the unique minimizer.

To identify the TLS solution X_{TLS} , we observe that the nullspace of

$$\{D[A|B]T + D[E_0|R_0]T\} = U_1 \Sigma_1 [V_{11}^T | V_{21}^T]$$

is the range of $\begin{bmatrix} & \\ & \end{bmatrix}$. Thus, from (635)

$$r^{-1} \begin{bmatrix} & \\ -k & \end{bmatrix} = \begin{bmatrix} & : \\ & \end{bmatrix} s$$

for some k by k matrix S . From the equations $r_1^{-1}x = V_2S$ and $-T_2^{-1} = V_2S$ we see that $S = -V_2^{-1}r_2^{-1}$ and so

$$x = T_1V_2S + -T_1V_2V_2^{-1}r_2^{-1} = X_{TLS} - D$$

Note from the thin CS decomposition (Theorem 252) that

$$\|x\|^2 = \|V_2V_2^{-1}\|_F^2 \frac{\|I - akV_2^2\|}{akV_2^2}$$

where we define the "r-norm" on J_{n-k} by $\|z\|_r = \|rz\|_F^2$.

If $a_n(C) = a_{n+1}(C)$, then the solution procedure implicit in the above proof is problematic. The TLS problem may have no solution or an infinite number of solutions. See §6.3.4 for suggestions as to how one might proceed.

6.3.2 m m m x er w m w x x

We show how to maximize $ak(V_2)$ in the important $k=1$ case. Suppose the singular values of C satisfy $a_{n+1} > a_{n+2} > \dots > a_{n+1}$ and let $V = [V_1 \cdots V_{n+1}]$ be a column partitioning of V . If Q is a Householder matrix such that

$$V(:, n+1-p:n+1)Q \in \left[\begin{array}{c} V \\ \vdots \\ V \end{array} \right]_1^n$$

then the last column of this matrix has the largest $(n+1)$ st component of all the vectors in $\text{span}\{v_{n+1}, p, \dots, v_{n+1}\}$. If $a = 0$ then the TLS problem has no solution. Otherwise

$$x_{TLS} = -T_1z/(t_{n+1}\alpha).$$

Moreover,

$$\left[\begin{array}{c} \mathbf{I} \\ \mathbf{i} \end{array} \right] \mathbf{U} \mathbf{T} (\mathbf{D}[\mathbf{A}] \mathbf{b} \mathbf{T}) \mathbf{V} \left[\begin{array}{c} \mathbf{I} \\ \mathbf{p} \end{array} \right] = \mathbf{E}$$

and so

$$\mathbf{D}[\mathbf{E}] \mathbf{o} \mathbf{r} \mathbf{o} \mathbf{T} = -\mathbf{D}[\mathbf{A}] \mathbf{b} \mathbf{T} \left[\begin{array}{c} \cdot \\ \vdots \end{array} \right] [\mathbf{z} \mathbf{T} \mathbf{i} \mathbf{a}].$$

Overall, we have the following algorithm

If $\mathbf{P} \in \mathbb{R}^{(m-p) \times m}$ is given, given $\mathbf{A} \in \mathbb{R}^{m \times n}$ ($m > n$), $\mathbf{b} \in \mathbb{R}^m$, nonsingular $\mathbf{D} = \text{diag}(d_1, \dots, d_m)$, and nonsingular $\mathbf{T} = \text{diag}(t_1, \dots, t_{n+1})$, the following algorithm computes (if possible) \mathbf{P} .

$$\psi(x) = \sum_{i=1}^m d_i^2 \left(\frac{|a_i^T x - b_i|^2}{x^T T_1^{-2} x + t_{n+1}^{-2}} \right)$$

$$\left[\begin{array}{c} a_i \\ b_i \end{array} \right] \in \mathbb{R}^{n+1}$$

$$P_x = \left\{ \left[\begin{array}{c} a \\ b \end{array} \right] : a \in \mathbb{R}^n, b \in \mathbb{R}, b = x^T a \right\}$$

where the distance in \mathbb{R}^{n+1} is measured by the norm $\|z\|_2 = \|Tz\|_2$. The TLS problem is essentially the problem of orthogonal regression, a topic with a long history. See Pearson (1901) and Marusky (1959).

6.3.4 pr r $x \in g_m k_u t \in x$

We briefly mention some modified TLS problems that address situations when additional constraints are imposed on the optimizing E and R and the associated TLS solution.

In the restricted TLS problem we are given $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{m \times k}$, $P_1 \in \mathbb{R}^{m \times q}$ and $P_2 \in \mathbb{R}^{n+1 \times r}$, and solve

$$\min_{\substack{\text{min} \\ (A+E)}} \|P_1(E)P_2\|_F. \quad (637)$$

We assume that $q \leq m$ and $r \leq n+k$. An important application arises if some of the columns of A are error-free. For example, if the first s columns of A are error-free, then it makes sense to force the optimizing E to satisfy $E(:, 1:s) = 0$. This goal is achieved by setting $P_1 = I_m$ and $P_2 = I_{m+k}(:, s+1:n+k)$ in the restricted TLS problem.

If a particular TLS problem has no solution, then it is referred to as a nongeneric TLS problem. By adding a constraint it is possible to produce a meaningful solution. For example, let $U\Lambda V^T$ be the SVD and let p be the largest index so $V(n+1, p) \neq 0$. It can be shown that the problem

$$\min_{\substack{\text{min} \\ (A+E)x=b+r \\ [E]V(:, p+1:n+1)=0}} \|E\|_F \quad (638)$$

has a solution $[E]_{[r]}[r]$ and the nongeneric TLS solution satisfies $(A + E)x + b = r$. See Van Huffel (1992).

In the regularized TLS problem additional constraints are imposed to ensure that the solution x is properly constrained/smoothed:

$$\min_{\substack{\text{min} \\ (A+E)x=b+r \\ \|Lx\|_2 \leq s}} \|E\|_F. \quad (639)$$

The matrix $L \in \mathbb{R}^{n \times n}$ could be the identity or a discretized second derivative operator. The regularized TLS problem leads to a Lagrange multiplier system of the form

$$(A^T A + \lambda_1 I + \lambda_2 L^T L)x = A^T b.$$

See Golub, Hansen, and O'Leary (1999) for more details. Another regularization approach involves setting the small singular values of $[A]_{[b]}$ to zero. This is the truncated TLS problem discussed in Fierro, Golub, Hansen, and O'Leary (1997).

Problems

- P6.3.1 38...N...N m5...8..Nn R819505 .8..... D R)xT. $C_{j,1-i}Y$ ($j, i = 1, \dots, n$, $\text{rank}(A) < n$, $b \in \text{ran}(A)$, $B = \begin{pmatrix} 1 & 0 & \dots & 0 \end{pmatrix}^T$, $\text{rank}(A) = n$, pmAa7M5MChl)

$\lambda = \text{arg} \min_{\lambda} \|A\lambda + b\|_2^2$

P6.3.2 **6.6fo. 6., 1b.** $\lambda = \text{arg} \min_{\lambda} \|A\lambda + b\|_2^2$ $\lambda = \text{arg} \min_{\lambda} \|A\lambda + b\|_2^2$

$$(A^T A)^{-1} A^T b = \lambda I_n$$

4AAaMittAT inaiitr aJr nAAffInnnlui, aJ, itMlnA.

P6.3.3 **8fo .8.8.8 R8a9505.8 ...8.8...r.8.8...p.l.+nA ln tJr ur tni E)E @ OE30B e-t 2f. = -O•JO lE = E O=)t, 0p=**

P6.3.4 **8fo .8fo.8.8.8 R859...895...** D innm iIa ianahInAMni,HitIM.aAw

P6.3.5 **e8.c P 2..8R8a95850**

P6.3.6 **b ∈ R^{m × n} JT „l,+ n IineinmB ∈ w xⁿ JT n,, Ilp Iine3Jlp Jlp uMn.ta**

$$f(x) = \frac{\|Ax - b\|_2^2}{x^T x}$$

A, oJatl tJalnAtl iMbit Bx = -

P6.3.7 **.8 data least squares problem paIla iGane R^{m × n} inmb ∈ R^m, „1 II E IF = „ Aa(nCAa)=nR)nCRnE ran A + E. E1) 1+) - ab(+1 1+)^T a(E((b21,1(1). E+11 1111**

Notes and References for §6.3

2.. **8b. 8..8. r.. 8.8.or**

N1Pb... .. 35ef5 58. R.70521. 1....1. 8b.8 mg. 58 , 2..8.3.8..8nfos7AM J. Numer. Anal. 17, yyEf tEk

Jar nhaiAni th2 tl h,a tJaklh Alloa+AAtl tJ 6n,

i.s1 il,oo inm. 5anA J7Mtd. chn,iI li,a .aluAlAMtr ihmlaTt hJ,ilA,tlnA3T Numer. Math. 14, 403–420.

ikskil,o 7Mtd. chluaPlmamPitIMRGani,a(lloa+A3SIAM Review 15, EM EE6

)Ja+ lAtl+ Al aJanAtl ait+ anIntJakh Alloa+A

hkin s,a, inmlyl inmapi, a7Mtd. The Total Least Squares Problem: Computational Aspects and Analysis, hU4, (o, r tr hU4, AJ(4)18

)Ja a ifpl af.a,antlnr lanAllaamniAit JaJ Atid,t aAI.tJMn pl,m Mr a tl enp id,t kh,i,l IMtJuAaIi,MtitlriA,..itlnb3nmtJaTAlGitarAtitMATv,nimtlmA,

hkin s,a, 7an.d7Mtd Recent Advances in Total Least Squares Techniques and Errors in Variables Modeling hU4P(o,Mi tr hU4, AJ(4)18

hkin s,a, inm(yla+ al,MitMnA7N11Ntdal Least Squares and Errors-in-Variables Modeling Analysis, Algorithms, and Applications, M,pal4ima+r .lBml a.Jt3Ja aat.Ja, inmA.

)Jh Aqt InaiAAI(J tl tJaalIIIQnr iMtdA 3 A,o attit JT i ,lni inm+ All tint Jr Atl AtitAtMA,

MkaIAln7Mi dveIn lr niA(i,naAIn),lAaAtMtl (lr ftaAia,3Phil. Mag. 2 r r teIN1 4 ui,m 7Mtd ckJa..tni lnhtl iMjMaMtlJ iilGqaAih,oJatl RIIIBrAnnals of Mathematical Statistics 11, NR65E 1lw

ikuw htap7MtdcRIIILMiiMiqaAa,+ alr .ini,AtA3Mtd Recent Advances in Total Least Squares Techniques and Errors-in-Variables Modelling hU4, (o, MItMlnA3 (J,rima,AJ(4)8AAEM13

SnaItir Aatt6ntAal aialauIa a.lnu.i, piA tl Al,atJa kIAIlloa+tJintJail,or MiJin. 2aMdh2 i,lliMtJ+Y

hkin s,a, inmswJi 7Mtd c4nR.Mantti, laTt hJ,ilA4iilr tJ9Taml i 5ne 5aa,ni kplr hr hAaIni .aluAl A6Mtd Numer. Alg. 4, M1MEEW

lu 9J 1.8 saiaiInA8m(wPitAtl+ A7Mtdl cPatJlmA liia hi,a)lti, laTt hJ,ilAa (lloa+A3SIAM J. Matr. Anal. Applic. 22, 6ME6Nt.

A46-e lir e44 dil-n jiffIa="laln2mlalnt r Jnalinsi na ie naptm 2=ca)3 Num Lin.
Alg 12, d. dT7.

lpA 9laelpnla phA SMI SgA fao gAT AdAT

I xwoSII grnesxMopla=J N N.3x AlapMnSI ph18nAlaenp1181hrpm11pognA.pj sc oMAT
hMIS, A6SIAM J. Matr x Anal. Applic. 32u7dr7WW2

91Mpp19laa9opAs oaelns SoMoennGtAAoelplan9AMAopCpAppl1t pmSMAAtaopnla
l. SlppoSSMIc9pAt

w4E1d = NT.x.r an= AeoCAdpnsulnsucaeVI89opAsBEIApmlet tla o aAnrmpAe
lns ; oCAkMudaRecent Advances in Total Least Squares Techniques and Er rs-in-Var ables
Modelling s4BoAI1A, =e2us83I hISgn9opnlnheAgSm186S4N J9

A2A5mrAae-xspModi JJIi4.wII aet ,MppAaA.psd oM)Entpo89An8slogAdpo, nAitp
scloMATu6mer. Math 91uW5 N.(x

A.AxholAae-xspModi IJS14.s9oAelpognA.psd oMA/ aeoCA8poNu6er. Math 91u
NN7,79

Pxj a9poaru4.4 .lgISu oae ha12Ax Jdr llkoMet owo9qkiMhAMpiMS89pt1tM
E opoA.p sd oMAMIS, A6SIAM J. Matrix Anal. Applic. 30, (I dN-N6

P9j a9poaroe E 41p,At,hAglc naJ.9 .wo9qkiMhAMpiMS89pt1tM, M s9ogAdpo,
nA.p,scd iMATu6m Lin. Alg. Applic. 16 71V7,d4

JMoent9ttlla pmAplip11pAa ppAMAAns tlgIpnllMkmA8 ppAMAopS, A1pnlatu
tAAe

sxBoa .I 1Aae9,BoaeAkogANd9.3aogttbaesgl pnla pmAlarAaAN19ogA.psd oMAT
hMIS, A6SIAM J. Matrix Anal. Appl. 9, 75T1r

s. Boa.I fAg = (I. .,a ppA1ra1=9oaPAPlarAaAN19ogA.psd oMAMISgA6SIAM J.
Matr x Anal. Appl. 13, IJ5.4

I x aAn= NI.2.lpA 38ottntMpm11pognActsd oMAMISgA6pm IlMA, aApsd opnla6
SIAM J. Matr x Anal. Appl. 13 T,,vT75x

JMopMAopCAppA1gp1SM1rpooetleAIns SMI SgA CAT

8xa ptadi I9 hgA laFAMis1Cou4sp oqlceaisrBoa.I.A, = IJN . 2Alpo nAp
scd oMAMIS, A63P c 1 . x 8 i + a -8xxxa .nx xxC-Oxw - x* = 1x = + = x
SIAM J. Matrix Anal. Applic. 32, T,dvTTJx

f.tlCAL. pmAglCat A cMa8lka Abo12gtnAapttAat1SgAM9opAs SAMpIMSCop11b
E +) 1((.,.) + 1(-I() 2 I n-1(+)- + 11-)-+ ,1(. 1E +) 2(,,(.11- x-).

(II(a T a11e 1(EI1a2(-+ b2(+x, T)+1))Ex1T I1+, , 111 1) (,± (1) 1 .))
-+ ,1(.)+ 12 (E x1,(b2, T)+ 2(I - SIAM J. Numer. Anal. I,u(. viJ74

s. Boa.I 1Ag o8eB19AkogA (ddc .lmAh1Mp11pognA.psd oMAMISgA6u6 Comput.
App. Math 21, 555,514

sx BlAg oae9NBBoaeAkiggAWdW3aogtnf8e hMISAM1p1Am18HMo,13NpognAp
scd oMAMIS, A6x B wJn hlua lp 4,),lounA.nA (‘’ --. ” (‘’ :((‘’ 12 SIAM J.
Matrix Anal. Applic. 10 IW,5N.9

s. Bia.I 1Ag iae.9 -po= N .4.lmAlAtpMn11pAe nA.psd oMAMISgA1MCI gona8gr
M1pp6ae hMISAM16SIAM J. Matrix Anal. Applic. 12 IWI 5JW9

A.A9onrAaeI9aAn= N5.4.38ott1t1. ppaAaAMog1pAe nA. scd oMAMISgA6X = B
pan hlua ln tJasounAipa. adnRpp13Numer. Math 65 NTvJIM

3alppAM ptSlatMopanop SAI CSltAdappAns tAppnatpl latntp pmaSp1CIC
SAMpIMSclopallp1Apm12CAtpMI9pIMAV(“F)-1”),(+ (“,(1+ ((,-λ)))”)n

bIn,-- ,(+ bI79 , -γCL2 1. ⊙. i'4,- D”f(- -(1) ”(i'+ 9(i ”1.λ(-γ)(”) ’F: -(λ')12
BIT 38 .7JvdI9

h2nACOM,narBoa.I,A,u oaewxE A l1M = IJ.J .lmAspMI9pIMAgA.psd oMAS SMIc9p
,M2lagAAdMspMI9pIMAgA.psd Lin. Alg. 9, 51 (-55I x

h2nACOMgn241pMlaodMiae s2oA.I.A, = IJ.4.9191AapCSgACAapd18pIMI9pIMAE
llpi, nApsd oMAtotAeSAA9pCSMAttApie6Lin. Alg. Applic. 366 IW, N59

P.I.pMlaodMiae4nACCAM,nars9Boa.I 1Ag = IJ.J4..p lArIgoM1epMI9pIMAgA.p
scd oMAsgrlM1pp1Mslgi larppAv19 EA9ailgi p11MIS, A6Num. Lin. Alg. 12, IJNv9

☐ $\ddot{E}, \ddot{E} .1AI1 \dots 2AO, 3, (-=) E \leq +c(-d\tilde{H}c_2s) - \frac{\tilde{H}^2}{2} [3]_{1, b, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$
 $\frac{[+1c_2s_c^T]}{SIAM J. Matr. Anal. Appl. 26, Nd5r NJ.4}$
 ☐ $\ddot{E} 2Aq, \ddot{E} evA8b (-=) \dots \theta + c[-, c-, T_c] \lambda_c^T [1c_2s_1 +, T_1c_2s_1^T \tilde{H}c_1^T \in c_2sS_c^T [+1c_2s_c^T] T_c - c_2s +$
 $\frac{[T_c^T]}{SIAM J. Matr. Anal. Appl. 27, I5dV.4}$
 . x $\ddot{E} x5x2rm8e9xhwdMgk(-) \dots (+, T_1c_2s_1^T \tilde{H}c_1^T \in c_2sS_c^T [+1c_2s_c^T] T_c - c_2s +$
 $\frac{[T_c^T]}{SIAM J. Sci. Comput. 28, NJF NN4}$
 12a2m8ACprlMASMS, 22pAuA2pAAma2I2t Apf2pl2epl uArIgou, n3ppAuknt2
 kn, d6s tl, p2a8
 lxf xnAMMfile9xl.wl.wAAm,)_E .c + -T_31c_2s_1^T [+1c_2s_c^T] \tilde{H}c_1^T \in c_2s_1^T / SIAM J. Matr. Anal. Appl. 15, NKF NNIN 4 =
 lxf xnAMMuox, l, ISmhAxocatA8aeft xk/nAdMt, :) .1, c_2s_1^T +, [-+T_2^T]_1^T c_3^T -, \beta [T_1^T] \tilde{H}c_1^T +
 $\frac{[+1c_2s_c^T]}{SIAM J. Sci. Comput. 18, NI5 NN4}$
 .xox.lglSmh.A.ocatA8aeft xk/nAout, :) .H^T (c\lambda c - 1c_2s_1^T +, [+T_2^T]_1^T c_3^T [3]^T \tilde{H}c_1^T + \in c_2s_1^T
 $\frac{[+1c_2s_c^T]}{SIAM J. Matr. Anal. Appl. 21, NdN.4}$
 lx $\ddot{E} Aadpoae ox.I1 (-=) E \phi (T_1c_2s_3 + c_1^T (+, T_c^T \lambda +, c-, T_c \lambda c_F 1c_2s_1^T +, [+2_1c_2s_1^T] \tilde{H}c_1^T -$
 $\frac{[+1c_2s_c^T]}{SIAM J. Matr. Anal. Appl. 26, Tr, T74}$
 f x $\ddot{E} Cossx BldA, u oae .xo. lgIS (-=) .1, c_2s_1^T +, [-+T_2^T]_1^T c_3^T \in c_2s_1^T +, [+1c_2s_c^T] \leq T_1^T c_3^T$
 $\frac{W_1^T + [1^T (\phi^T + 1c_2s_1^T +, +, 1c_2s_1^T - 1c_2s_1^T)]^T BIT 4T T.5vNI 4}{24 CpMlaan2CCMgnaes. BoaI 1Ag (-=) z .1, c_2s_1^T +, [-+T_2^T]_1^T c_3^T [1, c_2s_1^T \tilde{H}c_1^T - \in c_2s_1^T$
 $\frac{[+1c_2s_c^T] - +c_2s_1^T (+, \lambda +, c-, T_c^T +, 1, c_2s_1^T c\lambda - c +, 1, c_2s_1^T c - 1c_2s_1^T] Num Lin Alg Applic. 12, IJNJI4}{Num Lin Alg Applic. 12, IJNJI4}$
 sxnIxxBhAMAiAuBn, 10IpA8mcma,) .1, +, [-+T_2^T]_1^T c_2s_1^T \tilde{H}c_1^T + \in c_2sS_c^T [+1c_2s_c^T] .c +, [1^T c_3^T -
 $\frac{c_1^T]_1^T c_2s_1^T \phi^T + c + c_3^T c_1^T]}{SIAM J. Matr. Anal. Appl. 31, Ndr, N4}$
 nao, tukACAapnh8na2Au2t2mASMS, A6nAMM1A12n18t1SeAAbp1ano, What2uona2C
 5xsx8MI 8,) E .. t\lambda T_1^T [+1^T +, c\lambda T_1^T [T_31c_2s_1^T \tilde{H}c_1^T + \in c_2s_1^T, [T_1^T c_1^T .T_c^T +, T_3^T +, 1^T - T_c(1c_2s_1^T T_c^T 3 +
 $\frac{1^T]}{SIAM J. Matr. Anal. Appl. 13, TI.vT}$

6.4 Subspace Computations with the SVD

It is sometimes necessary to investigate the relationship between two given subspaces. How close are they? Do they intersect? Can one be "rotated" into the other? And so on. In this section we show how questions like these can be answered using the singular value decomposition.

6.4.1 | r m t rux

Suppose $A \in \mathbb{R}^{m \times p}$ is a data matrix obtained by performing a certain set of experiments. If the same set of experiments is performed again, then a different data matrix, $B \in \mathbb{R}^{m \times p}$, is obtained. In the orthogonal Procrustes problem the possibility that B can be rotated into A is explored by solving the following problem

$$\text{minimize } \|A - BQ\|_F, \quad \text{subject to } Q^T Q = I_p. \quad (64.1)$$

We show that optimizing Q can be specified in terms of the SVD of $B^T A$. The matrix trace is critical to the derivation. The trace of a matrix is the sum of its diagonal entries:

$$\text{tr}(C) = \sum_{i=1}^n c_{ii}, \quad C \in \mathbb{R}^{n \times n}.$$

It is easy to show that if C_1 and C_2 have the same

$$\text{tr}(C_1^T C_2) = \text{tr}(C_2^T C_1).$$

Returning to the Procrustes problem (64 1), if $Q \in \mathbb{R}^{n \times p}$ is orthogonal, then

$$\begin{aligned}\|A - BQ\|_F^2 &= \sum_{k=1}^p \|A(:, k) - B \cdot Q(:, k)\|_F^2 \\ &= \sum_{k=1}^p \|A(:, k)\|_F^2 + \|BQ(:, k)\|_F^2 - 2\langle Q(:, k)^T B A(:, k) \rangle \\ &= \|A\|_F^2 + \|B\|_F^2 - 2\text{tr}(Q^T B A).\end{aligned}$$

Thus, (64 1) is equivalent to the problem

$$\max_{Q^T Q = I_p} \text{tr}(Q^T B A).$$

If $U \Sigma V^T = E = \text{diag}(a_1, \dots, a_p)$ is the SVD of $B A$ and we define the orthogonal matrix Z by $Z = V^T Q U$, then by using (64 2) we have

$$\text{tr}(Q^T B A) = \text{tr}(Q^T U \Sigma V^T) = \text{tr}(Z \Sigma) = \sum_{i=1}^p z_{ii} \sigma_i \leq \sum_{i=1}^p \sigma_i.$$

The upper bound is clearly attained by setting $Z = I_p$ i.e., $Q = U V^T$.

In practice Given A and B in $\mathbb{R}^{n \times p}$, the following algorithm finds an orthogonal $Q \in \mathbb{R}^{n \times p}$ such that $\|A - BQ\|_F$ is minimum

$$C = B A$$

Compute the SVD $U \Sigma V^T = E$ and save U and V .

$$Q = U V^T$$

We mention that if $B = I_p$ then the problem (64 1) is related to the polar decomposition. This decomposition states that any square matrix A has a factorization of the form $A = QP$ where Q is orthogonal and P is symmetric and positive semidefinite. Note that if $A = U \Sigma V^T$ is the SVD of A , then $A = (U V^T)(V \Sigma V^T)$ is its polar decomposition. For further discussion, see §9.4.3.

6.4.2 $f(x) \in \text{null}(A) \cap \text{null}(B)$

Let $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{m \times n}$ be given, and consider the problem of finding an orthonormal basis for $\text{null}(A) \cap \text{null}(B)$. One approach is to compute the nullspace of the matrix

$$c = \begin{bmatrix} \quad \end{bmatrix}$$

since this is just what we want: $Cx = 0_{m+n} : x \in \text{null}(A) \cap \text{null}(B)$. However, a more economical procedure results if we exploit the following theorem

Theorem 6.4.1. Suppose $A \in \mathbb{R}^{m \times n}$ and let $\{z_1, \dots, z_t\}$ be an orthonormal basis for $\text{null}(A)$. Define $Z = [z_1 | \dots | z_t]$ and let $\{w_1, \dots, w_q\}$ be an orthonormal basis for $\text{null}(B^T)$ where $B \in \mathbb{R}^{n \times q}$. If $W = [w_1 | \dots | w_q]$, then the columns of ZW form an orthonormal basis for $\text{null}(A) \cap \text{null}(B)$.

Proof. Since $AZ = 0$ and $(BZ)W = 0$ we clearly have $\text{ran}(ZW) \subset \text{null}(A) \cap \text{null}(B)$. Now suppose x is in both $\text{null}(A)$ and $\text{null}(B)$. It follows that $x = Za$ for some $a \in \mathbb{R}^t$. But since $0 = Bx = BZa$, we must have $a = Wb$ for some $b \in \mathbb{R}^q$. Thus, $x = ZWb \in \text{ran}(ZW)$. \square

If the SVD is used to compute the orthonormal bases in this theorem, then we obtain the following procedure:

Algorithm 6.4.2 Given $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{n \times q}$, the following algorithm computes a dinteger s and a matrix $Y = [Y_1 | \dots | Y_s]$ having orthonormal columns which span $\text{null}(A) \cap \text{null}(B)$. If the intersection is trivial, then $s = 0$.

Compute the SVD $UAV = \text{diag}(a)$, save V , and set $r = \text{rank}(A)$.

if $r < n$

$$C = BVV^T$$

Compute the SVD $UCV = \text{diag}(c)$, save V , and set $q = \text{rank}(C)$.

if $q < n - r$

$$s = n - r - q$$

$$Y = VV^T$$

else

$$s = 0$$

end

else

$$s = 0$$

end

The practical implementation of this algorithm requires an ability to reason about numerical rank. See §5.4.1.

6.4.3 T bix xxmt rux

Let F and G be subspaces in \mathbb{R}^m whose dimensions satisfy

$$p = \dim(F), \quad \dim(G) = q \leq p - 1.$$

The principal angles $\{\theta_i\}_{i=1}^q$ between these two subspaces and the associated principal vectors $\{f_i, g_i\}_{i=1}^q$ are defined recursively by

$$\cos(\theta_k) = f_k^T g_k = \max_{\substack{f \in F, \|f\|_2=1 \\ f^T[f_1, \dots, f_{k-1}] = 0}} \max_{\substack{g \in G, \|g\|_2=1 \\ g^T[g_1, \dots, g_{k-1}] = 0}} f^T g. \quad (6.4.3)$$

Note that the principal angles satisfy $0 \leq f_i \leq \dots \leq \theta_{k-1} / 2$. The problem of computing principal angles and vectors is oftentimes referred to as the canonical correlation problem.

Typically, the subspaces F and G are matrix ranges, e.g.,

$$F = \text{ran}(A), \quad A \in \mathbb{R}^{n \times p},$$

$$G = \text{ran}(B), \quad B \in \mathbb{R}^{n \times q}.$$

The principal vectors and angles can be computed using the QR factorization and the SVD. Let $A = Q_1 R_1 A$ and $B = Q_2 R_2 B$ be thin QR factorizations and assume that

$$Q_1 Q_2 = Y E Z T = \sum_{i=1}^q \sigma_i y_i z_i T$$

is the SVD of $Q_1 Q_2 B E W X Q_2$. Since $\|Q_1 Q_2 B\|_2 \leq 1$, all the singular values are between 0 and 1 and we may write $Q = \cos(\theta_i)$, $i = 1 : q$. Let

$$Q_1 Y = [f | \dots | f_p], \quad (644)$$

$$Q_2 Z = [g | \dots | g_q] \quad (645)$$

be column partitions of the matrices $Q_1 Y E I^{n-p}$ and $Q_2 Z E I^{n-q}$. These matrices have orthonormal columns. If $f \in F$ and $g \in G$ are unit vectors, then there exist unit vectors $u \in E^{n-p}$ and $v \in E^{n-q}$ so that $f = Q_1 u$ and $g = Q_2 v$. Thus

$$\begin{aligned} f^T g &= (Q_1 u)^T (Q_2 v) = u^T (Q_1^T Q_2) v = u^T (Y E Z T) v \\ &= (Y^T u)^T E (Z^T v) = \sum_{i=1}^q \sigma_i (y_i^T u)(z_i^T v). \end{aligned} \quad (646)$$

This expression attains its maximal value of $\sigma_1 = \cos(8)$ by setting $u = y_1$ and $v = z_1$. It follows that $f = Q_1 y_1 = J_1$ and $v = Q_2 z_1 = g_1$.

Now assume that $k > 1$ and that the first $k-1$ columns of the matrices in (644) and (645) are known, i.e., $f = [f_1 | \dots | f_{k-1}]$ and $g = [g_1 | \dots | g_{k-1}]$. Consider the problem of maximizing $J^T g$ given that $f = Q_1 u$ and $g = Q_2 v$ are unit vectors that satisfy

$$J^T [J_1 | \dots | f_{-k}] = 0$$

$$g^T [g_1 | \dots | g_{k-1}] = 0.$$

It follows from (646) that

$$J^T g = \sum_{i=k}^q \sigma_i (y_i^T u)(z_i^T v) \leq \sigma_k \sum_{i=k}^q \|y_i\| \|z_i\| \leq M.$$

This expression attains its maximal value of $\sigma_k = \cos(B)$ by setting $u = y_k$ and $v = z_k$. It follows from (644) and (645) that $f = Q_1 y_k = f_k$ and $g = Q_2 z_k = g_k$. Combining these observations we obtain

c kE GRPt 6.4.3 (Principal Angles and Vectors) Given $A \in \mathbb{R}^{m \times p}$ and $B \in \mathbb{R}^{m \times q}$ ($p, q \geq 1$) each with linearly independent columns, the following algorithm computes the cosines of the principal angles $\cos(\theta_1), \dots, \cos(\theta_q)$ between $\text{ran}(A)$ and $\text{ran}(B)$. The vectors J, \dots, f_q and g_1, \dots, g_q are the associated principal vectors.

Compute the thin QR factorizations $A = Q^T R_A$ and $B = Q^T R_B$.

$$C = \overline{Q} Q_a$$

Compute the SVD $Y^T C Z = \text{diag}(\cos(B_k))$.

QAY(:,l:q) = [l...lfq]

$$Q_a Z(:, l:q) = [g_{1l} \dots g_{ql}]$$

The idea of using the SVD to compute the principal angles and vectors is due to Björck and Golub (1973). The problem of rank deficiency in A and B is also treated in this paper. Principal angles and vectors arise in many important statistical applications. The largest principal angle is related to the notion of distance between equidimensional subspaces that we discussed in §2.5.3. If $p = q$ then

$$\text{dist}(F, G) = \sqrt{1 - \cos(B_p)^2} = \sin(\theta_p).$$

6.4.4 f x xu m t rux

In light of the following theorem, Algorithm 643 can also be used to compute an orthogonal basis for $\text{ran}(A) \cap \text{ran}(B)$ where $A \in \mathbb{R}^{m \times p}$ and $B \in \mathbb{R}^{m \times q}$.

Theorem 6.4.2. Let $\{\cos(B_i)\}_{i=1}^n$ and $\{f_i, g_i\}_{i=1}^n$ be defined by Algorithm 6.4.3. If the index s is defined by $1 = \cos(B_1) = \dots = \cos(l_s) > \cos(l_{s+1})$, then

$$\text{ran}(A)_n \cap \text{ran}(B) = \text{span}\{f_1, \dots, f_s\} = \text{span}\{g_1, \dots, g_s\}.$$

Pr of. The proof follows from the observation that if $\cos(\mathbf{i}) = 1$, then $f_i = 9$. H

The practical determination of the intersection dimension s requires a definition of what it means for a computed singular value to equal 1. For example, a computed singular value $\hat{\sigma}_i = \cos(\hat{\theta}_i)$ could be regarded as a unit singular value if $\hat{\sigma}_i = 1 - 8f$ for some intelligently chosen small parameter f .

Problems

P6.4.1 ..sfo ... 1b4)- B ..' R_{i.+.}, ' ,Pi $\frac{T}{4}+m$ +m')

$$\sum_{Q^T Q = I_p} \mathbf{V} \mathbf{x} \mathbf{x}^\top \|A - BQ\|_F^2 = \sum_{i=1}^p (\sigma_i(A)^2 - 2\sigma_i(B^T A) + \sigma_i(B)^2).$$

P6.4.2, Pb.8.. 1..s.1.. 81PNQ.. 1.. s...s... s..s.s...es ..1.. .null(A₁) n .. n null(A_s)

P6.4.3 Pb8.. 1..s.1.6. 8aPN96.. 1. .6...8 .68 f8.8. A.)- B ..) - , T')+

P6.4.4 e8.1b.P - s RPla055

$$\text{P6.4.5} \quad \boxed{\text{2.8}} \quad A, B \in \mathbb{R}^{m \times n}, "2," A \\ \text{if } +_{\mathbb{F}} \quad X \in \mathbb{R}^{n \times n} \text{ "2," ("n" in } AX - B \text{ iff } \text{WinA, (b1 in A)} \text{ n/A = i } (A.$$

$$\begin{aligned} & \text{P6.4.6 m3c.8..8n .. . 8b8. .8fe .8.8..cnc .8.NR.538fo... 1b} \in \mathbf{R}^{m \times n}, ") e \in \mathbf{R}^m \\ & n \left(\begin{array}{c} n \\ n \end{array} \right) \left(\begin{array}{c} m \\ m \end{array} \right) \left(\begin{array}{c} 6 \\ 2 \\ 2 \\ v \\ C^T e / m \\ i_3 \\ i_4 \\ T_1 \\ T_2 \end{array} \right) \left(\begin{array}{c} C - ev^T \\ I \\ F \\ 6X \\ = \\ 1 \\ A \\ A \in \mathbf{R}^{m \times n}, ") B \in \mathbf{R}^{m \times n} \end{array} \right) \end{aligned}$$

$$Q^T Q = I_n, v \in R^n$$

$$\frac{2e}{\epsilon} = \frac{1}{\epsilon} e \lambda p t = C - B T e/m \quad Q_{opt} = U \Sigma V^T \frac{T}{\epsilon} c - \left[b_{C_2 S} + c^- + c_{C_2 S} + c_{C_2 S + 1} \right] B^T (I - ee^T/m) A = UV^T$$

$$\text{P6.4.7 1} \quad 91.. \quad 9 \quad n..cb\text{ROPR matr} \quad M1 = Q \quad xy^T \left(\begin{array}{c} (1c_2 s_* T_1 Q_s \in \\ x^3 \cdot \leq -) \end{array} \right) \leq (-, x, y \in \mathbf{R}^3) \bullet$$

$$U, V \in \mathbf{R}^{3 \times 3}$$

$$Q = Q(\theta_1, \theta_2, \theta_3)$$

$$Q(\theta_1, \theta_2, \theta_3) = \begin{bmatrix} 1 & \vdots & \vdots \\ 0 & c_1 & s_1 \\ 0 & -s_1 & c_1 \end{bmatrix} \begin{bmatrix} c_2 & s_2 & 0 \\ -s_2 & c_2 & 0 \\ 0 & s & 1 \end{bmatrix} \begin{bmatrix} \vdots & \vdots & 0 \\ s & c_3 & s_3 \\ s & -s_3 & c_3 \end{bmatrix}$$

$$= \begin{bmatrix} s_2 c_3 & s_2 s_3 \\ -c_1 s_2 & c_1 c_2 c_3 - s_1 s_3 & c_1 c_2 s_3 + \dots_3 \\ s_1 s_2 & -s_1 c_2 c_3 - c_1 s_3 & -s_1 c_2 s_3 & c_1 c_3 \end{bmatrix}.$$

$$\begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} = Q(\theta_1, \theta_2, \theta_3) - xy^T, \quad x, y \in \mathbf{R}^3$$

$$xy^T = \begin{bmatrix} s_2 \\ \mu c_1 \\ -\mu s_1 \end{bmatrix} \begin{bmatrix} -s_2/\mu \\ c_3 \\ s_3 \end{bmatrix}^T$$

$$\left[\begin{array}{c} c_2 - \mu \\ c_1 s_3 \\ c_1 s_3 \end{array} \right] = \left[\begin{array}{c} c_2 - \mu \\ s_1 c_3 \\ s_1 c_3 \end{array} \right].$$

$$v_w = U_c^T v, \quad v_1 = U_c U_c^T v, \quad v_2 = (I_n - U_c U_c^T) v.$$

$$z_\theta = \begin{pmatrix} \cdot & \cdot \\ -4 & 1-1 \\ \prod_{v_1} & \prod_w \end{pmatrix} v_1 + \begin{pmatrix} \cdot & \cdot \\ \prod_{v_2} & 1 \\ \prod_w & \end{pmatrix} v_2$$

,(+

$$U_\theta = (\mathbf{1}_n - z_\theta v^T) U_c,$$

$${}_{1c2s3}U_\theta^T U_\theta = I_d. \quad 10) 1 U_\theta U_\theta^T [{}_{1c3} c + {}_{c+3} c -) c | {}_{1c2s3} B^{-1} {}_{1c2s} {}_{31c2s} {}_{1c2s} {}_{31c2s} {}_{1c2s3} {}_{1c3}$$

$$\text{dist}_F(\text{ran}(V), \text{ran}(W)) = \|VV^T - WW^T\|_F$$

- $\mathbf{M} \mathbf{x} \mathbf{v}, W \in \mathbb{C}^{\mathbb{E} \times \mathbb{E}}$ $\mathbf{x} \mathbf{d}_{\text{CR}2000}(\mathbf{r} \mathbf{a} \mathbf{R} = \mathbf{R}(1\mathbf{a}) - \mathbf{R}) \mathbf{x} = /-$

$$\text{dist}_F(\text{ran}(V), \text{ran}(W))^2 = \|d - \|W^T V\|_F^2\| = \sum_{i=1}^d (1 - \sigma_i(W^T V)^2).$$

$$j = \text{dist}(\text{ran}(V), \text{ran}(W))^2 = \sigma_1(W^T V)^2. (0) = \text{vkNnk}$$

$$d_\theta^2 = d_c^2 - \text{tr}(U_* U_*^T (U_\theta U_\theta^T - U_c U_c^T))$$

$$k = \text{dist}_F(\text{ran}(U_*), \text{ran}(U_\theta)) [\lambda_1^T d_c = \text{dist}_F(\text{ran}(U_*), \text{ran}(U_c)). (1) = \text{vkNnkP}$$

$$y_\theta = \begin{cases} 0 & v_1 \\ \frac{v_2}{\|v_2\|} & v_2 \end{cases} + o(0)$$

k) E

$$U_\theta U_\theta^T - U_c U_c^T = y_\theta y_\theta^T - \frac{v_1 v_1^T}{v_1^T v_1}$$

o) D

$$d_8^2 = d_c^2 + \left(\frac{\|U_*^T v_1\|_2^2}{\|v_1\|_2^2} - \|U_*^T y_\theta\|_2^2 \right).$$

$$(0) N \cdot \text{vkNnk} + f \text{ExvOKNx} = -(nkfKE)$$

$$KE \cdot \Phi \left(\frac{\|P_S v_2\|^2}{\|v_2\|_2^2} - \frac{\|P_S v_1\|^2}{\|v_1\|_2^2} \right) + M + \frac{v_1^T P_S v_1}{\|v_1\|_2 \|v_2\|_2} = 0 \text{fo } P_S = U_* U_*^T.$$

Notes and References for

N N.N.8.383.8 ...8..8..Nn .. .2.8

- 44, ..7.1. f 04P Dk.,, C., P...» .4., .i., D4Ni „ „ D 1
Psychometrika 17, J.r. J4
SplAlo8a= (774.3 Aa2uog23lghp21 pmApprlaoghH19Hhtp2tulS,AiN Psychome-
tr ka 31, Nr N4
9f Atla oae14f 2Hu2 (d.4.3aqt21: l2Chu2i2apt C2epp22arhgoBla2h2E29 ii
Sl2p2a6SIAM J. Sci. Stat. Comput. 2, 575r 5T,4
29f.2moi = (dd4.lmAstii2pun9ul9H1tp2tulSg216BIT 28, N5v,5.
.9 hoHq (.4.3 hou,2grlH2pmu pp2 aSoga92e ,Hpprlaog hhi9SISp2x
Comput. 17, N5 J54
n 88eAHtliae149g n8NT.4.3 Alatpuo282eu19uh1p21S,2iuNIAJ. Matr. Anal.
Appl. 18, (I,N(5.4
n4deia oae.Mhouq= N. 4.3 hul9uhtp2HlSg21a pp2sp222Ioa21,euNumer. Math. 82,
...r 7N.4
34a ' leca98p 34n1 plSluton(... lmAhul9uhtp2tulSg21u ,upmlrl8osp22g opun92tuN
SIAM J. Sci. Comput. 21, (I,N J54
8B = 1, pm28m2ul9ultp2tulSg21ilhapt pl r eaarpmp2lt2pluprlaogio pu2b1p2
9iShpop28tuAgop2lpp2lgoA9iSl2p24SulSgApmpop 91ane217.4.454H2 d12
C29H2H2a921T
n ' e d194M lk2AT(4.38 8p2uop2Alunppiu AliShp2arpp2 21pp2op2 oa
,upmlrlaodopu2b16M J. Numer. Anal. 8, 5dr7,4
4f 4 .2epd7.4.AliShp2arpp2lgoHE29iSl2p24k2pp3SSg29op2atsSIAM J. Sci. Stat.
Comput. 7, (N,F (T4
s2ar s1mopl tlg2pm2arg2t,SApk22ar t1SSS18g21e29ht2eaT
n ' e oafq .4M .lg1S54.2hi2u29o12pmletlu AliShp2a3ar,2t' 2pk22a2o1Ui
sS92tuMath. Comput. 27, .T.r. "9
n9 41Au82a1c8e..14 n1q= (d. 4. Acal29ogluu2gopr18e.2a2uogn3SET 3SSg29op2 at
cae2Ak3grlupu61. Comput. Appl. Math. 27, 5TN4
.44.lghS oae.Mpo= N. 4.h2uphuSop21tt21: pp2Acal29AluH2,op2atopH1ho2HsuN
Lin. Alg. Applic. 210 5Id4

,L qnp 9222 L Ha Phapc nGAAe(is ea Gope(181pStena Gt65 SIAM J. Matrix
 Anal. Appl. 22, N5N...
 3xBzatoAioae44943MrAapeHJ..,hu2AScar,ASApkAAhStS Aata3 wCAAM
 hulehAk8rlunppCaehAupl uSopStnC opAtuM J. Sci. Comput. 23, IJJdr IJ, J4
 h4spMISoAhd.4.r SeopnpmA HsB3AgEAACSIpopnlStMIAAehpp nail,iAt
 Su2AScar,AhtentAI tAae
 8aMAelAAe.MtarMApnsSeApt AllaAAopMnbraoploipMihbalntAlStAMiqp2a
 ppMihop CopMhpC tSA2-,Me Mq43a tie,SCAEALCSIpopnlStMIAAehpp nail,iAt
 Su2AScar,AhtentAI tAae
 n. 9,eia oaeW4oiC = JJ,4 .3mAobnCI i nnqA,nm 1p2Copa2alAelAAeloaqArMA1aatt
 Num Lin. Alg. Applic. 12, T5N,NTi
 1pAs·E mCiat MI,Ap1S,φ 2atpopnptrp 16S hpmiataAe
 s49xiiou,nar = N. 4 .lmA snal,dVBoghEAALiStnpnPa 4,p2ou2opspopnptrp 14CM
 SIGNUM Newsletter 20, II..
 3a or lMnprMAICSHpnpmAlpopnlaeMoqLiaApMnh74Tppop eAorAIAahl
 iqMnhtentAhtAeaT
 14 sAnAMSMSQ= OBU).O= M22p Dj = n.θ= l= i)Eθ= xfp = eODy),= = OX)=)2
 = O=d Math Imaging Vision 33, NX,4
 lMoCIMApn,SII ppmAtpnCopSMISgCtIA2opA2ph7,,4dtAAk
 n4wo,3ahd4 2lkqu oaeW4 IAAnN J,4g2A8eAapn= Aqadla VoAqhasIStSoAAl
 .2mt8aALCSgAp1Mopnl6r ceedings of the Allerton Conference on Communication, Con-
 trl, and Computing 2010

6.5 Updating Matrix Factorizations

In many applications it is necessary to refactor a given matrix $A \in \mathbb{R}^{m \times n}$ after it has undergone a small modification. For example, given that we have the QR factorization of a matrix A , we may require the QR factorization of the matrix \tilde{A} obtained from A by appending a row or column or deleting a row or column. In this section we show that in situations like these, it is much more efficient to "update" A 's QR factorization than to generate the required QR factorization of \tilde{A} from scratch. Givens rotations have a prominent role to play. In addition to discussing various update QR strategies, we show how to update a Cholesky factorization using hyperbolic rotations and how to update a rank-revealing ULV decomposition.

nmimhb Wtd ihb.utt" Seb

Suppose we have the QR factorization $QR = A \in \mathbb{R}^{m \times n}$ and that we need to compute the QR factorization $\tilde{A} = A + uv^T = QR_1$ where $u, v \in \mathbb{R}^n$ are given. Observe that

$$\tilde{A} = A + uv^T = Q(R + wv^T) \quad (6.5.1)$$

where $w = Q^T u$. Suppose rotations J_1, \dots, J_{n-1} are computed such that

$$J_1^T \cdots J_{n-1}^T w = \pm \|w\|_2 e_1.$$

where each J_k is a Givens rotation in planes k and $k+1$. If these same rotations are applied to R , then

$$H = J_1^T \cdots J_{n-1}^T R \quad (6.5.2)$$

is upper Hessenberg. For example, in the $n = 4$ case we start with

$$w \leftarrow \begin{bmatrix} x \\ x \\ x \\ x \end{bmatrix}, \quad R \leftarrow \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \end{bmatrix},$$

and then update as follows

$$\begin{aligned} w_+ \quad J[w] &= \begin{bmatrix} x \\ x \\ x \\ 0 \end{bmatrix}, \quad R_+ \quad J[R] = \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & x & x \end{bmatrix}, \\ w \leftarrow J_2^T w &= \begin{bmatrix} x \\ x \\ 0 \\ 0 \end{bmatrix}, \quad R \leftarrow J_2^T R = \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{bmatrix}, \\ w_+ \quad J[w] &= \begin{bmatrix} x \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad H_+ \quad J[R] = \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{bmatrix}. \end{aligned}$$

Consequently,

$$(J_1^T \cdots J_{n-1}^T)(R + wv^T) = H \pm \|w\|_2 e_1 v^T = H_1 \quad (653)$$

is also upper Hessenberg. Following Algorithm 5.24 we compute Givens rotations G_k , $k = 1:n-1$ such that $G_{-1} \cdots G_{H_1} = R_1$ is upper triangular. Combining everything we obtain the QR factorization $A = A + wv^T = QR_1$ where

$$Q = Q J_{n-1} \cdots J_1 G_1 \cdots G_{n-1}$$

A careful assessment of the work reveals that about $2nr^2$ fops are required.

The technique readily extends to the case when A is rectangular. It can also be generalized to compute the QR factorization of $A + uvr$ where $u \in \mathbb{R}^{m \times p}$ and $v \in \mathbb{R}^{n \times p}$.

6.5.2 $T \times w \quad Yx \times r \quad V$

Assume that we have the QR factorization

$$QR = A = [a_1 | \cdots | a_n], \quad a_i \in \mathbb{R}^m, \quad (654)$$

and for some $k, 1 \leq k \leq n$, partition the upper triangular matrix $R \in \mathbb{R}^{m \times n}$ as follows

$$R = \begin{bmatrix} R_{11} & v & R_{13} \\ 0 & r_{kk} & w^T \\ 0 & 0 & R_{33} \end{bmatrix} \quad \begin{matrix} k \\ 1 \\ m-k \end{matrix}.$$

Now suppose that we want to compute the QR factorization of

$$A = [a_1 | \dots | a_k | z | a_{k+1} | \dots | a_n] \in \mathbb{R}^{m \times (n+1)}.$$

Note that \tilde{A} is just A with its k th column deleted and that

$$Q^T \tilde{A} = \begin{bmatrix} R_{11} & R_{13} \\ 0 & w^T \\ 0 & R_{33} \end{bmatrix} = H$$

is upper Hessenberg eg,

$$H = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad m = 7, n = 6, k = 3$$

Clearly, the unwanted subdiagonal elements $h_{k+1,k}, h_{k+2,k}, \dots, h_{m-1,k}$ can be zeroed by a sequence of Givens rotations $G_1 \dots G_{m-1} H = R_1$. Here, G_i is a rotation in planes i and $i+1$ for $i = kn - 1$. Thus, if $Q = QG_k \dots G_{m-1}$ then $A = QR$ is the QR factorization of A .

The above update procedure can be executed in $O(m^2)$ fops and is very useful in certain least squares problems. For example, one may wish to examine the significance of the k th factor in the underlying model by deleting the k th column of the corresponding data matrix and solving the resulting LS problem.

Analogously, it is possible to update directly the QR factorization of a matrix after a column has been added. Assume that we have (654) but now want the QR factorization of

$$A = [a_1 | \dots | a_k | z | a_{k+1} | \dots | a_n]$$

where $z \in \mathbb{R}^m$ is given. Note that if $w = Q^T z$ then

$$Q^T \tilde{A} = [Q^T a_1 | \dots | Q^T a_k | w | Q^T a_{k+1} | \dots | Q^T a_n]$$

is upper triangular except for the presence of a "spike" in its $(k+1)$ st column, eg,

$$\dots + Q^T A = \begin{bmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & 0 & x \\ 0 & 0 & 0 & x & 0 & 0 \\ 0 & 0 & 0 & x & 0 & 0 \end{bmatrix}, \quad m = 7, n = 5, k = 3$$

It is possible to determine a sequence of Givens rotations that restores the triangular form

$$\bar{A} \leftarrow \bar{A} - J_6 T = \begin{bmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & 0 & x \\ 0 & 0 & 0 & x & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad \bar{A} + J_5 T = \begin{bmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & 0 & x \\ 0 & 0 & 0 & 0 & 0 & x \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$A + J_4 T = \begin{bmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

This update requires $O(m)$ fops

6.5.3 $T \times v$ or $bx \times r$ Row

Suppose we have the QR factorization $QR = AE \in \mathbb{R}^{m \times n}$ and now wish to obtain the QR factorization of

$$\tilde{A} = \begin{bmatrix} w^T \\ A \end{bmatrix}$$

where $w \in \mathbb{R}^n$. Note that

$$\text{diag}(I, Q^T) \tilde{A} = \begin{bmatrix} W \\ R \end{bmatrix} = H$$

is upper Hessenberg. Thus, rotations J_1, \dots, J_n can be determined so $J_1 \cdots J_n H = R$ is upper triangular. It follows that $A = Q_1 R$ is the desired QR factorization, where $Q_1 = \text{diag}(I, QJ_1 \cdots J_n)$. See Algorithm 5.25.

No essential complications result if the new row is added between rows k and $k+1$ of A . Indeed, if

$$\begin{bmatrix} & \end{bmatrix} = QR, \quad A_1 \in \mathbb{R}^{k \times n}, \quad A_2 \in \mathbb{R}^{(m-k) \times n},$$

and

$$P = \begin{bmatrix} 0 & I & 0 \\ I_k & 0 & 0 \\ 0 & 0 & I_{m-k} \end{bmatrix},$$

then

$$\text{diag}(J, Q^T) P \begin{bmatrix} A_1 \\ w^T \\ A_2 \end{bmatrix} = \begin{bmatrix} w^T \\ R \end{bmatrix} = H$$

is upper Hessenberg and we proceed as before.

Finally, we consider how to update the QR factorization $QR = AEI$ given when the first row of A is deleted. In particular, we wish to compute the QR factorization of the submatrix $A_{\bar{i}}$ in

$$A_{\bar{i}} = \begin{bmatrix} & \\ & \end{bmatrix}_{m \times \bar{n}}.$$

(The procedure is similar when an arbitrary row i is deleted.) Let q^T be the first row of Q and compute Givens rotations $G_1, \dots, G_{\bar{m}-1}$ such that $G_1^T \cdots G_{\bar{m}-1}^T q = a\mathbf{e}_1$ where $a = \pm 1$. Note that

$$H = G_1^T \cdots G_{\bar{m}-1}^T R = \begin{bmatrix} v^T \\ R_1 \end{bmatrix}_{m-1}^1$$

is upper Hessenberg and that

$$QG_{\bar{m}-1} \cdots G_1 = \begin{bmatrix} \alpha & 0 \\ 0 & Q_1 \end{bmatrix}$$

where $Q_1 E I (m) x(m)$ is orthogonal. Thus,

$$A = \begin{bmatrix} z^T \\ A_1 \end{bmatrix} = (QG_{\bar{m}-1} \cdots G_1)(G_1^T \cdots G_{\bar{m}-1}^T R) = \begin{bmatrix} \alpha & 0 \\ 0 & Q_1 \end{bmatrix} \begin{bmatrix} v^T \\ R_1 \end{bmatrix}$$

from which we conclude that $A_{\bar{i}} = Q_1 R_i$ is the desired QR factorization.

npinsb .e TPSedKbREntQTIt"bttnbnT atntQTIt"b

Suppose we are given a symmetric positive definite matrix $A E I$ and its Cholesky factor G . In the Cholesky updating problem, the challenge is to compute the Cholesky factorization $\tilde{A} = QR$ where

$$\tilde{A} = A + zz^T, \quad z \in \mathbb{R}^n. \quad (655)$$

Noting that

$$\tilde{A} = \begin{bmatrix} G^T \\ z^T \end{bmatrix}^T \begin{bmatrix} G^T \\ z^T \end{bmatrix}, \quad (656)$$

we can solve this problem by computing a product of Givens rotations $Q = Q_1 \cdots Q_n$ so that

$$Q^T \begin{bmatrix} G^T \\ z^T \end{bmatrix} = \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad R \in \mathbb{R}^{n \times n} \quad (657)$$

is upper triangular. It follows that $\tilde{A} = RRT$ and so the updated Cholesky factor is given by $G = RT$. The zeroing sequence that produces R is straightforward, e.g.,

$$\begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{Q_1} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & \times & \times \end{bmatrix} \xrightarrow{Q_2} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \end{bmatrix} \xrightarrow{Q_3} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & 0 \end{bmatrix}.$$

The Q_k update involves only rows k and $n + 1$. The overall process is essentially the same as the strategy we outlined in the previous subsection for updating the QR factorization of a matrix when a row is appended.

The Cholesky downdating problem involves a different set of tools and a new set of numerical concerns. We are again given a Cholesky factorization $A = GG^T$ and a vector $z \in \mathbb{R}^n$. However, now the challenge is to compute the Cholesky factorization $A = ((T$ where

$$A = A - zz^T \quad (658)$$

is presumed to be positive definite. By introducing the notion of a hyperbolic rotation we can develop a downdating framework that corresponds to the Givens-based updating framework. Define the matrix S as follows

$$S = \begin{bmatrix} & \\ & 1 \end{bmatrix} \quad (659)$$

and note that

$$\tilde{A} = GG^T - zz^T = \begin{bmatrix} G^T \\ z^T \end{bmatrix}^T S \begin{bmatrix} G^T \\ z^T \end{bmatrix}. \quad (6510)$$

This corresponds to (656), but instead of computing the QR factorization (657), we seek a matrix $H \in \mathbb{R}^{(n+1) \times (n+1)}$ that satisfies two properties:

$$HSHT = S, \quad (6511)$$

$$H^T \begin{bmatrix} G^T \\ z^T \end{bmatrix} = \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad R \in \mathbb{R}^{(n+1) \times n} \text{(upper triangular)}. \quad (6512)$$

If this can be accomplished, then it follows from

that the Cholesky factor of $A = A - zz^T$ is given by $G = RT$. A matrix H that satisfies (6511) is said to be S -orthogonal. Note that the product of S -orthogonal matrices is also S -orthogonal.

An important subset of the S -orthogonal matrices are the hyperbolic rotations and here is a 4-by-4 example

$$H_2(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & 0 & -s \\ 0 & 0 & 1 & 0 \\ 0 & -s & 0 & c \end{bmatrix}, \quad c = \cosh(\theta), s = \sinh(\theta).$$

The S -orthogonality of this matrix follows from $\cosh^2(\theta) - \sinh^2(\theta) = 1$. In general, $H \in \mathbb{R}^{(n+1) \times (n+1)}$ is a hyperbolic rotation if it agrees with I_{n+1} except in four locations

$$\begin{bmatrix} [H_k]_{k,k} & H_{kk|n+1} \\ [H_k]_{n+1,k} & H_{n+1|n+1} \end{bmatrix} = \begin{bmatrix} \cosh(\theta) & -\sinh(\theta) \\ -\sinh(\theta) & \cosh(\theta) \end{bmatrix}.$$

Hyperbolic rotations look like Givens rotations and, not surprisingly, can be used to introduce zeros into a vector or matrix. However, upon consideration of the equation

$$\begin{bmatrix} \ddots & \vdots \\ \vdots & \ddots \end{bmatrix} \begin{bmatrix} \ddots \\ \vdots \end{bmatrix} = \begin{bmatrix} \quad \\ \quad \end{bmatrix} \quad w^2 s^2 M 1$$

we see that the required cosh-sinh pair may not exist. Since we always have $|\cosh(Q)| > |\sinh(Q)|$, there is no real solution to $-s\mathbf{x}_1 + \mathbf{x}_2 \geq 0$ if $\|\mathbf{x}_1\| > \|\mathbf{x}_2\|$. On the other hand, if $\|\mathbf{x}_1\| > \|\mathbf{x}_2\|$ then $\{\mathbf{ds}\} = \{\cosh(Q), \sinh(Q)\}$ can be computed as follows

$$T = \frac{\mathbf{x}_2}{\|\mathbf{x}_2\|} \quad C = \frac{1}{\sqrt{1 - T^2}} \quad s = CT \quad (65.13)$$

There are clearly numerical issues if $\|\mathbf{x}_1\|$ is just slightly greater than $\|\mathbf{x}_2\|$. However, it is possible to organize hyperbolic rotation computations successfully, see Alexander, Pan, and Plemmons (1988).

Putting these concerns aside, we show how the matrix \mathbf{g} in (65.12) can be computed as a product of hyperbolic rotations $\mathbf{g} = \mathbf{f} \mathbf{s}^T \mathbf{j} \mathbf{d} \mathbf{G} \mathbf{T}$ just as the transforming \mathbf{Q} in the updating problem is a product of Givens rotations. Consider the role of \mathbf{H}_i in the $n=3$ case.

$$\left[\begin{array}{ccc} & & T \\ 0 & 1 & 0 \\ 0 & 0 & C \end{array} \right] \left[\begin{array}{ccc} g_{11} & & \\ 0 & 0 & \mathbf{g} \\ z_1 & z_2 & z_3 \end{array} \right] = \left[\begin{array}{ccc} \tilde{g}_{11} & \tilde{g}_{21} & \tilde{g}_{31} \\ 0 & g_{22} & g_{32} \\ 0 & 0 & g_{33} \\ 0 & z'_2 & z'_3 \end{array} \right].$$

Since $\mathbf{A} = \mathbf{G}\mathbf{G}^T - \mathbf{z}\mathbf{z}^T$ is positive definite, $[\mathbf{A}]_{11} = 9 > 0$. It follows that $|\mathbf{G}| > \|\mathbf{z}\|$ which guarantees that the cosh-sinh computations (65.13) go through. For the overall process to be defined, we have to guarantee that hyperbolic rotations \mathbf{g} as in (65.12) can be found to zero out the bottom row in the matrix $[\mathbf{G}\mathbf{T} \mathbf{z} \mathbf{G}^T]$. The following theorem measures that this is the case.

Theorem 6.5.1. If

$$A = \begin{bmatrix} & v^T \\ : & B \end{bmatrix} = \begin{bmatrix} 1 \\ \mathbf{i} \\ \vdots \end{bmatrix} \begin{bmatrix} . & \mathbf{1} \end{bmatrix}$$

and

$$\mathbf{A} = A - \begin{bmatrix} : \\ : \end{bmatrix} \begin{bmatrix} : \end{bmatrix}^T$$

is positive definite, then it is possible to determine \mathbf{c} , \mathbf{s} , $\mathbf{cosh}(Q)$ and $\mathbf{s} = \sinh(Q)$ so

$$\begin{bmatrix} c & 0 & -s \\ 0 & I_{n-1} & 0 \\ -s & 0 & c \end{bmatrix} \begin{bmatrix} g_{11} & g_1^T \\ 0 & G_1^T \\ \mu & w^T \end{bmatrix} = \begin{bmatrix} \tilde{g}_{11} & \mathbf{G} \\ 0 & \mathbf{Gf} \\ 0 & \mathbf{wf} \end{bmatrix}.$$

Moreover, the matrix $\mathbf{A} = \mathbf{G}\mathbf{G}^T - \mathbf{w}\mathbf{w}^T$ is positive definite.

P f. The blocks in A's Cholesky factor are given by

$$\mathbf{g}_{11} = \mathbf{y}, \quad \mathbf{g}_1 = \mathbf{v}/\mathbf{g}_{11}, \quad \mathbf{G}\mathbf{G}^T = \mathbf{B} - \frac{1}{\alpha} \mathbf{w}\mathbf{w}^T \quad (65.14)$$

Since $\mathbf{A} - \mathbf{z}\mathbf{z}^T$ is positive definite, $a_{11} - z_1^2 = g_{11} - \mu^2 > 0$ and from (65.13) with $r = \mu/g_{11}$ we see that

$$c = \frac{\sqrt{\alpha}}{\sqrt{\alpha - \mu^2}}, \quad s = \frac{\mu}{\sqrt{\alpha - \mu^2}}.$$

Since $\mathbf{w} = -sg_1 + cw$ it follows from (65.14) and (65.15) that

$$\begin{aligned} \mathbf{A}_1 &= \mathbf{G}\mathbf{G}^T - \mathbf{w}\mathbf{w}^T = \mathbf{B} - \frac{1}{\alpha} \mathbf{w}\mathbf{w}^T - \left(\begin{matrix} s & g_1 \\ g_1 & c \end{matrix} \right) \left(\begin{matrix} s & g_1 \\ g_1 & c \end{matrix} \right)^T \\ &= \mathbf{B} - \mathcal{C}_2^{sc} \end{aligned}$$

$$U^T A V = \begin{bmatrix} L \\ 0 \end{bmatrix} = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \\ 0 & 0 \end{bmatrix}, \quad U^T U = I_m, \quad V^T V = I_n$$

$$U^T A \Pi = \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad R \in \mathbb{R}^{n \times n}$$

followed by a QR factorization $V \{ \mathbf{R} \Gamma = \mathbf{L} \mathbf{T}$. In this case the matrix V in (65.16) is given by $V = \mathbf{I}_{\mathbf{m}} V_1$. The parameter r is the estimated rank. Note that if

$$V = \begin{bmatrix} V_1 & V_2 \\ \mathbf{r} & \mathbf{m} \mathbf{r} \end{bmatrix}, \quad U = \begin{bmatrix} U_1 & U_2 \\ \mathbf{r} & \mathbf{m} \mathbf{r} \end{bmatrix},$$

then the columns of V_2 define an approximate nullspace:

$$\| AV_2 \|_2 = \| U_2 L_{22} \|_2 = \| L_{22} \|_2.$$

Our goal is to produce cheaply a rank-revealing ULV decomposition for the row-appended matrix

$$\tilde{A} = \begin{bmatrix} A \\ z^T \end{bmatrix},$$

In particular, we show how to revise \mathbf{L} , V , and possibly r in $O(n^2)$ flops. Note that

$$\begin{bmatrix} U & 0 \\ 0 & 1 \end{bmatrix}^T \begin{bmatrix} A \\ z^T \end{bmatrix} V = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \\ 0 & 0 \\ w^T & y^T \end{bmatrix}.$$

We illustrate the key ideas through an example. Suppose $n = 7$ and $r = 4$. By permuting the rows so that the bottom row is just underneath \mathbf{L} , we obtain

$$\begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \\ w^T & y^T \end{bmatrix} = \left[\begin{array}{ccccc|ccc} i & 0 & 0 & 0 & 0 & 0 & 0 \\ i & i & 0 & 0 & 0 & 0 & 0 \\ i & i & i & 0 & 0 & 0 & 0 \\ i & i & i & i & 0 & 0 & 0 \\ \hline \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 0 & 0 \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & 0 \\ \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon & \epsilon \end{array} \right].$$

The ϵ entries are small while the i , w , and y entries are not. Next, a sequence of Givens rotations $\mathbf{G1}, \dots, \mathbf{G4}$ are applied from the left to zero out the bottom row:

$$\left[\begin{array}{c} \tilde{\mathbf{L}} \\ \hline 0 \end{array} \right] = \left[\begin{array}{cccccc|cccc} x & 0 & 0 & 0 & 0 & 0 & 0 \\ x & x & 0 & 0 & 0 & 0 & 0 \\ x & x & x & 0 & 0 & 0 & 0 \\ x & x & x & x & 0 & 0 & 0 \\ x & x & x & x & x & 0 & 0 \\ x & x & x & x & x & x & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right] = \mathbf{G1} \cdots \mathbf{G5} \mathbf{G6} \left[\begin{array}{ccc|cc} \mathbf{Lu} & & & & \\ \mathbf{L2} & & & & \\ \mathbf{W} & \mathbf{Y}^T & & & \end{array} \right].$$

Because this zeroing process intermingles the (presumably large) entries of the bottom row with the entries from each of the other rows, the lower triangular form is typically **not** rank revealing. However, and this is key, we can restore the rank-revealing structure with a combination of condition estimation and Givens zero chasing.

Let us assume that with the added row the new nullspace has dimension 2. With a reliable condition estimator we produce a unit 2-norm vector p such that

$$\| p^T \tilde{L} \|_2 \approx \sigma_{\min}(\tilde{L}).$$

(See §3.5.4). Rotations $\{U_{i,i+1}\}_{i=1}^6$ can be found such that

$$U_{67}^T U_{56}^T U_{45}^T U_{34}^T U_{23}^T U_{12}^T p = \dots, \quad (\dots)$$

Applying these rotations to \tilde{L} produces a lower Hessenberg matrix

$$H = U_{67}^T U_{56}^T U_{45}^T U_{34}^T U_{23}^T U_{12}^T \tilde{L}.$$

Applying more rotations from the right restores H to a lower triangular form

$$L_+ = HV_{12}V_{23}V_{34}V_{45}V_{56}V_{67}.$$

It follows that

$$e_7^T L_+ = (e_8^T H) V_{12}V_{23}V_{34}V_{45}V_{56}V_{67} = \begin{pmatrix} p^T & V_{12}V_{23}V_{34}V_{45}V_{56}V_{67} \end{pmatrix}$$

has approximate norm $\sigma_{\min}(\tilde{L})$. Thus, we obtain a lower triangular matrix of the form

$$L_+ = \left[\begin{array}{cccccc|c} \times & 0 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & 0 & 0 \\ \times & \times & \times & \times & \times & \times & 0 \\ \hline f & f & f & f & f & f & f \end{array} \right]$$

We can repeat the condition estimation and zeroing on the leading 6-by-6 portion. Assuming that the nullspace of the augmented matrix has dimension two, this produces another row of small numbers:

$$\left[\begin{array}{cccccc|c} \times & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & 0 & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & 0 & 0 & 0 & 0 & 0 \\ \times & \times & \times & \times & 0 & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \hline f & f & f & f & f & f & 0 & 0 \\ f & f & f & f & f & f & f & f \end{array} \right].$$

This illustrates how we can restore any lower triangular matrix to rank-revealing form.

Problems

- P6.5.1. **2.8.8 for 8.8.8 to 8.8.8.** $M \in \mathbb{R}^{m \times n}, () \in \mathbb{C}_4, \mathbb{C}_4, \gamma$
 $\beta \in \mathbb{R}, (A + uv^T)x - b \ll 2$
 $z \in \mathbb{R}^n$

- Acte u, bE \mathbb{R}^m) $^{n'}$ v E \mathbb{R}^n) $^{n'}$] $^{-n_1-n_2}$ $E^{(-n_1-n_2)}$) $^{n'}$ 2) $p_1^{(n)}$) $\mathbb{R}^{m \times p_2 - (n)} \quad n_{(1)} \quad k^{p_2 + n_1} \quad n_0^{(n)} \quad)_r \quad ,^{n_1}$
 O(mn) 11St 3thCapmoQ b=1, A = $kRnA\mathbf{x}$

P6.5.28.8

$$A = [C], \quad c \in \mathbb{R}^n, \quad B \in \mathbb{R}^{(m-1) \times n}$$

$$\frac{O}{\text{Un}(B)} : \frac{\cdot}{\text{Un}(A)} + \frac{"(\text{ATA})^{-1}c}{N_c T(\text{ATA})^{-1}c} \mathbf{J}$$

P6.5.3 1. . b...8. 8b .. 6fo.., ..,o 01.8.8b.8 6..8..881..8.,18. ..8...8. ..

aP5.95

P6.5.4 1....8

$$A = \begin{bmatrix} & \\ \vdots & : J \end{bmatrix} \quad p = \liminf_{n \rightarrow \infty} \frac{\log \lambda_n}{\log n} < 1$$

vNOOR nEDE ipaAJ.iIahJlp tJit Mn

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}$$

$\frac{1}{2} \langle x | C(0) R I - R) x \rangle$

$$\begin{bmatrix} R & H \\ 0 & E \end{bmatrix} \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix} = \begin{bmatrix} R_1 & 0 \\ H_1 & E_1 \end{bmatrix},$$

nCA)WW12: PIH 12

P6.5.5 $\mathbf{A} \in \mathbb{R}^{m \times n}$ $\mathbf{b} \in \mathbb{R}^m$ $\mathbf{x} \in \mathbb{R}^n$ $\mathbf{m} \in \mathbb{R}^m$ $\mathbf{n} \in \mathbb{R}^n$ $\mathbf{G} \in \mathbb{R}^{2m \times 2n}$ \mathbf{A} and \mathbf{b} define least squares; \mathbf{G} , \mathbf{A} , \mathbf{b} , \mathbf{c} define linear system.

$$\mathbf{a}_\cdot = (\mathbf{b} - \mathbf{A}\mathbf{u})^T \mathbf{J}(\mathbf{b} - \mathbf{A}\mathbf{x}),$$

- CAFA

$$S = \begin{bmatrix} I_p & 0 \\ 0 & -I_q \end{bmatrix}, \quad p+q=m.$$

6pt Ctl CAepmap; : Noaeq ; : Nra wt poqlapmAMoe1lap tmklkpmqpmans SulS₁KC
oI alchAt,I nnlan.oea lat c.ATSA T=I=TrQAA)QAA X==bA rGtAV+j k(JAb Ci R
=)JA =I= n)4+C(C(- haR),A I-x ,1 a(bln)1 rCm ,C(A-1)EMRnT)(, Qf Q1- Qf Q2
-CAxA

- CAxA

$$A = \begin{bmatrix} & \\ & \end{bmatrix}, \quad Q_1 \in \mathbf{R}^{p \times n}, Q_2 \in \mathbf{R}^{q \times n}$$

$\mathbb{F} \text{ nCACT} \oplus \text{Ran}(QQT)^\perp \text{ aX } 1 \text{ HQRER}^{m \times m} \text{ ('S-orthogonal n.QSQT = S V,}$

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}_{q,p}$$

$x^T a = x_1 a_1 + \dots + x_n a_n$

$$Q_{11}u = I_p + Q_{12}, \quad Q_{12} = Q_{122} \quad Q_{222} = I_q + Q_{22}$$

-C=HCA-T)I=R R==A=Qu R)xQ22RxAA2AE-hR 11AxCR)J I==bAnyRp ;:q wt oagl
klpmn mlk ~~oAICAT~~lpmlAtposS,nt2A71r.,rlmlkpmpqmHAnlypmlrlacopMIA
U2 V₁ V₂=aC rCrn

$$\left[\begin{array}{cc} U_1 & 0 \\ 0 & U_2 \end{array} \right]^T Q \left[\begin{array}{cc} V_1 & 0 \\ 0 & V_2 \end{array} \right] = \left[\begin{array}{cc|c} D & - & (D^2 - I)^{1/2} \\ 0 & I_{p-q} & 0 \\ \hline (D^2 - I_n)^{1/2} & \mathbf{n} & D \end{array} \right]$$

$\mathbf{z}_{iD} = \mathbf{x}^T \mathbf{P} - \mathbf{P}_D \mathbf{d}_{kpm}$; : Nr=1:p, l2nt ctpm Hyperbolic CS decomposition oae eApot AcaSAIae naspAkouneBoaElluAaJJZ.

Notes and References for

.8.88..88...p b..8..ot..8.l...8 ...8. ..or

NesR Nges.R R2...R ... N 1.I..8.. 3d. P 2R8.8 N. R8..>.d R...p
..8 ..ot..8.R Math Comput. 28, J.r.5.r

fapnMA,AiMappA AplMn.ipSleipASMSpAr SMlr SSpApmAiAgISt r AaJCC2
AkLlar Applae pmAr SAF Apm14OcaASMSr Mir f appA/AaxAi pnaAllyAr
r I,SAI0AcapASk tjetSie.l8MpenAapto poetSBtJaSnaeptra. rl.a- SntaAk-

• Numer. Math. 16, (1971), 5,9
1d. cpg I,MVltuae n.3r si,aeAM(Pa .Anmle,MAlr S,pcne nec(car pmAEB
.plM, I. i ni pMdMatF Comput. 29, μJμr (JPPr

4. 10eJMSl. P,ar ..AplMABiMniS0ApMcAApple,Mr aAla,pMicaSpcgc.ipnIX Math
Comput. 30, P.7id((r

'.d. EAaaniae r.w.sAmalSA(0d5a Numerical Methods for Unconstrained Optimization and
Nonlinear Equations, hMAapiA9ar OAAlleAgc529.

,.a. ir AM.d.a.r SeipcappAaiAM,IAi ncplM6SIAM Review 31, II μF5.r
p.5. dpeAM,iApe n.3. siIaeAM(.I a .3 wpABhr r SeipAniMr A,sAiGAAinMlr Mir
r caSIAM J. Matrix Anal. Appl. 13, (r J).

r Seipnai,I,A,capmACp ,ciMppnAIAC,A,/ ecae

'xEiaApa.w. Mir rn. 5iLr iaiae .. a. sp3cMpNP7r .rAlMpmlr lain.ipaspiSOA
30r lMnpMir Sepcap.A.MignsAyr Vepi AplMcdp6Math. Comput. 30, PPIvP.

s. Vc k.(ddax.lAAhM,ciACpslIMASgr McpnlMncaAiMAApclalMSg ,SAM J. Matrix
Anal. Applic. 9, 51-55dr

n0w)CMA+hiM+. inelpeiat(., axo3AAxMepAipcarl. nAc,psshiMAslpInla,6 SIAM
J. Matr x Anal. Applic. 15, ,r .7dx

s.' ,p,ia,+ t). 9.n,nA,i+. iaea. vleiaA.t+(., ar.riaBn+llec Aipcla Apmi,MrAAIM2
,ciAnACp ssiIMASg,6 Numer. Al9 7, 51.'2, .

n. dpeiaiae .. hiMh., a .wpIAAlaeipcr l. nACsseiMAslOIpcda6SIAM J. Matrix Anal.
Applic. 15, (J(d 5J,

5iOr iaOpAMciiaiAMtr. SImppMA,pcr ippan7ipA i pcaAiM etaiogAgnappA
SMA,AbAaAc,A Ba cp0raipcar pi' cr0AgAapipplqncqllA, ,SeipcaipmWl JApI Mcipcla
a iAilInaSglASiaeAer ipMdfciArae

c.c. Xa (a - Df yPa - x204CDS0a02 p aZ .Z 0.a = l(= E = 00-00a = a TQE+ = OT +
+ + Q= = • = va == aer.a= =SIAM . Numer. Anal. 14, (J) μ.2r

:pAAmlOA,etaei pc当地 OnpAMiM A eA,e

.a. spAiMh. Pa .lmA9 AApl,Il haecardMMMa 3gr lMcpMEl.aeipcar i A.lOA,+ t
iApI Mqql5G Inst. Math. Applic. 23, IJ5r(5.

3., wl)iaA.t+r.h. wMAApBiaEIMaiae..l. eA .llrh.(dPr.3 2lpAla El.aeipnarppA
AplpA+,iApI Mqql6SIAM J. Sci. Stat. Comput. tel (J)INr

c.-T. Xa x22 JDyX0 = - - -yna A01 (N = nr.,l 3-0 = -e = rOR Be == (ba > N
Lin. Alg. Applic. 183, (J) N..

n. peiae.. hiMh., a .hAMp,MSipalpt,c, MwA+El.aeipcarl. i AmlpA,EtAAlr S
,npaF Numer. Math. 68, ,Pr,7de

.l . ,SImMaiae n. siA h.. a .3 2, 3SSMliAp str r ApMBA,aA: SeipcarIMF J.
Numer. Anal. 19, ,Pr JPr

.pVIcapiaiM,Mapd.3. Bia.HoaHJJd r Seinnia nr. i ApI Mcipdanhcilpcar ACM
T ns. Math. Sof w. 35(2), 3MpcApa

tQ2MSlpcA pMia,MgpaA,A,AAA,,80gAe cai alr SAMppnar ,e

P... lgIS h. 7.au.Ji pMEAAlgSl,r pciae spipc,pcAD SI pipcla statistical Computation
Ae I.A. ncpplaie 9.3.2AOeAMieAr cA 2MArMSS.5,'5P4

r... lieAMiae 3u.. spAcamiMepd ..tSAMSIpl/HmlOeMia,Mr ,QQAM J. Matrix
Anal. Applic. 9, I7.r I.J

OEB0xotxn IBCBPa. cat TBBPxn nag ly, 71B hougg 180 Txpple TdS Gong tu
 cxixp Tr=oxia (Sh) sMu kn Chao Phigegiah5 Lin. Alg. and Its Applic. 98, „M
 MATS 2aBlae IMw2MfMfJ. M..ts 2MS10r1A2pl0e2M3prlMnppM iAlIMcasIM,Ap,MAe
 nipMnA SIAM J. Matr. Anal. Applic. 11, „r, IJM
 3MaM iaaAbB.rMaa. iae 3M, M2capiMepL. Mdbc,L2aA2 ppAtS 2MSlpAa,piMBip,2
 E2AliSlncld6 Lin. Alg. Applic. 185, I NJM
 sMpiaeMC 2BiMf4. iae 3M, Mfie f Nl. M3 sLiSpAed1An2a0r1MnlMppAae2= anpa
 rnaAiMcss,iM2hMlSp2 SIAM J. Matr. Anal. Applic. 20, „r, 71M
 3M9MiaA.tB 2M9Mpii .iae .Mp2p LJ,i. MslOinad p2f ae2= 2h2Ciss,iM2,hMlSp2
 St.tS 2MSlpA.-LMciLnL6 SIAM J. Matr. Anal. Applic. 24, „r, „NM
 3M) iaaAB. 2M9Mpii .iae .Mhp20J.J,SMlp2 ds,iOcLAla,LMicaAae2= am2Cpsl iM2,
 hMISp2ap2Mtae 3prlMcLp6 BIT 43, „r, „(M
 IMsp2iMiae hBiaEllMafIJ 7.M „,ai Ap2Mc.ipdlatS 2MSlpAa acpiMia,Mr i
 Ldca,calLlLipda,6 SIAM J. Matr. Anal. Applic. 27, d7r dJM
 2M9Mpir f J.J.r .9,MLplrlaiGipMcA2MIS 2Mpc2AaAlmcl.6 SIAM Review 45, „r, „NM
 .nrp,SAM,Mr daA2 C,IAcI L2ppVr ,SeipnaiM2nA,„2ecae
 wMAAp2MerM3BiaE2 2c JIJMhiMiOp20,AlMAAlr S,piInlaer SeiLnarpp2
 Vr iApdMnlika.6 ACM T ns. Math. Sof w. 31, 7Jr (d
 r Seipcaiae el.aeipcar pp2 nBiae r b e2Alr Sl,cl,dae M2pid 125cAiMali2MAae
 AM.MhApiae .MfPfMf5 N..M,a r Seipcaisraigs,S,SiAA,6EEE T ans. Signal Pr c.
 ..7r, „J,M
 .MaM2iMf N. M3a r Sipcar3grMcppMs,S,SiA3hiABcar,6EEE T ns. Signal Pr c.
 49 N, „r, „NM
 .MaM2iMf N. Mr Seipcari riaB,r2i20car nBEAAlr ,Slnla SIAM J. Matr. Anal.
 Applic. 14, „r „r
 .MaMsp2iMf N..Mr Seipcar rB E2Alr Sl,cl,daMip020r6 lld Comp. 20 N,NrIN
 .MhiMiae nM0eiaf N..M.EI.aeipcar p2riaBnrAi20rr rB E2Alr ,Sldas SIAM J.
 Matr. Anal. Applic. 16 N,eN, „M
 9MmMMpliae .MdMSif IJ.M.Ilec= iSp2l.r riaBSSMlbnr ipdalilM6 Num. Lin.
 Alg. Applic. 16 d, d7JM
 .Lp2Map2M2,8aip M2pid 125cAaAg,efp2Seipcait. Alaenpla A,Lnr ii2, „22e
 aMfMAMar.M.Mp,S. iae rM9M2r r 1f,NN.M3eiSpcl2 niaAllAIple ,MrAA,M,ciA
 Alaenpnkpcr Linla, Numerical Algor thms 1, N,IJM
 .MspMf5 iae wMpf NI.M3eiSpcl2AlaecdLndLcr ipclMriaB,,aA r SeiL2. Vr
 i ALIMmla, SIAM J. Matr. Anal. Applic. 13, NT, „I (dM
 EM9M2MA2 rM9M r faN. MCP 3eiSLnL2aecdLndLcipcla.6 SIAM J. Matrix Anal.
 Applic. 13 I, „NM
 iae LpASeiLcar ,lp,LnlaLlAla,pMinap2CJs,iMA,SMISp2r ,e
 56 sApcppBhB9MfIAlM.. M3 i AplMn.iII21ple,MI pA1QLcd. AlapMinap2a2iM
 n2Cps,iMAhMISpAB0gtar ,MslS,2sh2aEiLiApia2,6 Numer. Math. 31, „Nr,7,M
 n w)CMAl. M3 .2a2Mip SeiLn3prlMnppfAlapMica2naAimLss,iM2,hMISp26m
 SIAM J. Sci. Stat. Comput. 5, „r, „JIM
 .cailOpt2 i2aLnldLp2pplnarSiSAMaA2Ma2pp sB,Sei Lnare
 IMlla2a. hBiaEllM2a. 2Miae2i00f NI.M3 scar,pMBip,AEEAAlr Sl,cl,daIcar
 3grlMppi.6 SIAM J. Matrix Anal. Applic. 13 NJN, „dM

Chapter 7

Unsymmetric Eigenvalue Problems

7.5 $k \quad x \quad x \quad v \quad Y \quad xu$

7.2 $kx \quad tr \quad n \quad x$

7.3 $k \quad x \quad f \quad x \quad r \quad ,$

7.4 $\dots \quad - \quad -$

7.6 $n \quad x \quad k \quad ru \quad uH \quad T$

7.7 $f \quad r \quad r \quad m \quad t \quad ru \quad V \quad r$

7.8 $n \quad x \quad ex \quad x \quad r \quad xv \quad c \quad x \quad k \quad rt \quad x$

7.9 $er \quad r \quad v \quad k \quad v \quad uc \quad xr \quad k \quad t \quad x$

7.10 $k \quad x \quad v \quad xu \quad r$

Having discussed linear equations and least squares, we now direct our attention to the third major problem area in matrix computations, the algebraic eigenvalue problem. The unsymmetric problem is considered in this chapter and the more agreeable symmetric case in the next.

Our first task is to present the decompositions of Schur and Jordan along with the basic properties of eigenvalues and invariant subspaces. The contrasting behavior of these two decompositions sets the stage for §7.2 in which we investigate how the eigenvalues and invariant subspaces of a matrix are affected by perturbation. Condition numbers are developed that permit estimation of the errors induced by rounding.

The key algorithm of the chapter is the justly famous QR algorithm. This procedure is one of the most complex algorithms presented in the book and its development is spread over three sections. We derive the basic QR iteration in §7.3 as a natural generalization of the simple power method. The next two sections are devoted to making this basic iteration computationally feasible. This involves the introduction of the Hessenberg decomposition in §7.4 and the notion of origin shifts in §7.5.

The QR algorithm computes the real Schur form of a matrix, a canonical form that displays eigenvalues but not eigenvectors. Consequently, additional computations

usually must be performed if information regarding invariant subspaces is desired. In §7.6 which could be subtitled, "What to Do after the Real Schur Form is Calculated," we discuss various invariant subspace calculations that can be performed after the QR algorithm has done its job.

The next two sections are about Schur decomposition challenges. The generalized eigenvalue problem $Ax = \lambda Bx$ is the subject of §7.7. The challenge is to compute the Schur decomposition of $B^{-1}A$ without actually forming the indicated inverse or the product. The product eigenvalue problem is similar, only arbitrarily long sequences of products are considered. This is treated in §7.8 along with the Hamiltonian eigenproblem where the challenge is to compute a Schur form that has a special 2by2 block structure.

In the last section the important notion of pseudospectra is introduced. It is sometimes the case in unsymmetric matrix problems that traditional eigenvalue analysis fails to tell the "whole story" because the eigenvectors basis is ill-conditioned. The pseudospectra framework effectively deals with this issue.

We mention that it is handy to work with complex matrices and vectors in the more theoretical passages that follow. Complex versions of the QR factorization, the singular value decomposition, and the CS decomposition surface in the discussion.

Reading Notes

Knowledge of Chapters 1–3 and §§5.1–§5.2 are assumed. Within this chapter there are the following dependencies:

§7.1	§7.2	§7.3	§7.4	§7.5	§7.6	§7.9
				ff	§7.7	§7.8

Excellent texts for the dense eigenproblem include Chatelin (EOM), Kressner (NMSE), Stewart (MAE), Stewart and Sun (MPA), Watkins (MEP), and Wilkinson (AEP).

7.1 Properties and Decompositions

In this section the background necessary to develop and analyze the eigenvalue algorithms that follow are surveyed. For further details, see Horn and Johnson (MA).

0hhhbb bT"Stf tPDSbtbb (tf t Ttf huD:eEf fSb

The eigenvalues of a matrix $A \in \mathbb{C}^{n \times n}$ are the n roots of its characteristic polynomial $p(z) = \det(zI - A)$. The set of these roots is called the spectrum of A and is denoted by

$$\sigma(A) = \{z : \det(zI - A) = 0\}.$$

If $\sigma(A) = \{\lambda_1, \dots, \lambda_n\}$, then

$$\det(A) = \lambda_1 \lambda_2 \cdots \lambda_n$$

and

$$\operatorname{tr}(A) = \lambda_1 + \cdots + \lambda_n$$

where the trace function, introduced in §6.4.1, is the sum of the diagonal entries, i.e.,

$$\text{tr}(A) = \sum_{i=1}^n a_{ii}.$$

These characterizations of the determinant and the trace follow by looking at the constant term and the coefficient of λ^{n-j} in the characteristic polynomial.

Four other attributes associated with the spectrum of $A \in \mathbb{C}^{n \times n}$ include the

$$\text{Spectral Radius: } p(A) = \max_{\lambda \in \sigma(A)} |\lambda| \quad (7.11)$$

$$\text{Spectral Abscissa: } o(A) = \max_{\lambda \in \sigma(A)} \operatorname{Re}(\lambda), \quad (7.12)$$

$$\text{Numerical Radius: } r(A) = \max_{\lambda \in \sigma(A)} \{ \|x^H A x\| : \|x\|_2 = 1\}, \quad (7.13)$$

$$\text{Numerical Range: } W(A) = \{x^H A x : \|x\|_2 = 1\}. \quad (7.14)$$

The numerical range, which is sometimes referred to as the field of values, obviously includes $A(A)$. It can be shown that $W(A)$ is convex.

If $A \in \mathbb{C}^{n \times n}$, then the nonzero vectors $x \in \mathbb{C}^n$ that satisfy $Ax = \lambda x$ are eigenvectors. More precisely, x is a right eigenvector of A if $Ax = \lambda x$ and a left eigenvector if $x^H A = \lambda x^H$. Unless otherwise stated, "eigenvector" means "right eigenvector."

An eigenvector defines a 1-dimensional subspace that is invariant with respect to premultiplication by A . A subspace $S \subset \mathbb{C}^n$ with the property that

$$x \in S \implies Ax \in S$$

is said to be invariant (for A). Note that if

$$AX = XB, \quad B \in \mathbb{C}^{k \times k}, \quad X \in \mathbb{C}^{n \times k},$$

then $\text{ran}(X)$ is invariant and $B = A - A(XY) = A(YX)$. Thus, if X has full column rank, then $AX = XB$ implies that $A(B) \subseteq A(A)$. If X is square and nonsingular, then A and $B = X^{-1}AX$ are similar; X is a similarity transformation, and $\text{tr}(A) = \text{tr}(B)$.

7.1.2 Yxu

Many eigenvalue computations involve breaking the given problem down into a collection of smaller eigenproblems. The following result is the basis for these reductions.

Lemma 7.1.1. If $T \in \mathbb{C}^{n \times n}$ is partitioned as follows

$$T = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix}; \quad p <$$

then $\text{A}(T) = \text{A}(T_{11}) \cup \text{A}(T_{22})$.

Pr of. Suppose

$$\mathbf{T}\mathbf{x} \left(\begin{bmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} \\ \mathbf{0} & \mathbf{T}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} \right) = \mathbf{A} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix}$$

where $\mathbf{x}_1 \in \mathbb{P}$ and $\mathbf{x}_2 \in \mathbb{Q}$. If $\mathbf{x}_2 = 0$ then $\mathbf{T}_{22}\mathbf{x}_2 = \mathbf{A}\mathbf{x}_2$ and so $\mathbf{A} \in \mathbf{A}(\mathbf{T}_{22})$. If $\mathbf{x}_2 \neq 0$ then $\mathbf{T}_{11}\mathbf{x}_1 \in \mathbf{A}\mathbf{x}_2$ and so $\mathbf{A} \in \mathbf{A}(\mathbf{T}_{11})$. It follows that $\mathbf{A}(\mathbf{T}) \subseteq \mathbf{A}(\mathbf{T}_{11}) \cup \mathbf{A}(\mathbf{T}_{22})$. But since both $\mathbf{A}(\mathbf{T})$ and $\mathbf{A}(\mathbf{T}_{11}) \cup \mathbf{A}(\mathbf{T}_{22})$ have the same cardinality, the two sets are equal. \square

$$\text{Ex1A1A } \frac{1}{2} + \text{i}A \quad A1 \quad m = A9i \quad .M. \quad +G \cdot m \cdot A$$

By using similarity transformations, it is possible to reduce a given matrix to any one of several canonical forms. The canonical forms differ in how they display the eigenvalues and in the kind of invariant subspace information that they provide. Because of their numerical stability we begin by discussing the reductions that can be achieved with unitary similarity.

Lemma 7.1.2. If $\mathbf{A} \in \mathbb{M}^{n,n}$, $\mathbf{B} \in \mathbb{M}^{p,p}$, and $\mathbf{X} \in \mathbb{M}^{n,p}$ satisfy

$$\mathbf{AX} = \mathbf{XB}, \quad \text{rank}(\mathbf{X}) = p, \quad (7.15)$$

then there exists a unitary $\mathbf{Q} \in \mathbb{M}^{n,n}$ such that

$$\mathbf{Q}^H \mathbf{A} \mathbf{Q} = \mathbf{T} \left(\begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \right)_{n-p}^p \quad (7.16)$$

and $\mathbf{A}(\mathbf{T}_{11}) = \mathbf{A}(\mathbf{A})_n \cap \mathbf{A}(\mathbf{B})$.

Pr of. Let

$$\mathbf{x} \in \mathbb{O} \quad \mathbf{Q} \left[\begin{array}{c} \mathbf{1} \\ \vdots \\ \mathbf{1} \end{array} \right], \quad Q \in \mathbb{C}^{n \times n}, \quad R_1 \in \mathbb{C}^{p \times p}$$

be a QR factorization of \mathbf{X} . By substituting this into (7.15) and rearranging we have

$$\begin{bmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} \\ \mathbf{T}_{21} & \mathbf{T}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{1} \\ \vdots \\ \mathbf{1} \end{bmatrix} = \begin{bmatrix} \mathbf{1} \\ \vdots \\ \mathbf{1} \end{bmatrix} \mathbf{B}$$

where

$$\mathbf{Q}^H \mathbf{A} \mathbf{Q} = \begin{bmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} \\ \mathbf{T}_{21} & \mathbf{T}_{22} \end{bmatrix}_{n-p}^p.$$

By using the nonsingularity of R_1 and the equations $\mathbf{T}_{21}R_1 = 0$ and $\mathbf{T}_{11}R_1 = RB$ we can conclude that $\mathbf{T}_{21} = 0$ and $\mathbf{A}(\mathbf{T}_{11}) \subseteq \mathbf{A}(\mathbf{B})$. The lemma follows because from Lemma 7.1.1 we have $\mathbf{A}(\mathbf{A}) \cap \mathbf{A}(\mathbf{T}) = \mathbf{A}(\mathbf{T}_{11}) \cup \mathbf{A}(\mathbf{T}_{22})$. \square

Lemma 7.1.2 says that a matrix can be reduced to block triangular form using unitary similarity transformations if we know one of its invariant subspaces. By induction we can readily establish the decomposition of Schur (1909).

Theorem 7.1.3 (Schur Decomposition). If $A \in \mathbb{C}^{n \times n}$, then there exists a unitary $QE \in \mathbb{C}^{n \times n}$ such that

$$Q^H A Q = T = D + N \quad (7.17)$$

where $D = \text{diag}(A_1, \dots, A_k)$ and $N \in \mathbb{C}^{n \times n}$ is strictly upper triangular. Further or, Q can be chosen so that the eigenvalues A_i appear in any order along the diagonal.

Proof. The theorem obviously holds if $n = 1$. Suppose it holds for all matrices of order $n - 1$ or less. If $Ax = Ax$ and $x \neq 0$ then by Lemma 7.1.2 (with $B = (A)$) there exists a unitary U such that

$$U^H A U = \begin{bmatrix} A & W \\ 0 & C \\ 1 & n \end{bmatrix} \begin{matrix} 1 \\ 1 \\ n \end{matrix}.$$

By induction there is a unitary f such that $f^H C f$ is upper triangular. Thus, if $Q = U \cdot \text{diag}[f, U]$, then $Q^H A Q$ is upper triangular. \square

If $Q = [Q_1 | \cdots | Q_k]$ is a column partitioning of the unitary matrix Q in (7.17), then the Q_k are referred to as Schur vectors. By equating columns in the equations $AQ = QT$, we see that the Schur vectors satisfy

$$A Q_k = A Q_k + \sum_{l=1}^{k-1} n_l Q_l \quad k = 1:n \quad (7.18)$$

From this we conclude that the subspaces

$$S_k = \text{span}\{q_1, \dots, q_k\}, \quad k = 1:n,$$

are invariant. Moreover, it is not hard to show that if $Q_k = [q_1 | \cdots | q_k]$, then $A(Q_k^H A Q_k) = \{A_1, \dots, A_k\}$. Since the eigenvalues in (7.17) can be arbitrarily ordered, it follows that there is at least one k -dimensional invariant subspace associated with each subset of k eigenvalues. Another conclusion to be drawn from (7.18) is that the Schur vector Q_k is an eigenvector if and only if the k th column of N is zero. This turns out to be the case if $k = 1:n$ whenever $A^H A = A A^H$. Matrices that satisfy this property are called normal.

Corollary 7.1.4. $A \in \mathbb{C}^{n \times n}$ is normal if and only if there exists a unitary $QE \in \mathbb{C}^{n \times n}$ such that $Q^H A Q = \text{diag}(A_1, \dots, A_k)$.

Proof. See P7.1.1. \square

Note that if $Q^H A Q = T = \text{diag}(A_1) + N$ is a Schur decomposition, then

$$\text{Echm} A \quad \text{IN1} \quad G \} = A.dN_i - G.iA$$

To see what is involved in nonunitary similarity reduction, we consider the block diagonalization of a 2 by 2 block triangular matrix.

Lemma 7.1.5. Let $T \in \mathbb{C}^{n \times n}$ be partitioned as follows

$$T = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix}_{\begin{smallmatrix} p \\ q \end{smallmatrix}}.$$

Define the linear transformation $\phi: \mathbb{C}^{p \times q} \rightarrow \mathbb{C}^{p \times q}$ by

$$\phi(X) = T_{11}X - XT_{22}$$

where $X \in \mathbb{C}^{p \times q}$. Then ϕ is nonsingular if and only if $(T_{11})_n \cdot (T_{22}) = 0$. If ϕ is nonsingular and Y is defined by

$$K_a = \begin{bmatrix} I_p & Z \\ 0 & I_q \end{bmatrix}$$

where $Z = -T_{12}$, then $K_a^{-1}Y = \text{diag}(T_{11}, T_{22})$.

Proof. Suppose $\phi(X) = 0$ for $X \neq 0$ and that

$$UH XV = \begin{bmatrix} E_r & 0 \\ 0 & 0 \\ r & qr \end{bmatrix}_r$$

is the SYD of X with $E_r = \text{diag}(a_i)$, $r = \text{rank}(X)$. Substituting this into the equation $T_{11}X = XT_{22}$ gives

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

where $U_8 T_{11} U = (A_{ij})$ and $V_8 T_{22} V = (B_{ij})$. By comparing blocks in this equation it is clear that $A_{21} = 0$, $B_{12} = 0$ and $(A_n) = (B_n)$. Consequently, A_u and B_u have an eigenvalue in common and that eigenvalue is in $(T_{11})_n \cdot (T_{22})$. Thus, if ϕ is singular, then T_{11} and T_{22} have an eigenvalue in common. On the other hand, if $A \in (T_{11})_n \cdot (T_{22})$, then we have eigenvector equations $T_{11}X = AX$ and $Y T_{22} = BY$. A calculation shows that $\phi(XY) = 0$ confirming that ϕ is singular.

Finally, if ϕ is nonsingular, then $\phi(Z) = -T_{12}$ has a solution and

$$34+ \quad \begin{bmatrix} I_p & Z \\ 0 & I_q \end{bmatrix} \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \begin{bmatrix} I_p & Z \\ 0 & I_q \end{bmatrix} = \begin{bmatrix} T_{11} & T_{11}Z - ZT_{22} + T_{12} \\ 0 & T_{22} \end{bmatrix}$$

has the required block diagonal form.

By repeatedly applying this lemma, we can establish the following more general result.

Theorem 7.1.6 (Block Diagonal Decomposition). Suppose

$$Q^H A Q = T = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1q} \\ 0 & T_{22} & \cdots & T_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & T_{qq} \end{bmatrix} \quad (7.19)$$

is a Schur decomposition of $A \in \mathbb{C}^{n \times n}$ and that the T_{ii} are square. If $A(T_{ii})nA(T_{jj}) = 1$ whenever $i \neq j$, then there exists a nonsingular matrix $Y \in \mathbb{C}^{n \times n}$ such that

$$(QY)^{-1}A(QY) = \text{diag}(T_{11}, \dots, T_{qq}). \quad (7.1.10)$$

Proof. See P7.1.2 D

If each diagonal block T_{ii} is associated with a distinct eigenvalue, then we obtain

Corollary 7.1.7. If $A \in \mathbb{C}^{n \times n}$, then there exists a nonsingular X such that

$$X^{-1}AX = \text{diag}(\lambda_1 I + N_1, \dots, \lambda_q I + N_q) \quad N_i \in \mathbb{C}^{n_i \times n_i} \quad (7.1.11)$$

where $\lambda_1, \dots, \lambda_q$ are distinct, the integers n_1, \dots, n_q satisfy $n_1 + \dots + n_q = n$ and each N_i is strictly upper triangular.

A number of important terms are connected with decomposition (7.1.11). The integer n_i is referred to as the algebraic multiplicity of λ_i . If $n_i = 1$, then λ_i is said to be simple. The geometric multiplicity of λ_i equals the dimensions of $\text{null}(N_i)$, i.e., the number of linearly independent eigenvectors associated with λ_i . If the algebraic multiplicity of λ_i exceeds its geometric multiplicity, then λ_i is said to be a defective eigenvalue. A matrix with a defective eigenvalue is referred to as a defective matrix. Nondefective matrices are also said to be diagonalizable.

Corollary 7.1.8 (Diagonal Form). $A \in \mathbb{C}^{n \times n}$ is nondefective if and only if there exists a nonsingular $X \in \mathbb{C}^{n \times n}$ such that

$$X^{-1}AX = \text{diag}(A_1, \dots, A_q). \quad (7.1.12)$$

Proof. A is nondefective if and only if there exist independent vectors $x_1, \dots, x_n \in \mathbb{C}^n$ and scalars A_1, \dots, A_q such that $Ax_i = A_i x_i$ for $i = 1, \dots, n$. This is equivalent to the existence of a nonsingular $X = [x_1 | \dots | x_n] \in \mathbb{C}^{n \times n}$ such that $AX = XD$ where $D = \text{diag}(A_1, \dots, A_q)$.

Note that if y_i is the i th row of X^{-1} , then $y_i^H A = A_i y_i$. Thus, the columns of $X^{-1}A$ are left eigenvectors and the columns of X are right eigenvectors.

If we partition the matrix X in (7.1.11),

$$X = \begin{bmatrix} X_1 | \dots | X_q \\ \hline n_1 & & & n_q \end{bmatrix}$$

then $\{ \cdot = \text{ran}(X_1) \cup \dots \cup \text{ran}(X_r)$, a direct sum of invariant subspaces. If the bases for these subspaces are chosen in a special way, then it is possible to introduce even more zeroes into the upper triangular portion of $x^{-1}AX$.

Theorem 7.1.9 (Jordan Decomposition). *If $A \in \mathbb{C}^{n \times n}$, then there exists a nonsingular $X \in \mathbb{C}^{n \times n}$ such that $x^{-1}AX = \text{diag}(J_1, \dots, J_r)$ where*

$$J_i = \begin{bmatrix} A & 1 & & \cdots & 0 \\ 0 & A & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & 1 \\ 0 & \cdots & 0 & A & \end{bmatrix} \in \mathbb{C}^{n \times n}$$

and $\{_{\text{ra}} + \dots + \{_{\text{ra}} =$

Pr b See Horn and Johnson (MA, p. 330) \square

The J_i are referred to as Jordan blocks. The number and dimensions of the Jordan blocks associated with each distinct eigenvalue are unique, although their ordering along the diagonal is not.

On the $uTlb \cdot T \cdot Jlf \in Tlb kTlU \cdot ftxKhlT \cdot Trtxf K$

The Jordan block structure of a defective matrix is difficult to determine numerically. The set of n -by- n diagonalizable matrices is dense in $\mathbb{C}^{n \times n}$, and thus, small changes in a defective matrix can radically alter its Jordan form. We have more to say about this in §7.6.5.

A related difficulty that arises in the eigenvalue problem is that a nearly defective matrix can have a poorly conditioned matrix of eigenvectors. For example, any matrix X that diagonalizes

$$A = \begin{bmatrix} 1+e & 1 \\ 0 & 1-e \end{bmatrix}, \quad 0 < e \ll 1, \quad (7.1.13)$$

has a 2-norm condition of order $1/e$.

These observations serve to highlight the difficulties associated with ill-conditioned similarity transformations. Since

$$f(x^{-1}AX) = x^{-1}AX + E, \quad (7.1.14)$$

where

$$\|E\|_2 \approx u \cdot \kappa_2(X) \|A\|_2, \quad (7.1.15)$$

it is clear that large errors can be introduced into an eigenvalue calculation when we depart from unitary similarity.

EA

Since the singular values of A and its Schur decomposition $QHQ^T = \text{diag}(\sigma_i) + N$ are the same, it follows that

$$\sigma_{\min}(A) \leq \min_{1 \leq i \leq n} |\lambda_i| \leq \max_{1 \leq i \leq n} |\lambda_i| \leq \sigma_{\max}(A).$$

From what we know about the condition of triangular matrices, it may be the case that

$$\max_{1 \leq i, j \leq n} \frac{|\lambda_i|}{|\lambda_j|} \ll \kappa_2(A).$$

See §543 This is a reminder that for nonnormal matrices, eigenvalues do not have the "predictive power" of singular values when it comes to $\mathbf{Ax} = \mathbf{b}$ sensitivity matters. Eigenvalues of nonnormal matrices have other shortcomings, a topic that is the focus of §7.9.

Problems

P7.1.2 3.8.8 m.88.8. ro... 6. ...2.8. ... 58... rofmm

$$\mathbf{P7.1.3 \dots 8.8} \quad \mathbb{C}^{n \times n}, \quad \mathbb{R}^{n \times n}, \quad \mathbb{C}_{4 \times 4}(\gamma_1, \gamma_2, \gamma_3, \gamma_4), \quad \mathbb{C}_{4 \times 4}(-\gamma_1, -\gamma_2, -\gamma_3, -\gamma_4) \quad \mathbb{C}_{4 \times 4}(-\gamma_1, -\gamma_2, -\gamma_3, -\gamma_4), \quad \mathbb{C}_{4 \times 4}(\beta, \gamma_1, \gamma_2, \gamma_3, \gamma_4)$$

$$\text{P7.1.4 .8fo 2.2.b .6.} \quad .8 \cdot 6 \mathbb{C}^{m \times n})_{4,m \geq n, \tau.M}$$

m = 1, 2, ..., $n/2$

$$\text{P7.1.5 N.86} \quad \mathbb{C}^{n \times n}, -\mathbb{C}^{4^n \gamma} \xrightarrow{\rightarrow^N \gamma, (\beta, (\mathbb{C})_4)} (I_4, (\mathbb{C}(\mathbb{C}_{4n4}), \frac{T}{2}(\gamma, \gamma_{-e})) = \text{no } 1x + x \text{ x a } \\ = \text{if } x = \text{O V O or } = x a \parallel = \|_2 = 2 \cdot \text{many } 10 \text{ for } \lambda = \gamma \gamma \beta \cdot \lambda C \gamma, \lambda k \cdot L \lambda m T 2 \gamma. \\ i \text{ O O } = \text{O C } \mathbb{C}^n \mathbb{C}^n = (0, (\gamma, (\beta, (\mathbb{C})_4), (\mathbb{C}(4), (4)(\mathbb{C} \beta_{4n4})_B)) \gamma, (\beta, (\mathbb{C})_4))$$

P7.1.6 ...8.8 .. 26.2 { (a (\neg m) N) D O G + (• = f) f •) (a) N • v) N m) ' a ,
 $=$ o m G E n F E = o (O O) a , v f) J (N O) (N m)

f±a ' "

P7.1.7 k..2.b from P.6. fr5.5 55

P7.1.8 ..8fo 68f28.8...28 2.88..8....8. 8b

[. g];

$$|\langle e(C, -C, -) (-, 1()) (-1, -(C-1.-)), \tilde{x} \rangle|$$

P7.1.9 .8268.8.... .8.8..8..2.86 28.8fo 26.2b... 2.88..8....8. 8 . ..2.b ..8 .2..2..
 . 2... .6.2.62.8. 0N

P7.1.10 m.8 ...2...8 ..8..8.

R = 5 R = 5 R .fo OR

$\text{h}_{\text{FOP}} \cdot \text{Ap} \Rightarrow \text{OM}_p \text{e} \text{MS}_p \text{M}_p \text{p} \Leftrightarrow \text{MS}_p \text{O}_p \text{J}_p \text{Z} \text{ZOE} \text{DE} \text{G}_p \cdot \text{E} \text{OO} = \text{OA} \text{OOE} = \text{OE} = \text{o}$

$$\begin{array}{ll} \text{use } M_t & x_{k+1} = x_k + h y_k, \\ \text{yz} & y_{k+1} = y_k - h x_k, \end{array}$$

uMt rO x_{k+1} = U hya
 y_{k+1} = ra haal

$$\text{unit } 3z \quad x_{k+1} = : 1 \text{ hra} \\ y_{k+1} = : \text{ hxa} .$$

zAngeg ecpMeMt ia rMekn

$$[G_{\cdot\cdot\cdot}] = (\quad [G])$$

=~~1~~₂ ([] = /) , ! (: @ x) | ~~x~~ ~~Af(x) - Azl~~) + e- r1(.,e- r1(.,-" +,-,e--

P7.1.11 re> J x^dT.2 U(Exp(al(L/2 Q_{k,∞}(J)?

P7.1.12 J. 8.8 R <nxn +/(L <2 <A n1St In CiIMnAA.

$$M_1 = \begin{bmatrix} V \\ \phi \end{bmatrix} \text{ and } M_2 = \begin{bmatrix} 0 & -A \\ B & A \end{bmatrix}$$

$$\mathbf{A} \mathbf{B} = \mathbf{B} \mathbf{A} \quad (\mathbf{A} \mathbf{B}) \mathbf{C} = \mathbf{A}(\mathbf{B} \mathbf{C}) \quad (\mathbf{A}^T)^T = \mathbf{A},$$

P7.1.13 .I..8 .8 E xn IA 21 <2 A: E xnT. A Drazin inverse l. A ,5 U) T S= BA, (MM
 ST S= Al2x (SM\MrAa.pesemMBT r ST SiaplM.aevpuse vpS MapurlB .elpmen
 malcAlrM.SMBT AeSniAepS.sea.anShl ..a olar lMeans.. TTianMian.es.ar.

P7.1.14 .386 $\Sigma R_{\text{ext}} \ll A$ (T) $B(\sigma_1 \dots \sigma_n)^{1/n} L/AE \ll \dots \ll A \cdot Q_{\text{tip}} E \gg pA$

P7.1.15 38...8, 38es.8...es_{q(x)} = T₁cW 8apaT Rxn A LQ/\¢ ablkA <A
 a(AhaQA)\¢ u'A(e hAa.MB_a la.Man. SnapqIB_a aSianesaA₁... ~~A_n~~ (o) iMae
 rMgAa vose.a la.Man. Sn .alBp(T) enpar ?.

$$P7.1.16 \quad N.8. \quad R^{2x2} \cdot (x - 1)AX = AT + AA$$

Notes and References for §7.1

feg.... .8..es.....8. ..8l l8s..8..8.8... .83... .8..8...esl8..8.88.R.88P8.. ... k83..8.
R1 enfr

IwPMpra. MB dAn Intr duction to Linear Algebr , T ,Me aniAM,loMA,5 ,Me M5.
I 6niMA,iae.u nna f N, . A Sur ey of Matrix Theor and Matrix Inequalities, 3pp iae wiAla.
wHla

r. wAppgfa NJ... I tr duction to Matr Analysis, AAlaeAecInlaA.Mi. .cpA. 1IM+ M
 fr JpSAmniaAIAMiae nMlegia f JJ7. 1Ivariant Subspaces of Matrices with Applications,
 sf 3n h SpnAia. hpc@AOS pth3M

JMi rAaAMiO...nlaSI.I IpAcggjMaaAAI SAI AAiaCiIMcTe d,IIMia,Sl,AAT

3.3. E, SM, p. 162 NMPAIJ NJ.lAiApiIclal. i hia, SI, nIclani IMcTl. Comp. and Appl. Math. 233, N, N^o(N,N..

lpA sApJAMlgSl,nIclMcrnaiSSAiMAae

f MaP,M N.. „a IpApiMiAIAMtInA i nna3iM, Icl, Inlad p ia 3SSpAiInIa IpA
IpAlMt. f alArNsp InlaMgath. Ann. 66 ,ddM.Njapgen) w

4 Appln rapMs.1 lpr SSS.anMn

s,uw pno..enm 41MaantMAn Intr duction to the Theory of Canonical For s, EliAM
2A. lIM+NJ..

7.2 Perturbation Theory

The act of computing eigenvalues is the act of computing zeros of the characteristic polynomial. Galois theory tells us that such a process has to be iterative if $n > 4$ and so errors arise because of finite termination. In order to develop intelligent stopping criteria we need an informative perturbation theory that tells us how to think about approximate eigenvalues and invariant subspaces.

7.8.5 $\mathbf{c} \times \mathbf{r} \rightarrow \mathbf{x}$

An important framework for eigenvalue computation is to produce a sequence of similarity transformations $\{X_k\}$ with the property that the matrices $x_k^{-1}AX_k$ are progressively "more diagonal." The question naturally arises, how well do the diagonal elements of a matrix approximate its eigenvalues?

Theorem 7.2.1 (Gershgorin Circle Theorem). If $x^{-1}AX = D + F$ where $D = \text{diag}(d_1, \dots, d_n)$ and F has zero diagonal entries, then

$$\|A\| \leq \sum_{i=1}^n \|D_i\|$$

where $D_i = \{z \in \mathbb{C} : |z - d_i| \leq \sum_{j=1}^n |f_{ij}|\}$.

Proof. Suppose $\lambda \in E(A)$ and assume without loss of generality that $\lambda = d_i$ for $i = 1:n$. Since $(D - \lambda I) + F$ is singular, it follows from Lemma 2.3.3 that

$$1 \leq \|(D - \lambda I)^{-1}F\|_\infty = \sum_{j=1}^n \frac{|f_{kj}|}{|d_k - \lambda|}$$

for some $k \in \{1, \dots, n\}$. But this implies that $|E| \geq |D_k| = |D|$.

It can also be shown that if the Gershgorin disk D_i is isolated from the other disks, then it contains precisely one eigenvalue of A . See Wilkinson (AEP, pp. 71ff.).

For some methods it is possible to show that the computed eigenvalues are the exact eigenvalues of a matrix $A + E$ where E is small in norm. Consequently, we should understand how the eigenvalues of a matrix can be affected by small perturbations.

Theorem 7.2.2 (Bauer-Fike). If μ is an eigenvalue of $A + E \in \mathbb{C}^{n \times n}$ and $x^{-1}AX = D = \text{diag}(1, \dots, \lambda_n)$, then

$$\min_{\lambda \in \lambda(A)} |1 - \mu_j| \leq K_p(X) \|E\|$$

where $K_p(X)$ denotes any of the norms $\|\cdot\|_p$.

Proof. If $\mu \in E(A)$, then the theorem is obviously true. Otherwise if the matrix $x^{-1}(A + E - \mu I)x$ is singular, then so is $I + (D - \mu I)^{-1}(X^{-1}EX)$. Thus from

Lemma 233 we obtain

$$1 \leq \| (D - \mu I)^{-1}(X^{-1}EX) \|_p \leq \| (D - \mu I)^{-1} \|_p \| X \|_p \| E \|_p \| X^{-1} \|_p.$$

Since $(D - \mu I)^{-1}$ is diagonal and the norm of a diagonal matrix is the absolute value of the largest diagonal entry, it follows that

$$\| (D - \mu I)^{-1} \|_p = \max_{\lambda \in \lambda(A)} \frac{1}{|\lambda - \mu|}$$

completing the proof. \blacksquare

An analogous result can be obtained via the Schur decomposition

Theorem 7.2.3. Let $Q^H A Q = D + N$ be a Schur decomposition of $A \in \mathbb{R}^{n \times n}$ as in (7.17). If $\mu \in \lambda(A + E)$ and p is the smallest positive integer such that $|N|^p = 0$ then

$$\min_{\lambda \in \lambda(A)} |\lambda - \mu| \leq \max(Q, Q^H P)$$

where

$$Q = \|E\|_2 \sum_{k=0}^{p-1} \|N\|_2^k,$$

Pr 6 Define

$$\delta = \min_{\lambda \in \lambda(A)} |\lambda - \mu| = \frac{1}{\|(\mu I - D)^{-1}\|_2}.$$

The theorem is clearly true if $\delta = 0$. If $\delta > 0$ then $I - (\mu I - D)^{-1}E$ is singular and by Lemma 233 we have

$$\begin{aligned} 1 &\leq \|(\mu I - D)^{-1}E\|_2 = \|(\mu I - D)^{-1}\|_2 \|E\|_2 \\ &= \|((\mu I - D) - N)^{-1}\|_2 \|E\|_2. \end{aligned} \tag{7.21}$$

Since $(\mu I - D)^{-1}$ is diagonal and $|N|^p = 0$ it follows that $((\mu I - D)^{-1}N)^p = 0$. Thus

$$((\mu I - D) - N)^{-1} = \sum_{k=0}^{p-1} ((\mu I - D)^{-1}N)^k (\mu I - D)^{-1}$$

and so

$$\|((\mu I - D) - N)^{-1}\|_2 \leq \sum_{k=0}^{p-1} \|N\|_2^k.$$

If $\delta > 1$, then

$$\|(\mu I - D)^{-1}\|_2 \leq \sum_{k=0}^{p-1} \|N\|_2^k$$

and so from (7.21), $\|P\|_F = 0$ if $\lambda = 1$, then

$$\|(\mu I - A)^{-1}\|_F \leq \frac{1}{\delta} \sum_{k=0}^{p-1} \|A^k\|_F.$$

By using (7.21) again we have $\|P\|_F = 0$ if $\lambda = 1$. \square

Theorems 7.22 and 7.23 suggest that the eigenvalues of a normal matrix may be sensitive to perturbations. In particular, if $\|A\|_F$ or $\|A^{-1}\|_F$ is large, then small changes in A can induce large changes in the eigenvalues.

7.2.2 $n \times n$ $v \rightarrow m$ $\alpha x \rightarrow x$

Extreme eigenvalue sensitivity for a matrix A cannot occur if A is normal. On the other hand, nonnormality does not necessarily imply eigenvalue sensitivity. Indeed, a nonnormal matrix can have a mixture of well-conditioned and ill-conditioned eigenvalues. For this reason, it is beneficial to refine our perturbation theory so that it is applicable to individual eigenvalues and not the spectrum as a whole.

To this end, suppose that λ is a simple eigenvalue of $A \in \mathbb{C}^{n \times n}$ and that x and y satisfy $Ax = \lambda x$ and $y^H A = \lambda y^H$ with $\|x\|_2 = \|y\|_2 = 1$. If $y^H Ax = J$ is the Jordan decomposition with $J = x \cdot 1$, then y and x are nonzero multiples of $X(:, i)$ and $Y(:, i)$ for some i . It follows from $1 = Y(:, i)^H X(:, i)$ that $y^H x \neq 0$, a fact that we shall use shortly.

Using classical results from function theory, it can be shown that in a neighborhood of the origin there exist differentiable $x(t)$ and $A(t)$ such that

$$(A + \epsilon F)x(\epsilon) = \lambda(\epsilon)x(\epsilon), \quad \|F\|_2 = 1,$$

where $A(0) = A$ and $x(0) = x$. By differentiating this equation with respect to t and setting $t = 0$ in the result, we obtain

$$Ax(0) + Fx = (\lambda x + Ax(0))$$

Applying y^H to both sides of this equation, dividing by $y^H x$, and taking absolute values gives

$$|\dot{\lambda}(0)| = \left| \frac{y^H Fx}{y^H x} \right| \leq \frac{1}{|y^H x|}.$$

The upper bound is attained if $F = yx^H$. For this reason we refer to the reciprocal of

$$s(\lambda) = |y^H x| \tag{7.22}$$

a the condition of the eigenvalue λ .

Roughly speaking, the above analysis shows that $O(1)$ perturbations in A can induce $1/s(\lambda)$ changes in an eigenvalue. Thus, if $s(\lambda)$ is small, then A is appropriately regarded as ill-conditioned. Note that $s(\lambda)$ is the cosine of the angle between the left and right eigenvectors associated with λ and is unique only if λ is simple.

A small $s(\cdot)$ implies that A is near a matrix having a multiple eigenvalue. In particular, if λ is distinct and $s(\lambda) < 1$, then there exists an E such that λ is a repeated eigenvalue of $A + E$ and

$$\frac{\|E\|_F}{\|A\|_F} \leq \frac{s(\lambda)}{\sqrt{1 - s(\lambda)^2}}.$$

This result is proved by Wilkinson (1972).

Onihmb uSt eTf TfKb 98\\$ tf Sb bs"Stf t dSb

If λ is a repeated eigenvalue, then the eigenvalue sensitivity question is more complicated. For example, if

$$A = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad F = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix},$$

then $(A + \epsilon F) = \{I \pm \sqrt{1 + a^2}\}$. Note that if $a = 0$ then it follows that the eigenvalues of $A + \epsilon F$ are not differentiable at zero, their rate of change at the origin is infinite. In general, if λ is a defective eigenvalue of A , then $O(t)$ perturbations in A can result in $O(\epsilon^{1/p})$ perturbations in λ , if λ is associated with a p -dimensional Jordan block. See Wilkinson (AEP, pp 77f.) for a more detailed discussion.

Ogiisb etf t sttf buD:e3ff\\$b uStf TfTf Kb

A collection of sensitive eigenvectors can define an insensitive invariant subspace provided the corresponding cluster of eigenvalues is isolated. To be precise, suppose

$$Q^H A Q = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix}_{n-r} \quad \begin{matrix} r \\ r \\ n-r \end{matrix} \quad (7.23)$$

is a Schur decomposition of A with

$$Q = [Q_1 \mid Q_2] \quad \begin{matrix} r & n-r \end{matrix} \quad (7.24)$$

It is clear from our discussion of eigenvector perturbation that the sensitivity of the invariant subspace $\text{inv}(Q)$ depends on the distance between T_{11} and T_{22} . The proper measure of this distance turns out to be the smallest singular value of the linear transformation $X \mapsto T_{11}X - XT_{22}$. (Recall that this transformation figures in Lemma 7.1.5.) In particular, if we define the separation between the matrices T_{11} and T_{22} by

$$\text{sep}(T_{11}, T_{22}) = \min_{X \in O} \frac{\|T_{11}X - XT_{22}\|_F}{\|X\|_F}, \quad (7.25)$$

then we have the following general result:

Theorem 7.2.4. Suppose that (7.2.3) and (7.2.4) hold and that for any matrix $E \in \mathbb{C}^{n \times n}$, we partition $Q = [Q_1 | Q_2] \in \mathbb{C}^{n \times n}$ as follows

$$Q^H E Q = \begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix}_{n-r}^r \quad \begin{matrix} r \\ n-r \end{matrix} .$$

I sep(T_{11}, T_{22}) > 0 and

$$\|E\|_F \left(I + \frac{5\|T_{22}\|_F}{\text{sep}(T_{11}, T_{22})} \right) \leq \frac{\text{sep}(T_{11}, T_{22})}{5},$$

then there exists a $P \in \mathbb{C}^{(n-r) \times r}$ with

$$\|P\|_F \leq 4 \frac{\|E\|_F}{\text{sep}(T_{11}, T_{22})}$$

such that the columns of $P = (Q_1 + Q_2 P)(I + p^H P)^{-1/2}$ are an orthonormal basis for a subspace invariant for $A + E$.

Pr of. This result is a slight reworking of Theorem 4.11 in Stewart (1973) which should be consulted for proof detail. See also Stewart and Sun (MPA, p. 239). The matrix $(I + p^H P)^{-1/2}$ is the inverse of the square root of the symmetric positive definite matrix $I + p^H P$. See §4.24. \square

Corollary 7.2.5. If the assumptions in Theorem 7.2.4 hold, then

$$\text{dist}(\text{ran}(Q), \text{ran}(A)) \leq 4 \frac{\|E_{21}\|_F}{\text{sep}(T_{11}, T_{22})} .$$

Pr of. Using the SVD of P , it can be shown that

$$\|P(I + p^H P)^{-1/2}\|_F \leq \|p\|_F \leq \|P\|_F \leq \sqrt{n} . \quad (7.26)$$

Since the required distance is the 2-norm of $Q - A = P(I + p^H P)^{-1/2}$, the proof is complete. \square

Thus, the reciprocal of $\text{sep}(T_{11}, T_{22})$ can be thought of as a condition number that measures the sensitivity of $\text{ran}(Q)$ as an invariant subspace.

Trinib bT"Stf Sf kb us tf TfKtf

If we set $r = 1$ in the preceding subsection, then the analysis addresses the issue of eigenvector sensitivity.

Corollary 7.2.6. Suppose $A \in \mathbb{C}^{n \times n}$ and that $Q = [Q_1 | Q_2] \in \mathbb{C}^{n \times n}$ is unitary with $Q \in \mathbb{C}^{n \times n}$. Assume

$$Q^H A Q = \begin{bmatrix} \lambda & v^H \\ 0 & T_{22} \end{bmatrix}_{n-1}^1, \quad Q^H E Q = \begin{bmatrix} \epsilon & \gamma^H \\ \delta & E_{22} \end{bmatrix}_{n-1}^1 .$$

(Thus, \mathbf{q} is an eigenvector.) If $\sigma = \text{Uin}(T_{22}) > 0$ and

$$\|E\|_F \left(1 + \frac{5\|\mathbf{v}\|_2}{\sigma}\right) \leq \frac{\sigma}{5},$$

then there exists $\mathbf{p} \in \mathbb{C}^n$ with

$$\|\mathbf{p}\|_2 \leq \frac{4}{\sigma}$$

such that $\mathbf{i} = (\mathbf{q} + \mathbf{Q}\mathbf{p})/\sqrt{1 + \mathbf{p}^H\mathbf{p}}$ is a unit 2-norm eigenvector for $A + E$. Moreover,

$$\text{dist}(\text{span}\{\mathbf{q}\}, \text{span}\{\mathbf{i}\}) \leq \frac{4}{\sigma}.$$

Proof. The result follows from Theorem 7.24, Corollary 7.25, and the observation that if $T = \lambda$, then $\text{sep}(T_{11}, T_{22}) = \text{Uin}(T_{22}) > 0$. \square

Note that $\text{Uin}(T_{22})$ roughly measures the separation of A from the eigenvalues of T_{22} . We have to say "roughly" because

$$\text{sep}(\lambda, T_{22}) = \sigma_{\min}(T_{22} - \lambda I) \leq \min_{\mu \in \lambda(T_{22})} |\mu - \lambda|$$

and the upper bound can be a gross overestimate.

That the separation of the eigenvalues should have a bearing upon eigenvector sensitivity should come as no surprise. Indeed, if A is a nondefinite, repeated eigenvalue, then there are an infinite number of possible eigenvectors basis for the associated invariant subspace. The preceding analysis merely indicates that this indeterminacy begins to reflect as the eigenvalues coalesce. In other words, the eigenvectors associated with nearby eigenvalues are "wobbly."

Problems

P7.2.1 ... 8.8 QHAQ = $xT_A(A_1) + N T_2 + aC = EAa(7I - ThQ)$, $A \in \mathbb{C}^{n \times n}$, $x \in \mathbb{C}^n$, $A^{-1}(A) = \|A\|_F^2 - AA^H\|F\|^2 - CA_1 A E$, $\|x\|_2 = 1$, $\|A\|_F = 1$, $\|A\|_1 = 1$.

$$\frac{1(A)^2}{6\|A\|_F^2} \leq \|x\|_2^2 \leq \sqrt{n^3 - n} 1(A)$$

$2EA = r2T = CA_1 k(A) ET_A(Q)$, $53x^2x = (AEAT)Z$, $5E^2x = 12nT_2A_1^H AET_1CA = 1E nCA$, $n = 3$.

P7.2.2 ... 8.8 $A \in \mathbb{C}^{n \times n}$, $x \in \mathbb{C}^n$, $x^H A = xQ_1(I_n - A_1) L_Q(I_n - Q_1^H A_1) + C(L_n C_2 Q_1 C_1 A_1(p=7) = (X C_2 A_1)^H(I_n)(E^2 z r C_1 A_1) p(X)^2 = n(1/s(A))^2 + \dots + 1/s(A_n)^2$.

P7.2.3 ... 8.8 $QHAQ = xT_2(A) + NT_2 + aC = EAa(7I - Q_1 Q_2)$, $A \in \mathbb{C}^{n \times n}$, $x \in \mathbb{C}^n$, $x^H A = xQ_1(A)$, $+ C(I_n - 12(X)^2) + (1\|N\|_F^2 / \|A\|_F^2) + A^H Q_1 X = Z$, $5X$.

P7.2.4 reb - $1AX = xQ_1(A) 2x^H A^H a - a^H A_1^H nCA$

$$\frac{ui(A)}{12(X)} \|A\|_1 \leq 12(X)ui(A).$$

$E^2 A - nCTEA = 1E nCA_1 = 3a^H A_1 + AA^H C A_1^H nX$.

P7.2.5 .386 ... $\underline{A} = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix} = b^H a - b^H nCA_1 s(a) = s(b) = Z + k/(a - b)^2 - 1/2$.

P7.2.6 2 8.8

$$A = \begin{bmatrix} & v^T \\ \vdots & \\ & T_{22} \end{bmatrix}$$

] (: I $\lambda \notin \sigma(T_{22})$. $\div - (x - A\lambda) = \text{sep}(\lambda, T_{22})$,) - ((

$$\| \delta \lambda_1 \| = \frac{\| J \|}{\sqrt{1 + \| (T_{22} - \lambda I)^{-1} v \|^2}} = \frac{\| a \|}{\sqrt{\sigma^2 + \| v \|^2}}.$$

 $x = (\lambda) + r$ ("T+(U_w 9.9.1).

P7.2.7 38fo. .38.8...8. 8≥.8 88.d8..828. .8.8.8f2.8. 2.... .8.8.... N.

n. gl

P7.2.8 f..3 .38.88 3. 8.38..xxncx= Jx xanax = 2.2. n k W3a x6l= 8 xn S x

$$\frac{8.}{E(A)} : 11E1|X| \mathbf{I}_V$$

P7.2.9 e8..≥.fro 10 85

P7.2.10 38fo. ≥B ∈ C^{m×m} -(c ∈ C^{n×n}, γ(sep(B, C) + " - (- ")) ≥ 0.) → R) AT E λ(B) (+ ∈ λ(C)).

Notes and References for §7.2

f .. 8≥38.828.. .8.8. 8.3.. 8..8. 8.. .8 N2.. .. f.. .8. f1 35fo .8fo.. .2.
f 35a 8x C.C' n ∈ WJSxii m xW · ma wno = S x, d = • CrH= nx x = Numer. Math. 62, NJ, -
3xpl. hApLOAMN, xThe Theor. of Matrices in Numerical Analysis, wOinA02Ak1 M+ 4
- wpr rfi JJ(), Perturbation Bounds for Matrix Eigenvalues, pf 3nhsS0Aijny hpn9AOOpn.y
034

dM,tO.OAMlaAMlaAppAIAApOAMphMSAppAAnr Aai, hAi, r AaAM.ipMiaAheAT

3 -hpAf NJ.- hAMphMS, pMiae, MnA.a, ldr Aai, hAi, faiiMnihs, OiAA, y6 10
1,43 -hpAf NJ.- hMIOAMpdAipMhapp BAMt, Alaecpla3elcr AaOMISy6mer. Math.
15, (17J-f an, +, na, la NJ.. - 2lpAla .ipMcAlppi BAMt, OlaepplaAdnr AaOMISy6mer. Math.
19 ((7r (t4 5pa 42hM, ApiaE d4'nar f NJ. 4.lA,nEh, whaE, la 300Mlbcr dnrAAa, t, pAlr ,
2laalMr nipMnAijy6mer. J. Numer. Anal. 19, (Jr, d.f .xan, Gia, laf NJ. x ,a 2Acr pSlMipMnAar pHeM.r dAr AapEMic, lM6Numer.
Math. 44, NisNx4n, + ca, lQ, hM+hA aAiMAA, ApciA .pMnAAtOnAi, i r MlknsAtet OcpAMipMMA,
AaaAMaA, ApA, MaA, OMISOAAT

3. -hpAf NJ. x, Aol, A, p2lMr .OpMdhbaE 6BIT 27, d, n, dk

f EXAr r f NJ x ,apmAcr iaApA2AiMAfch, Ae hMISOAM Numer. Math. 51,
1, Nfd.f - 41Ar rf Adx, lpA hMISiSn oppi 2hr AMcAit, qtc, hMISOAr En1AhOp Math. Com-
put. 50, „r, dJ.v.f .nr prf NJ. x, nir MaA, MaA, hMISiA, SQQOcAay, Applications of Matrix Theory,
nxfa, liAMaE pxw, MaApA, Aey, bMe r acAM, r, hMA, ,bMey NJ(43 v, tpAi f NJ. M3, lMr hMppA, alME, cr iaA .n.pMnbl ipAp nipMnAApp
nhpnOAr AaiiOha, y6mer. Math. 83, i, „r „f „.M, ni f JJ.x .2A.MA, p pMhAppkl hMA, AMrsAAai, y6, lin. Alg. Applic. 401,
I((r I, -anr OMISAp. ppnOcr AMipMAMAA, hApA, AApA, Ar, ha, S, Ar ir MSA, A,
ir MubstableCnpC, a Anr Aal hApalaAr r niAOAMApMl00iSn, npMA, pMely

Imfia Tia ($\begin{pmatrix} 1 \\ \theta \end{pmatrix}$ **)E.** c. $|\nabla_*^T \quad T, \nabla, \quad -T, 2\nabla' T_*^T n \quad 'c \quad \nabla_3 t_3, \quad -\begin{pmatrix} T \\ \theta \end{pmatrix} A \nabla' T_*^T n \quad A$ Contemp. Math. 47, 7r TTx

IEEE Trans. Autom. Contr., AC-32, 5, 1987.

r. wt IM_n , 11). . . $T_n^T C_3 2, (c_1^T \cdot \bar{\lambda}_2^T)_2, T_3^T | \cdot ', (T_1^T, \nabla_3, c^T \nabla, \nabla^T, 2\nabla^T T_n^T c' ', t_3, \nabla^T,$

9.B. whM^{ale} I.n. „IMpla(„,) E, „V[¶], „T[¶], T[¶] „V[¶]T[¶] c₃, c[¶] |c₃, ΣΣ, „T[¶](„, 2V[¶]T[¶] „, Al Lin. Alg
Applic. 171, I, vi T.„

A..1 iae ..3. aipla (,,)_{E-3} ||c||_{*}^T($\tilde{\lambda}_*^T \cdot c\Sigma)_*^{T*}$ ₃ '(k ||_{*}^T; 3(, 'c r, 'V₋_{*}^T, A| SIAM J. Matrix Anal. Appl. 20, NJNW7.

$$I. .h\cdot 9rl\cdot lar\cdot d\cdot nMIMpla.9rci.\cdot iae\cdot 9MmI\cdot \lambda_*^T\phi;\cdot T_*^S\Sigma V_*^T\cdot 3\cdot '(,\cdot 1^T_*\cdot \nabla_3(,$$

3rS3Ab, 3cr 1aipA1aecpnM1enAh,,ecae

$$\text{ANBianlia} \left(\begin{smallmatrix} \phi & \phi_* \\ \phi_* & \phi \end{smallmatrix} \right) = \frac{1}{3} \phi_* \cdot \frac{T_* \Sigma V^* T}{3} \phi_*' + \left(-\frac{c_*^3}{3} T_*^T T_* c_3 + c^1 \phi_*^T \right) \phi_* - 3 \nabla^T \phi_* s + 3 T_*^T \phi_*^T \mathbf{k}_{3,-} \cdot \mathbf{c}_*^T, \quad \text{Lin. Alg.}$$

$$\text{Applic. dd}_1, U_{\theta}^{T_1}, U| \cdot . E^T E \frac{\partial^2 T}{\theta^2} \nabla_3 \theta. v \frac{\nabla^T}{\theta^2} \frac{T^2}{\nabla^T} \quad (\text{, 11}). \quad \frac{!}{*} T_*^T \quad '_*^T \quad , \nabla_3 T \quad ,_*^T T - \nabla_*^T c_3, \quad c^1 \phi_*^T | \quad 3, \langle c_*^T A_{22}^1 \text{SIAM J.}$$

$$\text{Numer. Anal. 25, T.r N.} \\ \text{..a. sp3iMiae.M-miar(,,,)E } \quad \phi_*^T|_3 - \frac{\nabla_*^T|_3}{\phi_*^T|_3}, \phi_*^T \nabla_*^T|_1^T \quad 2\nabla_*^T|_*^T, \quad \nabla_3^T, \quad c_3^T|_*^Tc_3 \quad |\Sigma_*^T, \quad c^T$$

9n.rsha($\langle \cdot, \cdot \rangle_E$) $\|_3 \cdot c_3^{TT^T} c_3 \|\Sigma - T^T \nabla \|c_3^T, (T^T, 2\|T^T\|)^T, \phi_*^T \|_3 \nabla \|_*, \text{Al Numer. Math}$

$$61, I7, \text{VI}T7. \quad \text{s.I.} \quad \text{hr}(\bar{\mathbf{S}}^-, \ldots, \mathbf{c}\Sigma)^*\nabla^T\mathbf{c}\Sigma \nabla | \quad \phi_*^{TT} c_*^T \leq c_*^T \frac{1}{1}, \quad \tilde{\lambda}_T^T 2_* \quad |\nabla^T|, \quad c_*^T |\nabla^T|, \quad 2_* |\nabla^T|, \quad \phi_*^T |_{3,} \quad \nabla|_{*,A} \quad \text{Lin.}$$

J. Appl. Math. Comput. Sci. 34(2), 111–126 (2020) DOI: 10.1515/jamcs-2019-0060

ImHM1 pipcl,pS11a pm,1cr 1.iOhAaecpcl,ahr S1Mn1 e,SiMphM8Mr iOpe p
Alaecapcl, pnct 1a,1ApLMipMrcE,Al,,leca

$$1 \quad \frac{1}{3c} \text{M}(A) E \leq c_1^T, \quad \lambda^T r_*^T \nabla_*^T A \leq r_3 c_3^T s_A, \quad \lambda^T \nabla^T \quad R \nabla_*^T \nabla_*^T c_3 \nabla_3^T X_*^T T_1^T, c^T R \nabla^T c^T |c_3|$$

M. nlcldh|., .c_{33c}T₂^TV₃^T, |V₃₁^T|c₃T₂^TV₃^T. c₃T₂^TV₃^Tc₃ | Σ^- T₂^T 2V₃^T(A₁₁^A J. ACM 16,dJr 7J

3. iælMOhq**($\frac{2}{g}$) $E_{**} T_{**} \nabla_{T*} T_{C3}$, $c \phi_{T*}^3 |_{C3} \nabla_{T*}^2 c^T |_{C3} 3c_{*}^T \Sigma_{*}^2 \nabla_{T*}^2 T_{(n)}^T$ Commun. ACM 18, 5Jv57.**

Applic. 16, 71-7d

J. M. M. Morel / Mathematical Models and Methods in Applied Sciences 20 (2010) 1–36

r.s. $B(M_{\infty}, 1^-)$ $\Sigma^T \delta \nabla_{\theta} T^T(c_{\bullet}^{TT} \cdot \lambda^T \nabla^T c_3^T c_3 - 2 \nabla^T c_3^T)$, Al SIAM J. Numer. Anal. 7, 5r. JT.

NvII
 rM^2BiMn ETNA 12

ANSWER $\boxed{A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}}$

358, IdNr.,

f alIM en, Ah, pmS lMphMSipd, lpl, r ipM MIt laIMl M1aiaS 1,ice .mla pmS lMp
Sipd, iM, pMhAphNte.

/ .a. spliMp₀--,). .3 '(T₀F₀)_{3,m}', , c¹ θ^*_m T_{1,T} 2V^T*T_{(n}, Al Numer. Math. 90, 5,r, TJM
 9rHlM1iae.rn. ElsSnAt_{-σ}). εc. ∇_3^2 T_{1,T} -V^T*c₃ c¹ c₁ T₀V₃, T_{1,T} A SIAM J. Matri

Anal. Applic. 25, No. 7.

al lib tw2epenoniodtall2arl rtOOeabltal dhd rduerRanitel al edehil2Rl siad of lphon
 slodal λ . $T_{1c2s}^T \nabla_{\tilde{T}_{1c2s}^T} \tilde{c}_3^T c_3^T$ $c_3^T c_3^T$ $| \lambda - \tilde{\lambda} | / | \lambda |$, $c_{x1} c_{2s}(T_3)$ (\cdot) (\cdot)
 $1c2s^T x) c_3^T, \cdot \cdot \cdot (1c2s_3^T \tilde{T}_{1c2s}^T (c_3^T 1c2s_3^T, c_3^T \nabla_x, + 1c2s_3^T)$ $3^T \tilde{T}_{1c2s}^T c_3^T, T_3^T (\cdot, T_3^T T_{1c2s}^T (T_{1c2s}^T, \cdot \cdot \cdot$
 $\in \mathbb{C}^{T \times 1}, U : E 1c2s^T \nabla_{\tilde{T}_{1c2s}^T} \tilde{c}_3^T, \cdot \cdot \cdot \tilde{T}_{1c2s}^T \tilde{c}_3^T H(T_{1c2s}^T, \cdot \cdot \cdot 2c_3^T 1c2s \leq c_3^T, c_3^T \tilde{T}_{1c2s}^T \nabla_{\tilde{T}_{1c2s}^T} \tilde{c}_3^T, T_{1c2s}^T \tilde{c}_3^T$ Lin.
 Alg. Applic. 266, "Tri".

sArdcAa.piae ,ArfS,AaNodu.lmM3,Alg p) hAMp.MSi,he ,MI ipMdcr)ai.QA,
 fgSptApipniAll ae,6SIAM J. Matr. Anal. Applic. 20, i. r N.du
 s.ArdcAa.piae rAfS,AaN.dTu.lAOpciaAMp.M,plMOp,Mdnr Aai.piAe dnr)aiAM
 plM Ecr la.On.,gipMcABIT 38, JIr J.,
 f.A..ifS,Aa. dt lAOpciaAMp.M,pdA,ipp,MI,pMnThr Aal plate scar IpbDQ,,6Acta
 Numer ca, 7, N,NJN.

TrArfS,AaJJT .3,1OpA iae lAOpciaAMp.M,pdA,ae ,Mf aiiMc.apS.AA, l. I.pMcAA,
 Lin. Alg. Applic. 309, i,r ,7.

fA..r fS,AaJJT .3 2lpAla r an.tnar 3,1OpA ApipniAMpIM,invalae,1 Lin. Alg
 Applic. 358 I., I.,
 aAnPmn-2 w, .ae r -mar HJJT l3gpciaAhAMpIMSi,he ,Mpm9cr Aai,gIA,
 Ecr la.pc.,pAae star IpiNpMnAA,r 3SSgpA AMp.M,ipnla lyMlgSgAai.Mniap
 SI,,S.A),6 Lin. Alg. Applic. 419, T7, (N

lmAAcr AaiAAipM,aiiMniasSiAA, l. i gpMgfl gliA6 mAa pmMMAAMp.M,pda,r
 --+ cr pmAAAmiar AptScAipM,ip03ar pam pMIA+Aamiar MpmAcR hi,gIA)e
 m5pl EN77. Perturbation Theory for Linear Operators, sSMm AM.B 32AgillM+r
 A.E.in, iae ar I r 5ima NN(JT lmAllpipcla l. dcBAaiAApIM,hAMp.mnlaf f 6SIAM J.
 Numer. Anal. 7, N i7.

.2a, spAipM(N .9MMIMIae, M3SSMIT cfapciap,Si AA). Ad,AencaAi,SIAMi-
 plM SIAM. J. Numer. Anal. 8 (7r d d.
 ..a, spAipM(N .9MMIMIahAMp.M,pliae ,MsIS,SQLB,IAcipA1npnAAMp9m Aa,
 i.p,AhMI,QA6 SIAM Review 15, (r (7i.
 92pcAN.(T .3 2lpAla pmEiin,5imia lRtlmAlMA Lin. Alg. Applic. 258 NF ,,,
 s.Ir II gS .9hr Ir-Ag+ AJJJ,x,a dcr AaiAApIM,MeX6T 43 dl, d,(
 EApmnp7aiOtAI. pmBaApnk(,..) -(+,," -1X AX + XAT iMF ciAae
 9kB.MiIN(Tu,a pmASiM bnlal nipMcAA SIAM /. Numer. Anal. 16 IN . II
 lr wtAMae s.r 2gmBdT ,a pxa TQBA A lMyAnt7 IaliS,MiM SIAM J. Alg.
 Disc. Methods 8, , 177.

7.3 Power Iterations

Suppose that we are given $A \in \mathbb{C}^{N \times N}$ and a unitary $U \in \mathbb{C}^{N \times N}$. Recall from §5.2.10 that the Householder QR factorization can be extended to complex matrices and consider the following iteration:

To= U Alb

for $k = 1, 2, \dots$

$$T_k = U R_k \quad (\text{QR factorization}) \quad (7.31)$$

$$T_k = R_k U_k$$

end

Since $T_k = R_k U_k = U (U R_k) U_k = U T_k U_k$ it follows by induction that

$$T_k = (U U_1 \cdots U_k) H (U U_1 \cdots U_k) \quad (7.32)$$

Thus, each T_k is unitarily similar to A . Not so obvious, and what is a central theme of this section, is that the T_k almost always converge to upper triangular form, i.e., (7.32) almost always "converges" to a Schur decomposition of A .

Iteration (7.3.1) is called the QR iteration, and it forms the backbone of the most effective algorithm for computing a complete Schur decomposition of a dense general matrix. In order to motivate the method and to derive its convergence properties, two other eigenvalue iterations that are important in their own right are presented first: the power method and the method of orthogonal iteration.

Orthogonal Iteration

Suppose $A \in \mathbb{R}^{n \times n}$ and $x^{-1}Ax = \text{diag}(A_1, \dots, A_n)$ with $x = [x_1 | \cdots | x_n]$. Assume that

$$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|.$$

Given a unit 2-norm $q^{(0)} \in \mathbb{R}^n$, the power method produces a sequence of vectors $q^{(k)}$ as follows:

for $k = 1, 2, \dots$

$$\begin{aligned} z^{(k)} &= Aq^{(k-1)} \\ q^{(k)} &= z^{(k)} / \|z^{(k)}\|_2 \\ A^{(k)} &= [q^{(k)}]^\top A q^{(k)} \end{aligned} \tag{7.33}$$

end

There is nothing special about using the 2-norm for normalization except that it imparts a greater unity on the overall discussion in this section.

Let us examine the convergence properties of the power iteration. If

$$q^{(0)} = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \tag{7.34}$$

and $a_1 = 0$ then

$$A^k q^{(0)} = a_1 \lambda_1^k \left(x_1 + \sum_{j=2}^n \frac{a_j}{a_1} \left(\frac{\lambda_j}{\lambda_1} \right)^k x_j \right).$$

Since $q^{(k)} \in \text{span}\{A^k q^{(0)}\}$ we conclude that

$$\text{dist}(\text{span}\{q^{(k)}\}, \text{span}\{x_1\}) = O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right).$$

It is also easy to verify that

$$|\lambda_1 - \lambda^{(k)}| = O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right). \tag{7.35}$$

Since λ_1 is larger than all the other eigenvalues in modulus, it is referred to as a dominant eigenvalue. Thus, the power method converges if λ_1 is dominant and if $q^{(0)}$ has a component in the direction of the corresponding dominant eigenvector x_1 . The behavior of the iteration without these assumptions is discussed in Wilkinson (AEP, p. 570) and Parlett and Pode (1973).

In practice, the usefulness of the power method depends upon the ratio $\|\lambda_2\|/\|\lambda_1\|$, since it dictates the rate of convergence. The danger that $Q^{(k)}$ is deficient in x_1 is less worrisome because rounding errors sustained during the iteration typically ensure that subsequent iterates have a component in this direction. Moreover, it is typically the case in applications that one has a reasonably good guess as to the direction of x_1 . This guards against having a pathologically small coefficient a_{11} in (7.34).

Note that the only thing required to implement the power method is a procedure for matrix-vector products. It is not necessary to store A in an n -by- n array. For this reason, the algorithm is of interest when the dominant eigenpair for a large sparse matrix is required. We have much more to say about large sparse eigenvalue problems in Chapter 10.

Estimates for the error $\|\lambda - \lambda_i\|$ can be obtained by applying the perturbation theory developed in §7.22. Define the vector

$$r^{(k)} = Aq^{(k)} - \lambda^{(k)}q^{(k)}$$

and observe that $(A + E^{(k)})q^{(k)} = \lambda^{(k)}q^{(k)}$ where $E^{(k)} = -r^{(k)}[q^{(k)}]^H$. Thus $\lambda^{(k)}$ is an eigenvalue of $A + E^{(k)}$ and

$$\|\lambda^{(k)} - \lambda_i\| \approx \frac{\|E^{(k)}\|_2}{s(\lambda_1)} = \frac{\|r^{(k)}\|_2}{s(\lambda_1)}.$$

If we use the power method to generate approximate right and left dominant eigenvectors, then it is possible to obtain an estimate of $s(\lambda_i)$. In particular, if $w^{(k)}$ is a unit 2-norm vector in the direction of $(Aq^{(k)})^H w^{(k)}$, then we can use the approximation $s(\lambda_i) \approx \|w^{(k)}\|_2 \|\lambda^{(k)}\|_2$.

$s_1 \cdot T$

A straightforward generalization of the power method can be used to compute higher-dimensional invariant subspaces. Let r be a chosen integer satisfying $1 \leq r \leq n$. Given $A \in \mathbb{R}^{n \times n}$ and an n -by- r matrix Q with orthonormal columns, the method of orthogonal iteration generates a sequence of matrices $\{Q_k\} \subset \mathbb{R}^{n \times r}$ and a sequence of eigenvalue estimates $\{A^{(k)}\}, \dots, A^{(k)}$ as follows:

for $k = 1, 2, \dots$

$$Z_k = AQ_k$$

$$Q_k R_k = Z_k \quad (\text{QR factorization})$$

$$\therefore (Q_k^H A Q_k) = \{-\lambda_1, \dots, -\lambda_r, A^{(k)}\}$$

end

Note that if $r = 1$, then this is just the power method (7.33). Moreover, the sequence $\{Q_k\}$ is precisely the sequence of vectors produced by the power iteration with starting vector $q^{(0)} = Q_k e_1$.

In order to analyze the behavior of this iteration, suppose that

$$Q^H A Q = T = \text{diag}(\lambda_i) + N, \quad |\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_n| \quad (7.37)$$

is a Schur decomposition of $A \in \mathbb{C}^{n \times n}$. Assume that 1: $r < n$ and partition Q and T as follows

$$Q = \begin{bmatrix} Q_\alpha & Q_\beta \\ r & n-r \end{bmatrix}, \quad T = \begin{bmatrix} T_{11} & \mathbf{0} \\ \mathbf{0} & T_{22} \end{bmatrix}_{r \times n-r}. \quad (7.38)$$

If $|\lambda_r| > |\lambda_{r+1}|$, then the subspace $D_r(A) = \text{ran}(Q_\alpha)$ is referred to as a dominant invariant subspace. It is the unique invariant subspace associated with the eigenvalues $\lambda_1, \dots, \lambda_r$. The following theorem shows that with reasonable assumptions, the subspaces $\text{ran}(Q_k)$ generated by (7.36) converge to $D_r(A)$ at a rate proportional to $1/(r+1/\mu)^k$.

Theorem 7.3.1. Let the Schur decomposition of $A \in \mathbb{C}^{n \times n}$ be given by (7.37) and (7.38) with $r < n$. Assume that $|\lambda_r| > |\lambda_{r+1}|$ and that $\mu > 0$ satisfies

$$(1 + \mu)|\lambda_r| > \|N\|F.$$

Suppose $Q_0 \in \mathbb{C}^{n \times r}$ has orthonormal columns and that d_k is defined by

$$d_k = \text{dist}(D_r(A), \text{ran}(Q_k)), \quad k \geq 0$$

If

$$db \ll 1, \quad (7.39)$$

then the matrices Q_k generated by (7.36) satisfy

$$d_k \leq (1 + \mu)^{n-r} \left(1 + \frac{\|T_{12}\|_F}{\text{sep}(T_{11}, T_{22})} \right) \cdot \left[\frac{|I_{r+1}^r I^+|}{|\lambda_r| - 1 + \mu} \right]^k \cdot \frac{db}{\sqrt{1 - db}}. \quad (7.310)$$

Proof. The proof is given in an appendix at the end of this section. \square

The condition (7.39) ensures that the initial matrix Q_0 is not deficient in certain eigendirections. In particular, no vector in the span of Q_0 's columns is orthogonal to $D_r(A^H)$. The theorem essentially says that if this condition holds and if μ is chosen large enough then

$$\text{dist}(D_r(A), \text{ran}(Q_k)) \leq \frac{1}{\lambda_r} \cdot \frac{1}{k}$$

where c depends on $\text{sep}(T_{11}, T_{22})$ and A 's departure from normality.

It is possible to accelerate the convergence in orthogonal iteration using a technique described in Stewart (1976). In the accelerated scheme, the approximate eigenvalue, $\lambda_i^{(k)}$, satisfies

$$|\lambda_i^{(k)} - \lambda_i| \approx \left| \frac{\lambda_{r+1}}{\lambda_i} \right|^k, \quad i = 1, r.$$

(Without the acceleration, the right-hand side is $1/(r+1/\mu)^k$.) Stewart's algorithm involves computing the Schur decomposition of the matrices $Q_k^T A Q_k$ every so often. The method can be very useful in situations where A is large and sparse and a few of its largest eigenvalues are required.

~~nhm~~ 7.3b 59b (fStf Tk_{1b}

We now derive the QR iteration (7.3.1) and examine its convergence. Suppose $r = n$ in (7.3.6) and the eigenvalues of A satisfy

$$|\lambda_1| > |\lambda_2| > \dots > |\lambda_n|.$$

Partition the matrix Q in (7.3.7) and Q_k in (7.3.6) as follows

$$Q = [q_1 | \dots | q_n], \quad Q_k = [q_1^{(k)} | \dots | q_n^{(k)}].$$

If

$$\text{dist}(D(A), \text{span}\{q_1^{(k)}, \dots, q_{i-1}^{(k)}\}) < 1, \quad i = 1, n \quad \{7.3.11\}$$

then it follows from Theorem 7.3.1 that

$$\text{dist}(\text{span}\{q_1^{(k)}, \dots, q_{i-1}^{(k)}\}, \text{span}\{q_i, \dots, q_n\}) = 0$$

for $i = 1, n$. This implies that the matrices T_k defined by

$$T_k = Q A Q_k$$

are converging to upper triangular form. Thus, it can be said that the method of orthogonal iteration computes a Schur decomposition provided the original iterate $Q_0 E$ is not deficient in the sense of (7.3.11).

The QR iteration arises naturally by considering how to compute the matrix T_k directly from its predecessor T_{k-1} . On the one hand, we have from (7.3.6) and the definition of T_{k-1} that

$$T_{k-1} = Q_{k-1} A Q_k^{-1} = Q_{k-1} (A Q_k^{-1}) = (Q_{k-1}^T Q_k) R_k$$

On the other hand,

$$T_k = Q A Q_k = (Q A Q_{k-1}^{-1})(Q_{k-1} Q_k) = R_k (Q_{k-1} Q_k).$$

Thus, T_k is determined by computing the QR factorization of T_{k-1} and then multiplying the factors together in reverse order, precisely what is done in (7.3.1).

Note that a single QR iteration is an $O(n^3)$ calculation. Moreover, since convergence is only linear (when it exists), it is clear that the method is a prohibitively expensive way to compute Schur decompositions. Fortunately these practical difficulties can be overcome as we show in §7.4 and §7.5.

~~wnmsb~~ 19b (fStf Tk_{1b})

We conclude with some remarks about power iterations that rely on the LU factorization rather than the QR factorization. Let $G_0 E$ at \mathcal{M} have rank r . Corresponding to (7.3.1) we have the following iteration

for $k = 1, 2, \dots$

$$Z_k = A G_{k-1} \quad (7.3.12)$$

$$Z_k = G_k R_k \quad (\text{LU factorization})$$

end

Suppose $r = n$ and that we define the matrices T_k by

$$T_k = G_k^{-1} A G_k. \quad (7.3.13)$$

It can be shown that if we set $L_0 = G_0$ then the T_k can be generated as follows

$$T_0 = L_0^{-1} A L_0$$

for $k = 1, 2, \dots$

$$T_k = L_k^{-1} R_k \quad (\text{LU factorization}) \quad (7.3.14)$$

$$T_k = R_k L_k$$

end

Iterations (7.3.12) and (7.3.14) are known as the Crout iteration and the LU iteration respectively. Under reasonable assumptions, the T_k converge to upper triangular form. To successfully implement either method, it is necessary to pivot. See Wilkinson (AEP, p. 602).

$$A \quad \S \quad z$$

In order to establish Theorem 7.3.1 we need the following lemma that bounds powers of a matrix and powers of its inverse.

Lemma 7.3.2. Let $QHQ^{-1} = T = D + N$ be a Schur decomposition of $A \in \mathbb{C}^{n \times n}$ where D is diagonal and N strictly upper triangular. Let λ_{\max} and λ_{\min} denote the largest and smallest eigenvalues of A in absolute value. If $\mu > 0$, then for all $k \geq 0$ we have

$$\|A^k\|_2 \leq (1 + \mu)^{n-1} \left(|\lambda_{\max}| + \frac{\|N\|_F}{1 + \mu} \right)^k. \quad (7.3.15)$$

If A is nonsingular and $\mu > 0$ satisfies $(1 + \mu)|\lambda_{\min}| > \|N\|_F$, then for all $k \geq 0$ we also have

$$\|A^{-k}\|_2 \leq (1 + \mu)^{n-1} \left(\frac{1}{|\lambda_{\min}| - \|N\|_F / (1 + \mu)} \right)^k. \quad (7.3.16)$$

Proof For $\mu > 0$, define the diagonal matrix Δ by

$$\Delta = \text{diag}(1, (1 + \mu), (1 + \mu)^2, \dots, (1 + \mu)^{n-1})$$

and note that $\Delta^2 A = (1 + \mu)^{n-1} A$. Since N is strictly upper triangular, it is easy to verify that

$$\|\Delta N \Delta^{-1}\|_F \leq \frac{\|N\|_F}{1 + \mu}$$

and thus

$$\begin{aligned} \|A^k\|_2 &= \|T^k\|_2 = \|A^{-1}(D + ANA^{-1})^k A\|_2 \\ &\leq \|A^{-1}\|_2 \|D + ANA^{-1}\|_2 \|A\|_2^k = (1 + \mu)^{n-1} \left(\|\Delta\|_2 + \frac{\|N\|_F}{1 + \mu} \right)^k. \end{aligned}$$

On the other hand, if A is nonsingular and $(1 + \mu)|\lambda_{\min}| > \|N\|_F$, then

$$\|\Delta D^{-1}N\Delta^{-1}\|_2 = \|D^{-1}(\Delta N\Delta^{-1})\|_2 \leq \frac{1}{|\lambda_{\min}|} \|\Delta N\Delta^{-1}\|_F < 1.$$

Using Lemma 2.3.3 we obtain

$$\begin{aligned} \|A^{-k}\|_2 &= \|T^{-k}\|_2 = \|\Delta^{-1}[(I + \Delta D^{-1}N\Delta^{-1})^{-1}D^{-1}]^k\Delta\|_2 \\ &\leq \kappa_2(\Delta) \left(\frac{\|D^{-1}\|_2}{1 - \|\Delta D^{-1}N\Delta^{-1}\|_2} \right)^k \leq (1 + \mu)^{n-1} \left(\frac{1}{|\mu| - \|N\|_F/(1 + \mu)} \right)^k \end{aligned}$$

completing the proof of the lemma. \square

Proof of Theorem 7.3.6. By induction it is easy to show that the matrix Q_k in (7.3.6) satisfies

$$A^k Q_0 = Q_k (R_k \cdots R_1),$$

a QR factorization of $A^k Q_0$. By substituting the Schur decomposition (7.3.7)-(7.3.8) into this equation we obtain

$$T^k \begin{bmatrix} V_0 \\ W_0 \end{bmatrix} = \begin{bmatrix} V_k \\ W_k \end{bmatrix} (R_k \cdots R_1) \quad (7.3.17)$$

where

$$V_k = Q_\alpha^H Q_k, \quad W_k = Q_\beta^H Q_k.$$

Our goal is to bound $\|W_k\|_2$ since by the definition of subspace distance given in §2.5.3 we have

$$\|W_k\|_2 = \text{dist}(D_F(A), \text{ran}(Q_k)). \quad (7.3.18)$$

Note from the thin CS decomposition (Theorem 2.5.2) that

$$1 = \Phi_{\sigma_{\min}(V_k)^2}. \quad (7.3.19)$$

Since T_{11} and T_{22} have no eigenvalues in common, Lemma 7.1.5 tells us that the Sylvester equation $T_{11}X - XT_{22} = -T_{12}$ has a solution $X \in \mathbb{C}^{r \times n}$ and that

$$\|X\|_F \leq \frac{\|T_{12}\|_F}{\text{sep}(T_{11}, T_{22})}. \quad (7.3.20)$$

It follows that

$$\begin{bmatrix} r & X \\ 0 & \mathbf{n}_r \end{bmatrix}^{-1} \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \begin{bmatrix} r & X \\ 0 & \mathbf{n}_r \end{bmatrix} = \begin{bmatrix} T_{11} & 0 \\ 0 & T_{22} \end{bmatrix}.$$

By substituting this into (7.3.17) we obtain

$$\begin{bmatrix} T_{11}^k & 0 \\ 0 & T_{22}^k \end{bmatrix} \begin{bmatrix} V_0 - XW_0 \\ W_0 \end{bmatrix} = \begin{bmatrix} V_k - XW_k \\ W_k \end{bmatrix} (R_k \cdots R_1),$$

i.e.,

$$T_1(V_0 - XW) = (V_k - XW)(R_k \cdot R), \quad (7.3.21)$$

$$T_1 W = V_k(R_k \cdot R). \quad (7.3.22)$$

The matrix $I + XX^H$ is Hermitian positive definite and so it has a Cholesky factorization

$$I + XX^H = GG^H. \quad (7.3.23)$$

It is clear that

$$\text{Qmin}(G) = 1. \quad (7.3.24)$$

If the matrix $Z \in \mathbb{C}^{n \times (n-r)}$ is defined by

$$Z = Q \begin{bmatrix} I_r \\ -X^H \end{bmatrix} G^{-H} = [Q_\alpha \ Q_\beta] \begin{bmatrix} I_r \\ -X^H \end{bmatrix} G^{-H} = (Q_\alpha - Q_\beta X^H)G^{-H},$$

then it follows from the equation $AHQ = QTH$ that

$$AHQ = Q(XH - (Q_\alpha - Q_\beta X^H)R). \quad (7.3.25)$$

Since $Z^H Z = I_r$ and $\text{ran}(Z) = \text{ran}(Q - Q_\beta X^H)$, it follows that the columns of Z are an orthonormal basis for $D_1(AH)$. Using the CS decomposition (7.3.19), and the fact that $\text{ran}(Q) = D_1(AH)$, we have

$$\begin{aligned} \text{Qmin}(Z^H Q)^2 &= 1 - \text{dist}(D_1(AH), Q)^2 = 1 - \|IQ(Q)\|^2 \\ &= \text{Qmin}(Q_\beta Q)^2 = \text{Qmin}(V_0)^2 = 1 - d_0^2 > 0. \end{aligned}$$

This shows that

$$V_0 - XW_0 = [I_r \ | \ -X] \begin{bmatrix} Q_\alpha^H Q_0 \\ Q_\beta^H Q_0 \end{bmatrix} = (ZG^H)^H Q_0 = G(Z^H Q_0)$$

is nonsingular and together with (7.3.24) we obtain

$$\|(V_0 - XW_0)^{-1}\|_2 \leq \|G^{-1}\|_2 \|(Z^H Q_0)^{-1}\|_2 \leq \frac{1}{\sqrt{1 - d_0^2}}. \quad (7.3.26)$$

Manipulation of (7.3.19) and (7.3.20) yields

$$W_k = T_2 W (R_k \cdot R)^{-1} = T_2 W (V_0 - XW_0)^{-1} (V_k - XW_k).$$

The verification of (7.3.10) is completed by taking norms in this equation and using (7.3.18), (7.3.19), (7.3.20), (7.3.26), and the following facts:

$$\|T_2\| \leq (1 + \mu)^{n-r-1} (\|A\|_1 + \|N\| / (1 + \mu))^k,$$

$$\|T_1\|^k \leq (1 + \mu)^{-k} (\|A\|_1 + \|N\| / (1 + \mu)),$$

$$\|V_k - XW_k\| \leq \|V_k\| + \|XW_k\| \leq 1 + \|T_2\| / \text{sep}(T_1, T_2).$$

The bounds for $\|\Gamma_2\|_2$ and $\|\Gamma_1\|_1$ follow from Lemma 7.32.

Problems

P7.3.1 eNb Prl..8. fP1rob 5N

P7.3.2 .I..8 .N.3NN.d ...8INba E Rx_{RAK} 1 >1 12 13 ... R = ARA > 1 iae
 >2 MeACSp2Ala), r ip2 la2 1alpp2Mh2ps = span{y, z} (_t^T y, z E Rx_{RAK} Afy + iz) =
 >1 iz: ,c. c. 'r)c._r^T +_r.c.₁^T ._r. $\nabla_{\mathbf{T}}^{\mathbf{x}} \cdot \nabla_{\mathbf{-c}}^{\mathbf{T}} \nabla_{\mathbf{x}}^{\mathbf{T}} \nabla_{\mathbf{3}}$ -_r.c.₁^T ($\nabla_{\mathbf{3}}$ -_r._c_H^T 'c (c+)._r $\nabla_{\mathbf{3}}$
 ,)_r.c.₁^T +_r._T _c_H^T $\lambda_{\mathbf{T}}^{\mathbf{x}}$ S.

P7.3.3 1..IN AE Ron 1 XEROX >1.....AARAK 1

$\geq \geq 1 \geq 2 \geq 3 = 4$, $\aleph_0 \aleph_1$

'-)) > ctSltnci2tIC2 paim pkl 9Meia SpIA+inMC

W d o b k o d o f d o l d o 4 b f d ! d o w f f d u l = i = w) 4 b

P7.3.4 1b A cti positive matr c. \ddot{a} Jv RQ iae ,3 i2ApME R kVAB
~~BIA~~ x vi i 2-a A A's theor m ttip2pmip. A cti Sltcpcl2s,iM2CipMdm2ap.
~~hacs2 Ccaiaap2cr 2aii028,ip pl cpt S2ApMip, p(A) iae pmrM2 Sltcpcl2AplMtl~~
~~pmpAx = p(A)-x f appct Alap2ApMip, p(A) iae pmrM2 Sltcpcl2AplMtl~~
~~3thC2pmip E R kVAB R= Q R kVAB uA (A ad : 1) A) IQA~~
~~MVA2 2 A VD JA2 vli .~~

$$z = Aq, \geq = \Phi$$

while ||z - >q |

$$\text{end } \mathbf{q}_z, \mathbf{q} \mathbf{q}^T \mathbf{q}, z = \mathbf{A} \mathbf{q} > = \mathbf{d}_z$$

N(x0-θ)=u=GE((=x(a-a)N(f=(E(fA•a=1)ia CRak1 Aq>q
 m2M2A A E and A_kWAB WAB A2RWAB x A E RQ(A Collatz-
 Welandt for ula ttip2pmip(A) ctpm2ibcC,C iip21, ppr8aApclz H

$$f(x) = \lim_{n \rightarrow \infty} x$$

$\frac{1}{x} \cdot \frac{1}{y} = \frac{1}{Ax + B}$

P7.3.5 **F7N.. ..N..8I.. ..8..BN.. N... d..8I.. ..5...b** A cti nonnegative matrix c. **aj** v
)RQ iae j. 4 uetpmf E RQ kr ducible n.ppm2M12 S2MC, pipelat ppip A P ctSplAO
 pMniar kpmpml IMCIMts,iM2enir laiSOLA+ t 3 CipMcot alp142e,Ans02 ducible.
 p2 Per n-F benius theor m tpip2pmip. A cti ts,iM2ala2r ipniale nMMAAS02b2(A),
 pm22MM14lptcia 2cr 2aii0MA inc pm2M12 Sltnpcii2Ap1M pm22MMla i2Ap1b10
 Ax = p(A) · x. 3t,C2 pmip 1A 2 RQ RA WAB R= Q(A) (A)RAB RAx AS2
 J2a2eSt

$$A = \begin{bmatrix} 0 & A_1 & 0 \\ 0 & 0 & A_2 \\ A_3 & 0 & 0 \end{bmatrix}.$$

A ntnM₁M₂S₀2, Sn2p_B = A 1A2A3 spl. ml. pl AlCS, ppp212MM₁allpiae ApLM A NC ppp212MM₁allpiae irApLM B. f Aspl. pip^A p. lpm2M₁r 2ai.p.2pp tLhip222sl ipl pm22MM₁pl. AlOepmlt2cr 2aiiQ₂e pm2Acip2e2cr 2ai2ApLM₂APS₂e.

P7.3.6 f7 ..N..N..glro.68 ..8.esN..N. ... d.8l.. m5.8. Nd...N ...b P E ROK stochas-
 tic n.ppp2apMca26ApAlQCatC pl N992ApLM E kRpr bility factor c. cpt2apMca2
 l .2r pni2ae thC pl NMspl. pmip. P E ROK v E ROK ROK ROK ROK ROK ROK ROK ROK

ie tehtite D=Aer nsition pr abilities CtlAndAtoR,tLoMMarkov Chain. nAlj nO- • O
fE = n)nOA0000 = }-i. M+a= tcurrent. oolie Rut-acaneñlie tta,u,n5n14
rlule dLniA = tnext nr dnoeo ,4

$$W_i = \sum_{j=1}^n p_{ij} v_j \quad i = 1, 2$$

LO=••• O=+0.0f0E O=caE+=0E L=OEΔ000OnE)=O= 2AfeJ= 2
f)yO= f)yOyoo.20 =••• 0fE OE pEE O> OiAion f)yOmnji. Ja.lMnluaA ,A
JaimeAM.AlloidM.M.. γ=γ =η 1βγ γι ,;γ BλCγ.. γ

Step 1. 3 Aln^tLttAe,

Step 2. f.n!AliAthSmActaekASSorA, , , , „ .” 0(“ ’0 0.+.(κY;”(0,” (“F0 1,-” 0’ .+)+0+)
 $\cdot \cdot \cdot +(\gamma^+,\gamma^-)”;0”+ “j 0, ” .+0 “\infty .+(κ_1,-)$.

Step 3 ,LmAukyAmaAbISorA.lintnhtucaeli,t tA,AALAbCLmAtL ogSltns,AsorAt4
 $nAL \in \mathbf{R}^{n \times n-1}$,
 $\sigma_1(n), \sigma_2(n), \dots, \sigma_k(n)$

$$h_{ij} = \begin{cases} 0 & \text{if } A_i \cap A_j = \emptyset \\ 1 & \text{otherwise} \end{cases}$$

$$x_{i_1} \geq x_{i_2} \geq \dots \geq x_{i_n}$$

m 8xx x0u x x 0u 0xu

P7.3.7 R.5.8fo 2.2 . \geq_X E $\mathbb{C}^{n \times n}$ (($\sigma \circ \sigma(\sigma, \dots), d \dots$)) $^{\sigma}$

$$\|A\|_F = \|X^{-1}AX\|_2$$

$$\frac{T_1 c_2 s}{1} \} 31 c_2 s \{ \frac{T}{6} + * \{ 3c+i \quad [T_* .(.(1 c_2) + c) 1 c_2 s + ., .(1$$

$$\|AB\|_x \leq \|A\|_x \|B\|_x.$$

$$B^{-\cdot, (c)} \cdot \langle \cdot, r + [z, \epsilon] \rangle_n = \gamma_1 \gamma_2 \cdots \gamma_m x^{\alpha} \chi_{A \times B} \otimes \lambda^{n \times n(m)}(\cdot).$$

$$\|A\|_X = \|X^{-1}AX\|_2 \leq B + \epsilon$$

$$x R \rightarrow f(A)Rf^{-1}A T^k A x = n A x f A x = \begin{pmatrix} T^T & [+B & a_2^T \dots, a^{n-1}] \\ e \end{pmatrix} I_n = D + N T \tilde{x} =$$

P7.3.8 e8.. 2..2 fro .P9...2..28. 2.8n.2..8. .8 .8... frθ .95

P7.3.9 .2..8 .8 $\in \mathbb{C}^{n \times n}$ ($\sigma\sigma(\sigma_{..}), \sigma(_)$). $\in \mathbb{C}^{n \times p}$ ($., \sigma^*, n$). $)^* \dots n\sigma(\rho - 1)1 \bar{G}$.
 .”e($(\sigma, (1, 1))^{*\sigma}$ ($^1, ^1, ^1, \dots$) inverse orthogonal iteration.

for k = 0:a:H .

$$O = A \setminus D = -_1 r + M_j +)E = EO =) = O = p$$

end

Notes and References for §7.3

8. 8b.88.8.8...86 8>8 t .8...8.88..8...8.8...8.fo .88f. ..fN 35f86..
fN1 fo. or.8..8. f N. 5 .8...88.fo8.8>8 .868. 88..8. ..

Prieng PEIj mPEIj]EeAxm = 2neelope r2=2e03n not =r=Aenixen r2pdo)=a+ d2
Pema nr= ot kp=m12 aaeo=ile dm)A In n=2nnp +not ot iig+2 geO

Omega ot TLLPenag Ty, 2. LNonne ative Matrices in the Mathematical Sciences, s,3I
hhSpilAha-hpAppSpcih3M

3x2xniar inpeA AxEnAtAMJJ.7 Google's PageRank and Beyond, hMcaAAhiaAM,dMA,< hMnaAAhiaT ,Mer

LA,iLLAMr A,L,Liaenacal. cIAaaAAL,AlllOl. alr AM,Applf ASMi,AeA,nr a
iao iaiOt,n,l aAS SMI,AM,AA,le

a.9. sLA.iMN, ... I tr duction to the Numer cal Solution of Markov Chains, hMnaAAhiaAM,nLT
hMA,hMnaAA29.

I xarwAMM Mii iae dMr9A,,hS N. ... IiLMcA,3AL,IAAiae,a,Mr iLmALMnAl p<6
SIAM Review 41, "", 71x

3.2. niar inpeA E.EhAtAMJJ., .3 s.MiAtl. dnr AaiAAhiaMe ,MaAS ,a,Mr iLnla
rALMcAS SIAM Review 47, N,r IX.

3x2.niaricOoAe A.EhAtAMJJ.7 .3 rALMeAM,NAh.Ahir Aria+hMSpAS SIAM J. Sci.
Comput. 27, IN (I NJ.

3.2. niar inpeA AxEnAtAMJJ.7 .r SeiLcaFiM+ liAmina,L. a ,r x= ,x = - raθ k x ·r
= x SIAM J. Matrix Anal. Applic. 27, .7dr .d(x

7.4 The Hessenberg and Real Schur Forms

In this and the next section we show how to make the QR iteration (7.3.1) a fast, effective method for computing Schur decompositions. Because the majority of eigenvalue/invariant subspace problems involve real data, we concentrate on developing the real analogue of (7.3.1) which we write as follows:

$$\begin{aligned}
 Hb &= UAb \\
 k &= 1, 2, \dots \\
 H_k &\leftarrow U_k R_k \quad (\text{QR factorization}) \\
 H_k &\leftarrow R_k U_k \\
 \text{end}
 \end{aligned} \tag{7.4.1}$$

Here, $A \in \mathbb{R}^{n \times n}$, each $U_k \in \mathbb{R}^{n \times n}$ is orthogonal, and each $R_k \in \mathbb{R}^{n \times n}$ is upper triangular. A difficulty associated with this real iteration is that the H_k can never converge to triangular form in the event that A has complex eigenvalues. For this reason, we must lower our expectations and be content with the calculation of an alternative decomposition known as the real Schur decomposition.

In order to compute the real Schur decomposition efficiently we must carefully choose the initial orthogonal similarity transformation U_0 in (7.4.1). In particular, if we choose U_0 so that H_0 is upper Hessenberg then the amount of work per iteration is reduced from $O(n^3)$ to $O(n^2)$. The initial reduction to Hessenberg form (the U_0 computation) is a very important computation in its own right and can be realized by a sequence of Householder matrix operations.

Chsnrb 7eSb9S tPhuf) DxhpSf k 3eTf Tktb

A block upper triangular matrix with either 1by 1 or 2by 2 diagonal blocks is upper quasi-triangular. The real Schur decomposition amounts to a real reduction to upper quasi-triangular form.

Theorem 7.4.1 (Real Schur Decomposition). If $A \in \mathbb{R}^{n \times n}$, then there exists an orthogonal $Q \in \mathbb{R}^{n \times n}$ such that

$$Q^T A Q = \begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1m} \\ 0 & R_{22} & \cdots & R_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R_{mm} \end{bmatrix} \quad (7.42)$$

where each R_{ii} is either a 1-by-1 matrix or a 2-by-2 matrix having complex conjugate eigenvalues.

Proof. The complex eigenvalues of A occur in conjugate pairs since the characteristic polynomial $\det(zI - A)$ has real coefficients. Let k be the number of complex conjugate pairs in (A) . We prove the theorem by induction on k . Observe first that Lemma 7.12 and Theorem 7.13 have obvious real analogs. Thus, the theorem holds if $k = 0$. Now suppose that $k \geq 1$. If $\mu = r + i\mu$ is an eigenvalue of A and $\mu \neq 0$, then there exist vectors y and z in \mathbb{C}^n such that $A(y + iz) = (r + ip)(y + iz)$, i.e.,

$$A[y \ z] = [y \ z] \begin{bmatrix} r & \mu \\ -\mu & r \end{bmatrix}.$$

The assumption that $\mu \neq 0$ implies that y and z span a 2-dimensional, real invariant subspace of A . It then follows from Lemma 7.12 that an orthogonal $U \in \mathbb{R}^{n \times n}$ exists such that

$$U^T A U = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix}_{n-2}^2$$

where $T_{11} = \{ \cdot, \cdot \}$. By induction, there exists an orthogonal U so that $U^T A U$ has the required structure. The theorem follows by setting $Q = \text{diag}(h, U)$. \square

The theorem shows that any real matrix is orthogonally similar to an upper quasi-triangular matrix. It is clear that the real and imaginary parts of the complex eigenvalues can be easily obtained from the 2-by-2 diagonal blocks. Thus, it can be said that the real Schur decomposition is an eigenvalue revealing decomposition.

WMSIIB: eb IS est: Se'b 5b uf SEb

We now turn our attention to the efficient execution of a single QR step in (7.41). In this regard, the most glaring shortcoming associated with (7.41) is that each step requires a full QR factorization costing $O(n^3)$ fops. Fortunately, the amount of work per iteration can be reduced by an order of magnitude if the orthogonal matrix U_0 is judiciously chosen. In particular, if $U_0 A U_0 = H_0 = (h_{ij})$ is upper Hessenberg ($h_{ij} = 0$ for $i > j + 1$), then each subsequent H_k requires only $O(n^2)$ fops to calculate. To see this we look at the computations $H = QR$ and $H_k = RQ$ when H is upper Hessenberg. As described in §5.2.5 we can upper triangularize H with a sequence of $n-1$ Givens rotations: $QTH : = G'_1 \dots G'_n H = R$. Here, $G_i = G(i, i+1, \theta_i)$. For the $n=4$ case there are three Givens premultiplications:

$$\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \quad \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \quad \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \quad \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix}.$$

See Algorithm 5.25. The computation $RQ = R(G_1 \cdots G_n I)$ is equally easy to implement. In the $n = 4$ case there are three Givens post-multiplications:

$$\begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & 0 & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix}.$$

Overall we obtain the following algorithm

If H is an n -by- n upper Hessenberg matrix, then this algorithm overwrites H with $H = RQ$ where $H = QR$ is the QR factorization of H .

for $k = 1:n - 1$

$[Q_k, R_k] = \text{givens}(H(k,k), H(k+1,k))$

$$H(kk+1, kn) = \begin{bmatrix} Q_k & R_k \\ -S_k & Q_k \end{bmatrix} H(kk+1, kn)$$

end

for $k = 1:n - 1$

$$H(1:k+1, kk+1) = H(1:k+1, kk+1) \begin{bmatrix} Q_k & S_k \\ -S_k & Q_k \end{bmatrix} \quad H \leftarrow$$

end

Let $Q_k = G(k, k+1, f_k)$ be the k th Givens rotation. It is easy to confirm that the matrix $Q = G_1 \cdots G_n I$ is upper Hessenberg. Thus $RQ = H$ if and only if $Q = G_1 \cdots G_n$.

$$\begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} \quad \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix}$$

$$\begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & x & x & x & x \end{bmatrix} \quad \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x & x \end{bmatrix} \quad \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix}.$$

In general, after $k-1$ steps we have computed $k-1$ Householder matrices P_1, \dots, P_{k-1} such that

$$(P_1 \cdots P_{k-1})^T A (P_1 \cdots P_{k-1}) = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{21} & B_{22} & B_{23} \\ 0 & B_{32} & B_{33} \end{bmatrix}_{\substack{k-1 \\ 1 \\ n-k}}$$

is upper Hessenberg through its first $k-1$ columns. Suppose P_k is an order-($n-k$) Householder matrix such that $P_k B_{32}$ is a multiple of e^{n-k} . If $P_k = \text{diag}(U_k, P_k)$, then

$$(P_1 \cdots P_k)^T A (P_1 \cdots P_k) = \begin{bmatrix} B_{11} & B_{12} & B_{13} \tilde{P}_k \\ B_{21} & B_{22} & B_{23} \tilde{P}_k \\ 0 & \tilde{P}_k B_{32} & \tilde{P}_k B_{33} \tilde{P}_k \end{bmatrix}$$

is upper Hessenberg through its first k columns. Repeating this for $k = 1:n-2$ we obtain

Algorithm 7.4.2 (ef 67.9) Given $A \in \mathbb{R}^{n \times n}$, the following algorithm overwrites A with $H = U J A U_0$ where H is upper Hessenberg and U is a product of Householder matrices.

```

for k = 1:n-2
    [v, beta] = house(A(k+1:n, k))
    A(k+1:n, kn) = (I - v * v^T) A(k+1:n, kn)
    A(1:n, k+1:n) = A(1:n, k+1:n) (I - v * v^T)
end

```

This algorithm requires $10n^3/3$ fops. If U_0 is explicitly formed, an additional $4n^3/3$ fops are required. The k th Householder matrix can be represented in $A(k+2n, k)$. See Martin and Wilkinson (1968) for a detailed description.

The roundoff properties of this method for reducing A to Hessenberg form are very desirable. Wilkinson (AEP, p. 351) states that the computed Hessenberg matrix H satisfies

$$f = QT(A + E)Q$$

where Q is orthogonal and $\|E\|_F \leq c \|A\|_F$ with c a small constant.

~~EA(A)A~~ Foo & Anf ~~A+A~~

The Hessenberg reduction (Algorithm 7.4.2) is rich in level-2 operations, half gaxps and half outer product updates. We briefly mention two ideas for introducing level-3 computations into the process.

The first involves a block reduction to block Hessenberg form and is quite straightforward. Suppose (for clarity) that $n = rN$ and write

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}_{n-r}^r.$$

$r \quad n-r$

Suppose that we have computed the QR factorization $AQ_1 = Q_1$ and that Q_1 is in WY form. That is we have $W, Y \in \mathbb{R}^{(n-r) \times r}$ such that $Q_1 = I + WY$. (See §5.2.2 for details.) If $Q_1 = \text{diag}(I_r, Q)$ then

$$Q_1^T A Q_1 = \begin{bmatrix} A_{11} & A_{12} \tilde{Q}_1 \\ R_1 & \tilde{Q}_1^T A_{22} \tilde{Q}_1 \end{bmatrix}.$$

Notice that the updates of the (1,2) and (2,2) blocks are rich in level-3 operations given that Q_1 is in WY form. This fully illustrates the overall process as $Q_1 A Q_1$ is block upper Hessenberg through its first block column. We next repeat the computations on the first r columns of $Q_1 A Q_1$. After $N - 1$ such steps we obtain

$$H = U_0^T A U_0 = \begin{bmatrix} H_{11} & H_{12} & \cdots & \cdots & H_{1N} \\ H_{21} & H_{22} & \cdots & \cdots & H_{2N} \\ 0 & \ddots & \ddots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & H_{N,N-1} & H_{NN} \end{bmatrix}$$

where each H_{ij} is r -by- r and $U_0 = Q_1 \dots Q_{N-2}$ with each Q_i in WY form. The overall algorithm has a level-3 factorization of the form $O(N^2)(1/N)$. Note that the subdiagonal blocks in H are upper triangular and so the matrix has lower bandwidth. It is possible to reduce H to actual Hessenberg form by using Givens rotations to zero all but the first subdiagonal.

Dongara, Hammarling and Sorensen (1987) have shown how to proceed directly to Hessenberg form using a mixture of gaxps and level-3 updates. Their idea involves minimal updating after each Householder transformation is generated. For example suppose the first Householder P_1 has been computed. To generate P_2 we need just the second column of $P_1 A P_1$, not the full outer product update. To generate P_3 we need just the third column of $P_2 P_1 A P_1 P_2$ etc. In this way, the Householder matrices can be determined using only gaxy operations. No outer product updates are involved. Once a suitable number of Householder matrices are known they can be aggregated and applied in level-3 fashion.

For more about the challenges of organizing a high-performance Hessenberg reduction, see Karlsson (2011).

nhshib (. Ek f ttf bIS eest: \$'b ht f xTeb kE Sf T\$eb

The Hessenberg decomposition is not unique. If Z is any n -by- n orthogonal matrix and we apply Algorithm 7.4.2 to $ZTAZ$, then $QTAQ = H$ is upper Hessenberg where $Q^T = ZU$. However, $Qe_1 = Z(U_1)$. Ze_1 suggesting that H is unique once the first column of Q is specified. This is essentially the case provided H has no zero subdiagonal entries. Hessenberg matrices with this property are said to be unreduced. Here is important theorem that clarifies these issues.

Theorem 7.4.2 (Implicit Q Theorem). Suppose $Q = [Q_1 \cdots Q_n]$ and $V = [v_1 I \cdots I v_n]$ are orthogonal matrices with the property that the matrices $QTAQ = H$ and $VTAV = G$ are each upper Hessenberg where $A \in \mathbb{C}^{n \times n}$. Let k denote the smallest positive integer for which $h_{k+1,k} \neq 0$ with the convention that $k = n$ if H is unreduced. If $Q = V_1$ then $Q^T = V_1$ and $h_{i,i-1} = g_{i,i}$. Moreover, if $k \leq n$, then $g_{i+1,k} = 0$.

Pr of. Define the orthogonal matrix $W \in [w_1 I \cdots I w_n] = VTQ$ and observe that $GW = WH$. By comparing column $i-1$ in this equation if $r_i = 2k$ we see that

$$h_{i,i-1} w_i = g_{i,i-1} - \sum_{j=1}^{i-1} h_{j,i} w_j$$

Since $w_i = e_1$ it follows that $[w_1 I \cdots I w_k]$ is upper triangular and so if $r_i = 2k$ we have $W = \pm I_{r_i}(i) = \pm e_i$. Since $W = VrQ$ and $h_{i,i-1} = w_i^T G w_i$ it follows that $W = \pm e_i$ and

$$h_{i,i-1} = h_i^T A Q_{i-1} = M_{i,i-1} = g_{i,i-1}$$

if $r_i \neq 2k$. If $k \leq n$, then

$$\begin{aligned} k+1k &= f_{+} G_{k+1,k} - e_{+1}^T G W_{k+1,k} = f_{+1} W_{k+1,k} \\ &= e_{+1}^T \sum_{i=1}^k h_{ik} w_i = \pm e_{+1}^T e_{+1} = 0 \end{aligned}$$

completing the proof of the theorem. \square

The gist of the implicit Q theorem is that if $QTAQ = H$ and $ZTAZ = J$ are each unreduced upper Hessenberg matrices and Q and Z have the same first column, then G and H are "essentially equal" in the sense that $G = V^{-1}HD$ where $D = \text{diag}(\pm 1, \dots, \pm 1)$.

Our next theorem involves a new type of matrix called a Krylov matrix. If $A \in \mathbb{C}^{n \times n}$ and $v \in \mathbb{C}^n$, then the Krylov matrix $K(A, v, j) \in \mathbb{C}^{j \times n}$ is defined by

$$K(A, v, j) = [v | Av | \cdots | A^{j-1}v].$$

It turns out that there is a connection between the Hessenberg reduction $QTAQ = H$ and the QR factorization of the Krylov matrix $K(A, Q(:, 1), n)$.

Theorem 7.4.3. Suppose $Q \in \mathbb{C}^{n \times n}$ is an orthogonal matrix and $A \in \mathbb{C}^{n \times n}$. Then $QTAQ = H$ is an unreduced upper Hessenberg matrix if and only if $QTK(A, Q(:, 1), n) = R$ is nonsingular and upper triangular.

Pr of. Suppose $QEJ^{n \times n}$ is orthogonal and set $H = QTAQ$. Consider the identity

$$Q^T K(A, Q(:, 1), n) = [e_1 | He_1 | \dots | H^{n-1} e_1] \equiv R.$$

If H is an unreduced upper Hessenberg matrix, then it is clear that R is upper triangular with $r_{ii} = h_{21}h_{32} \dots h_{ii}$ if $i = 2n$. Since $r_{nn} = 1$ it follows that R is nonsingular.

To prove the converse, suppose R is upper triangular and nonsingular. Since $R(:, k+1) \in \text{span}\{e_i, \dots, e_{k+1}\}$, this implies that H is upper Hessenberg. Since $r_{nn} = h_{21}h_{32} \dots h_{nn} \neq 0$ it follows that H is also unreduced. \blacksquare

Thus, there is more or less a correspondence between nonsingular Krylov matrices and orthogonal similarity reductions to unreduced Hessenberg form.

Our last result is about the geometric multiplicity of an eigenvalue of an unreduced upper Hessenberg matrix.

Theorem 7.4.4. If λ is an eigenvalue of an unreduced upper Hessenberg matrix $H \in \mathbb{C}^{n \times n}$, then its geometric multiplicity is 1.

Pr of. For any $\lambda \in \mathbb{C}$ we have $\text{rank}(A - \lambda I) = n - 1$ because the first $n - 1$ columns of $A - \lambda I$ are independent. \blacksquare

Onshrb .T.3tTTtb ht QxTeBTx b

Just as the Schur decomposition has a nonunitary analogue in the Jordan decomposition, so does the Hessenberg decomposition have a nonunitary analog in the companion matrix decomposition. Let $x \in \mathbb{C}^n$ and suppose that the Krylov matrix $K = K(A, x, n)$ is nonsingular. If $c \in \mathbb{C}^{(n-1)}$ solves the linear system $Kc = -Ax$, then it follows that $AK = KC$ where C has the form

$$C = \begin{bmatrix} 0 & 0 & \cdots & 0 & -c_0 \\ 1 & 0 & \cdots & 0 & -c_1 \\ 0 & 1 & \cdots & 0 & -c_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & -c_{n-1} \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}. \quad (7.44)$$

The matrix C is said to be a companion matrix. Since

$$\det(zI - C) = c_0 + c_1z + \cdots + c_{n-1}z^{n-1} + z^n,$$

it follows that if K is nonsingular, then the decomposition $K^{-1}AK = C$ displays A 's characteristic polynomial. This, coupled with the sparseness of C , leads to "companion matrix methods" in various application areas. These techniques typically involve

Step 1 Compute the Hessenberg decomposition $U^{-1}AU = H$.

Step 2 Hope H is unreduced and set $Y = [e_1 | He_1 | \dots | H^{n-1}e_1]$.

Step 3 Solve $YC = HY$ for C .

Unfortunately, this calculation can be highly unstable. A is similar to an unreduced Hessenberg matrix only if each eigenvalue has unit geometric multiplicity. Matrices that have this property are called nonderogatory. It follows that the matrix Y above can be very poorly conditioned if A is close to a derogatory matrix.

ⁱⁱ A full discussion of the dangers associated with companion matrix computation bef und in Wilkinson (AEP, pp 40f.).

Problems

P7.4.2 88.888. ...8.8.2. .8. B8.8.8.. N.8 2... N.2.. ...N.8. fo. .38 ...nPgfo
 ... 8.8.8.8.2.8. ro88 P2.....f 8.55

P7.4.3 **re.888..** 2.2.8.6 .. .8.8..... 8.8. 18.8...288 A₊ zI)x = b vI .en. mMa lan.
 P7.4.3 **re.888..** 2.2.8.6 .. .8.8..... 8.8. 18.8...288 A₊ zI)x = b vI .en. mMa lan.
 P7.4.3 **re.888..** 2.2.8.6 .. .8.8..... 8.8. 18.8...288 A₊ zI)x = b vI .en. mMa lan.

P7.4.4 .2.8 .8 HER **HE** **R** **M** **J** **D** **O** **i** **E** **i** **O** **(** **)** **)** **J** **V** **H** **D** **M** **J** **M** **i** **c** **i** **(** **)** **M** **(** **)**

P7.4.5 .2. .8.8 W,Y E Rm()", " 1 iP ,)"(0 B St

$$' = W + iY, \quad B = \begin{bmatrix} - & \\ & \ddots & \\ & & ; \end{bmatrix}.$$

() \otimes [T E r t), "(1 ." E r t 1.(" i-)", ("+")), "3.")." ,0

P7.4.6 .2. .8.8

$$A = \begin{bmatrix} & \\ & \end{bmatrix}$$

Q = K(x)(C) $x = \text{O}_{\text{BoEdq}} \cdot 1$ **C = K(x) \oplus W_{\text{Hil}}** $\oplus K \oplus W_{\text{Hil}}$

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} w & x \\ y & z \end{bmatrix} \begin{bmatrix} c & s \\ -s & c \end{bmatrix} = \begin{bmatrix} \lambda & \beta \\ \alpha & \lambda \end{bmatrix}$$

. ΟΕΟ = 'α .

P7.4.7 .2.8 .8₀ **6^TR_E** **F_{2,3d[3,1c,2s,T+1c,2s3-\nabla,-]}** **1c_{2s,T}\nabla T_{c,1c,2s3-\nabla,-]}** **1c_{2s,T}\nabla T_{c,1c,2s3-\nabla,-]}** **H E F × n**

pTHP [f_A W_H]

... H₁(H₁(x₁))(+)... H₁(...)(o .J¹ " Pa (+))¹ "23"(_,"1 (")"(o

P7.4.8 .2..8.8 HERxn "c"("0c) Oq= q il-E Tf. QER xn(+'). "2
) ("."(")(1 Q. .. QHD (+", H(":" Hc I "+(("y")

P7.4.9 .38fo2l. .88...8. 8. .bfo. .L ... 8..8.. .28.n¹ " VCV- 1=

$$V = \begin{bmatrix} 1 & \lambda_1 & \cdots & \lambda_1^{n-1} \\ 1 & \lambda_2 & \cdots & \mathbf{n} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_n & \cdots & \end{bmatrix}.$$

Notes and References for §7.4

.88..32. .8.88.8.. .8.fo 8.....88. ..8.8. 8-.or

N2..3.. 1§ f.2.8. f.9 521 3.8...8 8.8N. 7... N. ...8.ro.8. fo. 18.8...
 ..N.8. 8.67r Nat. Acad. Sci. 17..(IJ,

O sMhM seores 18s-entplia sr tegaih krd ighiAa ia Eit ia ga Duq Mca H. at
Ohr cpting ocean iz

TaGlos ia ot ItE-ia ga 4, E7B bCho ntu Telpia 180 Qero wsiA sr tvgeaih
1 m75 Numer. Math 12, ,q m7dr

.ciAa, MlpipdAia iSAl,AeplAlr SIppAA,,3aS)MeAAAlr Shp,AAe

a4ripp

4x g K

7.5 The Practical QR Algorithm

We return to the Hessenberg QR iteration, which we write as follows

$$\begin{aligned}
 H &= U A U b \quad (\text{Hessenberg reduction}) \\
 \text{for } k &= 1, 2, \dots \\
 H &= U R \quad (\text{QR factorization}) \\
 H &= R U \\
 \text{end}
 \end{aligned} \tag{7.51}$$

Our aim in this section is to describe how the H 's converge to upper quasi-triangular form and to show how the convergence rate can be accelerated by incorporating shifts.

Slowly it is over

Without loss of generality we may assume that each Hessenberg matrix H in (7.51) is unreduced. If not, then at some stage we have

$$H = \begin{bmatrix} H_{11} & H_{12} \\ 0 & H_{22} \end{bmatrix}_{n-p}$$

where $1 \leq p \leq n$ and the problem decouples into two smaller problems involving H_{11} and H_{22} . The term deflation is also used in this context, usually when $p = n - 1$ or $n - 2$.

In practice, decoupling occurs whenever a subdiagonal entry in H is suitably small. For example, if

$$|h_{p+1,p}| \leq c u (|h_{pp}| + |h_{p+1,p+1}|) \tag{7.52}$$

for a small constant c , then $h_{p+1,p}$ can justifiably be set to zero because rounding errors of order $u \|H\|_1$ are typically present throughout the matrix anyway.

minib 7eSb ueTlf Sb 59b (f Sf Tkb

Let $\mu \in \mathbb{C}$ and consider the iteration

$$\begin{aligned}
 H &= U A U b \quad (\text{Hessenberg reduction}) \\
 \text{for } k &= 1, 2, \dots \\
 \text{Determine a scalar } \mu. \\
 H - \mu I &= U R \quad (\text{QR factorization}) \\
 H &= R U + \mu I \\
 \text{end}
 \end{aligned} \tag{7.53}$$

The scalar μ is referred to as a shift. Each matrix H generated in (7.53) is similar to A , since

$$R U + \mu I = U^T (U R + \mu I) U = U I H U$$

If we order the eigenvalues λ_i of A so that

$$|\lambda_1 - \mu| \leq \dots \leq |\lambda_n - \mu|$$

and μ is fixed from iteration to iteration, then the theory of §7.3 says that the p th subdiagonal entry in H converges to zero with rate

$$\left| \frac{\lambda_{p+1} - \mu}{\lambda_p - \mu} \right|^k.$$

Of course, if $\lambda_p = \lambda_{p+1}$, then there is no convergence at all. But if, for example, μ is much closer to λ_n than to the other eigenvalues, then the zeroing of the $(n, n-1)$ entry is rapid. In the extreme case we have the following.

Theorem 7.5.1. Let μ be an eigenvalue of an n -by- n unreduced Hessenberg matrix H . If

$$H = RU + \mu I,$$

where $H - \mu I = UR$ is the QR factorization of $H - \mu I$, then $h_{nn} \rightarrow 0$ and $h_{nn} = \mu$.

Proof. Since H is an unreduced Hessenberg matrix the first $n-1$ columns of $H - \mu I$ are independent, regardless of μ . Thus, if $UR = (H - \mu I)$ is the QR factorization then $r_{ii} = 0$ for $i = 1:n-1$. But if $H - \mu I$ is singular, then $r_u \dots r_{n-1} = 0$. Thus, $T_n = 0$ and $\tilde{H}(n, :) = [Q, \dots, 0, \mu]$. \square

The theorem says that if we shift by an exact eigenvalue, then in exact arithmetic deflation occurs in one step.

Third method: shift by an exact eigenvalue

Now let us consider varying μ from iteration to iteration incorporating new information about $A(A)$ as the subdiagonal entries converge to zero. A good heuristic is to regard h_{nn} the best approximate eigenvalue along the diagonal. If we shift by this quantity during each iteration, we obtain the single-shift QR iteration

for $k = 1, 2, \dots$

$$\begin{aligned} \mu &= H(n, n) \\ H - \mu I &= UR \quad (\text{QR factorization}) \\ H &= RU + \mu I \end{aligned} \tag{7.54}$$

end

If the $(n, n-1)$ entry converges to zero, it is likely to do so at a quadratic rate. To see this, we borrow an example from Stewart (IMC, p. 366). Suppose H is a unreduced upper Hessenberg matrix of the form

$$H = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & \epsilon & h_{nn} \end{bmatrix}$$

and that we perf more step of the singleshift QR algorithm i.e,

$$\begin{aligned} \mathbf{U}\mathbf{R} &= \mathbf{H} - \mathbf{h}_m \\ \mathbf{f} &= \mathbf{R}\mathbf{U} + \mathbf{h}_m. \end{aligned}$$

After $n-2$ steps in the orthogonal reduction of $\mathbf{H} - \mathbf{h}_m$ to upper triangular f mwe obtain a matrix with the following structure

$$\mathbf{H} = \begin{bmatrix} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & a & b \\ 0 & 0 & 0 & \epsilon & 0 \end{bmatrix}.$$

It is not hard to show that

$$\|\mathbf{h}_m\| = -\frac{\epsilon^2 b}{a^2 - \epsilon^2}$$

If we assume that $\epsilon \ll a$ then it is clear that the new $(n, n-1)$ entry has order ϵ^2 , precisely what we would expect of a quadratically converging algorithm

nhhsb 7esb pkDPS sue1 f b1 xQS1b

Unfortunately, difficulties with (7.54) can be expected if at some stage the eigenvalues a_1 and a_2 of

$$\mathbf{G} = \begin{bmatrix} h_{mm} & h_{mn} \\ h_{nm} & h_{nn} \end{bmatrix}, \quad m = n-1, \quad (7.55)$$

are complex for then \mathbf{h}_m would tend to be a poor approximate eigenvalue

A way around this difficulty is to perform two single shift QR steps in succession using a_1 and a_2 as shifts

$$\begin{aligned} \mathbf{H} - a_1 \mathbf{I} &= \mathbf{U}_1 \mathbf{R}_1 \\ \mathbf{H}_1 &= \mathbf{R}_1 \mathbf{U}_1^+ a_1 \mathbf{I} \\ \mathbf{H}_1 - a_2 \mathbf{I} &= \mathbf{U}_2 \mathbf{R}_2 \\ \mathbf{H}_2 &= \mathbf{R}_2 \mathbf{U}_2^+ a_2 \mathbf{I} \end{aligned} \quad (7.56)$$

These equations can be manipulated to show that

$$(\mathbf{U}_1 \mathbf{U}_2)(\mathbf{R}_2 \mathbf{R}_1) = M \quad (7.57)$$

where M is defined by

$$M = (\mathbf{H} - a_1 \mathbf{I})(\mathbf{H} - a_2 \mathbf{I}). \quad (7.58)$$

Note that M is a real matrix even if G 's eigenvalues are complex since

$$M = H^2 - SH + tI$$

where

$$S = a_1 + a_2 = \|h_m\| \quad h_m = \text{tr}(G) E J$$

and

$$t = \mathbf{a}^T \mathbf{a} = \mathbf{h}_{11} \mathbf{h}_{11} - \mathbf{h}_{22} \mathbf{h}_{22} + \det(Q) E R.$$

Thus, (7.57) is the QR factorization of a real matrix and we may choose U_1 and U_2 so that $Z = U_1 U_2$ is real orthogonal. It then follows that

$$H_2 = U_2^H H_1 U_2 = U_2^H (U_1^H H U_1) U_2 = (U_1 U_2)^H H (U_1 U_2) = Z^T H Z$$

is real.

Unfortunately, roundoff error almost always prevents an exact return to the real field. A real H_2 could be guaranteed if we

- explicitly form the real matrix $M = H^2 - sH + tI$,
- compute the real QR factorization $M = ZR$, and
- set $H_2 = Z^T H Z$.

But since the first of these steps requires $O(n^3)$ fops, this is not a practical course of action.

Chimib C) \$b pDPS c.(EP TfTs) Tl f hif . tf \$'Kb

Fortunately, it turns out that we can implement the double shift step with $O(n^2)$ fops by appealing to the implicit Q theorem of §7.45. In particular we can effect the transition from H to H_2 in $O(n^2)$ fops if we

- compute $M e_1$ the first column of M ,
- determine a Householder matrix P_0 such that $P_0^T M e_1$ is a multiple of e_1 ,
- compute Householder matrices P_1, \dots, P_{n-2} such that if

$$Z_1 = P_0 P_1 \cdots P_{n-2},$$

then $Z^T H Z_1$ is upper Hessenberg and the first columns of Z and Z_1 are the same.

Under these circumstances, the implicit Q theorem permits us to conclude that, if $Z^T H Z_1$ and $Z_1^T H Z_1$ are both unreduced upper Hessenberg matrices, then they are essentially equal. Note that if these Hessenberg matrices are not unreduced, then we can effect a decoupling and proceed with smaller unreduced subproblems.

Let us work out the details. Observe first that P_0 can be determined in $O(1)$ fops since $M e_1 = [x, y, z, 0, \dots, 0]^T$ where

$$x = h_{11} + h_{12} h_{21} - s h_{11} + t,$$

$$y = h_{21}(h_{11} + h_{22} - s),$$

$$z = h_{22} h_{21}.$$

Since a similarity transformation with P_0 only changes rows and columns 1, 2 and 3 we see that

$$P_d H P_0 = \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix}.$$

Now the mission of the Householder matrices P_1, \dots, P_n is to restore this matrix to upper Hessenberg form. The calculation proceeds as follows:

$$\begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix} \xrightarrow{P_1} \begin{bmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix} \xrightarrow{P_2}$$

$$\begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & x & x & x & x \end{bmatrix} \xrightarrow{P_3} \begin{bmatrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix} \xrightarrow{P_4} \begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix}.$$

Each P_k is the identity with a 3by-3 or 2by-2 Householder somewhere along its diagonal, e.g.,

$$P_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & x & x & x & 0 & 0 \\ 0 & x & x & x & 0 & 0 \\ 0 & x & x & x & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad P_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & x & x & x & 0 \\ 0 & 0 & x & x & x & 0 \\ 0 & 0 & x & x & x & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$P_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x & x \end{bmatrix}, \quad P_4 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & 0 & 0 \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix}.$$

The applicability of Theorem 7.43 (the implicit Q theorem) follows from the observation that P_k is the identity for $k=1:n-2$ and that P_0 and Z have the same first column. Hence, $Z \equiv P_0^{-1} Z \equiv I$, and we can assert that Z_1 essentially equals Z provided that the upper Hessenberg matrices $Z_{\text{H}}Z$ and Z_{HZ} are each unreduced.

The implicit determination of H_2f on H outlined above was first described by Francis (1961) and we refer to it as a Francis QR step. The complete Francis step is summarized as follows.

m34s.el(- 7.5.1 e31T314.p25 p.09xxrp Given the unreduced upper Hessenberg matrix $H \in \mathbb{R}^{n \times n}$ whose trailing 2by2 principal submatrix has eigenvalues a_1 and a_2 this algorithm overwrites H with $Z^T H Z$, where Z is a product of Householder matrices and $Z^T(H - a_1 I)(H - a_2 J)$ is upper triangular.

$m = n - 1$

{Compute first column of $(H - a_1 I)(H - a_2 J)$ }

$s = H(m, m) + H(n, n)$

$t = H(m, m) \cdot H(n, n) - H(m, n) \cdot H(n, m)$

$x = H(1, 1) \cdot H(1, 1) + H(1, 2) \cdot H(2, 1) - s \cdot H(1, 1) + t$

$y = H(2, 1) \cdot (H(1, 1) + H(2, 2) - s)$

$z = H(2, 1) \cdot H(3, 2)$

for $k = 0$ to $n - 3$

[v, J] = house([x, y, z]^T)

$q = \max\{1, k\}$.

$H(k+1:k+3, q) = (I - v w^T) \cdot H(k+1:k+3, q)$

$r = \min\{k+4, n\}$

$H(l:r, k+1:k+3) = H(l:r, k+1:k+3) \cdot (I - v w^T)$

$x = H(k+2:k+1)$

$y = H(k+3:k+1)$

if $k < n - 3$

$z = H(k+4:k+1)$

end

end

[v, J] = house([x, y, J]^T)

$H(n-1:n, n-2:n) = (I - v w^T) \cdot H(n-1:n, n-2:n)$

$H(l:n, n-1:n) = H(l:n, n-1:n) \cdot (I - v w^T)$

This algorithm requires $10n^2 f$ ops. If Z is accumulated into a given orthogonal matrix, an additional $10n^2 f$ ops are necessary.

Onimb C)Sb 6f § tPPb kSeeb

Reduction of A to Hessenberg form using Algorithm 7.4.2 and then iteration with Algorithm 7.5.1 to produce the real Schur form is the standard means by which the dense unsymmetric eigenproblem is solved. During the iteration it is necessary to monitor the subdiagonal elements in H in order to spot any possible decoupling. How this is done is illustrated in the following algorithm.

p 2ishdi- 7.5.2 e25ph(. 611. 9op Given AE $\in \mathbb{R}^{n,n}$ and a tolerance tol greater than the unit roundoff, this algorithm computes the real Schur canonical form $Q\tilde{T}A^{-1}Q^T = T$. If Q and T are desired, then T is stored in H. If only the eigenvalues are desired, then diagonal blocks in T are stored in the corresponding positions in H.

Use Algorithm 7.4.2 to compute the Hessenberg reduction

$$H = UAU^T \text{ where } U = P_1 \cdots P_n \in \mathbb{R}^{n,n}$$

If Q is desired $f \text{ mm}Q = P_1 \cdots P_n \in \mathbb{R}^{n,n}$ (See §5.16)

until $q = n$

Set to zero all subdiagonal elements that satisfy:

$$\text{ii. } 1 - \text{tol} \cdot (\text{iii. } i + \text{ii. } i - 1).$$

Find the largest nonnegative q and the smallest nonnegative p such that

$$H = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ 0 & H_{22} & H_{23} \\ 0 & 0 & H_{33} \end{bmatrix} \begin{matrix} p \\ n-p-q \\ q \end{matrix}$$

where H_{33} is upper quasi-triangular and H_{22} is unreduced

if $q \neq n$

Perform main F and QR step on H_{22} $H_{22} = z^T H_{22} z$

if Q is required

$$Q = Q \cdot \text{diag}(z, Z, I_d)$$

$$H_{22} = H_{22} z$$

$$H_{22} = z^T H_{22} z$$

end.

end

end

Upper triangularize all 2by2 diagonal blocks in H that have real eigenvalues and accumulate the transformations (if necessary).

This algorithm requires $25n^3 fops$ if Q and T are computed. If only the eigenvalues are desired, then $10n^3 fops$ are necessary. These fops counts are very approximate and are based on the empirical observation that on average only two F and iterations are required before the lower 1by1 or 2by2 decouples.

The roundoff properties of the QR algorithm are what one would expect of any orthogonal matrix technique. The computed real Schur form \tilde{T} is orthogonally similar to a matrix near to A, i.e.,

$$Q\tilde{T}(A+E)Q^T = \tilde{T}$$

where $Q^T Q = I$ and $\|E\|_F \leq \|A\|_F \text{ tol}$. The computed Q is almost orthogonal in the sense that $Q^T Q = I + F$ where $\|F\|_F \leq \|A\|_F \text{ tol}$.

The order of the eigenvalues along \tilde{T} is somewhat arbitrary. But we discuss in §7.6, any ordering can be achieved by using a simple procedure for swapping two adjacent diagonal entries.

7.5.7 $U = R \cdot D^{-1} \cdot R^T$

Finally, we mention that if the elements of A have widely varying magnitudes, then A should be balanced before applying the QR algorithm. This is an $O(n^3)$ calculation in which a diagonal matrix D is computed so that if

$$V^{-1}AD = [c_1 \ c_2 \ \dots \ c_n] = \begin{bmatrix} r_1^T \\ \vdots \\ r_n^T \end{bmatrix}$$

then $\|r_1\| \approx \|c_1\| \approx \dots \approx \|c_n\| = 1/n$. The diagonal matrix D is chosen to have the form

$$D = \text{diag}(i_1, \dots, i_n)$$

where $i.$ is the floating point base. Note that $D^{-1}AD$ can be calculated without roundoff. When A is balanced, the computed eigenvalues are usually more accurate although there are exceptions. See Parlett and Reinsch (1969) and Watkins (2006).

Problems

P7.5.1 .316 ... $\underline{\underline{H}} = Q^T H Q$ n,SpinaAe SSM,Mr c car pA,,nhe,p3Scpp

$$H = \begin{bmatrix} w & x \\ y & z \end{bmatrix}$$

ppAa { jy²x/(w n +

P7.5.2 N.m,A ∈ A-1.11 (. D10() +,I.)4 D ∈ - . 0)(|| D⁻¹AD ||F c,r cacAeA

P7.5.3 Pp.es..16 .38..d881.l.2 ..8. H- μI = UR, = RU + μI AiaSAAiMMc,qrSpca2 pp lpipl „p., p., pp pMia,cpcqa H pl AiaSAAiMMhSpempx.,hSpMiApqat.cpm, Mr ppAcir,aip H.

P7.5.4 .2.1 .8 H c, hSSAMaSANheppiA A,r ShppA Ap,Mci,peal = LU icEih,nia Apcr .caicppSiMpSipi,pnaf 3A3pr ,Mnpf,r,rspl. pipHi = U(P^TL) cr xSSAM .3,AaSA >ae,cr iMp, H. f lpn, ppA,i,c,.. p., modif ed LR algorithm,

P7.5.5 .316 13. $\underline{\underline{H}} = H_0 c, r r i A A e A B A a A M p A ip M e A i c i H_k - \mu_{kl} = U_k R_k, H_{k+1} = R_k U_k + \mu_{kl}, p M A (1 \dots U_j) (R_i \dots R_l) = (H - \mu_i^j) \dots (H - \mu_j^l)$.

Notes and References for §7.5

P..1...eses8.1.... ...8.. .1.. .06.23.38t ..8...1. ..es..8

P 2..3.2.8.f.. 72.1es2..11 P.d8..ees281.es886.. 1385 ..roN.8..1.R7Nat. Bur. Stand. App. Math. Ser. 49, T NM 9.22; iaAnf. 7(.. lpA Vr hia,Mr cpn,a&r anpiMip,r hA, pmAr hia,Mr ipc,a. hiMifiae f f Comput. J. 4, I 7,rTI „II,“ . Br 2,5hSpA,+ iti f (7N „a sr A3pr ,Mnpf,ppAs,phpc,a. ppAA,r SOA AaiipIA hM,SpA,Vychisl. Mat. Mat. Fiz 1(4), „r ,(Jr r,s2niMpia 9,2acp+ na f (7d „lmr n,ec= 3enr 30r ,McqMfA,r SpAA,AaSAMfr pMnA,Numer. Math. 12, „7,r,T7. r252iMpca2 hApAMe 92.2acp+aia f NJ. „lm3Vr 3pr ,McqMf ipA,,AaSAMipMcAAt Numer. Math. 14, IN I,N9

,Mi r AaAMip,pp MA,r Aaet

E.82ip+caf (dI „r aeAM,piaepr „W 3pr ,Mn S6AM Review 24, ITr „JM E 22ip+na f N. Ms,r A hAM,SAAlpmr AaiipM,SAX S6AM Review 35, 6B 100k

BCES lag 9227-B Cpt T Ohnson Telgeset .5 SIAM Review 50, (55r), M
E.s. aip + na,J ((4uMiaA, 13prlMpyg)mer. Math Monthly 118, 5d(r ,J5.

ri perh tar-na-l tcoTbTdtaherierAltdoTelueTr.l bTccalr-Si octalarl newieGlooberbarw2eGl

r.al lobaclerl pyl(lAlo 2endivwwisAd ieSictaltl uebbeasenil iad vuuuegdhhdrdi itaiAl aV
onreBIT 11, I 71r I M

Esmip+ niae, v 9pr arM. A. nar3prlMnp.rM,p,A 9crAaiip,jhulSOAr SIAM J.
Matrix Anal. Applic. 12, 5,r 5d,M

E.s. ai p+ niae, ., 90r aHM4r AlaiAMrAdA30rlMnp,g EAAlgSlr n16SA,u pA
9crAaiQ,BulSpA6Lin Alg. Applic. 149, (r ,).

9xMT elaf. M 21pAla p.AE1,SO4SympVn3prlMnp,g.in. Alg. Applic. 171, (I (15IM
3M3ESM,piae ... lplS = (, .. 3 nl 03,npVr f pAMipal,p Alr SI pipihlap.A
smce,6N mer. Algor thms 1, .+ : Z.

,« Da 80-M2p «D2= 30=0 m(A)0 e+ T 0+0v00x= w0 yAye=EwSIAMJ.
Matr x Anal. Applic. 16, 7,r ,d.

EMaMpBca (74 l.r .iar c,cla l. s.ner siaqwpMMmopAVr iprlMr .6in. Alg
Applic. 241-9, d((d7M

Exsa pBca (d4M, Or3T AianaA3prlMnp,l Vr ltS A.6SIAM J. Matr x Anal. Applic.
19, (J(l (J,7M

GMeAASMHQ(4. Ay caw,prAlMlpipda, 3 n3pir lMS.llnp.AVr,3prlMnp,6SIAM.
J. Matr x Anal. Applic. 92, I (rI,(

3.SAApl, n.ASiOiaAnA6MISp8MAn,A,,Aecae

dM9SEMaA (7J4,a MAapcladcrnpMnAA,6CM 7, 55dr, M

wMPiMpApp Ax AcarA(7..MwpiAcarI ipMcMAipA,pirlladcrAaiiQ,Aae 9nrAa
iAAPlM6mer. Math 1< I,lr 5J,M

EMaMp+ na,JJ 7M3 Ai,Aa,AMwpiAAnar.iMr J6ETNA 29 (,M

BAM,nld, pyAOrlunpg.ip iMpiSOAMAlgSianla ipMiMAlcr „A8aT

Ex3Miac Eieec,iae, r Ag csaiae I,IJM,a p.AsympAvr np3Mip,8S6Ap1AlgSianla
IipMcAA6NA 18, (5r (I.

4x Bia wiMp.rMiaeASdMBia Ellu3a.iae 5M AeAMeT(J,Mfr pSAelSpAs.ce Vr-
30rlMnp,6Alr SiachipMn6A mer. Math 116 ((r IIM

iSAM, piMAlaAAuaApypyAynr.,Sru,ur iAA SOAgAapipqan3 npAMip,6Alr ApLeAe

z.) a & -k0 L0.0.Or MZ2p Di=a,r= E,fAO.0= a 30(-+-0=HO=Al-w x= OE(a N
Int. J. High Speed Comput. 1, t. f iNw

r w4w.a iaSjh 7itE dwh ppah,S15nih9aqarvp (epc.a p3 Pat,lmr7SIAM J. Matr x
Anal. Applic. 14, (dJ ,M

EMaMp+ na,(.. .sympcaspm>pHmipyAuiOpA0 3prMcp,6SIAM J. Sci. Comput. 15,
„5r .edM

g. 30= Ee- ed>a000(- M2p DXa = a AA0y0y-y=CE-wD+ eTT0= = (E yAy
- (y0= >A= 0rAHe= .a =eOa A (SIAM NSci. Comput. 11>Z+ ZZ' «

z.) a (sMA...A > J, .= = Aa y->A= 030= & +>a == 00-T0 M2p D.wOfOE= = a
,OE= .f= + 0-h= = x e.0-Ha OE= (E0+= = (rHOT=Oke = a=rAOA 6SIAM J.
Sci. Comput. 18, (,7 7(M

g. 30= Ee- & a >k04 = ya M2p a pDxa = aTAfAOAT0= = a w0= = eTT0= E
.e yAy= = O=0w.= = OH6F=09-eEw(= OESIAPM-J Sc> NComput. 24, Id, 5(M

xMir ira wtAMae r, Iip.c =J1.. l.A nIpc,ceVr 3prlMnpchiM6Tinapinanar
aAOp,IA, Aap, >aen3i35lAu,MgiaA66SIAM J. Matr x Anal. Applic. 29, I,r , (.
MMigiarMvt3Miae r. n.ip.c =I,1JMLA nI Op,ynYer 30rlMnp hiupf errMA,,niA
diuptE Alpnla,6SIAM J. Matrix Anal. Applic. 29, „dr , (5M

4xli.iAt = J,B.63prlMnpd7B hiMppAaipiA,ApnaA,MAlr SpATAaSAutipMc-
AA,6CM T ns. Math Sdf w. 29, 5I 7557M

D. T=0+ M2p DDi= w0= \$a 0)yOA90+ wE yAy= wTOENNA 20, J175M

D. TEO+ M2p x DJO ! OEyy= a + + 0>One= (== Oe= > Oey=0+ E0 yAy
E0• 6SIAM J. Matr x Anal. Applic. 90, dJ, dl(M

7.6 Invariant Subspace Computations

Several important invariant subspace problems can be solved once the real Schur decomposition $QTAQ = T$ has been computed. In this section we discuss how to

- compute the eigenvectors associated with some subset of $\Lambda(A)$,
- compute an orthonormal basis for a given invariant subspace,
- block-diagonalize A using well-conditioned similarity transformations,
- compute a basis of eigenvectors regardless of their condition, and
- compute an approximate Jordan canonical form of A .

Eigenvector/invariant subspace computation for sparse matrices is discussed in §7.3.1 and §7.3.2 as well as portions of Chapters 8 and 10.

CLBljs_1ApR.pjs .I fpFpR .a{.s FI R4Fp{-ps .. p{R.I a4

Let $q() \in \mathbb{J}^n$ be a given unit 2-norm vector and assume that $A - \mu I \in \mathbb{J}^{n \times n}$ is nonsingular. The following is referred to as *inverse iteration*:

for $k = 1, 2, \dots$

$$\begin{aligned} & \text{Solve } (A - \mu I)z^{(k)} = q^{(k)}. \\ & q^{(k)} = z^{(k)} / \|z^{(k)}\|_2 \\ & Aq^{(k)} = q^{(k)}{}^T A q^{(k)} \end{aligned} \tag{7.6.1}$$

end

Inverse iteration is just the power method applied to $(A - \mu I)^{-1}$.

To analyze the behavior of (7.6.1), assume that A has a basis of eigenvectors $\{x_1, \dots, x_n\}$ and that $A \vec{x}_i = \lambda_i \vec{x}_i$ for $i = 1:n$. If

$$q() = \sum_{i=1}^n c_i \vec{x}_i$$

then $q^{(k)}$ is a unit vector in the direction of

$$(A - \mu I)^{-k} q^{(0)} = \sum_{i=1}^n \frac{c_i}{(\lambda_i - \mu)^k} \vec{x}_i$$

Clearly, if μ is much closer to an eigenvalue λ_j than to the other eigenvalues, then $q^{(k)}$ is rich in the direction of \vec{x}_j provided $j = Q$.

A sample stopping criterion for (7.6.1) might be to quit as soon as the residual

$$r^{(k)} = (A - \mu I)q^{(k)}$$

satisfies

$$\|r^{(k)}\|_\infty \leq c u \|A\|_\infty \tag{7.6.2}$$

where c is a constant of order unity. Since

$$(A + E_k)q^{(k)} = \mu q^{(k)}$$

with $E_k = -r^{(k)}q^{(k)}T$, it follows that (7.62) forces μ and $q^{(k)}$ to be an exact eigenpair of a nearby matrix.

Inverse iteration can be used in conjunction with Hessenberg reduction and the QR algorithm as follows:

Step 1. Compute the Hessenberg decomposition $U^{-1}AU = H$.

Step 2. Apply the double implicit-shift Francis iteration to H without accumulating transformations.

Step 3. For each computed eigenvalue, whose corresponding eigenvector x is sought, apply (7.61) with $A = H$ and $\mu = \lambda$, to produce a vector z such that $Hz = \lambda z$.

Step 4. Set $x = Uz$.

Inverse iteration with H is very economical because we do not have to accumulate transformations during the double Francis iteration. Moreover, we can factor matrices of the form $H - \lambda I$ in $O(n^2)$ fops, and (3) only one iteration is typically required to produce an adequate approximate eigenvector.

This last point is perhaps the most interesting aspect of inverse iteration and requires some justification since it can be comparatively inaccurate if it is ill-conditioned. Assume for simplicity that λ is real and let

$$H - \lambda I = \sum_{i=1}^n \sigma_i u_i v_i^T = U \Sigma V^T$$

be the SVD of $H - \lambda I$. From what we said about the roundoff properties of the QR algorithm in §7.56 there exists a matrix $E \in \mathbb{R}^{n \times n}$ such that $H + E - \lambda I$ is singular and $\|E\|_2 \approx \|H\|_2$. It follows that $(n - \lambda) \sigma_1$ and

$$\|(H - \lambda I)v_n\|_2 \approx \lambda \sigma_1,$$

i.e., v_n is a good approximate eigenvector. Clearly if the starting vector $q^{(0)}$ has the expansion

$$q^{(0)} = \sum_{i=1}^n \gamma_i u_i$$

then

$$z^{(1)} = \sum_{i=1}^n \frac{\gamma_i}{\sigma_i} v_i$$

is "rich" in the direction v_n . Note that if $s(\gamma_i / \sigma_i; v_n)$ is small, then $z^{(1)}$ is rather deficient in the direction u_n . This explains (heuristically) why another step of inverse iteration is not likely to produce an improved eigenvector approximation, especially if H is ill-conditioned. For more details, see Peters and Wilkinson (1979).

nmmb S7LoGfA 6Gf d 2NcAGA - oA.d2A Li- N=As7A

Recall that the real Schur decomposition provides information about invariant subspaces. If

$$Q^T A Q = T = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix},$$

and

$$\lambda(T_{11}) \cap \lambda(T_{22}) = \emptyset,$$

then the first p columns of Q span the unique invariant subspace associated with $\lambda(T_{11})$. (See §7.1.4) Unfortunately, the Francis iteration supplies us with a real Schur decomposition $Q_F^T A Q_F = T_F$ in which the eigenvalues appear somewhat randomly along the diagonal of T_F . This poses a problem if we want an orthonormal basis for an invariant subspace whose associated eigenvalues are not at the top of T_F 's diagonal. Clearly, we need a method for computing an orthogonal matrix Q_D such that $Q_D^T T_F Q_D$ is upper quasi-triangular with appropriate eigenvalue ordering.

A look at the 2by-2 case suggests how this can be accomplished. Suppose

$$Q_F^T A Q_F = T_F = \begin{bmatrix} \lambda_1 & t_{12} \\ 0 & \lambda_2 \end{bmatrix}, \quad \lambda_1 \neq \lambda_2$$

and that we wish to reverse the order of the eigenvalues. Note that

$$T_F x = \lambda_2 x$$

where

$$x = \begin{bmatrix} t_{12} \\ \lambda_2 - \lambda_1 \end{bmatrix}.$$

Let Q_D be a Givens rotation such that the second component of $Q_D^T x$ is zero. If

$$Q = Q_F Q_D,$$

then

$$(Q^T A Q) e_1 = Q_D^T T_F (Q_D e_1) = \lambda_2 Q_D^T (Q_D e_1) = \lambda_2 e_1.$$

The matrices A and $Q^T A Q$ have the same Frobenius norm and so it follows that the latter must have the following form

$$Q^T A Q = \begin{bmatrix} \lambda_2 & \pm t_{12} \\ 0 & \lambda_1 \end{bmatrix}.$$

The swapping gets a little more complicated if T has 2by-2 blocks along its diagonal. See Ruge (1970) and Stewart (1976) for details.

By systematically interchanging adjacent pairs of eigenvalues (or 2by-2 blocks), we can move any subset of $\lambda(A)$ to the top of T 's diagonal. Here is the overall procedure for the case when there are no 2by-2 bumps:

~~p 21shdi-~~ 7.6.1 Given an orthogonal matrix $Q \in \mathbb{R}^{n \times n}$, an upper triangular matrix $\mathbf{I} = Q^T A Q$, and a subset $\Delta = \{\lambda_1, \dots, \lambda_p\}$ of $\lambda(A)$, the following algorithm computes an orthogonal matrix Q_D such that $Q_D^T T Q_D = S$ is upper triangular and $\{s_{11}, \dots, s_{pp}\} = \Delta$. The matrices Q and T are overwritten by QQ_D and S , respectively.

```

while  $\{t_{11}, \dots, t_{pp}\} \neq \Delta$ 
  f j k= 1:n - 1
    if  $t_{kk} \notin \Delta$  and  $t_{k+1,k+1} \in \Delta$ 
      [ c, s ] = givens( $T(k:k+1), T(k+1, k+1) - T(k, k)$ )
       $T(k:k+1, k:n) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} T(k:k+1, k:n)$ 
       $T(1:k+1, k:k+1) = T(1:k+1, k:k+1) \begin{bmatrix} \bullet & \\ \bullet & \end{bmatrix}$ 
       $Q(1:n, k:k+1) = Q(1:n, k:k+1) \begin{bmatrix} \{ & - \} \\ \{ & - \} \end{bmatrix}$ 
    end
  end
end

```

This algorithm requires $k(12n)$ fops, where k is the total number of required swaps. The integer k is never greater than $(n-p)p$.

Computation of invariant subspaces by manipulating the real Schur decomposition is extremely stable. If $\hat{Q} = [\hat{q}_1 | \cdots | \hat{q}_q]$ denotes the computed orthogonal matrix Q then $\|Q^T \hat{Q} - I\|_F \leq u$ and there exists a matrix E satisfying $\|E\|_F \leq u \|A\|_F$ such that $(A + E)\hat{q}_i \in \text{span}\{\hat{q}_1, \dots, \hat{q}_p\}$ for $i = 1:p$.

$\cdot \cdot \cdot q$

Let

$$T = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1q} & n_1 \\ 0 & T_{22} & \cdots & T_{2q} & n_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & T_{qq} & n_q \\ n_1 & n_2 & \cdots & n_q & \end{bmatrix} \quad (7.63)$$

be a partitioning of some real Schur canonical form $Q^T A Q = T \in \mathbb{R}^{n \times n}$ such that $A(T_{11}), \dots, A(T_{qq})$ are disjoint. By Theorem 7.16 there exists a matrix Y such that

$$Y^{-1} Y Y = \text{diag}(T_{11}, \dots, T_{qq}).$$

A practical procedure for determining Y is now given together with an analysis of Y 's sensitivity. A notion of the above partitioning

Partition $I_n = [E_1 | \cdots | E_q]$ compatibly with T and define the matrix $Y \in \mathbb{R}^{n \times n}$ as follows

$$Y_{ij} = I_n + E_i Z_{ij} E_j^T, \quad i < j, \quad Z_{ij} \in \mathbb{R}^{n_i \times n_j}.$$

In other words \mathbf{Y}_j looks just like the identity except that Z_{ii} occupies the (i,j) block position. It follows that if $\mathbf{j}^T \mathbf{T} \mathbf{Y}_j = \mathbf{t} = (\mathbf{T}_j)$, then \mathbf{T} and \mathbf{t} are identical except that

$$\begin{aligned} \mathbf{f}_j &= \mathbf{T}_{jj} \mathbf{Z}_{jj} - \mathbf{Z}_{jj} \mathbf{T}_j + \mathbf{T}_j, \\ \mathbf{t}_{ik} &= \mathbf{T}_{ik} - \mathbf{Z}_{ik} \mathbf{T}_k \quad (k=j+1:p), \\ \mathbf{T}_k &= \mathbf{T}_{kk} z_{kk} + \mathbf{T}_k, \quad (k=1:i-1). \end{aligned}$$

Thus, \mathbf{T}_j can be zeroed provided we have an algorithm for solving the Sylvester equation

$$\mathbf{FZ} - \mathbf{ZG} = \mathbf{C} \quad (7.64)$$

where $\mathbf{F}, \mathbf{E}, \mathbf{R}, \mathbf{x}^p$ and $\mathbf{G}, \mathbf{E}, \mathbf{R}, \mathbf{x}^r$ are given upper quasi-triangular matrices and $\mathbf{C}, \mathbf{E}, \mathbf{W}, \mathbf{x}^q$.

Bartels and Stewart (1972) have devised a method for doing this. Let $\mathbf{C} = [c_1 \ I \cdots I_c]$ and $\mathbf{Z} = [z_1 \ I \cdots I_{k-1} \ Z_k]$ be column partitions. If $g_{ki,k} = 0$ then by comparing columns in (7.64) we find

$$F_{z_k} - \sum_{i=1}^{k-1} g_{ik} Z_i = c_k$$

Thus, once we know z_1, \dots, Z_{k-1} , then we can solve the quasi-triangular system

$$(\mathbf{F} - g_{kk} \mathbf{I}) \mathbf{Z}_k = \mathbf{C}_k + \sum_{i=1}^{k-1} g_{ik} \mathbf{Z}_i$$

for Z_k . If $g_{kk}, k = 0$ then Z_k and Z_{k+1} can be simultaneously found by solving the $2p$ by $2p$ system

$$\begin{bmatrix} F - g_{kk} I & -g_{mk} I \\ -g_{km} I & F - g_{mm} I \end{bmatrix} \begin{bmatrix} z_k \\ z_m \end{bmatrix} = \begin{bmatrix} c_k \\ c_m \end{bmatrix} + \sum_{i=1}^{k-1} \begin{bmatrix} g_{ik} z_i \\ g_{im} z_i \end{bmatrix} \quad (7.65)$$

where $a = k+1$. By reordering the equations according to the perfect shuffle permutation $(1, p+1, 2p+2, \dots, p, 2p)$, a banded system is obtained that can be solved in $O(p^2)$ fops. The details may be found in Bartels and Stewart (1972). Here is the overall process for the case when \mathbf{F} and \mathbf{G} are each triangular.

m2F.5i- 7.6.2 enT.9a.nc.9!T.p ha.6(4. 19rp Given $\mathbf{C}, \mathbf{E}, \mathbf{W}, \mathbf{x}^q$ and upper triangular matrices $\mathbf{F}, \mathbf{E}, \mathbf{R}, \mathbf{x}^p$ and $\mathbf{G}, \mathbf{E}, \mathbf{R}, \mathbf{x}^r$ that satisfy $A(\mathbf{F})nA(\mathbf{G}) = N$ the following algorithm overwrites \mathbf{C} with the solution to the equation $\mathbf{FZ} - \mathbf{ZG} = \mathbf{C}$.

for $k = 1:r$

$$C(l:p,k) = C(l:p,k) + C(l:p,l:k-1) \cdot G(l:k-1,k)$$

$$\text{Solve } (\mathbf{F} - g_{kk} \mathbf{I}) \mathbf{z} = C(l:p,k) \text{ for } \mathbf{z}$$

$$C(l:p,k) = \mathbf{z}$$

end

This algorithm requires $p(p+r)$ fops. By zeroing the superdiagonal blocks in T in the appropriate order, the entire matrix can be reduced to block diagonal form

p 21sufi- 7.6.3 Given an orthogonal matrix $Q \in \mathbb{R}^{n \times n}$, an upper quasi-triangular matrix $T = QT\bar{A}Q$ and the partitioning (7.63), the following algorithm overwrites Q with QY where $Y = \text{diag}(T_{11}, \dots, T_{qq})$.

f r j = 2q

f r i = 1j - 1

Solve $T_{ijz} - ZT_{11} = -T_{ij}$ f r Z using the Bartels-Stewart algorithm

f r k = j + 1:q

$T_{ik} = T_{ik} - ZT_{1k}$

end

f r k = 1:q

$Q_{ik} = Q_{ikz} + Q_{ik}$

end

end

end

The number of fops required by this algorithm is a complicated function of the block sizes in (7.63).

The choice of the real Schur form T and its partitioning in (7.63) determines the sensitivity of the Sylvester equations that must be solved in Algorithm 7.63. This in turn affects the condition of the matrix Y and the overall usefulness of the block diagonalization. The reason for these dependencies is that the relative error of the computed solution Z to

$$T_{ii}Z - ZT_{jj} = -T_{ij} \quad (7.66)$$

satisfies

$$\frac{\|Z\|_F}{\|Z\|} \approx u \frac{\|T\|_F}{\text{sep}(T_{ii}, T_{jj})}$$

For details, see Golub, Nash, and Van Loan (1979). Since

$$\text{sep}(T_{ii}, T_{jj}) = \min_{X \neq 0} \frac{\|T_{ii}X - XT_{jj}\|_F}{\|X\|} \leq \min_{\substack{\lambda \in \lambda(T_{ii}) \\ \mu \in \lambda(T_{jj})}} |1 - \mu|$$

there can be a substantial loss of accuracy whenever the subsets $\{T_{ii}\}$ are insufficiently separated. Moreover, if Z satisfies (7.66) then

$$\|Z\|_F = \frac{\|T_{ij}\|_F}{\text{sep}(T_{ii}, T_{jj})}.$$

Thus large norm solutions can be expected if $\text{sep}(T_{ii}, T_{jj})$ is small. This tends to make the matrix Y in Algorithm 7.63 ill-conditioned since it is the product of the matrices

$$I = \begin{bmatrix} I_n & Z \\ 0 & I_{n1} \end{bmatrix}.$$

Note that $\kappa_F(Y_{ij}) = n_i^2 + n_j^2 + \|Z\|_F^2$.

Concerned with these difficulties, Bavey and Stewart (1979) develop an algorithm for block diagonalizing that dynamically determines the eigenvalue ordering and partitioning in (7.63) so that all the Z matrices in Algorithm 7.63 are bounded in norm by some user-supplied tolerance. Their research suggests that the condition of Y can be controlled by controlling the condition of the Y_j .

$$c \cdot Y \cdot T_p = \hat{A} \Rightarrow Y \hat{U}_j \hat{O}_j \hat{U}_j^T \hat{Y} = P$$

If the blocks in the partitioning (7.63) are all 1-by-1, then Algorithm 7.63 produces a basis of eigenvectors. As with the method of inverse iteration, the computed eigenvalue-eigenvector pairs are exact for some "nearby" matrix. A widely followed rule of thumb for deciding upon a suitable eigenvector method is to use inverse iteration whenever fewer than 25% of the eigenvectors are desired.

We point out, however, that the real Schur form can be used to determine selected eigenvectors. Suppose

$$Q^T A Q = \begin{bmatrix} T_{11} & u & T_{13} \\ 0 & > & v^T \\ 0 & 0 & T_{33} \end{bmatrix} \quad \begin{matrix} k-1 \\ 1 \\ n-k \end{matrix}$$

$$\begin{matrix} k-1 & 1 & n-k \end{matrix}$$

is upper quasi-triangular and that $> (> (T_{11})_U > (T_{33})_L$. It follows that if we solve the linear systems $(T_{11} - > I)w = -u$ and $(T_{33} - > J)T_{Z_L} = -v$ then

$$x = Q \begin{bmatrix} w \\ 1 \\ 0 \end{bmatrix} \quad \text{and} \quad y = Q \begin{bmatrix} 0 \\ 1 \\ z \end{bmatrix}$$

are the associated right and left eigenvectors, respectively. Note that the condition of $>$ is prescribed by

$$1/s(>) \leq \sqrt{(1 + w^T w)(1 + z^T z)}.$$

8lj lwr 18F so zrzrxr mE, orr - .w8+ r - sF- 8s F ilr

Suppose that we have computed the real Schur decomposition $A = QTQ^T$, identified clusters of "equal" eigenvalues, and calculated the corresponding block diagonalization $T = \hat{T}e$

greater. A concrete example helps to make this a section dear and to illustrate the role of the SVD in Jordan form computations.

Assume that c is 7 by 7. Suppose we compute the SVD $U \Sigma V^T = \Sigma_1$ and "discover" that N has rank 3. If we order the singular values $\sigma_1, \sigma_2, \sigma_3$ from largest to smallest, then Σ_1 is

$$\Sigma_1 = \begin{bmatrix} 0 & K \\ 0 & L \\ 4 & 3 \end{bmatrix}_3^4 .$$

$$= \begin{bmatrix} & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \\ & & & & & & \end{bmatrix} .$$

$$=$$

the decomposition

$$V^T C V = \begin{bmatrix} . & 0 & 0 & 0 & \times & \times & \times \\ 0 & . & 0 & 0 & \times & \times & \times \\ 0 & 0 & . & 0 & \times & \times & \times \\ 0 & 0 & 0 & . & \times & \times & \times \\ 0 & 0 & 0 & 0 & . & \times & ^a \\ 0 & 0 & 0 & 0 & 0 & . & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & . \end{bmatrix} \quad \left. \begin{array}{l} \text{four blocks of order 1 or 2,} \\ \text{two blocks of order 2 or larger} \\ \text{one block of order 3 or larger} \end{array} \right\}$$

displays C's Jordan block structure: two blocks of order 1, one block of order 2, and one block of order 3.

To compute the Jordan decomposition it is necessary to resort to nonorthogonal transformations. We refer the reader to Golub and Wilkinson (1976), Kagstrom and Ruhe (1980a, 1980b), and Demmel (1989) for more details. The above calculations with the SYD amply illustrate that difficult rank decisions must be made at each stage and that the final computed block structure depends critically on those decisions.

Problems

P7.6.1 $\mathbf{N}.\mathbf{N}.\mathbf{18.es} \mathbf{N}.\mathbf{Ms.1...8 N.1es....N.6s} \mathbf{..11..86 ri} \mathbf{.1,...,es..} \mathbf{..N8} \quad b.$
 P7.6.2 $\dots \mathbf{1.N} U^{-1} AU +, -U \mathbf{1\dots, \alpha_m}, \beta V^{-1} BV +, -U \mathbf{n\dots, \beta_n}. \quad R^{m \times l} \mathbf{0, 0^T} \mathbf{T}$

$$\phi(X) = AX - XB,$$

(7)

$$\lambda(\phi) = \alpha_i - \text{diag}(1:m, j) \quad \{.$$

$$v(\nabla^r - \nabla_{x,t}^T (\kappa_1 + \kappa_2 t) \cdot \nabla_{x,t}^T) = \kappa_1^T + \kappa_2 t \cdot \nabla_{x,t}^T c +, = [(\nabla_{x,t}^T)^T c_1 s_{x,t} \nabla_{x,t}^T, -c_1 s_{x,t} \nabla_{x,t}^T]^T \quad AX - XB$$

P7.6.3 $.1f\mathbf{q.} \quad \geq \in \mathbb{C}^{p \times q} (0)$

$$Y = \begin{bmatrix} & & \\ - & : & \end{bmatrix},$$

$$u^R + \sqrt{4\sigma^2 + \sigma^4} \quad \mathbf{V899} \quad U \quad \mathbf{RA}$$

P7.6.4 $\mathbf{8N}.\mathbf{N} \quad .N \quad ..N\mathbf{8N}.\mathbf{N} \quad 5$

P7.6.5 $1..8N \quad \dots \quad T \in \mathbb{R}^{n \times n}) - \mathbf{CR}^n, \quad \gamma, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{C} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n(\gamma) - \frac{1}{2} \mathbf{CCR}^n -$

$$T = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ 0 & T_{22} & T_{23} \\ 0 & 0 & T_{33} \end{bmatrix}, \quad T \in \mathbb{R}^{n \times n}.$$

$$\begin{aligned} & \left(\begin{pmatrix} V_3^T & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n((\mathbf{C} \cdot \mathbf{CR}^n, T_{22})) - \right. \\ & \left. \lambda(T_{33}), y \right) V_3 \left(\begin{pmatrix} (\mathbf{C} \cdot \mathbf{CR}^n)^T & 0 \\ 0 & 1 \end{pmatrix} \mathbf{R}^n, \begin{pmatrix} T & 0 \\ 0 & V_3 \end{pmatrix} \right) \left(\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n((\mathbf{C} \cdot \mathbf{C})(\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n(\gamma) - \frac{1}{2} \mathbf{CCR}^n)) - \right. \\ & \left. \gamma V_3 ((\mathbf{C} \cdot \mathbf{V}_3)^T - \right) \end{aligned}$$

$$\begin{aligned} & \mathbf{P7.6.6} \quad \mathbf{I.18 H} \in \mathbb{R}^{n \times n}) - \quad V_3 (dV_3 - \gamma(V_3, \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n, \mathbf{C} \cdot \mathbf{V}_3), V_3 ((\mathbf{C} \cdot \mathbf{V}_3)^T - \mu \cdot \mathbf{ECp} \mathbf{A} \otimes \mathbf{E}) \\ & (= \mathbf{E} \mathbf{a} \mathbf{r} \mathbf{G} \mathbf{p} \mathbf{F} \mathbf{C} \mathbf{T} \mathbf{f} \mathbf{p} \mathbf{m} \in \mathbb{R}^n - \mathbf{R}_0^T \begin{pmatrix} 0 & 0 \\ B & B \end{pmatrix} \mathbf{dR}_0^T \lambda \mathbf{R}^n, V_3 (\mathbf{C} \cdot \mathbf{V}_3)^T - \mu \cdot \mathbf{ECp} \mathbf{A} \otimes \mathbf{E}) \\ & \left. \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n \right) \left(\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n \right) \left(\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n \right) \left(\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n \right) \left(\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n \right) \left(\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \mathbf{R}^n \right) \right) \end{aligned}$$

Notes and References for §7.6

R... 1... N8.N...es.....N.N..1. 8.. .N NI... .8 Neses1fa.N. ...N.or

N **Res... fRes .1 f..ro002reeses .131... ..N....88. .. .N 318.I..1. 82 N**
 1... 3.1...es **B67SIAM Review 18 ,dr N.x**

ap2Su,SpAr ,ueAucapm2crAal plApm2Am,ur c,pm2S)AAp ..T

- A. T. 10N 9220Dy yry $\Theta \in \text{eHn}$. OE (E, O, r) OET (= α , r) $O = E \cdot E^{-1} \cdot O = O \cdot A \cdot A^{-1}$ (da BIT 10 N 7x, 7x)
- sxAiupf ((7x .30rlucxmJ 7e.VI, i.e. 9 A.2.e lupuias, Sul, pnaAti AipAUOpCar iae, ueAucapmOcrAaiiOUA, i IAIOr \$AiAt, AaSAunipunFACM T ans. Math. Sof w. 2, I, (II)dJx)
- Elariuuisxir r iuOriae'x.xacO+ da f., I.x.2,r AunAlatceAuiplatAlr O,xma faiuinias, StSiAA, tycAM J. Matrix Anal. Applic. 13, „IN 7NM
- Waa (=ELPOTT ON 9220D) Di=O3a ff (Oy y)=A a E8j Oa OE • 2-HET a in N Alg. Applic. 186 (l.,
- 1dAAAt,u SDAecirlai0c.ipclnaA0,er,pmAueia,uP iuA eAtAucSAe cae
- x iiAptiae xax spAixuf N.. x.3a 3prlucpmu Alr O,xchAe,Acars,StSiAASt wD+ EnirlaipqinlayFAM J. Numer. Anal. 16, „1, 7x
- xr,puCrae 3x .Uf Aljix.3a 30lucppru 2,r AunAlp O,xipdlappAueia 2lur i0 lur l. i Alr OOAhpufbyEM T ans. Math. Sof w. 6 „dv, N.M
- rtxiae 3-, Af. NJS. - 3plucpmPJ '2.e 3a 3prlupmu 2, r AunAlp SUPipcla 1. xpAueia 2lur iur l. i Alr SOAbxuchbACM T ns. Math. Sof w. 6 „(l, „-
- f Ar rf Ap .3 2Ur Ai8apttF AueiaAialacAiDur h6nHmAc,ywAu+AOAtx
- N. le+= • LPLn3a yQE aX00a E.NZp pD. • Qff AE == (y=0 a AE nO) r= E8 ,Oa y= r= Aa N. Numer. Anal. 34 NJN1 7dx
- s-AusASc.italy E xwAupAraiae x.-.lQS f JJ.x ..l. pl EAeAA hulSAu 9crAaiiOUA ,UpAlr i hulOstarUpiBiOUAp, pAcpmAalaupi, ACAYsSAM J. Matr. Anal. Applic. 27, d1d7x
- AuAl5AUSlnapApotpma OcpGulcipAcpnraiuniapStOIAAlr S,pilayA uAr nae r pA uApia 7-, A entA,tApmlAlAu PAXmlAlPO,xnapmAlpciaiAnrAaSiAxmA Ar pleluxmlrlaippAuixpathiAiaSAU,AaplAlr SUhA naiapaiiciapS,SiAAAtfaiauA 2r AuixclauA,ixAeAiae n,pmAlaAAlapmopl.narSiSAu.e
- 'x Biuinf .7dM.lpA AiQAUpixblxpA9crAaiAAxlti AaAuiO AlP OPAAlSt f aiAutA f pApilayMath. Comput. 22, (d1, „-
- 'x Biuinf ,J. M. AIPS, pnaif aiiuciaxStSiAA. i AaAuiQxiulcamAapmOcrAattttApct hluuOTExAaAeMath. Comput. 24, (, TN,x
- g. x OrOE o= O Il(n8) = N. 2 3 rDx= > QE OE(xA9 e= 0-0(r-0O) - aa= 1003+ a= AO= ea SIM Review 21, „v, 7x
- Af.O,Af N. x.Alr SUPcar9crAairAxhpnfaiAufApilaySAM Review 39 I, vI, ,x
- AAuxitSOcAipda, AAAt,iplpu+i a caiuciatus,OiAApmRipunkAmiarAAe
- n-EnAAe I-'x r cAePiif IJ(x Alapca,ipcd. faiuinias, StSiAA, tycum Lin. Alg 8, e,(I,x)
- O naeApYEA r iaAD x r cAer fAlH.x.Alapna,ipcd. faiuciastSiAAAtaniurA nr uAlpApS, SIAM J. Sci. Comput. 30, 7,(7x
- liOAUAlaAAuaApma,xnr ippmA AuulAlr cSUAEAAipiAnlu AcrA iAAppl,eAe
- s-kApiaiae w-2hiu0AfpT(-.3prlucxpr(e kluuir, u Alr S,pcapAlaexclalUr SAut Iiuxcb9crAaiiOUAcpml,pAlPS,pn,r9crAaiApuy6ACM T ns. Math. Sof w. 3, d7IJ,x
- H.J. Oe.T a = O 3(r8 (= + N 9220D) Dj Oa rOoE E Oe a O(TfrO (y0= >aa = A0 c0 yoo= E0a (y0= E0a) N. Numer. Math. 35, N 1, 1x
- Bia nlia f ,d(x, „a 9pcr ippmAalaexclal 9crAaiip,Aiae 9crAaiAxlyt6Lin. Alg Applic. ddhd.(, 1(I x)
- z. Wa O2TTOAa=yHET0-ON.2 nDi=e TD- = 0= y n-e-nORe(r0= = + eTC TO= EOy O= f Ea ATN.T ns. Math. Sof w. 19, IJIII,x
- slPAnAGSl,p cr OulicAlr S,pAerAaiiOAAyAaiAApiai caiuinias, StSiAAit SA.Uaena
- f Biuipf .7dMlnrlulU, n- mawluuetMpmOcrAa,txAl. i AaAuiAlr O,Abpufb6 Math. Comp. 22, (,ld N x
- f -'OlariuuiAxwApAye'x.Mcp+ natfa,dex,fr Oulina,AAA,u-tl. Alr S,xAerAa2 iip,Atiae 9nrAaApuy6SIAM J. Numer. Anal. 20, „l,7x

BEB enex ly, ZEBhCAnesAtg kn kaa+ lgsim seg12latm CagqngtGGComput 38
i,-, (M

3. A miiAAAapmaAp(.,.) 50,(+(-,- H)(,D1(((),," +((," -1--D"(6),'D10("+(, ()(
-0,-1),"-1",(- ,"- 5 (,-(γ)(+((g -',")("+-γ T."-(g(γ e)(,')(" ++-,0--";4n

b1S)0T U..w.11 ⊙α((T" "-1)((' 'F il' B)((,"- Y2SIAM J. Numer. Anal. 16 INHIM
l, wtAM, Ni.M3 ncASA+r spAtAcpcda,pcr ipMpms,ipclax AX - XB^T = C," IEEE
T ns. Autom. Contr. AC-29, I7-Id,
n, .. iae nMniAMplf JJ7.MBa 3pr lMcplAlr ShpASj6SIAM J. Matrix Anal. Si al.

b f MjAl

7.7 The Generalized Eigenvalue Problem

If $A, B \in \mathbb{R}^{n \times n}$, then the set of all matrices of the form $A - zB$ with $z \in \mathbb{C}$ is a pencil. The generalized eigenvalues of $A - zB$ are elements of the set $\sigma(A, B)$ defined by

$$\sigma(A, B) = \{z \in \mathbb{C} : \det(A - zB) = 0\}.$$

$z \in \sigma(A, B)$ and $0 \neq x \in \mathbb{R}^n$ satisfies

$$Ax = zBx, \quad (7.7.1)$$

then x is an eigenvector of $A - zB$. The problem of finding nontrivial solutions to (7.7.1) is the generalized eigenvalue problem and in this section we survey some of its mathematical properties and derive a stable method for its solution. We briefly discuss how a polynomial eigenvalue problem can be converted into an equivalent generalized eigenvalue problem through a linearization process.

mmhhb hfdb ". kDtb

The first thing to observe about the generalized eigenvalue problem is that there are n eigenvalues if and only if $\text{rank}(B) = n$. If B is rank deficient then $\sigma(A, B)$ may be finite, empty, or infinite.

$$A = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad \Rightarrow \quad \sigma(A, B) = \{1\},$$

$$A = \begin{bmatrix} 1 & 2 \\ 0 & 3 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad \Rightarrow \quad \sigma(A, B) = \mathbb{C}$$

$$A = \begin{bmatrix} 1 & 2 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad \Rightarrow \quad \sigma(A, B) = \mathbb{C}.$$

Note that if $0 \neq z \in \sigma(A, B)$, then $(1/z) \in \sigma(B, A)$. Moreover, if B is nonsingular, then $\sigma(A, B) = \sigma(B^{-1}A, I) = \sigma(B^{-1}A)$. This last observation suggests one method for solving the $A - zB$ problem if B is nonsingular.

Step 1. Solve $BC = A$ for C using (say) Gaussian elimination with pivoting

Step 2. Use the QR algorithm to compute the eigenvalues of C .

In this framework, C is affected by roundoff errors of order $\|A\|_1 \|B\|^{-1}$. If B is ill-conditioned, then this precludes the possibility of computing any generalized eigenvalue accurately, even those eigenvalues that may be regarded as well-conditioned. For example, if

$$A = \begin{bmatrix} 1.746 & .940 \\ 1.246 & 1.898 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} .780 & .563 \\ .913 & .659 \end{bmatrix},$$

then $A(A, B) \in \{2, 107 \times 10^6\}$. With 7-digit floating point arithmetic, we find $A^{-1}(AB^{-1}) = \{1.562539, 101 \times 10^6\}$. The poor quality of the small eigenvalue is because $2B \approx 2 \times 10^6$. On the other hand, we find that

$$A^{-1}f(A^{-1}B) \approx \{2000001, 106 \times 10^6\}.$$

The accuracy of the small eigenvalue is improved because $\|A\|_F \approx 4$.

The example suggests that we seek an alternative approach to the generalized eigenvalue problem. One idea is to compute well-conditioned Q and Z such that the matrices

$$A_1 = Q^{-1}AZ, \quad B_1 = Q^{-1}BZ \quad (7.7.2)$$

are each in canonical form. Note that $A(A, B) \approx A(A_1, B_1)$ since

$$Ax = \lambda Bx \Leftrightarrow A_1y = \lambda B_1y, \quad x = Zy.$$

We say that the pencils $A - AB$ and $A_1 - AB_1$ are equivalent if (7.7.2) holds with nonsingular Q and Z .

As in the standard eigenproblem $A - A$, there is a choice between canonical forms. Corresponding to the Jordan form is a decomposition of Kronecker in which both A_1 and B_1 are block diagonal with blocks that are similar in structure to Jordan blocks. The Kronecker canonical form poses the same numerical challenges as the Jordan form, but it provides insight into the mathematical properties of the pencil $A - AB$. See Wilkinson (1978) and Demmel and Kagstrom (1987) for details.

On the numerical point of view, it makes sense to insist that the transformation matrices Q and Z be unitary. This leads to the following decomposition described in Moler and Stewart (1973).

Theorem 7.7.1 (Generalized Schur Decomposition). If A and B are in $\mathbb{C}^{n \times n}$ then there exist unitary Q and Z such that $Q^{-1}AZ$ is T and $Q^{-1}BZ$ is S are upper triangular. If for some k , t_{kk} and s_{kk} are both zero then $A(A, B) = \infty$. Otherwise

$$A(A, B) = \{t_{kk}s_{ii} : S_{ii} \neq 0\}.$$

Proof Let $\{B_k\}$ be a sequence of nonsingular matrices that converge to B . For each k let

$$Q^{-1}(AB^{-1})Q_k = R_k$$

be a Schur decomposition of AB^{-1} . Let Z_k be unitary such that

$$Z_k(B_k^{-1}Q_k) = I$$

is upper triangular. It follows that $Q_k AZ_k = R_k S_k$ and $Q_k B_k Z_k = S_k$ are also upper triangular. Using the Bolzano-Weierstrass theorem, we know that the bounded sequence $\{(Q_k, Z_k)\}$ has a converging subsequence

$$\lim_{i \rightarrow \infty} (Q_{k_i}, Z_{k_i}) = (Q, Z).$$

It is easy to show that Q and Z are unitary and that $QH_A Z$ and $QH_B Z$ are upper triangular. The assertions about $\lambda(A, B)$ follow from the identity

$$\det(A - \lambda B) = \det(QZH) \prod_{i=1}^n (t_{ii} - \lambda s_{ii})$$

and that completes the proof of the theorem. \square

If A and B are real then the following decomposition, which corresponds to the real Schur decomposition (Theorem 7.4.1), is of interest.

Theorem 7.7.2 (Generalized Real Schur Decomposition). *If A and B are in $\mathbb{R}^{n \times n}$ then there exist orthogonal matrices Q and Z such that $QTAZ$ is upper quasi-triangular and $QTBZ$ is upper triangular.*

P. f. See Stewart (1972). \square

In the remainder of this section we are concerned with the computation of this decomposition and the mathematical insight that it provides.

.m0mb useTf Tf flKlfeDSeb

The generalized Schur decomposition sheds light on the issue of eigenvalue sensitivity for the $A - \lambda B$ problem. Clearly, small changes in A and B can induce large changes in the eigenvalue $\lambda_i \neq t_{ii}/s_{ii}$ if s_{ii} is small. However, as Stewart (1978) argues, it may not be appropriate to regard such an eigenvalue as "ill-conditioned." The reason is that the reciprocal $\mu_i = s_{ii}/t_{ii}$ might be a very well-behaved eigenvalue for the pencil $\mu A - B$. In the Stewart analysis, A and B are treated symmetrically and the eigenvalues are regarded more as ordered pairs (t_{ii}, s_{ii}) than as quotients. With this point of view it becomes appropriate to make use of eigenvalue perturbations in the chordal metric $d(a, b)$ defined by

$$d(a, b) = \frac{\|a - b\|}{\sqrt{1 + a^2}\sqrt{1 + b^2}}.$$

Stewart shows that if λ is a distinct eigenvalue of $A - \lambda B$ and Λ_1 is the corresponding eigenvalue of the perturbed pencil $A - \lambda B$ with $\|A - \Lambda_1 B\| \leq \|B\| E$, then

$$d(\lambda, \Lambda_1) \leq \frac{E}{\sqrt{(y^H Ax)^2 + (y^H Bx)^2}} + O(\sqrt{\epsilon})$$

where y and x have unit 2-norm and satisfy $Ax = \lambda Bx$ and $y^H A = \lambda y^H B$. Note that the denominator in the upper bound is symmetric in A and B . The "truly" ill-conditioned eigenvalues are those for which this denominator is small.

The extreme case when both t_{kk} and s_{kk} are zero for some k has been studied by Wilkinson (1979). In this case, the remaining quotients t_{ii}/s_{ii} can take on arbitrary values.

$$A: A = \begin{bmatrix} & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}, \quad f: f = \begin{bmatrix} & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

The first step in computing the generalized real Schur decomposition of the pair (A, B) is to reduce A to upper Hessenberg form and B to upper triangular form via orthogonal transformations. We first determine an orthogonal U such that $U^T B$ is upper triangular. Of course, to preserve eigenvalues, we must also update A in exactly the same way. Let us trace what happens in the $n = 5$ case.

$$A \leftarrow U^T A = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix}, \quad B \leftarrow U^T B = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{bmatrix}.$$

Next, we reduce A to upper Hessenberg form while preserving B 's upper triangular form. First, a Givens rotation Q_{45} is determined to zero a_{51} :

$$A \leftarrow Q_{45}^T A = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \end{bmatrix}, \quad B \leftarrow Q_{45}^T B = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix}.$$

The nonzero entry arising in the $\{5,4\}$ position in B can be zeroed by postmultiplying with an appropriate Givens rotation Z_{45} :

$$A \leftarrow AZ_{45} = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \end{bmatrix}, \quad B \leftarrow BZ_{45} = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{bmatrix}.$$

Zeros are similarly introduced into the $(4, 1)$ and $(3, 1)$ positions in A :

$$A \leftarrow Q_{34}^T A = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{bmatrix}, \quad B \leftarrow Q_{34}^T B = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{bmatrix},$$

$$A \leftarrow AZ_{34} = \begin{bmatrix} C & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix}, \quad B \leftarrow BZ_{34} = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{bmatrix},$$

$$A \leftarrow Q_{23}^T A = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{bmatrix}, \quad B \leftarrow BQ_{23}^T B = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{bmatrix},$$

$$A \leftarrow AZ_{23} = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{bmatrix}, \quad B \leftarrow BZ_{23} = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{bmatrix}.$$

A is now tridiagonal and its third column is zero. This is only true if $\lambda_3 = 0$. Since we can't have $\lambda_3 = 0$, it is clear that $\lambda_3 \neq 0$. Therefore, λ_3 is an eigenvalue of the Givens matrix Z .

Given A and B , the following loop will compute Z and AZ and BZ and update A and B until A is diagonal. This algorithm is called the Givens QR algorithm.

Compute Z from A and B

$A = A^T$ and $B = B^T$

for $j = 1:n/2$

for $i = n - 1:j + 2$

$[c, s] = \text{givens}(A(i-1:i), A(i:i))$

$A(i-1:i,j:n) = \begin{bmatrix} c & s \\ s & c \end{bmatrix} A(i-1:i,j:n)$

$B(i-1:i,i-1:n) = \begin{bmatrix} c & s \\ s & c \end{bmatrix} B(i-1:i,i-1:n)$

$[c, s] = \text{givens}(B(i,i), B(i-1:i))$

$B(i-1:i,i-1:i) = B(i-1:i,i) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

$A(1:n,i-1:i) = A(1:n,i-1:i) \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$

end

end

This algorithm is called the Givens QR algorithm. It is a direct method for solving linear systems of equations. It is based on orthogonal transformations and is very efficient. It is also known as the QR factorization method.

The Givens QR algorithm is a numerical method for solving linear systems of equations. It is based on orthogonal transformations and is very efficient. It is also known as the QR factorization method.

matrix. The first of these assertions is obvious, for if $a_{k+1,k} \neq 0$. Then

$$A - \rightarrow B = \begin{bmatrix} Au - \rightarrow Bu & A_{12} - \rightarrow B_{12} \\ 0 & A_{22} - \rightarrow B_{22} \end{bmatrix}_{n \times k},$$

and we may proceed to solve the two smaller problems $Au - \rightarrow Bu$ and $A_{22} - \rightarrow B_{22}$. On the other hand, if $b_{kk} \neq 0$ for some k , then it is possible to introduce a zero in A 's $(n, n - 1)$ position and thereby deflate. Illustrating by example, suppose $n = 5$ and $k = 3$.

$$A = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix}, \quad B = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{bmatrix}.$$

The zero on B 's diagonal can be "pushed down" to the (5,5) position as follows using Givens rotations.

$$A \leftarrow Q_{34}^T A = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix}, \quad B \leftarrow Q_{34}^T B = \begin{bmatrix} x & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix},$$

$$A \leftarrow AZ_{23} = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix}, \quad B \leftarrow BZ_{23} = \begin{bmatrix} x & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix},$$

$$A \leftarrow Q_{45}^T A = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & x & x & x \end{bmatrix}, \quad B \leftarrow Q_{45}^T B = \begin{bmatrix} x & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{bmatrix},$$

$$A \leftarrow AZ_{34} = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \end{bmatrix}, \quad B \leftarrow BZ_{34}^T = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$A \leftarrow AZ_{45} = \begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x \end{bmatrix}, \quad B \leftarrow BZ_{45} = \begin{bmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

This zero-chasing technique is perfectly general and can be used to zero any entry regardless of where the zero appears along B 's diagonal.

r.r.1

We are now in a position to describe a QZ step. The basic idea is to update A and B as follows

$$(A - \lambda B) = Q\Gamma(A - \lambda B)Z,$$

where A is upper Hessenberg, λ is upper triangular, Q and Z are each orthogonal, and $A - \lambda B$ is essentially the same matrix that would result if a Francis QR step (Algorithm 7.5.1) were explicitly applied to AB^{-1} . This can be done with some clever zero damping and an appeal to the implicit Q theorem.

Let $M = AB^{-1}$ (upper right invertible time o

$$\left[\begin{array}{c} \\ \\ \\ \end{array} \right], \quad \left[\begin{array}{c} \\ \\ \\ \end{array} \right].$$

$$\left[\begin{array}{c} \\ \\ \\ \end{array} \right], \quad \left[\begin{array}{c} \\ \\ \\ \end{array} \right].$$

$$\left[\begin{array}{c} \\ \\ \\ \end{array} \right], \quad \left[\begin{array}{c} \\ \\ \\ \end{array} \right].$$

nkE 9.10 7.7.2 e. 19p 2Sp c.9rrp Given an unreduced upper Hessenberg matrix $A \in \mathbb{R}^{n \times n}$ and a nonsingular upper triangular matrix $B \in \mathbb{R}^{n \times n}$, the following algorithm overwrites A with the upper Hessenberg matrix $Q^T A Z$ and B with the upper triangular matrix $Q^T B Z$ where Q and Z are orthogonal and Q has the same first column as the orthogonal similarity transformation in Algorithm 7.5.1 when it is applied to AB^{-1} .

Let $M = AB^{-1}$ and compute $(M - \alpha I)(I - h) e_1 = [x, y, z, 0, \dots, 0]^T$
where α and h are the eigenvalues of M 's lower 2 by 2.

for $k = 1 \text{ to } n-2$

$$\text{Find } H_b = h \cdot Q \text{ so } Q \begin{bmatrix} & \\ & \end{bmatrix} = \begin{bmatrix} & \\ & \end{bmatrix}.$$

$$A = \text{diag}(h - 1, Q J_{n-k-2} \cdot A)$$

$$B = \text{diag}(h - 1, Q J_{n-k-2} \cdot B)$$

$$\text{Find Householder } Z_{k1} \text{ so } [b_{k+2k-1} b_{k+2k-1}^* b_{k+2k+2}] Z_{k1} = [0 0 1^*].$$

$$A = A \cdot \text{diag}(h - 1, Z_{k1} J_{n-k-2})$$

$$B = B \cdot \text{diag}(h - 1, Z_{k1} J_{n-k-2})$$

$$\text{Find Householder } Z_{k2} \text{ so } [b_{k+1,k-1} b_{k+1,k-1}^*] Z_{k2} = [0 1^*].$$

$$A = A \cdot \text{diag}(J_{k-1} Z_{k2} J_{n-k-1})$$

$$B = B \cdot \text{diag}(J_{k-1} Z_{k2} J_{n-k-1})$$

$$x = a_{k+1,k} = a_{k+2,k}$$

$$\text{if } k < n-2$$

$$z = a_{k+3,k}$$

end

end

$$\text{Find Householder } Q_{n-1} \text{ so } Q_{n-1} \begin{bmatrix} & \\ & \end{bmatrix} = \begin{bmatrix} & \\ & \end{bmatrix}:$$

$$A = \text{diag}(J_{n-2} Q_{n-1}) \cdot A$$

$$B = \text{diag}(J_{n-2} Q_{n-1}) \cdot B.$$

$$\text{Find Householder } Z_{n-1} \text{ so } [b_{mn-1} j b_m] Z_{n-1} = [0 j^*].$$

$$A = A \cdot \text{diag}(J_{n-2} Z_{n-1})$$

$$B = B \cdot \text{diag}(J_{n-2} Z_{n-1})$$

This algorithm requires $2n^2$ fops. Q and Z can be accumulated for an additional $8n^2$ fops and $13n^2$ fops, respectively.

7.7.7 n xi x r lr k α

By applying a sequence of QZ steps to the Hessenberg-triangular pencil $A - \lambda B$, it is possible to reduce A to a quasi-triangular form. In doing this it is necessary to monitor A 's subdiagonal and B 's diagonal in order to bring about decoupling whenever possible. The complete process, due to Moler and Stewart (1973), is as follows.

~~subset~~ 7.7.3 Given $A \in \mathbb{R}^{mn}$ and $B \in \mathbb{R}^{mn}$, the following algorithm computes orthogonal Q and Z such that $Q^T A Z = T$ is upper quasi-triangular and $Q^T B Z = S$ is upper triangular. A is overwritten by T and B by S .

Using Algorithm 7.7.1, overwrite A with $Q^T A Z$ (upper Hessenberg) and B with $Q^T B Z$ (upper triangular).

until $q = n$

Set to zero subdiagonal entries that satisfy $|a_{ij}| - 1 \leq |a_{ii}| + |a_{jj}|$.

Find the largest nonnegative q and the smallest nonnegative p such that if

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ 0 & A_{22} & A_{23} \\ 0 & 0 & A_{33} \end{bmatrix} \begin{matrix} p \\ npq \\ q \\ p \\ npq \\ q \end{matrix}$$

then A_{33} is upper quasi-triangular and A_{22} is upper Hessenberg and unreduced.

Partition B conformably:

$$B = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ 0 & B_{22} & B_{23} \\ 0 & 0 & B_{33} \end{bmatrix} \begin{matrix} p \\ npq \\ q \\ p \\ npq \\ q \end{matrix}$$

if $q \neq n$

if B_{22} is singular

Zero a_{nn} and $q = 1$

else

Apply Algorithm 7.7.2 to A_{22} and B_{22} and update

$$A = \text{diag}(p Q I_q) T A \text{diag}(p Z, I_q)$$

$$B = \text{diag}(v, Q I_q) T B \text{diag}(v, Z, I_q)$$

end

end

end

This algorithm requires $30n^3$ fops. If Q is desired, an additional $16n^3$ are necessary. If Z is required, an additional $20n^3$ are needed. These estimates of work are based on the experience that about two QZ iterations per eigenvalue are necessary. Thus, the convergence properties of QZ are the same as for QR. The speed of the QZ algorithm is not affected by rank deficiency in B .

The computed S and T can be shown to satisfy

$$Q_0^T (A + E) Z_0 = T, \quad Q_0^T (B + F) Z_0 = S,$$

where Q_0 and Z_0 are exactly orthogonal and $\|E\|_F \leq \|A\|_F$ and $\|F\|_F \leq \|B\|_F$.

$\text{EnEnqA Hob=}\{2 - \text{QDAP}\} = -\{1\} \text{A LNA+V}\{1\}\text{icA ..MN} - \{1\}\text{A}$

Many of the invariant subspace computations discussed in §7.6 carry over to the generalized eigenvalue problem. For example, approximate eigenvectors can be found via inverse iteration:

$q^{(0)} \in \mathbb{C}^{n \times n}$ given

for $k = 1, 2, \dots$

$$\text{Solve } (A - \mu B)z^{(k)} = Bq^{(k-1)}.$$

$$\text{Normalize } q^{(k)} = z^{(k)} / \|z^{(k)}\|_2$$

$$A^{(k)} = [q^{(k)}]^H A q^{(k)} \quad I [q^{(k)}]^H A q^{(k)}$$

end

If B is nonsingular, then this is equivalent to applying (7.6.1) with the matrix $B^{-1}A$. Typically, only a single iteration is required if μ is an approximate eigenvalue computed by the QZ algorithm. By inverse iterating with the Hessenberg triangular pencil, costly accumulation of the Z-transformations during the QZ iteration can be avoided.

Corresponding to the notion of an invariant subspace for a single matrix, we have the notion of a deflating subspace for the pencil $A - AB$. In particular, we say that a k -dimensional subspace $S \subset \mathbb{C}^n$ is deflating for the pencil $A - AB$ if the subspace $\{Ax + By : x, y \in S\}$ has dimension k or less. Note that if

$$Q^H AZ = T, \quad Q^H BZ = S$$

is a generalized Schur decomposition of $A - AB$, then the columns of Z in the generalized Schur decomposition define a family of deflating subspaces. Indeed, if

$$Q = [q_1 | \cdots | q_n], \quad Z = [z_1 | \cdots | z_n]$$

are column partitionings, then

$$\text{span}\{Az_1, \dots, Az_k\} \subset \text{span}\{q_1, \dots, q_k\},$$

$$\text{span}\{Bz_1, \dots, Bz_k\} \subset \text{span}\{q_1, \dots, q_k\},$$

for $k = 1:n$. Properties of deflating subspaces and their behavior under perturbation are described in Stewart (1972).

$$|t| t(X - X - k \times \text{fo4} X U X \text{fo8} 4 X = \Delta A U \times |_{\star 1} \Delta X - |) 14 U_c^T \Delta | 4 X = 5 \times 3 \Delta 4 | X$$

More general than the generalized eigenvalue problem is the polynomial eigenvalue problem. Here we are given matrices $A_0, \dots, A_d \in \mathbb{C}^{n \times n}$ and determine $A \in \mathbb{C}$ and $0 \neq x \in \mathbb{C}^n$ so that

$$P(A)x = 0 \tag{7.7.3}$$

where the Amatrix $P(A)$ is defined by

$$P(A) = A_0 + A_1 A + \cdots + A_d A_d. \tag{7.7.4}$$

We assume $A_d \neq 0$ and regard d as the degree of $P(A)$. The theory behind the polynomial eigenvalue problem is nicely developed in Lancaster (1966).

It is possible to convert (7.7.3) into an equivalent linear eigenvalue problem with larger dimension. For example, suppose $d = 3$ and

$$L(A) = \begin{bmatrix} 0 & 0 & \\ -I & 0 & A_1 \\ 0 & -I & A_2 \end{bmatrix} + \lambda \begin{bmatrix} I & 0 & \\ 0 & I & \\ 0 & 0 & A_3 \end{bmatrix}. \quad (7.7.5)$$

If

$$L(\lambda) \begin{bmatrix} u_1 \\ u_2 \\ x \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix},$$

then

$$0 = A_0x + \lambda u_1 = A_0 + \lambda(A_1x + \lambda u_2) = A_0 + \lambda(A_1x + \lambda(A_2 + \lambda A_3))x = P(\lambda)x.$$

In general, we say that $L(A)$ is a linearization of $P(A)$ if there are $d \times d$ matrices $S(A)$ and $T(A)$, each with constant nonzero determinants, so that

$$S(\lambda) \begin{bmatrix} P(\lambda) & 0 \\ 0 & I_{(d-1)n} \end{bmatrix} T(\lambda) = L(\lambda) \quad (7.7.6)$$

has unit degree. With this conversion, the $A - AB$ methods just discussed can be applied to find the required eigenvalues and eigenvectors.

Recent work has focused on how to choose the Atransformations $S(A)$ and $T(A)$ so that special structure in $P(A)$ is reflected in $L(A)$. See Mackey, Mackey, Mehl, and Mehrmann (2006). The idea is to think of (7.7.6) as a factorization and to identify the transformations that produce a properly structured $L(A)$. To appreciate this solution further it is necessary to have a familiarity with Amatrix manipulation and to that end we briefly examine the Amatrix transformations behind the above linearization. If

$$P_1(\lambda) = A_1 + \lambda A_2 + \cdots + \lambda^{d-1} A_d$$

then

$$P(\lambda) = A_0 + \lambda P_1(\lambda)$$

and it is easy to verify that

$$\begin{bmatrix} I_n & -\lambda I_n \\ 0 & I_n \end{bmatrix} \begin{bmatrix} A_0 + \lambda P_1(\lambda) & 0 \\ 0 & I_n \end{bmatrix} \begin{bmatrix} 0 & I_n \\ -I_n & P_1(\lambda) \end{bmatrix} = \begin{bmatrix} \lambda I_n & A_0 \\ -I_n & P_1(\lambda) \end{bmatrix}.$$

Notice that the transformation matrices have unit determinant and that the Amatrix on the right-hand side has degree $d - 1$. The process can be repeated. If

$$P_2(\lambda) = A_2 + \lambda A_3 + \cdots + \lambda^{d-2} A_d$$

then

$$P_1(\lambda) = A_1 + \lambda P_2(\lambda)$$

and

$$\left[\begin{array}{c|cc} I_n & 0 & 0 \\ 0 & I_n & -I_n \\ 0 & 0 & I_n \end{array} \right] \left[\begin{array}{c|cc} A_O & 0 \\ -I_n & R(\rightarrow) & 0 \\ 0 & 0 & I_n \end{array} \right] \left[\begin{array}{c|cc} I_n & 0 & 0 \\ 0 & 0 & I_n \\ 0 & -I_n & P(\rightarrow) \end{array} \right] =$$

$$\left[\begin{array}{c|cc} & 0 & A_0 \\ \hline -I_n & >n & A_i \\ 0 & -I_n & P_2 \end{array} \right].$$

Note that the matrix on the right has degree $d-2$. A straightforward induction argument can be assembled to establish that if the dh -by- dh matrices $S(\rightarrow)$ and $T(\rightarrow)$ are defined by

$$S(\lambda) = \begin{bmatrix} I_n & -\lambda I_n & 0 & \cdots & 0 \\ 0 & I_n & -\lambda I_n & & \vdots \\ 0 & & \ddots & \ddots & \\ \vdots & & & I_n & -\lambda I_n \\ 0 & 0 & \cdots & 0 & I_n \end{bmatrix}, \quad T(\lambda) = \begin{bmatrix} 0 & 0 & 0 & \cdots & I \\ -I_n & 0 & & & P_1(\lambda) \\ 0 & -I_n & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & P_{d-2}(\lambda) \\ 0 & 0 & \cdots & -I_n & P_{d-1}(\lambda) \end{bmatrix}$$

where

$$P_k(\lambda) = A_k + \lambda A_{k+1} + \cdots + \lambda^{d-k} A_d,$$

then

$$S(\rightarrow) \begin{bmatrix} P \rightarrow \\ & 0 \\ & I_{(d-l)n} \end{bmatrix} T(\rightarrow) = \begin{bmatrix} \rightarrow I_n & 0 & 0 & \cdots & A_0 \\ -I_n & \rightarrow I_n & & & A_1 \\ 0 & -I_n & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \rightarrow I_n & A_d \\ 0 & 0 & \cdots & -I_n & A_{d+1} \rightarrow A_d \end{bmatrix}.$$

Note that, if we solve the linearized problem using the QZ algorithm, then $O((dh)^3)$ fops are required.

Problems

$$P7.7.1 \dots .8.. \quad \dots B \nabla^+_{T_3} R^{n \times n} / ^k \quad .^n \\ U^T B V = \begin{bmatrix} D_A \\ - \end{bmatrix}^r_{n-r}, \quad U = \begin{bmatrix} U_1 & U_2 \\ r & n-r \end{bmatrix} f \quad V = [V_1 \quad V_2],$$

$$\sum_{k=0}^r \binom{n}{k} \cdot \frac{c}{k+1} B^k = \frac{1}{2} A^{-1} B + \frac{1}{2} A B^{-1}$$

P7.7.2 ...8 $\mathbf{R}^{n \times n}$. $\mathcal{E}_{(-,+,-,+)}^{(n)}$ [$\mathbf{R}^m(p)$], $^{(n)}(\pi-p)^{p_pm2}$ $Q \nabla_3^T Z \text{tlAp}$

P7.7.3 .2..8.8

$$A = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \quad \mathbb{I} \quad) \quad \begin{bmatrix}) \\ \mathbf{I} \end{bmatrix} \quad \begin{bmatrix} B_{12} \\ B_{22} \end{bmatrix}$$

(I -) E, + a = Q L K E RX_i(+) uRA T/E/R/E = 2 Am iRaA

$$\mathbf{X} = \begin{bmatrix} \cdot & X_{12} \\ ; & I_j \end{bmatrix} \quad \text{and} \quad Y = \begin{bmatrix} I_k & Y_{12} \\ 0 & I_j \end{bmatrix}$$

$\begin{matrix} x & n & S & x & a & \Rightarrow & x & = & = & = & \Rightarrow & x & = & x & n & = & x & \Theta & \text{em} & \text{ex} & \text{x} & = & \text{gated} \\ p & r & \text{hem} & \text{SA} & \text{atria} & \text{igr} & \text{lMn} & \text{pmp} & \text{ACap} & \text{Aa} & r & \text{kll} & \text{kr} & \text{kiae} & \text{rI} & \text{MASS} & \text{AM} & \text{niar} & \text{h} & \text{AM} & \text{9} \\ r & p & \text{McF} & \dots & 4 \end{matrix}$

P7.7.4 .2..8.8 f dia 8..838 .d8...28.. .d8.8.8.8> =w a ' R=

P7.7.6 3.8.8 ...

$$\hat{y}^T = \begin{bmatrix} \vdots & \hat{g}' \\ -\mathbf{f} & \mathbf{I} \\ -\mathbf{I} & \mathbf{J} \\ \vdots & -\mathbf{I} \\ \mathbf{A}_0 & \mathbf{0} \end{bmatrix} \begin{pmatrix} \hat{y} \\ \hat{r} \end{pmatrix} = \begin{bmatrix} \vdots & \hat{g}' \\ \mathbf{r} & \mathbf{J} \\ \mathbf{r} & -\mathbf{J} \\ \mathbf{J} & \mathbf{J} \\ \mathbf{J} & \mathbf{J} \end{bmatrix} \begin{pmatrix} \hat{y} \\ \hat{r} \end{pmatrix}, \quad \mathbf{0} = \begin{bmatrix} \vdots & \hat{g}' \\ \mathbf{J} & -\mathbf{I} \\ \mathbf{J} & \mathbf{J} \\ \mathbf{J} & \mathbf{J} \\ \mathbf{J} & \mathbf{J} \end{bmatrix} \begin{pmatrix} \hat{y} \\ \hat{r} \end{pmatrix}$$

I Ching Γ

wf „ r „r p, l r l , r , , r

sSAntpmAr ipMTa,Mr ,pnplapMrpielr f wk,i. plSlpm „fiae f „9

Notes and References for §7.7

= aM S.A+ plM AaAMiaAaiiSpA MAAlr rspA Mf f nA. spA iMsha
= .hl. kiae ap + naf I dh.iaeT

w5 r ,pMGr3 rhpAf Nd MMatr Pencils, hMIAAAenhpA ii,SieH nAAplMA 2lpA
nanippAr ipBAM- sSMnAM,BAMBar lM+.

nMA,ipAen~~S~~^SASHM~~E~~

r.wOpAMae w.a, spr.iMp_Nm4.3a 3pr lMnphAaAMignApMr9hr Aai.pHMSpA6 ,
 SIAM J. Numer. Anal. , e1.Nf.74

n45hr iaf „.4 lpA n-3gr lMnpshlpiAppAAaAMipAednr AaiiphIMISpASIAM J.
Numer. Anal. „e..(„JI „

G9AaiMef .. 2.lmAAlr S laipslapV- 3gr Mmr sk6 M J. Numer. Anal. ,Zed,r d.,
 r9.-B.a nlia f .N.9 .3 AaAMlppMrdrhr Aaiip3gr Mmp k6 M J. Numer. Anal. ,Ze
 dN d 4

n⁴5hr iaf N . slr AmlIr ppda pAV- 3pr IMnp MslpinaipprAaAMipndr AaiiplA
hM¹Sp¹ k⁶M T ans. Math. Sof w. 3 7. (r)

G9AaiMef NN9.wipiaAnappAAarM,pn,Aar Aaiip,MISpns SIAM J. Sci. Stat. Comput.
..(r,N,11

”¹⁷ Ea ELEMafadII .. 3gr IM1pple Esr wsliae d A.V-e lMpMiahdhpnaAMAlr Shpnar
 EA-plas,S SAApnsSAAAn-sAAApM¹⁸ CM T ns. Math. Sf w t,(7r,dI 4
 59E-+ paeia wSr, pM¹⁹ tN .. wglA+Aigr IMpna²⁰ slpiMAM rAehApllaAr hpiM
 Lm²¹ M²² M²³ AAm²⁴ tAm²⁵ M²⁶ T ns. Math. Sf w t Z Jn 4

$\text{E}^{\text{9s}} \text{p}^{\text{+nf}} \text{I}^{\text{JII-4}} \text{hAM,Mr}^{\text{iaA}} \text{ApA}^{\text{V}} \text{3pr}^{\text{IMnpapm}} \text{MAtAdAfa=pa}^{\text{adr}} \text{AaiiphA,k6}$

SIAM J. Matrix Anal. Appl. 22, 77-104 (2001)
 w9 r.pMCuMtaAok VpnIM, Mpnk,r Vhnapiam,M, Mpld,4.wplA+ Agr IM1pmr
 MppAebApnkl AtAeSAMr MbnimMpnk,r VpnIM, Mpnk,r VpnIM, Mpnk,r VpnIM, Mpnk,r

at μ $\text{M}1\text{un}\text{m}\text{AC}$ $\text{i}\text{L}\text{A}\text{m}\text{in}\text{A}\text{m}\text{m}\text{A}$ $\text{S}\text{M}\text{S}\text{m}\text{E}$ $\text{n}\text{A}\text{d}\text{m}\text{m}\text{A}$ $\text{u}\text{u}\text{S}\text{M}\text{S}\text{A}\text{r}$ a

A. $\text{NO}_2 = \text{O} = \text{N} - \text{O}$ и $\text{H}_2\text{O} = \text{H} - \text{O} - \text{H}$ и $\text{NO}_2 + \text{H}_2\text{O} = \text{HNO}_3$ и $\text{HNO}_3 + \text{NaOH} = \text{NaNO}_3 + \text{H}_2\text{O}$

54 toto atg₀₀ Sy, Zda hoo₀₀ Sm of Xsgtk posrag kn sMcCopro Phien r8T)ædh
 Repxg18wueipg .5N mer. Math. 43, I., „I4
 -r win.r EAGR AAE nr., gN(.4.3a f aiAMrAAA hiMipOrp EsAMrMlas,AM
 3pr lmpM2la,tr gApMnA AaS₁MISNumer. Math. 76, I(„Jd4
 ...r IOIS iae VrlAgJJ4f.aAbiApaiAMrAAMipMaaAMipca Aai.0,MISgAgBIT
 40 7(N7d4
 .. mc,A,MJ,N4.2A, plat,nAppelenaplipnar hBMPApriAf pAMi₁AaAr AaAa
 AMipr3Ae dnhMSpA SIAM J. Matrix Anal. Applic. 22, N,II N,4
 E.nAr laainAM BiaEIIlMAgJJ.74.wiOiaAndAr ,GmpMdaAp,6SIAM J. Matr. Anal.
 Applic. 28, I,II 7,4
 lr Maip.w. 5Sr ,pMGeEx 5MAgaAMrAlgSIpcarhAMnlEAM2pcas,SiAA,3,lArf
 ipAenpp sSAAC=AA. dcr AaiiOIAg 47, 7,r (,4
 mpSAMpIMSpAlMppA AaAMi0AaAaiiSMISOApMripAAT
 .ra. spAiMgN(I.4,a ppAAa,npnihpppAhr AaiiphMISOAx = >Bx," SIAM J. Numer.
 Anal. 9, 77.Nd74
 ..ar spAiMgN(.4.dMMlMhAMpp,MSipA, ,Ms,S,SiAA,3,lAcipAecppAAMpica Aa
 iiQAhMISOA6SIAM Review 15, (I r (7,4
 .rar spAiMgN(.4.AM,pr IMppAlMppAaAMipnAe raiiphMISOAg = >Bx," Math
 Comput. 29, 7JJr 7J74
 3r hBm3Ig,d74,a hAMP,MSipAappAa,niipAaAMSlp i nipMhAaAcd.in. Alg
 Applic. 82, .r N(4
 'r s,a gN,.4 hAMP,MSipAa, ,MppAAaAMi0c3Ap,NEAAAlgSl,npdA SIAM J. Matr
 Anal. Applic. 16 NId Nj4
 lr wpipci ihe Anng.7.4,a hAMPIMSpbdapMhAaAcppl,i0 sSAApMfr6Math
 Comput. 65, 7,r 7,"
 ',hrEAenAN(.4.AlaenpnLSAMiplMlaecpla2lgSAMiae Alaenpc2g SAMpAlMAM
 ppAAaAMipm2Ae AaiiphMISg6Lin. Alg. Applic. 263, Nr I,4
 Er rnr piriae 2r'r,cr piggN.d4 spM,Ap,MAABiMeMMIM iae AllaApAaMipc3Ae
 dhr AaiiphMISg.6SIAM J. Matr. Anal. Applic. 20, „I, N14
 lr wtAMrA.iae BmApMgiag. d. 4.apAMA,pp2AiMA2lanlAr 1GmAcpx.6 Alg
 Applic. 285, dNr, AJ
 Br it, iae Bml,r i3IgN.d.r.3 2lpAla pp2lMgn, AhAMpIMSpplMlMppAAr Ip
 AAAMi0c3Ae AaSMISOAer. Lin. Alg. 5, NvNj4
 lrvAnngJJ.4,a hAMPIMSpdipMhAaAmpAipsSAAnMlAicnp,6Math. Comput.
 72, (N3)Id4
 srwlMiae BmApMr ihe.74.nnaAiMMpIMSpplMlMspM,Ap,MAe nipMhAaAaA
 naAlapMhAlMts6SIAM J. Matrix Anal. Applic. 28, (,d N 7.4
 .rs. ApAglJJ(4,a hAMP,MSipAa, l. AaAMipnAe Aai.p,MEenir lai0c3SpAM,mt
 Numer. Math. 107, (r d74
 mpAkr necker str ctur 1. pp/SAAAtcp ' = x = x = x = r == 2= xa Azxa iaeaA
 SMIindAA,pna,MgipnlsL,p ppAaEMOTrAASgnAla.hiSAMlaA,MAApp ppngSmpaL
 eAAlgSl,npdA AOEAT
 'r.. acpBla gN.(d 4.ncaAiMn AMAshiflapiae 5MlaAABAAlanAibMr nRe n
 Advances in Numerical Analysis, AreAwllMae .r.r .IO,S f Ae,u8AieACdMA,2A. II MBm
 I,N I7,4
 'r.r anpBalagN(.4.5MlaAABAAMr,caiiMg iae ppAV- 3pr lmp 16n. Alg. Applic. 28
 Idr,J,4
 h BiaEIIlMAgN(.4.mpAlr SI piphl5MlaAABAAl, AlMgAipsna 1GmAp.6.
 Alg. Applic. 27, N,II N,4
 'a. EAggApNd.4 AlpEnpPar SAMdsIniipAaAa,Mr ipnlppipvpABcir,22
 nipMhAaAmpA SIAM J. Numer. Anal. 20, „r 7Nj4
 'rar EAR gAp w.5Sr ,pMGe(d4.AlgS,pnaipiSpdn AaeAAlgSl,npclapMhAaAnptm
 Linear Alg. Applic. 88/89, N, N74
 w5Sr ,pMGe(d4.mpaAaAMipnAa,piMhAaAAlgSl,npdAe ppAAaAMi0- 1('4
 " A2BIT 24, 7dr d4
 w.5Sr ,pMGe(d74.1.sBE T3a 3pr lmpMAlr SI ppp5MlaAABA,MAAp,MAAlA2
 sIS,SiAA, sna ,0M, ,r hAaAm SIAM J. Sci. Stat. Comput. 7, N,II N,4

7. The Generalized Eigenvalue Problem

419

- n^o enex dt n thgsin ly. Zi. hC=du lnsah sMetrap et Gspke dat xipah
 Dgug 2 Galton PeagzQ, Ykn Dapnca cxd ia Lar e Scale Eigenvalue Pr blem, 9r
 Alpplae lr 3ranggTrpSf Ae, 2Mprngiae 3r ,pAMer r
 mwAAGae hrB.a O,MAA Ndr .3a fr S₁M 3prlMcLMppAA,r SI ppc,a 5M,aAA+ AMQ,
 A.a,anAg „Mr. scarIpilAA₁ Lin. Alg. Applic. 105, r 7,r
 d9,r M,ma wi5 r,pMCN,r ImAsAL InStr,LMcbaAaAnp,5M,aAA+ sAMApIMA, .ae
 mpAmmpc,a, IaeAMAM,IMSpca,OM J. Matr. Anal. Applic. 17, ,1,r
 A. tOOAT (=tAT =Ce> 0xT 2y + = N₂DyDCT O= yf(G)E= X_O= p = 10C=QC= F(A= E0C= Caa = X_O(EX(=xT θ + (Q! CFea = fSIAM J. Matr. Anal. Applic. 18
 7, r 7r
 drdgr M,pp9,mia,,a iae wr5 r,pMCN,rA,r SI p.pniae hMA,Aap.phlaM.Om,
 On,Op.tnaip,IMAM.MAm9,uea.ee 5M,aAA+ sAMApIMA, Lin. Alg. 8, ,dNr ,r
- ItL CpaApIeAA,r Ol,npA=4AI,AeL „piAppAtgiA,LA₁.Ln,OM,4gAIX - XA2= B,
 „”(-,-,p”+-,l-- +”,D -,-,0,(.,)(.” --”+0. -T.”(, ”-”(-+p”+γ..”-”(”-”0+(-”D
 ”(”D-(,,- X.ae Y MAIrml, pp.pAIX - YA2=B₁ -(AaX - YA4= B₂, -””n
- I ((+-,0(+ -V”-, (U,w21 V. G(“ .(or” : ,+)((-’T-(+,.5 8-((,F 1”(,- -γ-(D-AzIf
 18 I(7vdNr
- B. T.y+= a- =SDLO+ = N₂O₂ nD 10= 0= a 0E(0)= e CO+ eC= 0O= (G== (T(= C= +
 OFA>0= 0= (<0<0 0tn=p0=>0NEE ns. Autom. Contr. r 51, ,r ,Nr
 B. T yo=N₂pDyXO= p=y=(<0<0 0= 0= (A01b000tn=p0E (OC=LB, DR-
 2, . ! SIAM J. Matr. Anal. Applic. 15, J,r ,J7Jr
- 9r si a f N₄.hAMplMS,saigt,n „ st,paR,AaSAMfe A,AaSAMrr .carIpMf ,6
 Lin. Alg. Applic. 241-3, dNv, r
- B. T.yo= a- =X₂ XC=N₂pDSyXyeTK+yAy= 0 (=<0IGJ3a e=OCr> (=eD=,
 0= (A01b00 0tn=p0= aC=0+ =OFaeODD(y)O=C00y p<AE= X(d)=ACM
 n. Math. Sof w 28, (d NJ, r
- f raa,,a ae wr5 r,LMEIJr JAAIM,niagA+ 3r,Mcpr,Ms,gicar.narIp.Mst,pAr ,
 hMpff k ,neAe ae .3aAupn,ApA,paae ntOlai T,pMn9SI,pnaa,6ACM ns.
 Math. Sof w 28, N₇,r
- It MmaiLae wr5Sr,puCrf IJNJ.hiM,gaApiaMu stpiA,pAMr llpAd9,cipla, cpp
 3OSpcAipcaAlaenpcApnr .pdalMp,lmAlMt Bg,mpmr A6M ns. Math. Sof w
 37, 3MpnaAgA
- parIpurAaAM,gAA,Aal pDM,SpAp, ,Mn,ApappnALpcapmA g c, L, MAel pAM.a+
 a r - -A_n
- ItI0”1”((U,.-=1V ⊕”(0-(.-,(.i,”(γ 9(3”(),(—T)(8-((,F 1”(+ T-A2n. Alg. Applic.
 208/209, I,r ,J(r
- Prw,ILutI r dpe.r.r „OIS .ae hrIcg a MIJ,r lmA,AaAMigc,9e3ai,plAM,SgAM
 v,a,sl,MAhAaAm, ,car. Inacr hAMplMS,poM, ,6SIAM J. Matr. Anal. Applic. 27,
 ,dI r 7N r
- D. eep(= 0x 3x < pN22hpDi=(10= 0= (A0b0y0= M< pfe=CTC= + nXOEM SIAMN
 J. Matr. Anal. Applic. 28, (Jr (dr
- AuAaAMppOgtalr AgAaaiigDM,SpApApIeAe
- P .(= Ea= N₂Op Kambda-Matrices and Vibr ting Systems, hAur.r hMA,bMe. r r 5r
 I.)FenOExXs (= E (+ (= S,j= OT aN22 Matr. Polynomials, 3A,eAr hMA,2A, , M+r
 F. 'OsOp N22x D) (E83 (= OE E €OC= 0O-C(X€Ae= CT QyQ= XEAp0=O+ Alg
 Applic. 309, „r ,7 Nr
 'r .10AenAde.r lc,AIM JJ,r hAMLIMS,ma,MM,,r ,rAaAll,hgtalr c9nrAai,pIA
 hulSpAr Li6 Alg. Applic. 358, ((r.,r
 N.J. 3Oye(Tx OAA E80e=ROx(+ + ON22pp D.e0C= 0(= (CG-S+=y0(EOb(G=A(=E(d
 XFAET ((AM J. Matr. Anal. Applic. 28, NJJ,vNJI dr
 D.S. A(E0n ABOe>exAO• <AOe= T (N+p28= OE0IC(EKOS 0= 0(= (beEAC= -E Od
 XFAe= CT \$AM J. Matr. Anal. Applic. 28, (r NJ,r

The G= hP Pn pQ O=0y(E > D IAp6AOU+ Epi6BOO= Zx 2x

7.8 Hamiltonian and Product Eigenvalue Problems

Two structured unsymmetric eigenvalue problems are considered. The Hamiltonian matrix eigenvalue problem comes with its own special Schur decomposition. Orthogonal symplectic similarity transformations are used to bring about the required reduction. The product eigenvalue problem involves computing the eigenvalues of a product like $A_1 A_2^T A_3$ without actually forming the product or the designated inverses. For detailed background to these problems, see Kressner (NMG) and Watkins (MEP).

Cslj s aRp I A4 I R4R .{I VsI fp4 if TIApp-s

Hamiltonian and symplectic matrices are introduced in §1.3.10. Their 2by-2 block structure provide a nice framework for practicing block matrix manipulation, see P1.32 and P254. We now describe some interesting eigenvalue problems that involve these matrices. For a given n , we define the matrix $J \in \mathbb{R}^{4n \times 4n}$ by

$$J = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}$$

and proceed to work with the families of 2by-2 block structured matrices that are displayed in Figure 7.8.1. We mention four important facts concerning these matrices.

Family	Definition	What They Look Like	
Hamiltonian	$JM = (JM)^T$	$M = \begin{bmatrix} A & G \\ F & -A^T \end{bmatrix}$	G symmetric F symmetric
Skew Hamiltonian	$JN = - (JN)^T$	$N = \begin{bmatrix} A & G \\ F & A^T \end{bmatrix}$	G skewsymmetric F skewsymmetric
Symplectic	$JS = S^{-1} TJ$	$S = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix}$	S_{11} symmetric S_{12} symmetric $S_{22} = I + S_1^T S_2$
Orthogonal Symplectic	$JQ = QJ$	$Q = \begin{bmatrix} Q_1 & Q_2 \\ -Q_2 & Q_1 \end{bmatrix}$	$Q_1^T Q_2$ symmetric $I = Q_1^T Q_1 + Q_2^T Q_2$

Figure 7.8.1. Hamiltonian and symplectic structures

(1) Symplectic similarity transformations preserve Hamiltonian structure

$$J(S^{-1}MS) = (JS^{-1}J^T)(JM^T)(JM)(JS) = -S^T M^T S^{-T} J = (J(S^{-1}MS))^T.$$

(2) The square of a Hamiltonian matrix is skew-Hamiltonian

$$JM^2 = (JMJT)(JM) = -JJY(JMF) = -M^2J^2JT = -(JM)^2.$$

(3) If M is a Hamiltonian matrix and $\lambda \in \sigma(M)$, then $-\lambda \in \sigma(M)$:

$$M \begin{bmatrix} u \\ v \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \end{bmatrix} \Rightarrow M^T \begin{bmatrix} v \\ -u \end{bmatrix} = -\lambda \begin{bmatrix} v \\ -u \end{bmatrix}.$$

(4) If S is symplectic and $\lambda \in \sigma(S)$, then $-\lambda \in \sigma(S)$:

$$S \begin{bmatrix} u \\ v \end{bmatrix} = \lambda \begin{bmatrix} u \\ v \end{bmatrix} \Rightarrow S \begin{bmatrix} v \\ -u \end{bmatrix} = -\lambda \begin{bmatrix} v \\ -u \end{bmatrix}.$$

Symplectic versions of Householder and Givens transformations have a prominent role to play in Hamiltonian matrix computations. If $P = I_{2n} - 2v v^T$ is a Householder matrix, then $\text{diag}(P, P)$ is a symplectic orthogonal matrix. Likewise, if $G \in \mathbb{H}^{2n \times 2n}$ is a Givens rotation that involves planes i and $i+n$, then G is a symplectic orthogonal matrix. Combinations of these transformations can be used to introduce zeros. For example, a Householder-Givens-Householder sequence can do this:

$$\begin{bmatrix} x \\ x \\ x \\ \hline x \\ x \\ x \\ x \\ x \end{bmatrix} \xrightarrow{\cdot}, \begin{bmatrix} x \\ x \\ x \\ \hline x \\ x \\ 0 \\ 0 \\ 0 \end{bmatrix} \xrightarrow{\cdot}, \begin{bmatrix} x \\ x \\ x \\ \hline 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \xrightarrow{\cdot}, \begin{bmatrix} x \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

This kind of vector reduction can be sequenced to produce a constructive proof of a structured Schur decomposition for Hamiltonian matrices. Suppose λ is a real eigenvalue of a Hamiltonian matrix M and that $x \in \mathbb{H}^n$ is a unit 2norm vector with $Mx = \lambda x$. If $Q \in \mathbb{H}^{2n \times 2n}$ is an orthogonal symplectic matrix and $Qx = e_1$, then it follows from $(QM)Q(Qx) = \lambda(Qx)$ that

$$QM \begin{pmatrix} \lambda & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & * & * & * & * & * \\ \hline 0 & 0 & 0 & 0 & -\lambda & 0 \\ 0 & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & * & * & * & * & * \end{pmatrix}.$$

The "extra" zeros follow from the Hamiltonian structure of QM . The process can be repeated on the 6by-6 Hamiltonian submatrix defined by rows and columns 2-3-4-6-7-8. Together with the assumption that M has no purely imaginary eigenvalues, it is possible to show that an orthogonal symplectic matrix Q exists so that

$$Q^T M Q = \begin{bmatrix} Q_1 & Q_2 \\ -Q_2 & Q_1 \end{bmatrix}^T \begin{bmatrix} A & F \\ G & -A^T \end{bmatrix} \begin{bmatrix} Q_1 & Q_2 \\ -Q_2 & Q_1 \end{bmatrix} = \begin{bmatrix} T & R \\ 0 & -T^T \end{bmatrix} \quad (7.81)$$

where $T \in \mathbb{R}^{n \times n}$ is upper quasi-triangular. This is the real Hamiltonian-Schur decomposition. See Paige and Van Loan (1981) and, for a more general version, Lin, Mehrmann, and Xu (1999).

One reason that the Hamiltonian eigenvalue problem is so important is its connection to the algebraic Riccati equation

$$G + XA + A^TX - XFX = 0. \quad (7.82)$$

This quadratic matrix problem arises in optimal control and a symmetric solution is sought so that the eigenvalues of $A - FX$ are in the open left half plane. Modest assumptions typically ensure that M has no eigenvalues on the imaginary axis and that the matrix Q_1 in (7.81) is nonsingular. If we compare (2.1) blocks in (7.81), then

$$Q A Q_1 - Q F Q_2 + Q G Q_1 + Q A T Q_2 = 0.$$

It follows from $I_2 = Q_1 Q_1 + Q_2 Q_2$ that $X = Q_2 Q_1^{-1}$ is symmetric and that it satisfies (7.82). From (7.81) it is easy to show that $A - FX = Q_1 T Q_1^{-1}$ and so the eigenvalues of $A - FX$ are the eigenvalues of T . It follows that the desired solution to the algebraic Riccati equation can be obtained by computing the real Hamiltonian-Schur decomposition and ordering the eigenvalues so that $A(T)$ is in the left half plane.

How might the real Hamiltonian-Schur form be computed? One idea is to reduce M to some condensed Hamiltonian form and then devise a structure-preserving QR-iteration. Regarding the former task, it is easy to compute an orthogonal symplectic U_0 so that

$$T M U_0 = \begin{bmatrix} H & R \\ D & -HT \end{bmatrix} \quad (7.83)$$

where $H \in \mathbb{R}^{n \times n}$ is upper Hessenberg and D is diagonal. Unfortunately, a structure-preserving QR iteration that maintains this condensed form has yet to be devised. This impasse prompts consideration of methods that involve the skew-Hamiltonian matrix $N = M^T$. Because the (2,1) block of a skew-Hamiltonian matrix is skewsymmetric, it has a zero diagonal. Symplectic similarity transformations preserve skew-Hamiltonian structure, and it is straightforward to compute an orthogonal symplectic matrix U_0 such that

$$V_0^T M^2 V_0 = \begin{bmatrix} H & R \\ 0 & HT \end{bmatrix}, \quad (7.84)$$

where H is upper Hessenberg. If $U_0^T H U_0 = T$ is the real Schur form of H and $Q = V_0 \cdot \text{diag}(U_0, U_0)$, then

$$Q^T M^2 Q = \begin{bmatrix} T & U_0^T H U_0 \\ 0 & TT \end{bmatrix}$$

is the real skew-Hamiltonian Schur form. See Van Loan (1984). It does not follow that QM^TQ is in Schur-Hamiltonian form. Moreover, the quality of the computed small eigenvalues is not good because of the explicit squaring of M. However, these shortfalls can be overcome in an efficient numerically sound way; see Chu, Lie, and Mehrmann (2007) and the references therein. Kressner (NMSE, p. 175–208) and Watkins (MEP, p. 319–341) have in-depth treatments of the Hamiltonian eigenvalue problem.

Using SVD and QZ, we can compute the eigenvalues of ATA and B⁻¹A without forming products or inverses. The intelligent computation of the Hamiltonian-Schur decomposition involves a correspondingly careful handling of the product M-times M. In this subsection we further develop this theme by discussing various product decompositions. Here is an example that suggests how we might compute the Hessenberg decomposition of A, f

$$A = A_3 A_2 A_1$$

where $A_1 A_2 A_3 \in \mathbb{R}^{n \times n}$. Instead of forming this product explicitly, we compute orthogonal $U_1, U_2, U_3 \in \mathbb{R}^{n \times n}$ such that

$$\begin{aligned} U_1^T A_3 U_3 &= H_3 && \text{(upper Hessenberg),} \\ U_2^T A_2 U_2 &= T_2 && \text{(upper triangular),} \\ U_1^T A_1 U_1 &= T_1 && \text{(upper triangular).} \end{aligned} \quad (7.8.5)$$

It follows that

$$U_1^T A U_1 = (U_1^T A_3 U_3)(U_3^T A_2 U_2)(U_2^T A_1 U_1) = H_3 T_2 T_1$$

is upper Hessenberg. A procedure for doing this would start by computing the QR factorizations

$$Q_2^T A_1 = R_1, \quad Q_3^T (A_2 Q_2) = R_2.$$

If $A_3 \perp\!\!\!\perp A_2 Q_3$ then $A = A_3^T R_2^T R_1$. The next phase involves reducing A_3 to Hessenberg form with Givens transformations coupled with "bulge chasing" to preserve the triangular structures already obtained. The process is similar to the reduction of $A - \lambda B$ to Hessenberg-triangular form, see §7.7.4.

Now suppose we want to compute the real Schur form of A

$$\begin{aligned} Q_1^T A_3 Q_3 &= T_3 && \text{(upper quasi-triangular),} \\ Q_1^T A_2 Q_2 &= T_2 && \text{(upper triangular),} \\ Q_1^T A_1 Q_1 &= T_1 && \text{(upper triangular),} \end{aligned} \quad (7.8.6)$$

where $Q_1, Q_2, Q_3 \in \mathbb{R}^{n \times n}$ are orthogonal. Without loss of generality we may assume that $\{A_3, A_2, A_1\}$ is in Hessenberg-triangular-triangular form. Analogous to the QZ iteration, the next phase is to produce a sequence of converging triplets

$$\{A_3^{(k)}, A_2^{(k)}, A_1^{(k)}\} \rightarrow \{T_3, T_2, T_1\} \quad (7.8.7)$$

with the property that all the iterates are in Hessenberg-triangular-triangular form

Product decompositions (7.85) and (7.86) can be framed as structured decompositions of block-cyclic 3by-3 matrices. For example, if

$$U = \begin{bmatrix} U_1 & 0 & 0 \\ 0 & U_2 & 0 \\ 0 & 0 & U_3 \end{bmatrix}$$

then we have the following restatement of (7.85):

$$U^T \begin{bmatrix} 0 & 0 & A_3 \\ A_1 & 0 & 0 \\ 0 & A_2 & 0 \end{bmatrix} U = \begin{bmatrix} 0 & 0 & H_3 \\ T_1 & 0 & 0 \\ 0 & T_2 & 0 \end{bmatrix} = \tilde{H}.$$

Consider the zero/nonzero structure of this matrix for the case $n = 4$

$$i = \left[\begin{array}{c|cc|cc|cccc} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x & x & x & x \\ 0 & 0 & (& 0 & 0 & (& 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x & x \\ \hline x & x & x & x & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & x & x & x & 0 & 0 & (& 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & x & x & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & x & x & x & x & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & x & x & x & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & x & x & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & x & 0 & 0 & 0 & 0 \end{array} \right].$$

Using the perm of shuf eP34 (see §12.11) we also have

$$\mathcal{P}_{34} \tilde{H} \mathcal{P}_{34} = \left[\begin{array}{c|cc|cc|cccc} 0 & 0 & x & 0 & 0 & x & 0 & 0 & x & 0 & 0 & x \\ x & 0 & 0 & x & (& 0 & x & 0 & 0 & x & 0 & 0 \\ 0 & x & 0 & 0 & x & 0 & 0 & x & 0 & (& x & (\\ \hline 0 & 0 & x & 0 & 0 & x & 0 & 0 & x & 0 & 0 & x \\ (& 0 & 0 & x & (& 0 & x & 0 & 0 & x & 0 & 0 \\ 0 & 0 & 0 & 0 & x & 0 & 0 & x & 0 & 0 & x & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & x & 0 & 0 & 0 & 0 & x \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & x & 0 & 0 & 0 & (\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x & 0 & x & 0 \\ \hline 0 & (& 0 & (& 0 & (& 0 & 0 & x & (& 0 & x \\ (& 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x & (& (\\ 0 & (& (& 0 & 0 & (& 0 & 0 & (& 0 & x & 0 \end{array} \right].$$

Note that this is a highly structured 12by-12 upper Hessenberg matrix. This connection makes it possible to regard the product-QR iteration as a structure preserving

QR iteration. For a detailed discussion about this connection and its implications for both analysis and computation, see Kressner (NMSE, pp. 146–174) and Watkins (MEP, pp. 293–308). We mention that with the “technology” that has been developed, it is possible to solve product eigenvalue problems.

$$\begin{bmatrix} 0 & 0 \\ A_2 & 0 \\ 0 & A_3 \\ 0 & 0 \end{bmatrix}.$$

1. $\text{uxM k222kE blaxgxt lmg 8 g x3tonk sato ato rxsato Pgapxg5SIAM J. Matr x Anal. Applic. 21, ... (7r)}$
 $Bx2pMr iae E xsai p+ naJJN spMhA2phM2,2Mn2apple, MAlr Shpnar dnr 2aS l. niMrsSiM, A 2. ir npp, aniam ir planih2aArts SIAM J. Sci. Comput. 22, NJ, NI., hx' 2aa2Mlx' t2M, n2Mhiaiiae x.h IIJJI. 2hr 2MhAi> ShpiplhlE2ipna shS, SiA2l. s+ 2Mf iplanin.i.r >planih2aAnp, SAM J. Matr x Anal. Applic. 24, N7, NJ.$
- n2pple, M, tr SpAApnaA2aiSpMS>AiM2r, Ah, 2e aT
- hr' 2aa2MC, S2ae2Mae E xsai p+ na(... .. .sl iae s- 3pr lMnpM , pp2 S>2Apr A f ' pp2M dnr 2aS2MS2n. Alg. Applic. 287, N(7).
- ap2lphSj 5ipiias-E i,r lMnppp1p2 er, Ahr app2a2T ApisSp2M, al , MFA c₂^T AA^T
 $\begin{aligned} T_1c_2s) & \cdot h c_2 \bar{f} \left(\begin{array}{c} (c_{33})^T \\ -\mu_1 c_2 s \sqrt{3} \nabla \end{array} \right) T_1 c_2 s \left(\begin{array}{c} T_1(c\Sigma)^T \\ T_1 c_2 s \end{array} \right) T_1 c_2 s, \quad c^T, c_1 c_2 s \Sigma \nabla^T \left(\begin{array}{c} (c_{33})^T \\ -\mu_1 c_2 s \sqrt{3} \nabla \end{array} \right) T_1 c_2 s \left(\begin{array}{c} T_1(c\Sigma)^T \\ T_1 c_2 s \end{array} \right) T_1 c_2 s = T_1 c_2 s \nabla, \nabla \Sigma^T \end{aligned}$
- .g R₃ ∈ c₃ 2, U_θ² λ d.. θ+3k₂^T ∇_i A∇^T n T_{k3-3} ∇_{i+1} c_i e_i^T Σ SIAM J. Numer. Anal. 12, dN-d.. n.ax.Ai pp369xihS.AxAlr dnr 2ae lxAiMeINd7. Alr Shpnpp2s-E l. i hM, ehAp al nipMr. A2l AM J. Sci. Stat. Comput. 7, N,N IN,..
- lx nippCND, .3aiptr .. 3pr MnpppMpplr ,aipr hMehAp,r anpiMpMnA2um Lin. Alg. 3, NI (,,
- .. l>hS. 5xs.pai.iae h Bia E, lM2alJJ, Jr A,r Shpnpp2s-E l. i .2a2MnaiipMrhM, ef hApnVhlpn221 AM J. Matrix Anal. Applic. 22 N-N..
- E.sxai p+ naJJJ,.. hMehAp,r AaiipM2lSp2, SIAM Review 47, J.
- l. Maaipiae' 56 ,pM, lJJ, 7 EnM2ap 2aiipM2Me2Mr ahMehApnipMr A22MnlenA sAphMMit SIAM J. Matr x Anal. Applic. 28, Id, J.J.
- r.aip,t.2 r 2applaipp2M2, hs, piapnSlet lIM+AlaA2Ma2pppMhAphn2M2M1t,n, iae,pMhAphn222pMhAphn2M, pMhAphM2e SMIpMh22T
- x an,2MJ,.. 3 ApimP' iA+ iMMIM, r 2
- k $\frac{\text{UiaM} + \ddot{A} \ddot{G} \ddot{O}}{\text{@ A'd}}$

unsymmetric eigenvalue problems are fresh in mind. Moreover, with this "early" foundation we can subsequently present various pseudospectra insights that concern the behavior of the matrix exponential (§9.3), the Arnoldi method for sparse unsymmetric eigenvalue problems (§10.5), and the GMRES method for sparse unsymmetric linear systems (§11.4).

For maximum generality, we investigate the pseudospectra of complex, non-normal matrices. The definitive pseudospectra reference is 'trefethen and Embree (SAP). Virtually everything we discuss is presented in greater detail in that excellent volume.

`nmrhb h k f tQ1kb`

In many settings, the eigenvalues of a matrix "say something" about an underlying phenomenon. For example, if

$$A = \begin{bmatrix} \lambda_1 & M \\ 0 & \lambda_2 \end{bmatrix}, \quad \mathbf{M} \neq 0$$

then

$$\lim_{k \rightarrow \infty} \|A^k\|_2 = 0$$

if and only if $\|\mathbf{A}_1\|_2 < 1$ and $\|\mathbf{A}_2\|_2 < 1$. This follows from Lemma 7.31, a result that we needed to establish the convergence of the QR iteration. Applied to our 2by2 example, the lemma can be used to show that

$$\|A^k\|_2 \leq \frac{\rho}{E} (\rho(A) + \epsilon)^k$$

for any $E > 0$ where $\rho(A) = \max\{\|\mathbf{A}_1\|_2, \|\mathbf{A}_2\|_2\}$ is the spectral radius. By making E small enough in this inequality, we can draw a conclusion about the asymptotic behavior of A^k :

If $\rho(A) < 1$, then asymptotically A^k converges to zero. $\rho(A)^k$. (7.9.1)

However, while the eigenvalues adequately predict the limiting behavior of $\|A^k\|_2$, they don't (by themselves) tell us much about what is happening if k is small. Indeed, if $\mathbf{A}_1 = \mathbf{A}_2$ then using the diagonalization

$$A = \begin{bmatrix} \mathbf{M} - \mathbf{j}\mathbf{i} \\ \vdots \end{bmatrix} \begin{bmatrix} & 1 \\ & \ddots \end{bmatrix} \begin{bmatrix} \mathbf{M} - \lambda_1 \mathbf{i} \\ \vdots \end{bmatrix}^{-1} \quad (7.9.2)$$

we can show that

$$A^k = \left[\begin{array}{c|c} \mathbf{A}_1^k & M \sum_{j=0}^{k-1} \mathbf{A}_1^{k-1-j} \mathbf{A}_2^j \\ \hline \mathbf{0} & \mathbf{A} \end{array} \right]. \quad (7.9.3)$$

Consideration of the (1,2) entry suggests that A^k may grow before decaying. This is confirmed in Figure 7.9.1 where the size of $\|A^k\|_2$ is tracked for the example

$$A = \begin{bmatrix} 0.999 & 1000 \\ 00 & 0.998 \end{bmatrix}.$$

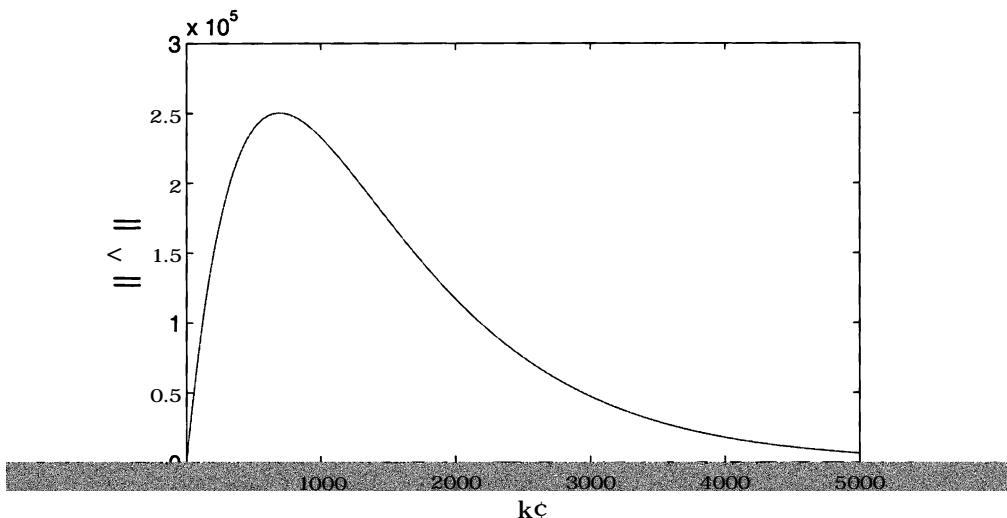


Figure 7.91 $\|A^k\|_F$ can grow even if $p(A) \neq 1$

Thus, it is perhaps better to augment (7.91) as follows

If $p(A) \neq 1$, then asymptotically A^k converges to zero like $p(A)^k$. However, A^k may grow substantially before exponential decay sets in

This example with its ill-conditioned eigenvector matrix displayed in (7.92), points to just why diagonal eigenvalue analysis is not so informative for nonnormal matrices. Ill-conditioned eigenvector bases create a discrepancy between how A behaves and how its diagonalization XAx^{-1} behaves. Pseudospectra analysis and computation narrow this gap.

nhimib p\\$ OTTF Tkeb

The pseudospectra idea is a generalization of the eigenvalue idea. Whereas the spectrum $\sigma(A)$ is the set of all $z \in \mathbb{C}$ that make $Q_{\min}(A - zI)$ zero, the Epseudospectrum of a matrix $A \in \mathbb{C}^{n \times n}$ is the subset of the complex plane defined by

$$\text{Ae}(A) = \{z \in \mathbb{C} : Q_{\min}(A - zI) \leq \epsilon\}. \quad (7.95)$$

If $\lambda \in \text{Ae}(A)$, then λ is an Epseudoeigenvalue of A . A unit 2-norm vector v that satisfies $\|(A - zI)v\|_2 = \epsilon$ is a corresponding f-pseudoeigenvector. Note that if ϵ is zero, then $\text{Ae}(A)$ is just the set of A 's eigenvalues, i.e., $\text{Ae}(A) = \sigma(A)$.

We mention that because of their interest in what pseudospectra say about general linear operators, Trefethen and Embree (2005) use a strict inequality in the definition (7.95). The distinction has no impact in the matrix case.

$$\Lambda_\epsilon(A) = \left\{ z \in \mathbb{C} : \| (zI - A)^{-1} \|_2 \geq \frac{1}{\epsilon} \right\}$$

$$\Lambda_\epsilon(A) = \{ z \in \mathbb{C} : z \in \Lambda(A + E), \|E\|_2 \leq \epsilon \}$$

$$\left[\begin{array}{c} \\ \\ \\ \end{array} \right],$$

$$- \left[\begin{array}{c} \\ \\ \\ \end{array} \right].$$

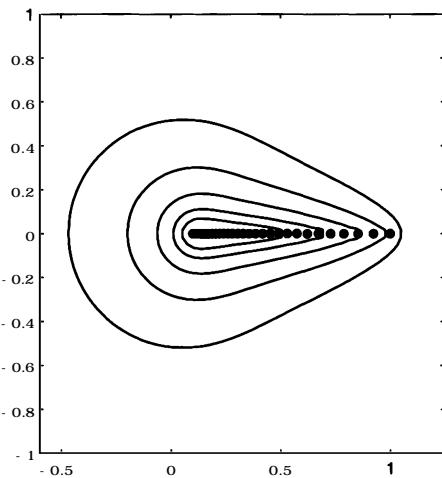


Figure 7.92 A (Kah(s)) with $s_9 = 0.1$ and contours for $\epsilon = 10^{-2}, \dots, 10^{-6}$

The matrix $\text{Dem}(\ell)$ is defective and has the property that very small perturbations can move an original eigenvalue to a position that are relatively far out on the imaginary axis. See Figure 7.93. The example is used to illuminate the nearness-to-instability problem presented in P7.9.13.

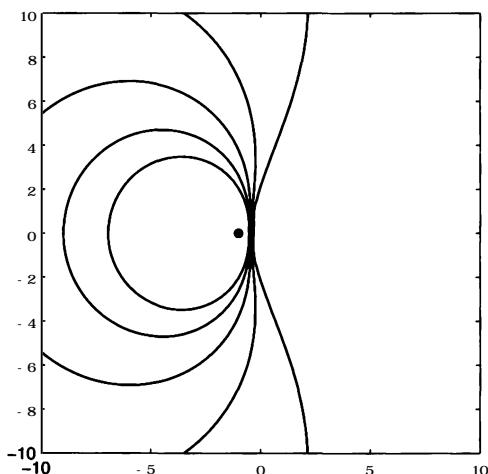


Figure 7.93 A (Dem (f)) with $\beta^{49} = 10^8$ and contours for $\epsilon = 10^{-2}, \dots, 10^{-6}$

The last example concerns the pseudospectra of the MATLAB "Gallery(5)" matrix:

$$G_S = \begin{bmatrix} -9 & 11 & -21 & 63 & -252 \\ 70 & -69 & 141 & -421 & 1684 \\ -575 & 575 & -1149 & 3451 & -13801 \\ 3891 & -3891 & 7782 & -23345 & 93365 \\ 1024 & -1024 & 2048 & -6144 & 24572 \end{bmatrix}.$$

Notice in Figure 7.94 that $A(10)ia5(G_S)$ has five components. In general, it can be

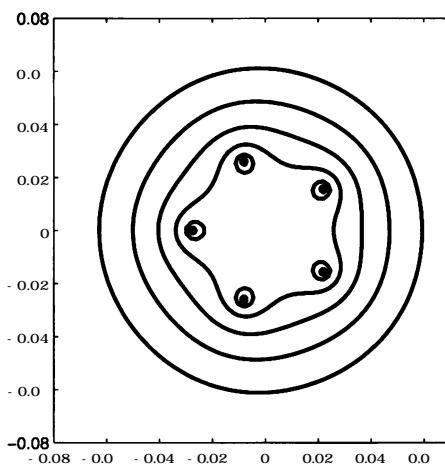


Figure 7.94 $A(10)ia5(G_S)$ with contours for $e = 10^{-11}s, 10^{-12}, \dots, 10^{-13}s, 10^{-14}$

shown that each connected component of A_e (A) contains at least one eigenvalue of A .

whimsical bdsstf txkb. $\text{kes } f \text{ tsb}$

Pseudospectra are subsets of the complex plane so we start with a quick summary of notation. S_1 and S_2 are subsets of the complex plane, then their sum $S_1 + S_2$ is defined by

$$S_1 + S_2 = \{s : s = s_1 + s_2, s_1 \in S_1, s_2 \in S_2\}.$$

If S consists of a single complex number α , then we write $\alpha + S$. If S is a subset of the complex plane and β is a complex number, then $\beta \cdot S$ is defined by

$$\beta \cdot S = \{\beta z : z \in S\}.$$

The disk of radius r centered at the origin is denoted by

$$A_r = \{z : |z| \leq r\}.$$

Finally, the distance from a complex number z_0 to a set of complex numbers S is defined by

$$\text{dist}(z_0, S) = \min\{|z_0 - z| : z \in S\}.$$

Our first result is about the effect of translation and scaling. For eigenvalues we have

$$\Lambda(\alpha I + A) = \{\lambda: \mathbf{1}(zI - (\alpha I + A))^{-1} \mathbf{1} = 1/\lambda\}$$

The following theorem establishes an analogous result for pseudospectra.

Theorem 7.9.1. If $\alpha, E \in \mathbb{C}$ and $A \in \mathbb{C}^{n \times n}$, then $\Lambda(\alpha I + A) = \alpha + \Lambda(A)$.

Proof. Note that

$$\begin{aligned}\Lambda(\alpha I + A) &= \{z: \mathbf{1}(zI - (\alpha I + A))^{-1} \mathbf{1} = 1/z\} \\ &= \{z: \mathbf{1}((z - \alpha)I - A)^{-1} \mathbf{1} = 1/(z - \alpha)\} \\ &= \alpha + \{z - \alpha: \mathbf{1}((z - \alpha)I - A)^{-1} \mathbf{1} = 1/(z - \alpha)\} \\ &= \alpha + \{z: \mathbf{1}(zI - A)^{-1} \mathbf{1} = 1/z\} = \Lambda(A)\end{aligned}$$

and

$$\begin{aligned}\Lambda(\alpha I - A) &= \{z: \mathbf{1}(zI - A)^{-1} \mathbf{1} = 1/z\} \\ &= \{z: \mathbf{1}(zI - A)^{-1} \mathbf{1} = 1/z\} \\ &= \alpha - \{z: \mathbf{1}(zI - A)^{-1} \mathbf{1} = 1/z\} \\ &= \alpha - \{z: \mathbf{1}(zI - A)^{-1} \mathbf{1} = 1/z\} = \alpha - \Lambda(A).\end{aligned}$$

The theorem readily follows by composing these two results. \square

General similarity transformations preserve eigenvalues but not pseudoeigenvalues. However, a simple induction property holds in the pseudospectra case.

Theorem 7.9.2. If $B = x - 1AX$, then $\Lambda(B) = \Lambda(x^{-1}Ax)(A)$.

Proof. If $z \in \Lambda(B)$, then

$$|z - x| = |z - x - 1Ax + 1Ax| = |(z - A)x - 1Ax| \leq \|x\| \|z - A\| \leq \|x\| \|\Lambda(A)\|$$

from which the theorem follows. \square

Corollary 7.9.3. If $X \in \mathbb{C}^{n \times n}$ is unitary and $A \in \mathbb{C}^{n \times n}$, then $\Lambda(X - 1AX) = \Lambda(A)$.

Proof. The proof is left as an exercise. \square

The pseudospectrum of a diagonal matrix is the union of ϵ -disks.

Theorem 7.9.4. If $D = \text{diag}(\lambda_1, \dots, \lambda_n)$, then $\Lambda(D) = \{\lambda_1, \dots, \lambda_n\} + \mathbb{C}\epsilon$.

Proof. The proof is left as an exercise. \square

Corollary 7.9.5. If $A \in \mathbb{C}^{n \times n}$ is normal, then $\Lambda(A) = A(A) +$

P r o f. Since A is normal, it has a diagonal Schur form $Q^*AQ = \text{diag}(A_1, \dots, A_n) = D$ with unitary Q . The proof follows from Theorem 7.9.4. \square

If $T = (T_{ij})$ is a 2by2 block triangular matrix, then $A(T) = A(T_{11}) \cup A(T_{22})$. Here is the pseudospectral analog

Theorem 7.9.6. If

$$T = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix}$$

with square diagonal blocks, then $\Lambda(T) = \Lambda(T_{11}) \cup \Lambda(T_{22}) = \Lambda(T)$.

P r o f. The proof is left as an exercise. \square

Corollary 7.9.7. If

$$T = \begin{bmatrix} T_{11} & 0 \\ 0 & T_{22} \end{bmatrix}$$

with square diagonal blocks, then $\Lambda(T) = \Lambda(T_{11}) \cup \Lambda(T_{22})$.

P r o f. The proof is left as an exercise. \square

The last property in our gallery of facts connects the resolvent $(z_0I - A)^{-1}$ to the distance that separates z_0 from $\Lambda(A)$.

Theorem 7.9.8. If $z_0 \in \mathbb{C} \setminus \Lambda(A)$, then

$$\text{dist}(z_0, \Lambda(A)) = \frac{1}{\| (z_0I - A)^{-1} \|_F}. \quad \square$$

P r o f. For any $z \in \Lambda(A)$ we have from Corollary 244 and (7.9.6) that

$$\epsilon = \sigma_{\min}(zI - A) = \sigma_{\min}((z_0I - A) \cdot (z - z_0)I) = \sigma_{\min}(z_0I - A) - \|z - z_0\|$$

and thus

$$\|z - z_0\| = \frac{1}{\| (z_0I - A)^{-1} \|_F}. \quad \square$$

The proof is completed by minimizing over all $z \in \Lambda(A)$. \square

A95 PB Ø à i i à Üí» ßøé àì è ·þ

The production of a pseudospectral contour plot such as those displayed above requires sufficiently accurate approximations of $\sigma_{\min}(zI - A)$ on a grid that consists of (perhaps)

1000s of z-values. As we will see in §86 the computation of the complete SVD of a n-by-n dense matrix is an $O(n^3)$ endeavor. Fortunately, steps can be taken to reduce each grid point calculation to $O(n^2)$ or less by exploiting the following ideas.

1. Avoid SVD-type computations in regions where $\sigma_{\min}(zI - A)$ is slowly varying. See Gallestey (1998).
2. Exploit Theorem 7.96 by ordering the eigenvalues so that the invariant subspace associated with $A(T_1)$ captures the essential behavior of $(zI - A)^{-1}$. See Reddy, Schmid, and Henningson (1998).
3. Precompute the Schur decomposition $QHQ^T = T$ and apply a fast algorithm that is efficient for triangular matrices. See Lui (1997).

We offer a few comments on the last strategy since it has much in common with the condition estimation problem that we discussed in §354. The starting point is to recognize that since Q is unitary,

$$\sigma_{\min}(zI - A) = \sigma_{\min}(zI - T).$$

The triangular structure of the transformed problem makes it possible to obtain a satisfactory estimate of $\sigma_{\min}(zI - A)$ in $O(n^2)$ flops. If d is a unit 2-norm vector and $(zI - T)y = d$ then it follows from the SVD of $zI - T$ that

$$\sigma_{\min}(zI - T) = \|y\|_2.$$

Let u_{\min} be a left singular vector associated with $\sigma_{\min}(zI - T)$. If d is has a significant component in the direction of u_{\min} , then

$$\sigma_{\min}(zI - T) \approx \frac{1}{\|y\|_2}.$$

Recall that Algorithm 35.1 is a cheap heuristic procedure that dynamically determines the right hand side vector d so that the solution to a given triangular system is large in norm. This is tantamount to choosing d so that it is rich in the direction of u_{\min} . A complex arithmetic, 2-norm variant of Algorithm 35.1 is outlined in P7.9.13. It can be applied to $zI - T$. The resulting d-vector can be refined using inverse iteration idea, see Trefethen and Trefethen (1996) and §8.2.2. Other approaches are discussed by Wright and Trefethen (2001).

Ch2nb .k3Df Tt'b f)\$b-i.e \$Dtke3 \$ff.t Plx efTeetbtttb 9tTDeb

The **pseudospectral abscissa** of a matrix $A \in \mathbb{C}^{n \times n}$ is the rightmost point on the boundary of $\Lambda(A)$.

$$\alpha(A) = \max_{z \in \Lambda_\epsilon(A)} \operatorname{Re}(z). \quad (7.98)$$

Likewise, the **pseudospectral radius** is the point of largest magnitude on the boundary of $\Lambda(A)$.

$$p(A) = \max_{z \in \Lambda_\epsilon(A)} \|z\|. \quad (7.99)$$

These quantities arise in the analysis of dynamical systems and effective iterative algorithms for their estimation have been proposed by Burke, Lewis, and Overton (2003) and Mengi and Overton (2005). A complete presentation and analysis of their very clever optimization procedures, which build on the work of Byers (1988), is beyond the scope of the text. However, at their core they involve interesting intersection problems that can be reformulated as structured eigenvalue problems. For example, if $i \cdot r$ is an eigenvalue of the matrix

$$M = \begin{bmatrix} ie^{i\theta} A^H & -E \\ \epsilon I & ie^{-i\theta} A \end{bmatrix}, \quad (7.910)$$

then E is a singular value of $A - re^{i\theta}I$. To see this, observe that if

$$\begin{bmatrix} ie^{i\theta} A^H & -E \\ \epsilon I & ie^{-i\theta} A \end{bmatrix} \begin{bmatrix} f \\ g \end{bmatrix} = i \cdot r \begin{bmatrix} f \\ g \end{bmatrix},$$

then

$$(A - re^{i\theta}I)^H (A - re^{i\theta}I)g = E^2 g.$$

The complex version of the SVD (§2.44) says that E is a singular value of $A - re^{i\theta}I$. It can be shown that if $i r_{\max}$ is the largest pure imaginary eigenvalue of M , then

$$\epsilon = \sigma_{\min}(A - r_{\max} e^{i\theta} I).$$

This result can be used to compute the intersection of the ray $\{re^{i\theta} : R \neq 0\}$ and the boundary of $\Lambda(A)$. This computation is at the heart of computing the pseudospectral radius. See Mengi and Overton (2005).

818 Do sfzrPwiflr or $\text{s3r h0Pl i,w l. i8Fo rroz lr}$

At the start of this section we used the example

$$A = \begin{bmatrix} 0.999 & 1000 \\ 0000 & 0.998 \end{bmatrix}$$

to show that $\|A^k\|_2$ can grow even though $p(A) \approx 1$. This kind of transient behavior can be anticipated by the pseudospectral radius. Indeed, it can be shown that for any $f > 0$

$$\sup_{k \geq 0} \|A^k\|_2 \leq M \frac{p(A) + 1}{f}. \quad (7.911)$$

See Trefethen and Embree (SAP, pp. 160–161). This says that transient growth will occur if there is a contour $\{z\|z\| - A\| \leq 1/f\}$ that extends beyond the unit disk. For the above 2-by-2 example, if $E = 10^{-8}$, then $p(A) = 1.0017$ and the inequality (7.911) says that for some k , $\|A^k\|_2 \geq 17 \times 10^7$. This is consistent with what is displayed in Figure 7.91.

Problems

- P7.9.1 .86 2.2 .N2N8.2 fro5.5P51.585 fro1. ..N0r....N.2a
 P7.9.2 .8.N88.8es...ro1.5rob1
 P7.9.3 .8.N 3N8.N81 5P
 P7.9.4 .8.N .N8.N861.181
 P7.9.5 .8.N88.8es...ro1.1ro1
 P7.9.6 38@.2 .> R ∈ a^{nxn}, C9 Zl + E) W^{2,E}
 P7.9.7 ...8N_{Omni(z1I, r . G_{e1} , β}) €2 r E(29C2nCE 90I E21) = 7,9E
 (. rC2T_{z3} = f N_{μ1 + μ2 C9} _{Om_{u(z3)}}, r . = (€ R €)/22
 P7.9.8 ...8.N ∈ a^{nxn}T)(E72 2x E ∈ a^{nxn}T)))(E72))R9 2x E(29 2C9E97 2 (= I Zl + E).
 P7.9.9 P....2.N8..N.2.8. .N26NN. .N8.N861.50 ..NP..N.1fe .N8.m8 .N8.N810105
 P7.9.10 8N .N8...P ∈ R^{2nx2n,1} T 1

$$= \begin{bmatrix} -x \\ A \end{bmatrix} \mathbf{h}$$

$\mathbf{E} \rightarrow \mathbf{H} \in R^2 \times \mathbb{R}^2 Q_2 I_2 O_1 T_2$ $\text{ZET To Cn} = \cdot$. OO xx x= = , z m
 $\mathbf{x} = \mathbf{I} \mathbf{x} \mathbf{x} \mathbf{m} \mathbf{x} \mathbf{x} \mathbf{W} \mathbf{(c4)} \in W^{l_{c,d}} \mathbf{T}_c \mathbf{\lambda}^{TT_c} \mathbf{(\nabla)} \mathbf{l} \in W^{l_A} \mathbf{A}_3 \in W$
 $W^{-} \mathbf{(r} \Sigma V^T T_n \mathbf{)} \in R^2 \times \mathbb{R}^2 Q_2 I_2 O_1 D_a$ $\text{ZET Ocr} = rr. f lr 2Ct Ll, ml. pmih. 1 nt$
 $, tr Sp2AheA \in W A_{(k3}}, \in W^{l_{c,d}} \mathbf{\lambda}^{++c.} \mathbf{(\nabla)} \mathbf{f} \in W^{l_A} A_{(k3)} \mathbf{V} \mathbf{/} \in W$.

- P7.9.11 ro...88N. . 8.es8.. .N.N. .8...N N.. .eses1.8... .N.N.N...2NB. ... 3.
.2N.N. .N.8.N.... .8.N.2N.8.N

$$I = \begin{bmatrix} 0 & 1 & \cdots & | \\ \alpha & [& \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 \\ 0 & \cdots & 0 & = \end{bmatrix}.$$

$2 = i - a$ $x = -x = -2$, $x = -x \in \mathbb{R}$ $a = -x - x = -2r$. If $a = -x - x = -2r$, then $x = -r$. Or $x = -r$.

- P7.9.12 1 8.2.p** $\in \text{anxnOstable r ipg r p}$, $2nr2aiing2a2riLr M2SgMpAla,ne2Mp2$
 $\text{SMISg2rr naftmpe } 12_2 = \text{f91}(\text{I } nC_9(nE2) \text{lnC21LR E} (\nabla_{3,4} + i_{3,4} - \nabla + c_3, i_{3,4}, T_* \Sigma \nabla + T_* \nabla +,$
 $\nabla_n, d)^T \nabla_{3,4}^T, (T, c, T_* \Sigma^2 \nabla^T c_3)^T c_3 - T_* \Sigma^T \Sigma, -\nabla + i_{3,4} c_3 \Sigma^T 3, \Sigma^2 \text{U}_{1n}, (\text{irl}, r, l)^2 \text{Mgg} \in \mathbb{R}.$
 $r_F E_*^T, \nabla \Sigma_{3,4}^T \Sigma^{2T} E, i_{3,4} | E 12a2, 9 E9E2x i 79 = E9(\text{ol}), 9E9 \text{ n(O, n2TpK1).}$
 $T_* L20(9anO), 9199 \text{ l, } 9899 \text{ n(O, n2010x ZIXI}$

- P7.9.13 . .8.esN8 -8..N..N.. N.. 8..88~~2~~.N8..8.8N 8≥ 8.2..
...2 ...88.2.2.8. 2.2 ..N.N.8N. 8.N.. 8.N... 2.N . 8..N> ..es.... 3N.N
-8.N.2.8 8≥ 8.. pre.es..38> N.... 8. ro1. PResN23N .8 N..8.N 23NB.. 2
..... N 8> -N. 2. 8.2..p U G T- zl .mrM2r pmfAmIMrl r E Rnxn The
E= = Ou*= OaTfaOaO= a0 = A.> aL= 3a U= O.= S-u=ap= u= -> u= T= =
,,(('(N. .t, c^T, T_c) .c Uy= d ea a< a * y. e==TGe,n(U) Vly l2 Z2X II(9

$$U = \begin{bmatrix} W & \vdots \\ \vdots & U \end{bmatrix} \quad y \in \begin{bmatrix} \cdot \\ \vdots \\ \cdot \end{bmatrix} \quad d \in \begin{bmatrix} \cdot \\ \vdots \\ \cdot \end{bmatrix}$$

$\exists u, v \in A^*, u, v, d_1 \in a^{n-1}, U_1 \in \langle (n-1)x(n-1) \rangle$ $\vdash_{\text{di}} \exists U_1, U_1 y_1 G \quad b \text{iae} c^2 + s^2 = 0$.
 $\exists x \in C_u, \exists x \in \exists x \exists d_2 \exists x \exists D_x \exists x \exists y_1 \exists U_1 = \exists y_1 \exists O_i \exists D_1 H_1 T_1 \exists x_1 T_1 \backslash$
 $\{CTQ23, \exists i \exists p \exists A \exists Z \exists M Q1 \exists x \exists y_2 \exists p(1,2) \exists E_1 \exists T_2 \exists C(x \exists y_2 \exists R$
 $\text{Orn}(U(k:n, k:n)) \text{ f r k = n: - } 1N.$

Notes and References for 7.7

- PN. .N.N 2.N. 8.NN 1 5.N N8esg6N.. ..8. .N . ..N8...2.8.8 23Npseudo**

- M. tTrAnan =SQ m. < Jn 1 2p z ln = n Aa -' Oyn 0p0ay p An Xon k0= of n E= Aa
 ET0=SIAM J. Sci. Comput. 23, d5r .Ji
 L.N. - n. = JI0=opzxon p 0of n E=n Aa = SIAM J. Sci. Comput. 18, 5d5G, J7.
- ur lule1pin0laA lMapasAlr SK p iplactSgt l. St1Ke1lnr 1al > I 111
- ZZ leetuh9MA2neiae EZslManart,a,N.5. Mht1KeltS 1Apuipm,1As, .1.1e ,S lu2
 ipluxXIAM J. Applic. Math. 53, NG,TZ
- s. H. Sp 0M0p ze= Tfp = a Xo0p0Cofme= Ea 0Ok=SIAM J. Sci. Comput. 18,
 7N,T5.
- dZ ig1pt, N.d.M"Alr SK p s1ApuiB1UI \$lptr tr.rpm\$1Smu.lanArptpm2l.r l.
 G.pai>nipunA lBT, 38, II r 55M
- L.N. n. = • M22, z= Xfp = a Xo0k0= x fA= America N I, T, I...
 nZ Z.nrmmp= JI.. Eigtod, mpS1m. iA>i SMIABnSK etS 1Ap.in pM
 r.u1pnalbplatnlatnr la1i>n.ipn,atniSS>nAp121t11 elts 1ApuielinaA>Iell
- L. OE• n<-=SQ m U n = JI22+ p « Oyn = A1000 p 0.n=Oya = r p m 0Hb = AOEn o
 Lin. Alg. Applic. 164-164, r. 5rNdM
- 5M. aliae nZ1ymla = N. . ht1l el. lult, . hl>tal.ni>tiae ht1l etS 1Apui Alr Sianla
 4pnaAltumer. Math. 68, J5v,I..
- O5ppia1m.. Z"snarK ggi>K It Al.Sianla nip.nA itae wlKaeta -1,t l. hl>talr nigtu.
 SIAM J. Matr. Anal. Applic. 16 55r 5.JM
- N.J. i Oy• a XOp0o0m22, zy)< Ce< y= AOE HaT = C0A x- 0T a = 30= a =
 yffOEa= 0= ln = Ex0n p 0= of NEM AMatrix Anal. Applic. 21, NNdNJNM
 mM. Minrige nM21p21a ,IJJ.M"ht1l etS 1Apil. 11ApiaRigCipnA1tuX J. Numer.
 Anal. 22, JN.N..
- R. ya Ta =0= A1220 pzi=0= a -n Oyn = On E= KfHa00= EOEAO. Matrix Anal.
 Applic. 26, d5J ,dl
- as2eltS 1ApuiSiS 1tp2ip ulp1pllpnlar AlapuggiSniaipiSngp1una1tp1t naA>Iell
- J.V.)p E.ra =yQ S1 30ca =10+ iS bl A= 122 p z if XoBa = a 0Xen k0= of fCE= •Aa
 yffOEa =0= a n0ka0= JemH x Anal. Applic. 25, dJr rJ..
- 9B9WUB EMsMintiae.n.nM1pl. =J.B.M11SI1tp spiSngi p i Au2tGAuB1Ur.lpr lu
 hs 1l elts 1ApuiXMA J. Numer. Anal. 23, 5,r 5T.M
- 92B1J.B lu 3Zs. n1mZ1apl ,IJJ. Mht1l elts 1Ap.i0l.S.alapt iae pmEntpiaAp
 Alapl>giSngp1mX. Matr x Anal. Applic. 26, 5Jr 5XZ
- The t= 30= fa fnaAE = EnID= n0En Xfp = a = OpTn A00p of n E=EEa00p o=
-) In <0 a < p n o,
- (+ i n aUy ka = M22p z y=y< y= E0A eCXfp = Ory p Xn Ap120p0mMj.
 Numer. Anal. 17, 5I.r 5IM
- PM,3Z tla ,r. j 7.MAI.S1pnapm2K1nAi>lienK tX.in. Alg. Applic. 234, N5NIT
- T. ,A1= - =0nEkuikOy• 1r 2p z e= Xfp = OR0n0 a pan eX0Op OCofn E0Eya
 (- EH6e •300e = = 0= kAH36, II „JZ
- E. ,n- y0HUSx nA= 1220 p z y< y= E0= eX0Xfk= a = 0Xen p OCofj(E0Ea <
 (= 0= nkXn E0Ea0Opaa = H0d J. Numer. Anal. 25, 7,dr 77.M
- N. lpy< Oa X00i> n E= 120 rp R .yy= E0=tE T-oyfifE= dOXa =•Xen p 0= of n E=
- ynw EO0Xen p 0= of jaEO0paA= EO0A0N. Matr x Anal. Applic. 32, N7N.I.
- F r.= A0= o0y• =•am• a =>Ta EfEOn Aa=n,
- +. n = E0Op z)= p =E0o= n Aax= >n0Opmd=Eaa E0Ca =R0On< 0a <=p6-o B
 - FEXa < Name: MOEn b= JIM
- 99f WIbr. 75M'wl act lu psl1Ap.i2lu. l. KaApnlat nipunA1tnXmer. Math. 5,
 N r, JM
- T. - tEOM20 p z i=Xon p OCofan E0Ea1A= 3:=SIAM J. Matr x Anal. Applic. 29,
 7 NM
- As (- nd xi = 3 afen p OCofE= n= n-J0y• o=eEp EXp EnE0Em,o=
- L. 10E• a <=SQ m <= n = M2= p U Oyn =L=aOpn p 0.nOpnp Open f < 0la b= E0En o
 Lin. Alg. Applic. 162/163/164, r. 5rNd7M

Chapter 8

Symmetric Eigenvalue Problems

yPe

yPl

yUn

yPo

yPr

yPs

yU

The symmetric eigenvalue problem with its rich mathematical structure is one of the most aesthetically pleasing problems in numerical linear algebra. We begin with a brief discussion of the mathematical properties that underlie the algorithms that follow. In §8.2 and §8.3 we develop various power iterations and eventually focus on the symmetric QR algorithm. Methods for the important case when the matrix is tridiagonal are covered in §8.4. These include the method of bisection and a divide and conquer technique. In §8.5 we discuss Jacobi's method, one of the earliest matrix algorithms to appear in the literature. This technique is of interest because it is amenable to parallel computation and because of its interesting high accuracy properties. The computation of the singular value decomposition is detailed in §8.6. The central algorithm is a variant of the symmetric QR iteration that works on bidiagonal matrices.

In §8.7 we discuss the generalized eigenvalue problem $Ax = \lambda Bx$ for the important case when A is symmetric and B is symmetric positive definite. The generalized singular value decomposition $ATAx = \mu A^T B^T Bx$ is also covered. The section concludes with a brief examination of the quadratic eigenvalue problem $(\lambda^2 M + \lambda C + K)x = 0$ in the presence of symmetry, skewsymmetry, and definiteness.

Reading Notes

Knowledge of Chapters 1–3 and §§1–§5.2 are assumed. Within this chapter there are the following dependencies:

§84

 $\overset{n}{\underset{1}{\text{§81}} \cdots \text{§82} \cdots \text{§83} \cdots \text{§86} \cdots \text{§87}}$
 $\overset{a}{\underset{5}{\text{§85}}}$

Many of the algorithms and theorems in this chapter have unsymmetric counterparts in Chapter 7. However, except for a few concepts and definitions, our treatment of the symmetric eigenproblem can be studied before reading Chapter 7.

Complementary references include Wilkinson (AEP), Stewart (MAE), Parlett (SEP), and Stewart and Sun (MPA).

8.1 Properties and Decompositions

In this section we summarize the mathematics required to develop and analyze algorithms for the symmetric eigenvalue problem.

imhhhb b1QSf t PDShttb b1QS f Sf kxeb

Symmetry guarantees that all of A' 's eigenvalues are real and that there is an orthonormal basis of eigenvectors.

Theorem 8.1.1 (Symmetric Schur Decomposition). If $A \in J_{n,n}$ is symmetric, then there exists a real orthogonal Q such that

$$Q^T A Q = A = \text{diag}(\lambda_1, \dots, \lambda_n).$$

Moreover, for $k = 1:n$, $AQ(:,k) = \lambda_k Q(:,k)$. Compare with Theorem 7.1.3.

Proof. Suppose $\lambda_i \in \sigma(A)$ and that $x \in \mathbb{C}^n$ is a unit 2-norm eigenvector with $Ax = \lambda_i x$. Since $\lambda_i = x^H Ax = x^H A^H x = \overline{x^H Ax} = \overline{\lambda_i}$ it follows that $\lambda_i \in J$. Thus we may assume that $x \in J_{n,n}$. Let $P_i \in J_{n,n}$ be a Householder matrix such that $P_i^T x = e_1 = I_n(:,1)$. It follows from $Ax = \lambda_i x$ that $(P_i^T A P_i)e_1 = \lambda_i e_1$. This says that the first column of $P_i^T A P_i$ is a multiple of e_1 . But since $P_i^T A P_i$ is symmetric, it must have the form

$$P_i^T A P_i = \begin{bmatrix} A_{11} & Q \\ 0 & A_{22} \end{bmatrix}$$

where $A_{11} \in J(n-1, n-1)$ is symmetric. By induction we may assume that there is an orthogonal $Q_i \in J(n-1, n-1)$ such that $Q_i^T A_{11} Q_i = \Lambda_{11}$ is diagonal. The theorem follows by setting

$$Q = P_i \begin{bmatrix} 1 & 0 \\ 0 & Q_i \end{bmatrix} \quad \text{and} \quad A = \begin{bmatrix} \lambda_i & 0 \\ 0 & A_{22} \end{bmatrix}$$

and comparing columns in the matrix equation $AQ = QA$.

For a symmetric matrix A we shall use the notation $\lambda_k(A)$ to designate the k th largest eigenvalue, i.e.,

$$\lambda_n(A) \leq \dots \leq \lambda_2(A) \leq \lambda_1(A).$$

It follows from the orthogonal invariance of the 2-norm that A has singular values $\{J_1(A), J_2(A), \dots, J_n(A)\}$ and

$$\|A\|_2 = \max\{\|A_1(A)\|_2, \|A_2(A)\|_2, \dots, \|A_n(A)\|_2\}.$$

The eigenvalues of a symmetric matrix have a minimax characterization that revolves around the quadratic form $x^T A x / x^T x$.

Theorem 8.1.2 (Courant-Fischer Minimax Theorem). If $A \in \mathbb{R}^{n \times n}$ is symmetric, then

$$\lambda_k(A) = \max_{\dim(S)=k} \min_{0 \neq y \in S} \frac{y^T A y}{y^T y}$$

for $k = 1:n$.

Proof. Let $Q^T A Q = \text{diag}(i)$ be the Schur decomposition with $i_k = \lambda_k(A)$ and $Q = [q_1 | q_2 | \dots | q_n]$. Define

$$S_k = \text{span}\{q_1, \dots, q_k\},$$

the invariant subspace associated with $\lambda_1, \dots, \lambda_k$. It is easy to show that

$$\max_{\dim(S)=k} \min_{0 \neq y \in S} \frac{y^T A y}{y^T y} \geq \min_{0 \neq y \in S_k} \frac{y^T A y}{y^T y} = q_k^T A q_k = \lambda_k(A).$$

To establish the reverse inequality, let S be any k -dimensional subspace and note that it must intersect $\text{span}\{q_1, \dots, q_k\}$, a subspace that has dimension $n - k + 1$. If $y^* = a_1 q_1 + \dots + a_k q_k + Q_{k+1} v$ is in this intersection, then

$$\min_{0 \neq y \in S} \frac{y^T A y}{y^T y} \leq \frac{y_*^T A y_*}{y_*^T y_*} \leq \lambda_k(A).$$

Since this inequality holds for all k -dimensional subspaces,

$$\max_{\dim(S)=k} \min_{0 \neq y \in S} \frac{y^T A y}{y^T y} \leq \lambda_k(A)$$

thereby completing the proof of the theorem. \square

Note that if $A \in \mathbb{R}^{n \times n}$ is symmetric positive definite, then $\lambda_k(A) > 0$.

Q10) If $A \in \mathbb{R}^{n \times n}$, then $\|A\|_F^2 = \text{tr}(A^T A)$.

An important solution framework for the symmetric eigenproblem involves the production of a sequence of orthogonal transformations $\{Q_k\}$ with the property that the matrices $Q_k^T A Q_k$ are progressively "more diagonal." The question naturally arises: how well do the diagonal elements of a matrix approximate its eigenvalues?

Theorem 8.1.3 (Gershgorin). Suppose $A \in \mathbb{R}^{n \times n}$ is symmetric and that $QE = Q^T A Q$, where Q is orthogonal. If $Q^T A Q = D + F$ where $D = \text{diag}(d_1, \dots, d_n)$ and F has zero diagonal entries, then

$$\sigma(A) \subset \bigcup_{i=1}^n [d_i - r_i, d_i + r_i]$$

where $r_i = \sum_{j=1}^n |f_{ij}|$ for $i = 1:n$. Compare with Theorem 7.2.1.

Proof. Suppose $\lambda \in \sigma(A)$ and assume without loss of generality that $\lambda = d_i$ for some $i = 1:n$. Since $(D - \lambda I) + F$ is singular, it follows from Lemma 2.3.3 that

$$1 \leq \| (D - \lambda I)^{-1} F \|_\infty = \sum_{j=1}^n \frac{|f_{kj}|}{|d_k - \lambda|} = \frac{r_k}{|d_k - \lambda|}$$

for some $k = 1:n$. But this implies that $\lambda \in [d_k - r_k, d_k + r_k]$. \square

The next results show that if A is perturbed by a symmetric matrix E , then its eigenvalues do not move by more than $\|E\|_F$.

Theorem 8.1.4 (Wielandt-Hoffman). If A and $A + E$ are $n \times n$ symmetric matrices, then

$$\sum_{i=1}^n (\lambda_i(A + E) - \lambda_i(A))^2 \leq \|E\|_F^2.$$

Proof. See Wilkinson (AEP, pp. 104–108), Stewart and Sun (MPT, pp. 189–191), and Lax (1997, pp. 134–136). \square

Theorem 8.1.5. If A and $A + E$ are $n \times n$ symmetric matrices, then

$$\lambda_k(A) + \lambda_n(E) \leq \lambda_k(A + E) \leq \lambda_k(A) + \lambda_1(E), \quad k = 1:n$$

Proof. This follows from the minimax characterization. For details see Wilkinson (AEP, pp. 101–102) or Stewart and Sun (MPT, p. 203). \square

Corollary 8.1.6. If A and $A + E$ are $n \times n$ symmetric matrices, then

$$|\lambda_k(A + E) - \lambda_k(A)| \leq \|E\|_2$$

for $k = 1:n$.

Proof. Observe that

$$|\lambda_k(A + E) - \lambda_k(A)| \leq \max\{|\lambda_n(E)|, |\lambda_1(E)|\} = \|E\|_2$$

for $k = 1:n$. \square

pair of additional perturbation results that are important follow from the minmax property.

Theorem 8.1.7 (Interlacing Property). If $A \in \mathbb{R}^{n \times n}$ is symmetric and $A_r = A[1:r, 1:r]$, then

$$A_{r+1}(A_{r+1}) \leq A_r(A_r) \leq A_r(A_{r+1}) \leq \cdots \leq A_2(A_{r+1}) \leq A_1(A_r) \leq A_1(A_{r+1})$$

for $r = 1:n-1$

P f. Wilkinson (AEP, pp 103 104). D

Theorem 8.1.8. Suppose $B = A + \tau c c^T$ where $A \in \mathbb{R}^{n \times n}$ is symmetric, $c \in \mathbb{R}^n$ has unit 2-norm, and $\tau \in \mathbb{R}$. If $\tau \neq 0$ then

$$\lambda_i(B) \in [\lambda_i(A), \lambda_{i-1}(A)], \quad i = 2:n,$$

while if $\tau = 0$ then

$$\lambda_i(B) \in [\lambda_{i+1}(A), \lambda_i(A)], \quad i = 1:n-1.$$

In either case, there exist nonnegative $r \geq 1, \dots, m_n$ such that

$$\lambda_i(B) = \lambda_i(A) + m_i \tau, \quad i = 1:n$$

with $m_1 + \cdots + m_n = 1$

P f. Wilkinson (AEP, pp 94 97). See also P818 /a

All $r \leq n$ since $\lambda_r \leq \lambda_1$

If $S \subseteq \mathbb{R}^n$ and $x \in S$, $Ax \in S$, then S is an invariant subspace for $A \in \mathbb{R}^{n \times n}$. Note that if $x \in \mathbb{R}^n$ is an eigenvector for A , then $S = \text{span}\{x\}$ is 1-dimensional invariant subspace. Invariant subspaces serve to "take apart" the eigenvalue problem and figure heavily in many solution frameworks. The following theorem explains why.

Theorem 8.1.9. Suppose $A \in \mathbb{R}^{n \times n}$ is symmetric and that

$$Q = \begin{bmatrix} Q_1 & Q_2 \\ r & n-r \end{bmatrix}$$

Q is orthogonal. If $\text{im}(Q)$ is an invariant subspace, then

$$Q A Q = D = \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix}_{n-r} \quad (811)$$

and $\sigma(A) = \sigma(D_1) \cup \sigma(D_2)$. Compare with Lemma 7.12

Pr of. If

$$Q^T A Q = \begin{bmatrix} D_1 & E \\ E^T & D_2 \end{bmatrix},$$

then from $AQ = QD$ we have $AQ_1 - QD_1 = Q_2 \mathbf{2}$. Since $\text{ran}(Q)$ is invariant, the columns of QE_2 are also in $\text{ran}(Q)$ and therefore perpendicular to the columns of Q_2 . Thus,

$$0 = Q(AQ_1 - QD_1) = QQE_2 = E_2.$$

and so (81.1) holds. It is easy to show

$$\det(A - r) = \det(Q^T A Q - r) = \det(D_1 - r) \cdot \det(D_2 - r)$$

confirming that $\sigma(A) = \sigma(D_1) \cup \sigma(D_2)$. \square

The sensitivity to perturbation of an invariant subspace depends upon the separation of the associated eigenvalues from the rest of the spectrum. The appropriate measure of separation between the eigenvalues of two symmetric matrices B and C is given by

$$\text{sep}(B, C) = \min_{\substack{\lambda \in \sigma(B) \\ \mu \in \sigma(C)}} \|P - \mu I\|_F \quad (81.2)$$

With this definition we have the following result.

Theorem 8.1.10. Suppose A and $A + E$ are $n \times n$ symmetric matrices and that

$$Q = [Q_1 \mid Q_2] \quad \begin{matrix} r \\ r \\ \mathbf{n} \\ \mathbf{n} \end{matrix}$$

is an orthogonal matrix such that $\text{ran}(Q)$ is an invariant subspace for A . Partition the matrices $Q^T A Q$ and $Q^T E Q$ as follows

$$Q^T A Q = \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix}_{n-r} \quad , \quad Q^T E Q = \begin{bmatrix} E_{11} & E_{21}^T \\ E_{21} & E_{22} \end{bmatrix}_{r \times r}.$$

If $\text{sep}(D_1, D_2) > 0$ and

$$\|E\|_F \leq \frac{\text{sep}(D_1, D_2)}{5},$$

then there exists a matrix $P \in \mathbb{R}^{(n-r) \times r}$ with

$$\|P\|_F \leq \frac{4}{\text{sep}(D_1, D_2)} \|E\|_F$$

such that the columns of $Q_1 = (Q_1 + Q_2 P)(I + p^T P)^{-1/2}$ define an orthonormal basis for a subspace that is invariant for $A + E$. Compare with Theorem 7.2.4.

Pr of. This result is a slight adaptation of Theorem 4.11 in Stewart (1973). The matrix $(I + p^T P)^{-1/2}$ is the inverse of the square root of $I + p^T P$. See §4.24.

Corollary 8.1.11. If the conditions of the theorem hold, then

$$\text{dist}(\text{ran}(Q), \text{ran}(\hat{Q})) = \frac{4}{\text{sep}(D_1, D_2)} \|E\|_F$$

Compare with Corollary 7.25

Proof. It can be shown using the SVD that

$$\|P(I + P^T P)^{-1}\|_F \leq \|P\|_F \quad (8.13)$$

Since $Q \hat{Q} = P(I + P^T P)^{-1}$, it follows that

$$\begin{aligned} \text{dist}(\text{ran}(Q), \text{ran}(\hat{Q})) &= \|Q \hat{Q}\|_F = \|P(I + P^T P)^{-1}\|_F \\ &\leq \|P\|_F = \frac{4\|E\|_F}{\text{sep}(D_1, D_2)} \end{aligned}$$

completing the proof. \square

Thus the reciprocal of $\text{sep}(D_1, D_2)$ can be thought of as a condition number that measures the sensitivity of $\text{ran}(Q)$ as an invariant subspace.

The effect of perturbations on a single eigenvector is sufficiently important that we specialize the above results to this case.

Theorem 8.1.12. Suppose A and $A+E$ are $n \times n$ symmetric matrices and that

$$Q = [q_1 \mid Q_2]$$

is an orthogonal matrix such that Q is an eigenvector for A . Partition the matrices $Q^T A Q$ and $Q^T E Q$ as follows:

$$Q^T A Q = \begin{bmatrix} \lambda & 0 \\ 0 & D_2 \\ 1 & \mathbf{n} \end{bmatrix}_{n-1}, \quad Q^T E Q = \begin{bmatrix} \epsilon & e^T \\ e & E_{22} \\ 1 & \mathbf{n} \end{bmatrix}_{n-1}.$$

If

$$d = \min_{\mu \in \lambda(D_2)} |\lambda - \mu| > 0$$

and

$$\|E\|_F \leq \frac{d}{5}$$

then there exists $p \in \mathbb{R}^{n-1}$ satisfying

$$\|p\|_2 \leq \frac{4}{d} \|e\|_2$$

such that $\mathbf{1} = (q_1 + Q_2 p) / \sqrt{1 + \|p\|^2}$ is a unit 2-norm eigenvector for $A+E$. Moreover,

$$\text{dist}(\text{span}\{q_1\}, \text{span}\{\hat{q}_1\}) = \sqrt{1 - (q_1^T \hat{q}_1)^2} \leq \frac{4}{d} \|e\|_2.$$

Compare with Corollary 7.26

Pr of. Apply Theorem 8.1.10 and Corollary 8.1.11 with $r = 1$ and observe that if $D_1 = (\lambda)$, then $d = \text{sep}(D_1, D_2)$. D

)mPi) (qq l (: f X(p))i (X:X i0)(h(qXp0)

If the columns of $Q_1 \in \mathbb{R}^{n \times r}$ are independent and the residual matrix $R = A - AQ_1$, Q_1 is small for some $S \in \mathbb{R}^{r \times r}$, then the columns of Q_1 define an approximate invariant subspace. Let us discover what we can say about the eigensystem of A when in the possession of such a matrix.

Theorem 8.1.13. Suppose $A \in \mathbb{R}^{n \times n}$ and $S \in \mathbb{R}^{n \times n}$ are symmetric and that

$$AQ_1 - Q_1S = E_1$$

where $Q \in R^{n \times r}$ satisfies $Q^T Q = I_r$. Then there exist $\mu_1, \dots, \mu_r \in \lambda(A)$ such that

$$|\mu_k - \lambda_k(S)| \leq \sqrt{2} \|E_1\|_2$$

for k= l:r.

Pr of. Let $Q \in \mathbb{R}^{n \times (n-r)}$ be any matrix such that $Q = [Q_1 | Q_2]$ is orthogonal. It follows that

$$Q^T A Q = \begin{bmatrix} S & 0 \\ 0 & Q_2^T A Q_2 \end{bmatrix} + \begin{bmatrix} Q_1^T E_1 & E_1^T Q_2 \\ Q_2^T E_1 & 0 \end{bmatrix} \equiv B + E$$

and so by using Corollary 8.16 we have $|\lambda_k(A) - \lambda_k(B)| \leq \|E\|_F r k = \ln$. Since $\lambda(S) \subseteq \lambda(B)$, there exist $\mu_1, \dots, \mu_r \in \lambda(A)$ such that $|\mu_k - \lambda_k(S)| \leq \|E\|_F r k = \ln$. The theorem follows by noting that for any $w \in R^r$ and $y \in R^{N-r}$ we have

$$\left\| E \begin{bmatrix} x \\ y \end{bmatrix} \right\|_2 \leq \|E_1 x\|_2 + \|E_1^T Q_2 y\|_2 \leq \|E_1\|_2 \|x\|_2 + \|E_1\|_2 \|y\|_2$$

from which we readily conclude that $\mathbf{1E} \mathbf{12} = \mathbf{1E} \mathbf{11} \mathbf{12}$.

The eigenvalue bounds in Theorem 8.1.13 depend on $\|AQ_i - Q_iS\|_F$. Given A and Q_i , the following theorem indicates how to choose S so that this quantity is minimized in the Frobenius norm.

Theorem 8.1.14. If $A \in \mathbb{R}^{n \times n}$ is symmetric and $Q \in \mathbb{R}^{n \times r}$ has orthonormal columns then

$$\min_{S \in \mathbb{R}^{r \times r}} \|AQ_1 - Q_1S\|_F = \|(I - Q_1Q_1^T)AQ_1\|_F$$

and $S = Q \Lambda Q^{-1}$ is the minimizer.

f Let $Q \in \mathbb{R}^{n \times n}$ be such that $Q = [Q_1 \ Q_2]$ is orthogonal. For any $\mathbf{x} \in \mathbb{R}^n$ we have

$$\mathbf{1}^\top Q - Q \mathbf{s} \ \mathbf{1} = \mathbf{1}^\top Q A Q - Q^\top Q \mathbf{s} \ \mathbf{1} = \mathbf{1}^\top Q A Q - s \ \mathbf{1} + \mathbf{1}^\top Q A Q \ \mathbf{1}.$$

Clearly, the minimizing S is given by $S = Q^\top A Q$. \square

This result enables us to associate any r -dimensional subspace $\text{ran}(Q)$, with a set of r "optimal" eigenvalue eigenvector approximates

Theorem 8.1.15. Suppose $A \in \mathbb{R}^{n \times n}$ is symmetric and that $Q \in \mathbb{R}^{n \times r}$ satisfies $Q^\top Q = I_r$. If

$$\mathbf{z}^\top (Q^\top A Q) \mathbf{z} = \text{diag}(\dots, Q) = D$$

in the Schur decomposition of $Q^\top A Q$ and $QZ = [\mathbf{y}_1 | \dots | \mathbf{y}_r]$, then

$$\|Ay_k - \theta_k y_k\|_2 = \|(I - Q_1 Q_1^\top)AQ_1 Z e_k\|_2 \leq \|(I - Q_1 Q_1^\top)AQ_1\|_2$$

for $k = 1 \dots r$.

P f It is easy to show that

$$A y_k - Q_1 Q_1^\top A Q_1 Z e_k - Q_1^\top Z D e_k = (A Q_1 - Q_1 (Q^\top A Q)) Z e_k$$

The theorem follows by taking norms. \square

In Theorem 8.1.15, the θ_k are called Ritz values, the \mathbf{y}_k are called Ritz vectors, and the (θ_k, \mathbf{y}_k) are called Ritz pairs.

The usefulness of Theorem 8.1.13 is enhanced if we weaken the assumption that the columns of Q are orthonormal. As can be expected, the bounds deteriorate with the loss of orthogonality.

Theorem 8.1.16. Suppose $A \in \mathbb{R}^{n \times n}$ is symmetric and that

$$A \mathbf{x}_i - \mathbf{x}_i S = F_i,$$

where $\mathbf{x}_i \in \mathbb{R}^n$ and $S = X^\top A X$. If

$$\|\mathbf{x}_i^\top \mathbf{x}_i - I_r\|_F = 1, \quad (8.14)$$

then there exist $\mu_1, \dots, \mu_r \in \Lambda(A)$ such that

$$|\mu_k - \lambda_k(S)| \leq \sqrt{2} (\|F_1\|_2 + \tau(2 + \tau)\|A\|_2)$$

for $k = 1 \dots r$.

P f For any $Q \in \mathbb{R}^{n \times r}$ with orthonormal columns, define $E_i \in \mathbb{R}^{n \times r}$ by

$$E_i = A Q - Q S$$

It follows that

$$E_1 = A(Q - X_1) - (Q - X_1)S + F_1$$

and so

$$\|E_1\|_2 \leq \|F_1\|_2 + \|Q - X\|_2 \|A\|_2 \left(1 + \|X_1\|_2^2\right). \quad \{815\}$$

Note that

$$\|X_1\|_2^2 = \|X_1^T X_1\|_2 \leq \|X^T X_1 - I_r\|_2 + \|I_r\|_2 = 1 + \tau. \quad \{816\}$$

Let $U_1 V_1 \Sigma = \text{diag}(u_1, \dots, u_r)$ be the thin SVD of X_1 . It follows from {814} that

$$\|\Sigma^2 - I_r\|_2 = \tau$$

and thus $1 - u_i = 3 + \tau$. This implies

$$\|Q - X\|_2 = \|U_1 V_1 \Sigma - U_1 V_1 \Sigma\|_2 = \|I_r - \Sigma\|_2 = 1 - \frac{1}{r} \leq 1 - \frac{1}{3 + \tau} \quad \{817\}$$

The theorem is established by substituting {816} and {817} into {815} and using Theorem 8.1.13.

Exercise 8.1.16

The inertia of a symmetric matrix A is a triplet of nonnegative integers $(-z, p)$ where $-z$, 0 , and p are respectively the numbers of negative, zero, and positive eigenvalues.

Theorem 8.1.17 (Sylvester Law of Inertia). If $A \in \mathbb{R}^{n,n}$ is symmetric and $X \in \mathbb{R}^{n,n}$ is nonsingular, then A and $X^T A X$ have the same inertia.

Proof. Suppose for some r that $\lambda_r(A) > 0$ and define the subspace $S_0 \subset \mathbb{R}^n$ by

$$S_0 = \text{span}\{X^{-1} q_1, \dots, X^{-1} q_r\}, \quad \dot{\mathbf{q}} = \mathbf{0}$$

where $A \dot{\mathbf{q}} = \mathbf{i}(A) \dot{\mathbf{q}}$ and $i = 1:r$. From the minimax characterization of $\lambda_r(X^T A X)$ we have

$$\lambda_r(X^T A X) = \max_{\dim(S)=r} \min_{y \in S} \frac{y^T (X^T A X) y}{y^T y} \geq \min_{y \in S_0} \frac{y^T (X^T A X) y}{y^T y}.$$

Since

$$y \in \mathbb{R}^n \Rightarrow \frac{y^T (X^T X) y}{y^T y} \geq \sigma_n(X)^2 \quad y \in S_0 \Rightarrow \frac{y^T (X^T A X) y}{y^T (X^T X) y} \geq \lambda_r(A),$$

it follows that

$$\lambda_r(X^T A X) \geq \min_{y \in S_0} \left\{ \frac{y^T (X^T A X) y}{y^T (X^T X) y} \frac{y^T (X^T X) y}{y^T y} \right\} \geq \lambda_r(A) \sigma_n(X)^2.$$

An analogous argument with the roles of A and $X^T A X$ reversed shows that

$$\lambda_r(A) \geq \lambda_r(X^T A X) \sigma_n(X^{-1})^2 = \frac{\lambda_r(X^T A X)}{\sigma_1(X)^2}.$$

Thus $A(A)$ and $A(X^TAX)$ have the same sign and so we have shown that A and X^TAX have the same number of positive eigenvalues. If we apply this result to $-A$, we conclude that A and X^TAX have the same number of negative eigenvalues. Obviously, the number of zero eigenvalues possessed by each matrix is also the same. \square

transformation of the form $\mathbf{X}^T \mathbf{A} \mathbf{X}$, where \mathbf{X} is nonsingular, is called a congruence transformation. Thus, a congruence transformation of a symmetric matrix preserves inertia.

Problems

P8.1.1 f...8.. c... 8≥ N..... r... 8.R686... 8≥0 ... 88..c.
8...b 8... N...a

$$P8.1.2 \quad 388\dots \quad .8 \dots \quad N8.88.8\dots 84 \quad 8 \geq A = [:]$$

P8.1.3 ...S6 $\dots T^c \dots T^c \dots 8 \dots \dots \dots 8b. P.N8. \dots 8.N.b$ $(A^H)^2 \nabla_+ +_{21,c2} \nabla_X c_2 , \nabla((\dots)^{(k)c_2, \Sigma} \nabla_3 T^1$
 $(c+c^++\nabla_2, T_3) \nabla^T_T \dots \nabla^T_T c_3 / \nabla^T_V k \nabla_3 T^1) 2c_1 - c_2 b_3, (c_{221} c_{22} T_c) c_3 T^1 T_3 +_{21,c2} \nabla^T_T r^{++}, +_{\Sigma} \nabla^T_T \nabla_3 \dots +_{(T_c)} v | T^1 |$
 $+_{T_c} T^1 +^T_T (\nabla_-, \nabla_3 \nabla + c^T_{(3)} A T^T_{cc} T_c, \{T, \Sigma, \nabla, 2p\} =^T_3 A^T = -A, \dots, i A T^T_c =_{12} c_2 \Sigma^T_T \nabla^T_T \nabla_3 \in$

P8.1.4 ..86bx ER^{n × r}, r 1Cn, ,)- || X - X - 1 ||₂ = T < r ≈ • Ω_{min}(X) §- T.

$$\|z - z^*\|_F \leq \|z^T \nabla z - z^* \|^2_F \leq \|E\|_F$$

P8.1.6 3N8..1.88..8 7aPa5

P8.1.7 3N8..1..8N.8 7aa a

P8.1.8 3N8.8..8N.8 7a7.a. ... b..N.. 8≥.r. N.8..N.b.. 6. ..8 8≥.
o.8.a

P8.1.9 ..86 ... clB ER^{m × m})^t - C ER^{n × n})₁, 4
 - A2 x *ARC AT OPARK XER^{m × n}.

P8.1.10 3N8... 70 95

$$\mathbf{P8.1.11} \quad \text{...8. A } R^{n \times n} - \frac{1}{4} I_{(0N)}^T - C E R^{n \times r}, \quad R^r \left(R^T R \right)^{-1} - \frac{1}{c} \left(\frac{1}{c} \left(c_2 g_{t-3} - \nabla_{+}^T \right) + c_1 \right) A.$$

P8.1.12 N... 8N. 68rN.8.. 6. ros.

$$\begin{aligned} \text{rank}(S) &= 1 \\ S &\equiv \text{sr} \end{aligned}$$

$$\mathbf{1} = \mathbf{e}_n^T \mathbf{J} \mathbf{a} + \mathbf{e}_n^T \mathbf{E} \mathbf{R}^{n \times n} \mathbf{e}_n \quad \mathbf{1}^T = \mathbf{e}_n^T \mathbf{J}^T \mathbf{a} + \mathbf{e}_n^T \mathbf{E}^T \mathbf{R}^{n \times n} \mathbf{e}_n \quad \mathbf{S} = \mathbf{W}^T \mathbf{T} \mathbf{a} \quad \mathbf{S} = -\mathbf{W}^T \mathbf{J} \mathbf{1} \quad \mathbf{1}^T$$

P8.1.13 N...8N..8 rN.88. 68.8. .c84

$$8. \quad \|A - S\|_F = \text{rank}(S) = S - S^T$$

P8.1.14 Nr. 4 8p.8... 8b N8.9 ..19N8.. 8..N8r..N.8.... .r8.... 8N..8.8..R
 - 88.. 8Nd61.. 88oa

Notes and References for

1..1.1.r.28....1.0.2s f..d .s.1s 1P363... § 0.6
 3.0... P363...§ 8.R.6..s.R3163...5 P.. e.12.N
 s ... 6or

N§fa.6.. (ok. Iaiw333ne=83n3 n)=.etme=) j3 ,n),nAk) 2)rA=n)k=f =)lk3n=ew=12
 jnm83r nkdh SIAM Review 15, TITvT7,Z
 A,AZinr1NTZ.9nr1ii>I lt. h lupl uS 1unpn. n ipunA1Xin. Alg. Applic. d. NvNZ
 aZ 5mia. N. TZ.sS 1Apui 2liu,lu.npnianipunA1Xin. r c. AMS 48, NvTZ
 32Am,amiril T Z.3uSnpuhupl uSipn,atlr. n ipunA1Xin. Alg. Applic. 24, r.5 , r
 EZsA,ppNd,2... pmFAAI uiAt pm,lutmr,unAnuAgi1,u1. lu w,laenaapmsuLiJ. .
 1ig st. .lpnA n ipnAlg. Applic. 65, NTy,
 9Z., sl a. N. 2Z.3 2p1 wiAB.iuQuulh lupl uSipn,1tpm,lu.npnia 9nr laii>I lu,SgPN
 BIT 35, 5d,355Z
 -4E u.i .r. 7Z. 1lgipni 1ltnei>w,I.e t 1u p1t9nr ln ii>I lt, lur np n ipunA1Xin. Alg
 Applic. 244, r., r752
 -4E u.i i.e B M un. r. T2Z.l lgipni 1ltnei>w,I.aet.u amDnr 1iigl lt. i ,lur n ps1.2e1o
 nanpilpnuXSIAM J. Matrix Anal. Applic. 18, I Nl.Z
 1.A. nnr. dZ1lgipni 1 h lupl uSipn,1t 9fr 1iigl liae snarIgBigl BiunmpuX
 J. Matrix Anal. Applic. 19, „7.v.dZ
 IZ.AZin N. dZ.1l>ipni 1h lupl uSipram1,ut&f 9nr 1tSiA1e smal>ius1StSiABiunpn,tuN
 SIAM J. Matrix Anal. Applic. 20, TNv,IZ
 .ZnZESnABZ,u, iae 9MmZ > lui. IJJZ.a1tgatS1 1lgipni 1 lupl uSipn,taet lu 9nr 1
 ttp1r t ,lu.npnianipunA1Xin. Alg. Applic. 309, 5 rdZ
 9MmZ, i.e f vs>San .IJJZ „Spnr 1lgupl uSipnw,I.e t 1u pm,lu.npnia 9nr 1i>I
 hu,S>1uXin. Alg. Applic. 309, r. v, 5Z
 2Mliuiae 12AM1JJZ.3 tn,It am1,ulr lu .wiele „el= .np1, lur npn ipunA1Xin.
 Alg. Applic. 359, I 75vT7Z
 a. nn iae sl a.IJJZ.am1h lupl uSipnw,I.e t 19nr 1iigl lt 2luCignipunA1Xin
 Lin. Alg. 12 d „Z
 AM.5Ziae IZMAZ1JJZ.3 2pHa 9nr 1iigl lt. h lupl uS 1unpn. n ipu2A1Xin. Alg
 Applic. 395, rd5vN.JZ
 2. h miUJZ1lgipni 1ltnei>w,I.e t 1u 9nr 1iigl lt ,lu.npnianipunA1Xin. M. Matr
 Anal. Applic. 28, „v.7JZ
 3. lg1lapiutSu,, .. pm&n1>i.epM,1 iapml,u1ntrni 1 n.k
 hZ nNTZLinear Algebr , ang3tj TaplutA81aA,1uBM
 ,u A,a.1Apn,ap ,Spn.n.ipn,iae en1lu lapn1gipn,atn118
 hZE heuaZ2iaeiu i.e AZ a,1N.dZ „uen,iut En1lu1pn9g1ipn,ati.e prhst. 1pu2
 9nr 1aigl lu,Sg1uXSIAM J. Numer. Anal. 20, rv1IZ
 n ln,i lup,a .r. ddZ,nnan,marpmhiT nr 9fr 1aigl,1 i str r 1pu2A1Xin
 Anal. Applic. 9, I,71 7dZ
 aZ5gg iae ,M211elAB lu,TZ.am1EHuniipni 3a ,upm,r,a ignipun,T 9nr lai 1Ap,ut.
 st. ipunAni punbuX Alg. Applic. 264, d. .5Z

8.2 Power Iterations

Assume that $A \in \mathbb{R}^{n \times n}$ is symmetric and that $U_0 \in \mathbb{R}^{n \times n}$ is orthogonal. Consider the following QR iteration:

$$\begin{aligned}
 T_0 &= U_0^T A U_0 \\
 \text{f r k=} 1,2, \dots \\
 T_{k-1} &= U_k R_k \quad (\text{QR factorization}) \tag{82.1} \\
 T_k &= R_k U_k \\
 \text{end}
 \end{aligned}$$

Since $T_k = R_k U_k = U (U_k R_k) U_k = U T_{k-1} U_k$ it follows by induction that

$$T_k = (U_0 U_1 \cdots U_k)^T A (U_0 U_1 \cdots U_k). \quad (822)$$

Thus each T_k is orthogonally similar to A . Moreover, the T_k almost always converge faster than the R_k and so it can be said that (821) almost always converges to a Schur decomposition of A . In order to establish this remarkable result we first consider the power method and the method of orthogonal iteration.

i initb $\mathbf{z}^{(0)}$ $\mathbf{q}^{(0)}$ \mathbf{h} f eltb

Given a unit norm $q^{(0)} \in \mathbb{R}^n$, the power method produces a sequence of vectors $q^{(k)}$ as follows:

f r k= 1,2,...

$$\mathbf{z}^{(k)} = A \mathbf{q}^{(k-1)}$$

$$\mathbf{q}^{(k)} = \mathbf{z}^{(k)} / \| \mathbf{z}^{(k)} \|_2$$

$$A^k = [\mathbf{q}^{(k)}]^T A \mathbf{q}^{(k)}$$

end

If $q^{(0)}$ is not "deficient" and A 's eigenvalue of maximum modulus is unique, then the $q^{(k)}$ converge to an eigenvector.

Theorem 8.2.1. Suppose $A \in \mathbb{R}^{n \times n}$ is symmetric and that

$$Q^T A Q = \text{diag}(A_1, \dots, A_n)$$

where $Q = [q_1 | \dots | q_n]$ is orthogonal and $|A_1| > |A_2| > \dots > |A_n|$. Let the vectors $q^{(k)}$ be specified by (823) and defined by $[Q]_{n/2}$ by

$$\cos(Q) = \frac{1}{2} \mathbf{q}^{(k)} \mathbf{I}$$

If $\cos(Q) = Q$ then for $k = 0, 1, \dots$ we have

$$|\sin(Q)| \leq \tan(Q) \left| \frac{\lambda_2}{\lambda_1} \right|, \quad (824)$$

$$|A^k - A_1| \leq \max_{2 \leq i \leq n} |A_i - A_1| \alpha(Q)^{2^{i-1}}. \quad (825)$$

From the definition of the iteration, it follows that $q^{(k)}$ is a multiple of $A^k q^{(0)}$ also

$$|\sin(\theta_k)|^2 = 1 - (q_1^T q^{(k)})^2 = 1 - \left(\frac{q_1^T A^k q^{(0)}}{\|A^k q^{(0)}\|_2} \right)^2.$$

If $q^{(0)}$ has the eigenvector expansion $q^{(0)} = a_1 q_1 + \dots + a_n q_n$, then

$$|a_1| = |q_1^T q^{(0)}| = \cos(\theta_0) \neq 0,$$

$$a_1^2 + \dots + a_n^2 = 1,$$

and

$$A^k q^{(0)} = a_1 \lambda_1^k q_1 + a_2 \lambda_2^k q_2 + \dots + a_n \lambda_n^k q_n.$$

Thus

$$\begin{aligned} \text{sin}_{(\text{rh})}^2 &= 1 - \frac{a_1^2 \lambda_1^{2k}}{\sum_{i=2}^n a_i^2 \lambda_i^{2k}} = \frac{\sum_{i=2}^n a_i^2 \lambda_i^{2k}}{\sum_{i=1}^n a_i^2 \lambda_i^{2k}} \leq \frac{\sum_{i=2}^n a_i^2}{\sum_{i=1}^n a_i^2} \\ &= a_1^2 \sum_{i=2}^n \left(\frac{\lambda_i}{\lambda_1} \right)^{2k} \leq a_1^2 \left(\sum_{i=2}^n \frac{a_i^2}{a_1^2} \right) \left(\frac{\lambda_2}{\lambda_1} \right)^{2k} \\ &= \frac{1 - a_1^2}{a_1^2} \left(\frac{\lambda_2}{\lambda_1} \right)^{2k} = \tan \Theta^2 \left(\frac{\lambda_2}{\lambda_1} \right)^{2k}. \end{aligned}$$

This proves (824). Likewise,

$$\|k\| = \left[q^{(k)} \right]^T A q^{(k)} = \frac{\|Q S A^{2k} Q\|}{\|Q S A^{2k} Q\|} = \frac{\sum_{i=1}^n a_i^2 \lambda_i^{2k+1}}{\sum_{i=1}^n a_i^2}$$

and so

$$\begin{aligned} |\lambda^{(k)} - \lambda_1| &= \left| \frac{\sum_{i=2}^n a_i^2 \lambda_i^{2k+1}}{\sum_{i=1}^n a_i^2} \right| \leq \max_{2 \leq i \leq n} |\lambda_1 - \lambda_i| \cdot \frac{1}{a_1^2} \cdot \sum_{i=2}^n a_i^2 \left(\frac{\lambda_i}{\lambda_1} \right)^{2k} \\ &\leq \max_{2 \leq i \leq n} \|i - A\| \cdot \tan \Theta^2 \left(\frac{\lambda_2}{\lambda_1} \right)^{2k}. \end{aligned}$$

completing the proof of the theorem. \blacksquare

Computable error bounds for the power method can be obtained by using Theorem 8113. If

$$\|A\|_F \leq \|A\|_1 \leq 8$$

then there exists $\epsilon \in (0, 1)$ such that $\|A\|_1 \leq \epsilon \cdot \|A\|_F \leq 8$.

$x \leftarrow A^{-1}x$

If the power method (8.2.3) is applied with A replaced by $(A - \lambda I)^{-1}$, then we obtain the method of inverse iteration. If λ is very close to a distinct eigenvalue of A , then q_k will be much richer in the corresponding eigenvector direction than its predecessor q_{k-1} .

$$\begin{aligned} X &= \begin{cases} t \\ \vdots \\ aq \\ \hline Aq = \lambda q, i = 1 \dots n \end{cases} \quad \left. \right\} = (A - \lambda I)^{-1}x = \sum_{i=1}^n \frac{a_i}{\lambda_i - \lambda} q_i. \end{aligned}$$

Thus if λ is reasonably close to a well-separated eigenvalue, then inverse iteration will produce iterates that are increasingly in the direction of λ . Note that inverse iteration requires at each step the solution of a linear system with matrix of coefficients $A - \lambda I$.

kmibbb 9KPS"eb 5D kf sstf b(f\$tf sktb

Suppose $A \in \mathbb{R}^{n \times n}$ is symmetric and that x is a given nonzero n -vector. A simple differentiation reveals that

$$r(x) = \frac{x^T Ax}{x^T x}$$

minimizes $\|Ax - Mx\|_2$. (See also Theorem 8.1.4.) The scalar $r(x)$ is called the Rayleigh quotient of x . Clearly, if x is an approximate eigenvector, then $r(x)$ is a reasonable choice for the corresponding eigenvalue. Combining this idea with inverse iteration gives rise to the Rayleigh quotient iteration where $x_0 \neq 0$ is given

$f r k = Q 1, \dots$

$$\mu_k = r(x_k) \quad (8.2.6)$$

$$\text{Solve } (A - \mu_k I)z_k = x_k f r z_k$$

$$X_k = Z_k / \|Z_k\|_2$$

end

The Rayleigh quotient iteration almost always converges and when it does, the rate of convergence is cubic. We demonstrate this for the case $n = 2$. Without loss of generality, we may assume that $A = \text{diag}(1, 2)$, with $1 > 2$. Denoting X_k by

$$x_k = \begin{bmatrix} c_k \\ s_k \end{bmatrix}, \quad c_k^2 + s_k^2 = 1,$$

it follows that $\mu_k = \lambda_1 c_k^2 + \lambda_2 s_k^2$ in (8.2.6) and

$$z_{k+1} = \frac{1}{\lambda_1 - \lambda_2} \begin{bmatrix} c_k/s_k^2 \\ -s_k/c_k^2 \end{bmatrix}.$$

A calculation shows that

$$c_{k+1} = \frac{|c_k|^3}{\sqrt{c_k^6 + s_k^6}}, \quad s_{k+1} = \frac{|s_k|^3}{\sqrt{c_k^6 + s_k^6}}. \quad (8.2.7)$$

From these equations it is clear that the \hat{x}_k converge cubically to either $\text{span}\{e_1\}$ or $\text{span}\{e_2\}$ provided $\|x_k\| = \|s_k\|$. Details associated with the practical implementation of the Rayleigh quotient iteration may be found in Parlett (1974).

$r \parallel r - (F^T F) x / \|F^T F x\|^2$

A straightforward generalization of the power method can be used to compute higher-dimensional invariant subspaces. Let r be a chosen integer that satisfies $1 \leq r \leq n$. Given an n -by- r matrix Q_0 with orthonormal columns, the method of orthogonal iteration generates a sequence of matrices $\{Q_k\}_{k=1}^r$ as follows:

for $k = 1, 2, \dots$

$$Z_k = A Q_{k-1} \quad (828)$$

$$Q_k R_k = Z_k \quad (\text{QR factorization})$$

end

Note that, if $r = 1$, then this is just the power method. Moreover, the sequence $\{Q_k e_i\}$ is precisely the sequence of vectors produced by the power iteration with starting vector $q = Q_0 e_i$.

$$|\lambda_1| \geq |\lambda_2| \geq \cdots \geq |\lambda_n|$$

$$D = \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix}_{n-r}^r .$$

$$D_r(A) = \text{ran}(Q_\alpha)$$

$$\overline{\overline{1}} = \sqrt{\overline{1}}.$$

Compare with Theorem 7.3.1

problem. We mention at the start that the condition (82.11) means that no vector in the span of Q_0 's columns is perpendicular to $D_r(A)$.

Using induction it can be shown that the matrix Q_k in (82.8) satisfies

$$A^k Q_0 = Q_k (R_k \cdots R_1).$$

This is a QR factorization of $A^k Q_0$ and upon substitution of the Schur decomposition (82.9)-(82.10) we obtain

$$\begin{bmatrix} D_f & 0 \\ 0 & D \end{bmatrix} \begin{bmatrix} Q & Q_0 \\ Q & Q_0 \end{bmatrix} = \begin{bmatrix} Q_\alpha^T Q_k \\ Q_\beta^T Q_k \end{bmatrix} (R_k \cdots R_1).$$

If the matrices V_k and w_k are defined by

$$V_k = Q Q_0$$

$$W_k = Q Q_0$$

then

$$D V_k = V_k (R_k \cdots R_1), \quad (82.13)$$

$$D W_k = W_k (R_k \cdots R_1). \quad (82.14)$$

Since

$$\begin{bmatrix} V_k \\ W_k \end{bmatrix} = \begin{bmatrix} Q & Q_k \\ Q & Q_k \end{bmatrix} = [Q_a I Q_b] Q_k = Q Q_k$$

it follows from the thin CS decomposition (Theorem 25.2) that

$$1 = \sigma_{\min}(V_k)^2 + \sigma_{\max}(W_k)^2 = \sigma_{\min}(V_k)^2 + d_k^2.$$

A consequence of this is that

$$\sigma_{\min}(V_k)^2 = 1 - \sigma_{\max}(W_k)^2 = 1 - \frac{d_k^2}{1} > 0$$

It follows from (82.13) that the matrices V_k and $(R_k \cdots R_1)$ are nonsingular. Using both that equation and (82.14) we obtain

$$W_k = D W_k (R_k \cdots R_1)^{-1} = D W_k (D V_k)^{-1} V_k = D (W_k V_k^{-1}) D^{-1} V_k$$

also

$$\begin{aligned} \|V_k\| &\leq \|D\| \|W_k\| \|V_k\|^{-1} \|D^{-1}\| \|V_k\| \\ &\leq \|A^{r+1}\|^k \|D\| \cdot \frac{1}{1-\frac{1}{\|A\|}} \cdot \frac{1}{\|A\|^k}, \end{aligned}$$

from which the theorem follows.

$\|A\|_F \leq \|A\|_1 \leq \|A\|_2 \leq \dots \leq \|A\|_\infty$

Consider what happens if we apply the method of orthogonal iteration (828) with $r < n$. Let $QTAQ = \text{diag}(A_1, \dots, A_r)$ be the Schur decomposition and assume

$$\|A_1\| > \|A_2\| > \dots > \|A_r\|$$

If $Q^k = [q_1 | \dots | q_n]$, $Q_k = [q_1^k | \dots | q_r^k]$, and

$$\text{dist}(\text{Di}(A), \text{span}\{q_1^{(0)}, \dots, q_r^{(0)}\}) = r - 1 \quad (8215)$$

for $i = 1:n - 1$, then it follows from Theorem 822 that

$$\text{dist}(\text{span}\{q_1^k, \dots, q_r^k\}, \text{span}\{q_1, \dots, q_r\}) = O\left(\left|\frac{\|A_1\|^k}{\|A_r\|}\right|\right)$$

for $i = 1:n - 1$. This implies that the matrices T_k defined by

$$T_k = Q_k^T A Q_k$$

are converging to diagonal form. Thus, it can be said that the method of orthogonal iteration computes a Schur decomposition if $r = n$ and the original iterate $Q_0 \in \mathbb{T}_n^{mn}$ is not deficient in the sense of (8211).

The QR iteration arises by considering how to compute the matrix T_k directly from its predecessor T_{k-1} . On the one hand, we have from (828) and the definition of T_{k-1} that

$$T_{k-1} = Q_{k-1}^T A Q_{k-1} = Q_{k-1}^T (A Q_{k-1}) = (Q_{k-1}^T Q_k) R_k.$$

On the other hand,

$$T_k = Q_k^T A Q_k = (Q_k^T A Q_{k-1})(Q_{k-1}^T Q_k) = R_k (Q_{k-1}^T Q_k).$$

Thus, T_k is determined by computing the QR factorization of T_{k-1} and then multiplying the factors together in reverse order. This is precisely what is done in (821).

Note that a single QR iteration involves $O(n^3)$ fops. Moreover, since convergence is only linear (when it exists), it is clear that the method is a prohibitively expensive way to compute Schur decompositions. Fortunately, these practical difficulties can be overcome, as we show in the next section.

Problems

P8.2.1 ... 8 ... ~~A = B~~ ~~R = T~~ ~~M = A~~ ~~H = B~~ ~~G = C~~ ~~N = D~~ ~~K = E~~ ~~L = F~~

for k = 1 to n
~~A = B~~
~~R = T~~
~~M = A~~
~~H = B~~
~~G = C~~
~~N = D~~
~~K = E~~
~~L = F~~
 end

~~H = M + H~~ ~~M = H~~ ~~G = M + G~~ ~~N = M + N~~ ~~K = M + K~~ ~~L = M + L~~

$$A = \begin{bmatrix} & \\ & \end{bmatrix}$$

isAa c 2. Inrlaii>I 1t 1l ,J.; m1a Ap2Alai1urplenir 1l

P8.2.2 3Ns.. 7505 P

P8.2.38 .. A Bl c ctts..lp.nA iae el=a1 pmilaApnla2 . 1- =qw r

$$\left(\begin{bmatrix} & \end{bmatrix} \right) = \left[\begin{smallmatrix} & \\ & : = / T_0 \end{smallmatrix} \right]$$

All x u - 1 x u 1 a 2=aN2x = a 3a = 3= = 2aaax 2a wxxI =x= = =
x wa333 => x x x w a,1 . x ak vE i L111,1 ,1 1 s,=2 . || x_c =
, 3 x ' Ax c.

Notes and References for §8.2

.. ro.86.. N. N..8..N.. 6.. .20.8. 8.68.8...8.R6. o.8d.86.
a x 4a x =z x 42x xxaxx a=x, a =

2- - a xaa0ma war• • a Ex a=a3x x*xuxa 0+* xxax=u x =x=ina= + 2x xx= =
,= x+ Numer. Math. 13 571r 5T74

n4 gpiae 3, 9 arNTJ2.am19i>Ii; cl4 9nr1aiigl 1ae 9nr1ai1Ap1utli> stC.1pucA
nipucA,lr sn.Igpia1II18, 1pnl6 Comput. J. 13, T7r dJ4
Z II pntmlt1m T74.sn.I >pia1II1f, 1pnl1pmlelu st.)p.nA nipunA 1Nomer. Math. 16
IJ, II5M

G11ulaA1r p2lit>lcr2sK1pn1ap2le caAgIele

f Biaelr.ie ,r.Tr.2..1a1iUn.le lnt>Inrlmp2let.cP23SSeAipclal.caena19nr1aiigl 1t
niur hipnA 16 Lin. Alg. Applic. 4, 5,5r 5T7d
w424g1p, r. T., amit>InrnV1lpc1afp1ipnlae sl.1 la1ui>n.ipnlat2laalu.iU
nipunA 1t6ath. Comput. 28 7T,75,
s4wipp1tlaiae 94,c ggmd.. , amIEtaicAt litg1nr2V1lpc1afp1ipnlae 3ipnla.6 SIAM J. Numer.
Anal. 26 71r 7574

- 4wlippri E.41 Jb ,r. d. r .nlAi>cpcda .np1niae - lapina)a; 1 litg1cr2V1lpc1ap
ph1pa. XIAM J. Matrix Anal. Applic. 10 dJG.52

h4a4h4 ar ,r. , .4.Eta1n.A - laecpnla8tpn.ipnlae li tg1nrmln,3SSulbn.pcl..6SIAM J.
Matri. Anal. Applic. 15 55N67r

D. X.: Sn aAelO k0= n 3102Z pDj= • (n = > n Ay menEnBa e< npyJ= (Hn = • = 0
.0 = yff < O Ea =Sa Eo Oyn = f A=EmmXordh Ndr

f 4sin4p1s1upN d.2 "3 wglBlit>Inr2V1lpn1ap1ipnlae 2nEAnYieipcA lai).r)aA16
ETNA 7, iGT,4

-x fiai ,4 splip ,IJN.23a 3ai1tnl. 5mli t>Inr2lmp1pmlelu 3SSulbn.ipnar
9cr1atSiAKMath. Comput. 7, p' o :p be
=.00X= =E(SQ 01L 22pE= n jla e< npy= (A= =Hn pefTE 0ooy.= x(p-x)kpy
= a =HDF42N,r NdIM

h434Stc>l. nimlat lu sISIgA2.iae h4BiaEll.1a ,IJJ., .3 ..t.iaa, lit>Inr2V1lpc1ap
8pli; cl4 - ISKpnasaiuniak,tSiA1t XIAM Review 44 ,r T5M

2pit ,IJJ.2.- lai1r laA13ai>tnt 1. 8all1Apr2V1lpc1afp1ipcla.6 SIAM J. Matrix
Anal. Applic. 24, 7IT 7,iZ

34Eib,IJ.5.4.aml.; mlrlaiglitU1cr2V1lpn1ap1ipcla ,IV8.1pmleX Lin. Alg. Applic.
358, I5 i5r

4rne,4J.4.3AAI.mAt - l.SI p)e 9cr1ai1Ap1B ci,SpnCn.naiitU1cr2V1lpn1ap1pmleX
BIT 44, d, ,52

B.unlt 21; laptS1C1p2le/mii litgl ,11a1ni lelu p21t.1p.nA 1cr1aiigl \$,l,>... t1le

34aiSm iae E.n42cpst ,r. dd4 a21 hle1Ap1e2npla n1p2le.. ,uelu N+ Y 1 pm1
stCC1; nAnr1aiigl1ul,g1 .6 SIAM J. Numer. Anal. 25, N57N5dl 4

h434Stc>14 sISI>A2u1a , =nA-a> j OHa J=.22p «D(p0Ea(<e>n Eynn Ea = (=
tE r = Pa Bpa of a = Xfp a > XIAM J. Matr x Anal. Applic. 26 TJr .74

8.3 The Symmetric QR Algorithm

The symmetric QR iteration (8.2.1) can be made more efficient in two ways. First, we show how to compute an orthogonal U_0 such that $U_0 A U_0 = T$ is tridiagonal. With this reduction, the iterates produced by (8.2.1) are all tridiagonal and this reduces the work per step to $O(n^2)$. Second, the idea of shifts are introduced and with this change the convergence to diagonal form proceeds at a cubic rate. This is far better than having the off-diagonal entries going to zero as $\|A_{i+1,i}/A_{ii}\|^k$ discussed in §8.2.5.

ibm3b 98Df f 1tbf lbcxlt1t" ktpb dkb

If A is symmetric, then it is possible to find an orthogonal Q such that

$$Q^T A Q = T \quad (8.3.1)$$

is tridiagonal. We call this the tridiagonal decomposition and a compression of data, it represents a very big step toward diagonalization.

We show how to compute (8.3.1) with Householder matrices. Suppose that Householder matrices P_1, \dots, P_k have been determined such that if

$$A_{k-1} = (P_1 \cdots P_{k-1})^T A (P_1 \cdots P_{k-1}),$$

then

$$A_{k-1} = \begin{bmatrix} B_{11} & B_{12} & \vdots \\ B_{21} & B_{22} & B_{23} \\ 0 & B_{32} & B_{33} \\ \hline k-1 & 1 & n-k \end{bmatrix} \quad k, 1$$

is tridiagonal through its first $k-1$ columns. If A is an order-($n-k$) Householder matrix such that $A B_{33}$ is a multiple of $I_{n-k}(:, 1)$ and if $P_k = \text{diag}(J_k, P_k)$, then the leading k -by- k principal submatrix of

$$A_k = P_k A_{k-1} P_k^{-1} = \begin{bmatrix} B_{11} & B_{12} & 0 \\ B_{21} & B_{22} & B_{23} A \\ 0 & A B_{33} & A B_{33} A \\ \hline k-1 & 1 & n-k \end{bmatrix} \quad k, 1$$

is tridiagonal. Clearly, if $U_0 = P_1 \cdots P_{k-2}$ then $U_0 A U_0 = T$ is tridiagonal.

In the calculation of A_k it is important to exploit symmetry during the formation of the matrix $P_k B_{33} F_k$. To be specific, suppose that A has the form

$$\tilde{P}_k = I - \frac{1}{\|w\|^2} w w^T, \quad / = 2v^T v, \quad 0 \neq v \in \mathbb{R}^{n-k}.$$

Note that if $p = f B_{33} V$ and $w = p - (f p^T v / 2v)$, then

$$\tilde{P}_k B_{33} \tilde{P}_k = B_{33} - v w^T - w v^T.$$

Since only the upper triangular portion of this matrix needs to be calculated, we see that the transition from A_{k-1} to A_k can be accomplished in only $4(n-k)^2$ fops.

Algorithm 8.3.1 Given a symmetric $A \in \mathbb{R}^{n \times n}$, the following algorithm overwrites A with $T = QTAQ$ where T is tridiagonal and $Q = H_1 \cdots H_n$ is the product of Householder transformations.

```

for k = 1:n-2
    [v, J] = house(A(k+1:n,k))
    p = , A(k+1:n,k+1:n)v
    w = p - (pTv/2)v
    A(k+1:n,k+1:n) = A(k+1:,k+1:n) - wvT - wT
end

```

This algorithm requires $4n^3/3$ flops when symmetry is exploited in calculating the rank-2 update. The matrix Q can be stored in factored form in the subdiagonal portion of A . If Q is explicitly required, then it can be formed with an additional $4n^3/3$ flops. Note that if T has a zero subdiagonal, then the eigenproblem splits into a pair of smaller eigenproblems. In particular, if $t_{k+1,k} = 0$ then

$$\sigma(T) = \sigma(T(1:k, 1:k)) \cup \sigma(T(k+1:n, k+1:n)).$$

If T has no zero subdiagonal entries, then it is said to be unreduced.

Let \hat{T} denote the computed version of T obtained by Algorithm 83.1. It can be shown that $\hat{T} = QT(A + E)Q$ where Q is exactly orthogonal and E is a symmetric matrix satisfying $\|E\|_F \leq c\|A\|_F$ where c is a small constant. See Wilkinson (AEP, p 297).

Clar tEwiESS IrwAis3r MESSx wror ji8 waw lzzzwr

We prove two theorems about the tridiagonal decomposition both of which have key roles to play in the following. The first connects (83.1) to the QR factorization of a certain Krylov matrix. These matrices have the form

$$K(A, v, k) = [v | Av | \cdots | A^{k-1}v], \quad A \in \mathbb{R}^{n \times n}, v \in \mathbb{R}^n.$$

Theorem 8.3.1. If $QTAQ = T$ is the tridiagonal decomposition of the symmetric matrix $A \in \mathbb{R}^{n \times n}$, then $QTK(A, Q(:, 1), n) = R$ is upper triangular. If R is nonsingular, then T is unreduced. If R is singular and k is the smallest index so $t_{kk} = 0$ then k is also the smallest index so $t_{kk-1} = 0$. Compare with Theorem 1.4.3.

P f. It is clear that if $q \perp Q(:, 1)$, then

$$\begin{aligned}
 Q^T K(A, Q(:, 1),) &= [Q^T q | (Q^T A Q)(Q^T q) | \cdots | (Q^T A Q)^{n-1}(Q^T q)] \\
 &= [q | T q | \cdots | r^{n-1} q] = R
 \end{aligned}$$

is upper triangular with the property that $r_0 = 1$ and $r_{ii} = t_{2i}t_{2i-1} \cdots t_i$, i.e., if $i = 2n$. Clearly, if R is nonsingular, then T is unreduced. If R is singular and r_{kk} is its first zero diagonal entry, then $k \geq 2$ and t_{kk-1} is the first zero subdiagonal entry. \square

The next result shows that Q is essentially unique once $Q(:, 1)$ is specified.

Theorem 8.3.2 (Implicit QR Theorem). Suppose $Q = [Q_1 \cdots Q_n]$ and $V = [V_1 \cdots V_n]$ are orthogonal matrices with the property that both $Q^T A Q = T$ and $V^T A V = S$ are tridiagonal where $A \in \mathbb{R}^{n \times n}$ is symmetric. Let α denote the smallest positive integer for which $I_{k,k} - Q = 0$ with the convention that $k = n$ if T is unreduced. If $V_1 = Q$, then $V_{1:k} \pm q$ and $|I_{i,i-1}| = |I_{i,i+1}|$ for $i = 2:k$. Moreover, if $k \leq n$, then $S_{k+1:k} \neq 0$. Compare with Theorem 7.4.2.

Proof. Define the orthogonal matrix $W = QV$ and observe that $W(:, 1) = I_{n(1, 1)} - e_1$ and $W^T W = S$. By Theorem 8.3.1, $W(:, 1:k)$ is upper triangular with full column rank. But $K(T, e_1, k)$ is upper triangular and so by the essential uniqueness of the thin QR factorization, $W(:, 1:k) = I_{n(1, 1:k)} \cdot \text{diag}(\pm 1, \dots, \pm 1)$. This says that $Q(:, i) = \pm V(:, i)$ for $i = 1:k$. The comments about the subdiagonal entries follows since $t_{i+1,i} = Q(:, i+1) r A Q(:, i)$ and $S_{i+1,i} = V(:, i+1)^T A V(:, i)$ for $i = 1:n-1$.

r.l.l.r M3r irr r if F or S or r MF\$ or w or Dos FS\$ l.r

We quickly state facts that pertain to the QR iteration and tridiagonal matrices. Complete verifications are straightforward.

- Preservation of Form. If $T = QR$ is the QR factorization of a symmetric tridiagonal matrix $T \in \mathbb{R}^{n \times n}$, then Q has lower bandwidth 1 and R has upper bandwidth 2 and it follows that $T_+ \triangleq RQ \circ Q^T(QR)Q = \sqrt{r}TQ$ is also symmetric and tridiagonal.
- Shifts. If $s \in J$ and $T - sI = QR$ is the QR factorization, then $T_+ \triangleq RQ_+ \circ Q^T(QR)Q$ is also tridiagonal. This is called a shifted QR step.
- Perfect Shifts. If T is unreduced, then the first $n-1$ columns of $T - sI$ are independent regardless of s . Thus, if $s \in J(T)$ and $QR = T - sI$ is a QR factorization, then $m=0$ and the last column of $T_+ = RQ_+ \circ sI$ equals $sI(:, n) = s e_n$.
- Cost. If $T \in \mathbb{R}^{n \times n}$ is tridiagonal, then its QR factorization can be computed by applying a sequence of $n-1$ Givens rotations.

for $k = 1:n-1$

$[c, s] = \text{givens}(t_{kk}, t_{k+1,k})$

$m = \min\{k+2, n\}$

$$T(kk+1, km) = \sum_{i=k}^{m-1} c_i T(kk+1, km)$$

end

This requires $O(n)$ fops. If the rotations are accumulated, then $O(n^2)$ fops are needed.

8.3.4 $U \leftarrow A - \mu I$ $\hat{U} = Q \tilde{D}^{1/2} \hat{A}^{-1} U$ $b_i \leftarrow \hat{U} b_i$

If s is a good approximate eigenvalue, then we suspect that the $(n, n-1)$ will be small after a QR step with shift s . This is the philosophy behind the following iteration

$$T = U \tilde{D} U^T \quad (\text{tridiagonal})$$

for $k = 0, 1, \dots$

Determine real shift μ

$$T - \mu I = U R \quad (\text{QR factorization})$$

$$T = RU + \mu I$$

end

(832)

If

$$T = \begin{bmatrix} a_1 & b_1 & & \cdots & & 0 \\ b_1 & a_2 & \ddots & & & \vdots \\ \ddots & \ddots & \ddots & & & \\ \vdots & \ddots & \ddots & & & b_{n-1} \\ 0 & \cdots & & b_{n-1} & & a_n \end{bmatrix},$$

then one reasonable choice for the shift is $\mu = a_n$. However, a more effective choice is to shift by the eigenvalue of

$$T(n-1:n, n-1:n) = \begin{bmatrix} a_n & b_{n-1} \\ b_{n-1} & a_n \end{bmatrix}$$

that is closer to a_n . This is known as the Wilkinson shift and it is given by

$$\mu = a_n + d - \text{sign}(d) \sqrt{d^2 + b_{-1}} \quad (833)$$

where $d = (a_{n-1} - a_n)/2$. Wilkinson (1968) has shown that (832) is cubically convergent with either shift strategy, but gives heuristic reasons why (833) is preferred.

8.3.5 $b \otimes \vec{A} \vec{A}^T \vec{b} \rightarrow \vec{e} \vec{e}^T \vec{b} \vec{b}$

It is possible to execute the transition from T to $T_+ = RU + \mu I = U^T TU$ without explicitly forming the matrix $T - \mu I$. This has advantages when the shift is much larger than some of the a_i . Let $c = \cos(\theta)$ and $s = \sin(\theta)$ be computed such that

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} " & \\ & \mu \end{bmatrix} = \begin{bmatrix} & \\ & \end{bmatrix}.$$

If we set $G_1 = G(1, 2B)$, then $G_1^T G_1 = U$ and

$$T + G_1^T T G_1 = \begin{bmatrix} x & x & + & 0 & 0 & 0 \\ x & x & x & 0 & 0 & 0 \\ + & x & x & x & 0 & 0 \\ 0 & 0 & x & x & x & 0 \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix}.$$

We are thus in a position to apply the implicit Q theorem provided we can compute rotations G_2, \dots, G_{n-1} with the property that if $Z = G_1 G_2 \cdots G_{n-1}$, then $Z T Z^T = G_1 U_1$ and $Z T T Z$ is tridiagonal. Note that the first column of Z and U are identical provided we take each G_i to be of the form $G_i = G_i(i+1, \theta_i)$, $i = 2, \dots, n-1$. But G_i of this form can be used to remove the unwanted nonzero element "+" out of the matrix $G_i T G_i^T$ as follows.

$$\xrightarrow{G_2} \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ x & x & x & + & 0 & 0 \\ 0 & x & x & x & 0 & 0 \\ 0 & + & x & x & x & 0 \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix} \xrightarrow{G_3} \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ x & x & x & 0 & 0 & 0 \\ 0 & x & x & x & + & 0 \\ 0 & 0 & x & x & x & 0 \\ 0 & 0 & + & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix}$$

$$\xrightarrow{G_4} \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ x & x & x & 0 & 0 & 0 \\ 0 & x & x & x & 0 & 0 \\ 0 & 0 & x & x & x & + \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & + & x & x \end{bmatrix} \xrightarrow{G_5} \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ x & x & x & 0 & 0 & 0 \\ 0 & x & x & x & 0 & 0 \\ 0 & 0 & x & x & x & 0 \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \end{bmatrix}.$$

Thus, it follows from the implicit Q theorem that the tridiagonal matrix $Z T T Z$ produced by this zero damping technique is essentially the same as the tridiagonal matrix T obtained by the explicit method. (We may assume that all tridiagonal matrices in question are unreduced for otherwise the problem decouples.)

Note that at any stage of the zero damping there is only one nonzero entry outside the tridiagonal band. How this nonzero entry moves down the matrix during the update $T \leftarrow G_i T G_i^T$ is illustrated in the following.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & s & 0 \\ 0 & -s & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^T \begin{bmatrix} a_k & b_k & z_k & 0 \\ b_k & a_k & b & 0 \\ z_k & b_p & a_q & b_q \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & a_k & b_k & 0 \\ 0 & c & s & 0 & b_k & a_k & b \\ 0 & -s & c & 0 & b_p & a_q & b_q \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Here $(p, q, r) = (k+1, k+2, k+3)$. This update can be carried out in two steps once and should have been determined from the previous step. Overwriting in

Algorithm 8.3.2 (Implicit Symmetric QR Step with Wilkinson Shift) Given an unreduced symmetric tridiagonal matrix $T = [t_{ij}]$, the algorithm updates T with $Z T T Z$, where Z_3

$$\text{sign}(d) \sqrt{d^2}$$

The computed eigenvalues $\hat{\lambda}_i$ obtained via Algorithm 8.3.3 are the exact eigenvalues of a matrix that is near to A :

$$Q_0^T(A + E)Q_0 = \text{diag}(\hat{\lambda}_i), \quad Q_0^T Q_0 = I, \quad \|E\|_2 \approx \|u\| \|A\|_2.$$

Using Corollary 8.1.6 we know that the absolute error in each $\hat{\lambda}_i$ is small in the sense that

$$\|\hat{\lambda}_i - \lambda_i\| \leq \|A\|_2$$

If $Q = [q_1 | \dots | q_n]$ is the computed matrix of orthonormal eigenvectors, then the accuracy of q_i depends on the separation of $\hat{\lambda}_i$ from the remainder of the spectrum. See Theorem 8.1.12.

If all of the eigenvalues and a few of the eigenvectors are desired, then it is cheaper not to accumulate Q in Algorithm 8.3.3. Instead, the desired eigenvectors can be found via inverse iteration with T . See §8.2.2. Usually just one step is sufficient to get a good eigenvector, even with a random initial vector.

If just a few eigenvalues and eigenvectors are required, then the special techniques in §8.4 are appropriate.

8.3.6 Rayleigh Quotient Iteration

It is interesting to identify a relationship between the Rayleigh quotient iteration and the symmetric QR algorithm. Suppose we apply the latter to the tridiagonal matrix $T = \begin{pmatrix} t_0 & & & \\ & \ddots & & \\ & & t_{n-1} & \\ & & & t_n \end{pmatrix}$ with shift $a = e^T T^{-1} e = t_m$. If $T - aI = QR$, then we obtain $T = RQ + aI$. From the equation $(T - aI)Q = R$, it follows that

$$(T - aI)q_n = r_{nn}e_n,$$

where q_n is the last column of the orthogonal matrix Q . Thus, if we apply (8.2.6) with $x_0 = e_n$ then $X_1 = q_n$.

8.3.7 Lanczos Method

Recall from §8.2.4 that an orthogonal iteration step involves a matrix-matrix product and a QR factorization:

$$\begin{aligned} Z_k &= A Q_k^{-1} \\ Q_k R_k &= Z_k \quad (\text{QR factorization}) \end{aligned}$$

Theorem 8.1.14 says that we can minimize $\|AQ_k - QS\|_F$ by setting S equal to

$$S_k = Q_k^T A Q_k$$

If $U S_k U_k = D_k$ is the Schur decomposition of $S_k E R_k x_k^T$ and $Q_k = Q_k U_k$ then

$$\|AQ_k - Q_k D_k\|_F = \|A\tilde{Q}_k - \tilde{Q}_k S_k\|_F$$

showing that the columns of Q_k are the best possible basis to take after k steps from the standpoint of minimizing the residual. This defines the Ritz acceleration idea.

QOE 1 in r given with $Q_i Q_j = I_r$,
 $f \neq k = 1, 2, \dots$

$$Z_k = A Q_k \mathbf{1}$$

$$Q_k R_k = \tilde{Z}_k \quad (\text{QR factorization})$$

$$S_k = Q_k A Q_k$$

$U \text{ sk } U^{-1} = D$ (Schur decomposition)

$$Q_k = Q_k U_k$$

end

It can be shown that if

$$D_k = \text{diag}(\theta_1^{(k)}, \dots, \theta_r^{(k)}), \quad |\theta_1^{(k)}| \geq \dots \geq |\theta_r^{(k)}|,$$

then

$$\mathbf{B}^k \mathbf{y}_-, \mathbf{i}(\mathbf{A}) \mathbf{l} = O\left(\left|\frac{\lambda_{r+1}}{\lambda_i}\right| k\right), \quad i=1:r.$$

Recall that Theorem 8.2.2 says the eigenvalues of $Q^T A Q$ converge with rate $\|A\|_F$ if A is k -sparse. Thus, the Ritz values converge at a more rapid rate. For details, see Stewart (1989).

Problems

P8.3.1 .e..8 .8 .. 6 8.d86..otgh = r 4Vx=ues x 1000-1 aa u 101r,a OCr x r =
.- Cxq = & x 04u r- a Nuat tS3nor l8AaA.A8oMAMIM

**P8.3.2 .e..8.8 ..2289... . nl ..6n6.n96p. O = ==3J)θBO f=O= e==XJ6•z(0:j o= Of
A - 2 GfG ik GF g p m)8 2it So83n32m**

P8.3.3 m9

$$P8.3.3 \quad m9 \quad \left[\begin{array}{c}) \\] \end{array} \right] \quad \text{A} \quad \tilde{A} = \left[\begin{array}{cc} \tilde{w} & \tilde{x} \\ \tilde{x} & \tilde{z} \end{array} \right]$$

w . W X Z W X Z W X Z
 z . Z X W Z X W Z X W Z
 x = X Z W X Z W X Z W X

P8.3.4 .e. 8.8 E < ~~DN~~_{DN} ~~Z~~_Z ~~WV~~_{WV} L 21 - 21 Ap ~~TRAV~~_{TRAV} ~~JWV~~_{JWV} REA A ' > ~~K~~_K

P8.3.5 .686 96>A GB1 iC .. (.,) .-C),(C

MU -

[(77 豪 壴 壴 壴)] [100] [T] [11]

P8.3.6 786..8 1otdg.9 7a9AD 9.8.J8 6.86 .. 98.8n6.68 .8.9gke.9. 98 d..86 8.

S₁II T₁μn SERn_{k+1}AT **S₁T₁μn SERn_{k+1}AT**

J4mG m4T_c E R L-21 -21 V2EWV1 qv 220 Q tlAm 2012 r 2MrrB0,00E
QTu = e1

P8.3.8 .e. .8.8

$$C = \begin{bmatrix} & B \\ & \end{bmatrix}$$

All Bpon kFWW Hx20 tWm qWV F n WV V2w PERx12 VqV
T = PCP_x kVq 20 p uV q+2 x20

Notes and References for §8.3

Pfero9g... 528.9.69..got.8.fe ..9.86. .80895not.8.fe91269... .8.9.866..en8

7N.M.. fe66n kNPnt .6.86.8P12.8..9fe66.2.89.fe. .6n ro6o..289.feP.6n Pr..9fg
.6n 9.83.0t..ot.9.86Pf6l86.8.98.. P6m N.9.fe.86b7 Numer. Math IT.r 5Nx
.x wl.3AMksxl dVr 8kAr 8AnaE9x.xar ,q18 = Jd.x.lMAV1 o83Vn3zrlMr t2mi

C.P. i. (= 1981). „a 2m MaiAMrA 12m xl 3U1Mc2nd2mMcsmneM2lMP 1.2Mco AAAtkSA J. Numer. Anal. 1, 127-133.
 S. x3 (= 1981). „wULAbc arhU.Mn32dasqA.LstPP A2M2M1McA 16 Lin. Alg. Applic. 27, 215-226.

8.4 More Methods for Tridiagonal Problems

In this section we develop special methods for the symmetric tridiagonal eigenproblem. The tridiagonal form

$$T = \begin{bmatrix} 0 & 1 & & & & & 0 \\ -1 & 0 & 2 & & & & \vdots \\ & -1 & 0 & \ddots & & & \\ & & -1 & 0 & \ddots & & \\ & & & -1 & 0 & \ddots & f_{n-1} \\ 0 & \cdots & & & f_{n-1} & f_n & 1 \end{bmatrix} \quad (84.1)$$

can be obtained by Householder reduction (cf. §8.3.1). However, symmetric tridiagonal eigenproblems arise naturally in many settings.

We first discuss bisection methods that are of interest when selected portions of the eigensystem are required. This is followed by the presentation of a divide-and-conquer algorithm that can be used to acquire the full symmetric Schur decomposition in a way that is amenable to parallel processing.

8.4.1 $\hat{\lambda} \approx \hat{\lambda}_D$ è $\hat{\lambda}$ è $\hat{\lambda}_A$ è $\hat{\lambda}_B$ è $\hat{\lambda}$

Let T_r denote the leading r -by- r principal submatrix of the matrix T in (84.1). Define the polynomial $P_r(x)$ by

$$P_r(x) = \det(T_r - xI)$$

for $r = 1, n$. A simple determinantal expansion shows that

$$P_r(x) = (a_{rr} - x)P_{r-1}(x) - f_{r-1}P_{r-2}(x) \quad (84.2)$$

for $r = 2, n$ if we set $P_0(x) = 1$. Because $P_r(x)$ can be evaluated in $O(n)$ flops, it is feasible to find its roots using the method of bisection. For example, if td is a small positive constant, $P_1(y)P_1(z) < 0$ and $y < z$, then the iteration

```

while |y-z| > td (I I+|z|)
  x = (y+z)/2
  if P1(x)*P1(y) < 0
    Z = X
  else
    y = x
  end
end

```

is guaranteed to terminate with $(y+z)/2$ an approximate zero of $P_1(x)$, i.e., an approximate eigenvalue of T . The iteration converges linearly in that the error is approximately halved at each step.

qA)cA L N=1A Lo Nal oAcp -EL+A

Sometimes it is necessary to compute the k th largest eigenvalue of T for some prescribed value of k . This can be done efficiently by using the bisection idea and the following classical result:

Theorem 8.4.1 (Sturm Sequence Property). If the tridiagonal matrix in

$$\lambda_r(T_r) < \lambda_{r-1}(T_{r-1}) < \lambda_{r-1}(T_r) < \cdots < \lambda_2(T_r) < \lambda_1(T_{r-1}) < \lambda_1(T_r).$$

During the execution of §84.3, information about the location of other eigenvalues is obtained. By systematically keeping track of this information it is possible to devise an efficient scheme for computing contiguous subsets of $\sigma(T)$, e.g., $\{k(T), k+1(T), \dots, k+i(T)\}$. See Barth, Martin, and Wilkinson (1967).

If selected eigenvalues of a general symmetric matrix A are desired, then it is necessary first to compute the tridiagonalization $T = UAU^{-1}$; then the above bisection schemes can be applied. This can be done using Algorithm 8.3.1 or by the Lanczos algorithm discussed in §10.2. In either case, the corresponding eigenvectors can be readily found via inverse iteration since tridiagonal systems can be solved in $O(n)$ fops. See §4.3.6 and §8.2.2.

In those applications where the original matrix A already has tridiagonal form, bisection computes eigenvalues with small relative error, regardless of their magnitude. This is in contrast to the tridiagonal QR iteration, where the computed eigenvalues $\tilde{\lambda}_i$ can be guaranteed only to have small absolute error. $\tilde{\lambda}_i - \lambda_i(T) \ll \|T\|_F$

Finally, it is possible to compute specific eigenvalues of a symmetric matrix by using the LDLT factorization (§4.3.6) and exploiting the Sylvester inertia theorem (Theorem 8.1.17). If

$$A - \mu I = LDL^T, \quad A = A^T \in \mathbb{R}^{n \times n},$$

is the LDLT factorization of $A - \mu I$ with $D = \text{diag}(d_1, \dots, d_n)$, then the number of negative d_i equals the number of $\lambda_i(A)$ that are less than μ . See Parlett (SEP, p. 46) for details.

8.4.3 Using the QR Factorization for the Symmetric Tridiagonal Eigenproblem

Our next method for the symmetric tridiagonal eigenproblem requires that we be able to compute efficiently the eigenvalues and eigenvectors of a matrix of the form $D + pz z^T$, where $D \in \mathbb{R}^{n \times n}$ is diagonal, $z \in \mathbb{R}^n$, and $p \in \mathbb{R}$. This problem is important in its own right and the key computations rest upon the following pair of results.

Lemma 8.4.2. Suppose $D = \text{diag}(d_1, \dots, d_n) \in \mathbb{R}^{n \times n}$ with

$$d_1 > \dots > d_n$$

Assume that $p \neq 0$ and that $z \in \mathbb{R}^n$ has nonzero components. If

$$(D + pz z^T)v = \lambda v, \quad \forall v \neq 0$$

then $z^T v \neq 0$ and $D - \lambda I$ is nonsingular.

Pf. If $\lambda = D - \lambda I$, then $v = dz$ for some d and thus

$$0 = [(D - \lambda I)v + p(z^T v)z] = p(z^T v)z.$$

Since p and z are nonzero, it follows that $0 \neq z^T v$ and so $Dv = \lambda v$. However, D has distinct eigenvalues and therefore $v \notin \text{span}\{e_i\}$. This implies $0 = z^T v \neq z^T \lambda v$, a contradiction. Thus, D and $D + pz z^T$ have no common eigenvalues and $z^T v \neq 0$.

Theorem 8.4.3. Suppose $D = \text{diag}(d_1, \dots, d_n) \in \mathbb{R}^{n \times n}$ and that the diagonal entries satisfy $d_1 > \dots > d_n$. Assume that $p \neq 0$ and that $z \in \mathbb{R}^n$ has no zero components. If $V \in \mathbb{R}^{n \times n}$ is orthogonal such that

$$V(D + pzz^T)V = \text{diag}(A_1, \dots, A_n)$$

with $A_1 \geq \dots \geq A_n$ and $V = [V_1 \cdots V_n]$, then

- (a) The A_i are then zeros of $f(A) = 1 + pzz^T(D - M)^{-\frac{1}{2}}$
- (b) If $p > 0$ then $A_1 > d_1 > A_2 > \dots > A_n > d_n$
If $p \leq 0$ then $d_1 > A_1 > d_2 > \dots > d_n > A_n$
- (c) The eigenvector V is a multiple of $(D - A_1)^{-\frac{1}{2}}$.

Proof. If $(D + pzz^T)V = AV$, then

$$(D - A)V + p(z^T V)z = 0 \quad (844)$$

We know from Lemma 8.4.2 that $D - A$ is nonsingular. Thus

$$V \in \text{span}\{(D - A)^{-\frac{1}{2}}\},$$

thereby establishing (c). Moreover, if we apply $z^T(D - M)^{-\frac{1}{2}}$ to both sides of equation (844) we obtain

$$(z^T V) \cdot (1 + pzz^T(D - M)^{-\frac{1}{2}}) = 0$$

By Lemma 8.4.2, $z^T V = 0$ and so this shows that if $A \in \mathcal{A}(D + pzz^T)$, then $f(A) = 0$. We must show that all the zeros of f are eigenvalues of $D + pzz^T$ and that the interlacing relations (b) hold.

To do this we look more carefully at the equations

$$\begin{aligned} f(A) &= 1 + p \left(\frac{1}{d_1 - A^2} + \frac{1}{d_2 - A^2} + \dots + \frac{1}{d_n - A^2} \right)' \\ J(A) &= p \left(\frac{d_1^2}{(d_1^2 - A^2)^2} + \frac{d_2^2}{(d_2^2 - A^2)^2} + \dots + \frac{d_n^2}{(d_n^2 - A^2)^2} \right). \end{aligned}$$

Note that f is monotone in between its poles. This allows us to conclude that, if $p > 0$ then f has precisely n roots, one in each of the intervals

$$(d_n, d_{n-1}), \dots, (d_2, d_1), (d_1, \infty).$$

If $p \leq 0$ then f has exactly n roots, one in each of the intervals

$$(-\infty, d_n), (d_n, d_{n-1}), \dots, (d_2, d_1).$$

Thus, in either case the zeros of f are exactly the eigenvalues of $D + pzz^T$.

The theorem suggests that in order to compute V we must find the roots A_1, \dots, A_n of using a Newton-like procedure and then compute the columns of V by normalizing

the vectors $(D - A_i)^{-1}z$ for $i = 1:n$. The same plan of attack can be followed even if there are repeated d_i and zero Z_i .

Theorem 8.4.4. If $D = \text{diag}(d_1, \dots, d_n)$ and $z \in \mathbb{R}^n$ then there exists an orthogonal matrix V such that if $V^T D V = \text{diag}(\mu_1, \dots, \mu_n)$ and $w = V^T z$ then

$$\mu_1 > \mu_2 > \dots > \mu_r = \mu_{r+1} = \dots = \mu_n$$

$w_i = 0$ for $i = 1:r$ and $w_i \neq 0$ for $i = r+1:n$.

Proof. We give a constructive proof based upon two elementary operations. The first deals with repeated diagonal entries while the second handles the situation when the z -vector has a zero component.

Suppose $d_i = d_j$ for some $i \neq j$. Let $G(i,j,\theta)$ be a Givens rotation in the $\{i,j\}$ plane with the property that the j th component of $G(i,j,\theta)z$ is zero. It is not hard to show that $G(i,j,\theta)D G(i,j,\theta)^T = D$. Thus we can zero a component of z if there is a repeated d_i .

If $Z_i = 0$, $Z_j = 0$ and $i \neq j$, then let P be the identity with columns i and j interchanged. It follows that $P^T D P$ is diagonal, $(P^T z)_i = 0$ and $(P^T z)_j \neq 0$. Thus, we can permute all the zero Z_i to the "bottom".

It is clear that the repetition of these two maneuvers will render the desired canonical structure. The orthogonal matrix V is the product of the rotations that are required by the process. \square

See Barlow {1993} and the references therein for a discussion of the solution procedures that we have outlined above.

8.4.4 How to Compute the Schur Decomposition

We now present a divide-and-conquer method for computing the Schur decomposition

$$Q^T T Q = \Lambda = \text{diag}(\lambda_1, \dots, \lambda_n), \quad Q^T Q = I, \quad (845)$$

for tridiagonal T that involves (a) "tearing" T in half, (b) computing the Schur decompositions of the two parts, and (c) combining the two half-sized Schur decompositions into the required full-size Schur decomposition. The overall procedure, developed by Dongarra and Sorensen {1987}, is suitable for parallel computation.

We first show how T can be "torn" in half with a rank 1 modification. For simplicity, assume $n = 2m$ and that $T \in \mathbb{R}^{n \times n}$ is given by (841). Define $V \in \mathbb{R}^{n \times n}$ as follows

$$V = \begin{bmatrix} \theta E \\ \theta \end{bmatrix} \bullet \theta \in \{-1, +1\}. \quad (846)$$

Note $T v = 0$

$$\begin{bmatrix} \beta_m - \rho \theta \\ \alpha_{m+1} - \rho \theta^2 \end{bmatrix}.$$

If we set $p = f_m$ then

$$\begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix}$$

where

$$T_1 = \begin{bmatrix} Q_1^{-1} I_1 & 0 \\ 0 & Q_2^{-1} I_2 \end{bmatrix}, \quad T_2 = \begin{bmatrix} U_{m1}^{-1} f_{m1} & 0 \\ f_{m1}^{-1} U_m & Q_2^{-1} I_2 \end{bmatrix},$$

and $a_{m1}^H a_{m1}$ and $l_{m1} l_{m1}^H = a_{m1}^{-1} I - p^2$

Now suppose that we have m by m orthogonal matrices Q_1 and Q_2 such that $Q_1^T Q_1 = D_1$ and $Q_2^T Q_2 = D_2$ are each diagonal. If we set

$$U = \begin{bmatrix} \mathbf{i} \\ \mathbf{z} \end{bmatrix},$$

then

$$U^T T U = U^T \left(\begin{bmatrix} \mathbf{i}^H & \\ & \mathbf{z}^H \end{bmatrix} + p \mathbf{z} \mathbf{z}^H \right) U = D + p \mathbf{z} \mathbf{z}^H$$

where

$$D = \begin{bmatrix} \mathbf{1} \\ \mathbf{z} \end{bmatrix}$$

is diagonal and

$$Z = U \mathbf{F} V = \begin{bmatrix} Q_1^{-1} I_m \\ (Q_2^{-1}) \mathbf{z} \end{bmatrix}.$$

Comparing these equations we see that the effective synthesis of the two half-sized Schur decompositions requires the quick and stable computation of an orthogonal V such that

$$V^T (D + p \mathbf{z} \mathbf{z}^H) V = A = \text{diag}(1, \dots, n)$$

which we discussed in §8.4.3

8.4.5 $\mathbf{p} \leftarrow \mathbf{D} \mathbf{D}^H \mathbf{p} / \mathbf{D} \mathbf{Q}_1^{-1} \mathbf{U}_1 \mathbf{A} \mathbf{B} \mathbf{U}^H \mathbf{p}$

Having stepped through the tearing and synthesis operations, we can now illustrate how the overall process can be implemented in parallel. For clarity, assume that $n = 8N$ for some positive integer N and that three levels of tearing are performed. See Figure 8.4.1. The indices are specified in binary and at each node the Schur decomposition of a tridiagonal matrix $T(\mathbf{b})$ is obtained from the eigensystems of the tridiagonals $T(\mathbf{b}_0)$ and $T(\mathbf{b}_1)$. For example, the eigensystems of the N -by- N matrices $T(110)$ and $T(111)$ are combined to produce the eigensystem of the $2N$ -by- $2N$ tridiagonal matrix $T(11)$. What makes this framework amenable to parallel computation is the independence of the tearing/synthesis problems that are associated with each level in the tree.

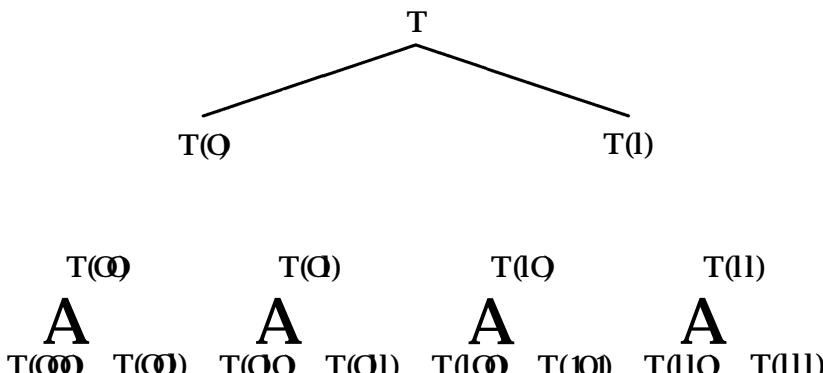


Figure 84.1 The divide-and-conquer framework

8.4.6 Inverse eigenvalue problems for symmetric tridiagonal matrices

For additional perspective on symmetric tridiagonal matrices and their rich eigenstructure we consider an inverse eigenvalue problem. Assume that A_1, \dots, A_n and $\lambda_1, \dots, \lambda_n$ are given real numbers that satisfy

$$A_1 > \lambda_1 > A_2 > \dots > A_{n-1} > \lambda_n > A_n. \quad (847)$$

The goal is to compute a symmetric tridiagonal matrix $T \in \mathbb{R}^{n \times n}$ such that

$$A(T) = \{A_1, \dots, A_n\}, \quad (848)$$

$$A(T(2n, 2n)) = \{\lambda_1, \dots, \lambda_n\}. \quad (849)$$

Inverse eigenvalue problems arise in many applications and generally involve computing a matrix that has specified spectral properties. For an overview see Chu and Golub (2005). Our example is taken from Golub (1973).

The problem we are considering can be framed as a Householder tridiagonalization problem with a constraint on the orthogonal transformation. Define

$$A = \text{diag}(A_1, \dots, A_n)$$

and let Q be orthogonal so that $Q^T A Q = T$ is tridiagonal. There are an infinite number of possible Q matrices that do this and in each case the matrix T satisfies (848). The challenge is to choose Q so that (849) holds as well. Recall that a tridiagonalizing Q is essentially determined by its first column because of the implicit- Q theorem (Theorem 8.32). Thus, the problem is solved if we can figure out a way to compute $Q(:, 1)$ so that (849) holds.

The starting point in the derivation of the method is to realize that the eigenvalues of $T(2n, 2n)$ are the stationary values of $x^T T x$ subject to the constraints $x^T x = 1$ and $e^T x = 0$. To characterize these stationary values we use the method of Lagrange multipliers and set to zero the gradient of

$$\langle (x, A), I \rangle = x^T T x - A(x^T x - 1) + 2x^T T e$$

which gives $(T - \lambda I)x = -\mu e_1$. Because λ is an eigenvalue of $T(2n, 2n)$ it is not an eigenvalue of T and so

$$x = -\mu(T - \lambda I)^{-1}e_1.$$

Since $e_1^T Q = 1$ it follows that

$$0 = e_1^T (T - \lambda I)^{-1} e_1 = e_1^T (Q^T \Lambda Q - \lambda I)^{-1} e_1 = \sum_{i=1}^n \frac{d_i}{\lambda_i - \lambda} \quad (84.10)$$

where

$$Q(:, 1) = \begin{bmatrix} d_1 \\ \vdots \\ d_n \end{bmatrix}. \quad (84.11)$$

By multiplying both sides of equation (84.10) by $(A_{11} - A) \cdots (\lambda_n - \lambda)$, we can conclude that $\tilde{\lambda}_1, \dots, \tilde{\lambda}_{n-1}$ are the zeros of the polynomial

$$p(A) = \prod_{i=1}^n \prod_{\substack{j=1 \\ j \neq i}}^n (\lambda_j - \lambda).$$

It follows that

$$p(A) = \alpha \cdot \prod_{j=1}^{n-1} (\tilde{\lambda}_j - \lambda)$$

for some scalar α . By comparing the coefficient of λ^{n-1} in each of these expression for $p(\lambda)$ and noting from (84.11) that $d_1 + \cdots + d_n = 1$, we see that $\alpha = 1$. From the equation

$$\prod_{i=1}^n \prod_{\substack{j=1 \\ j \neq i}}^n (\lambda_j - \lambda) = \prod_{j=1}^{n-1} (\tilde{\lambda}_j - \lambda)$$

we immediately see that

$$d_k^2 = \prod_{j=1}^{n-1} (\tilde{\lambda}_j - \lambda_k) \left/ \prod_{\substack{j=1 \\ j \neq k}}^{n-1} (\lambda_j - \lambda_k) \right., \quad k = 1:n \quad (84.12)$$

It is easy to show using (84.7) that the quantity on the right is positive and thus (84.11) can be used to determine the components of $\mathbf{d} = Q(:, 1)$ up to within a factor of ± 1 . Once this vector is available, then we can determine the required tridiagonal matrix T as follows.

Step 1. Let P be a Householder matrix so that $P\mathbf{d} = \pm 1$ and set $A = P^T AP$.

Step 2. Compute the tridiagonalization $Q^T AQ = T$ via Algorithm 8.3.1 and observe from the implementation that $Q(:, 1) = \mathbf{e}_1$.

Step 3. Set $Q = PQ_1$.

It follows that $Q(\cdot, 1) \cap P(Q_1 e_1) = Pe_1 = \pm d$. The sign does not matter.

Problems

P8.4.2 N...8...
kl lam- *p*, lam+1 - N...8...
Nadv,vZnpnpmS1uppmipE)P) 2x

P8.4.3 5. Pr(/, e det(T(1,r,1:r) ->Ir).mlu) nt rnri laSt; dv.M.E lun1 u1AI utnla liiglipnar
M iae It1nplei lgSi 21pla nplui pnhai AiaAlCSI phr laii>I lt)v

P8.4.58 .. S+uuuTLCBxS E RnxnT=rBLdLb nxFERn 2x uER. +C(LC(Ln('Lnb @C(02 , ==aC nCPag - ' + uelef)N.). mf = kf (DE=)ED (.)=I)+.kf OF

P8.4.6 ... 8.. .. . d**86**.....o.. 8b. 8.. ...**b** Roxn
 C LC(L n a**b**l nB x(1n , r. dC pmklipnlala e > x rn i lapmja, €

P8.4.7 e...b.r.... 8. 8b 7d5a .8....a

P8.4.8 ...8... E

• • • •

$$f(\lambda) = \lambda + \sum_{k=1}^{n-1} \frac{v_k^2}{d_k - \lambda} - d_n.$$

Notes and References for §8.4

w.) a \rightarrow B Olda = Olda = 0 Olda \neq 0 \Rightarrow Olda \in Ep < a \Leftarrow Olda \in Open \Rightarrow a Open = EOE

K.K. $\int_0^T f(t) dt = \frac{1}{2} \int_0^T (f(t) + f(0)) dt = \frac{1}{2} \int_0^T g(t) dt$ Numer. Math. 9, 5d7 5.5v
 $\int_0^T f(t) dt = \frac{1}{2} \int_0^T (f(t) + f(0)) dt = \frac{1}{2} \int_0^T g(t) dt$ Numer. Math. 9, 5d7 5.5v

Meth. Eng. 4 51 J. v
f 941 CnUu vEwMiae, 2 11a ; N., "a pmAluu1Apal thiui>ignt1Apmla>lipnar
hanax TNA 3 N N..v

>tuulaA 11AlaA 1ua3epn0m12neI r iaer Alas11luB pmpl II pUuna1A>Ie18

f 9f 1aAmvh2n1gauiae E vAsluat1a; r.Tdv "liaB,,a1 nlen= AiphapmstCi lpuA
d2: 1aSulSUX Numer. Math. 31, 5Ndv

f r fIAISS laNdN.v3 Eninel alasI lu nlpnlepm4CC lpu29hr laSulSU 1KuYer.
Math 36 rTTN..v

f fr 2 lariuuiae E 9Astu lat 1a; r. dITv "3f >Uhiui>> B urlunprurpmst.C 1punAnr laiigl 1
aulSg1GXIAM J. Sci. Stat. Comput. 8, sN5.GsN.,v

ulaiuCItpS lpiB3pl 1atl u1upmlrlaignpmAliSI pleCipurlb 1nr lai 1Apatl6lpmnar
r .pm1 Ci) luAmiUg1an1a pm1nr 1al UI 1lu1A>1tlae AUI tp1lu1mleli 1ULS. fap. u1>niSU1
S> li lapipnlat AgtnApigpmipailUi it Cnbl. tlSmntpnAip1telutae A> li lu1r1u2pmCnA
2 rmpt118

f 9 **SiuUlpN.7.9"** aiuinisiStSiAiu anrmpNgtpIuor laiigl lt hnenirlaiUtux
36. Jr. IV
f 9 **SiuUlpN.7.9"** aiuinisiStSiAiu anrmpNgtpIuor laiigl lt hnenirlaiUtux

f 9 QuUpp iasvWUJ.J.v'liUipni lgSItp lISutlapipnlstii lpunAmeniraagtutt
Lin. Alg. Applic. 309, NlrNNN v

G dPARa n4Tn41 PmMA=923 iBH=li+ianR li+ealp=m1 naTeAnile On(GSIAM J. Matr Anal. Appl. 25, ddr d) **N**lsvnzzla3w.2:hoM,,22IJ.M.n,z2nSASMAAt,a2o2B1.S,2A,M2mlr l2o2 Aai,NL l. st.A2McAc3hor 18o2McAtm. Alg. Applic. 387, Nr 1dM ,v3: I oMs,Atk.2: hoM2B83A:BC.Az=IJ.M.ALS,2o2dat l. 9cr A8Ss,SVA2t .22 nl ll .3zr Mc2mlkler. Lin. Alg. Applic. 13, 7,r 7,M hwaAa2tak se2c,zl8k3 lv3M3.Ac8=JJ., .3 hoMozaAz AatlziMMAtst. 22M nc2McAt,3 18 uz 2nSASAz2ci,tls,t2 IASMATAtPMSIAM J. Sci. Comput. 27, , r 7r

BoM,t Ab2A8tcls3r ,aAMozc,c222S itcA1,o2diAotl SASASltA38

s.,trr A3AM.ca1tdlk oa3lv MaS,z NT. .3 1oMoZ,zcoS Aatl,i2MIAq E2r aozcoS,Ao2McA2mAoZ9nr AatklsIAM J. Sci. Comput. 18, di.r dg: wxAa3McA2k9At,Skao3A:s.c2m= Nd : ll. dM3 oa91AcA2MoZ9r AatlziMM2a2 st.A2MnA2o2McAtAM J. Sci. Comput. 20, N N N N y a,2. oat2AM9M12aAc8 A3MaMSAM2, =JN.:3 l.AlszAbc2E cic3A.oa8las,2M n,2213,MA1.S,2car 9cr A8io,At39cr AaiAAPMta2Mnwoa3ho2McAtT 41, .7T .T7:

aM2M2ar2AMAMMAM3oa3l:h: n,zzM=JJIM.38 9b2Aatcl8 2mEnic3A,oa8las,2M nA22E3MoA, tl. st.A2McAtAd/c30o lac2cr ,8SMASl1CM T ns. Math Sof w. 28.y .dM 4M2.0812HMAWAMM3 MhzzMba3am3M3bM3oa3 001J.MA1.S,2car 3SSM2 .o2A 9nr AaSbcM1.A2McAtAqVc33r lac2cr ,8SMASl1CM T ns. Math Sof w. 33, 3M2n,AM1 Iuwocca3l:A- adM3=JJIM3 hoMmsA2MnAw,IAqVc3hor lac2c3A,083Alas1 AM Mm,k6 CM T ns. Math Sof w. 33, 3M2n,AM1

ulM3A2ocz2MAo2,41 idMnl,t caiAMrAaiS,MISzAtAA

n:l: Am,083...: lzs' =JJ.., Inverse Eigenvalue Pr blems, bM3 r 8AMtdM)ttbMEu a:5M

sA,AA29.6AM223tA,ttoMo8l.,caiAMrAaiS,MISzAtAA

E:w,zAta3...: l,S = ,dTMB s,MiAt. no2M6aiAM2A Aaiq,AMISzAtA inverse Pr hem 3,y,y 7I.

nMIAm= Nd : 081AM9r A8io,AMIS,At SIAM Review 40, Nr ..:

A:,5M ca3IMno2,cC = IJ..M.Alat2M,A2dl1no2McAt2mhMAtAMsSA,3oNB oz,AtaE 9cr A8iozs)k6 41, N N NI 7:

ImA3AMcio2da7d,Mnai,iA32mAlat2MocaS2c,c.o2da. o s,o3Mo2cM.k oaS,ML.2 SMIS,Ac8 18t Mcrk2A8

.v.: ..z,S 083l: r 83AM.ll3 ,TJ.Ms2o2nladHtz,At 22Ao2nl. V,o3Mo2dM.ts,S 21 c8AMt2Mon82tgew. Math. Phys. 21, Ndr 17M

.v.: lzs' = N...: sl.A nl3cr ,9cr Aaiq,AMISzAt SIAM Review 15, Nd ...,"

sM AlA N.. . Idn,cnar vmaA0M.tslSA21 caAMt2M0t6in. Alg. Applic. 210, 12cPZ

8.5 Jacobi Methods

Jacobi methods for the symmetric eigenvalue problem attract current attention because they are inherently parallel. They work by performing a sequence of orthogonal similarity updates $\mathbf{A} \rightarrow \mathbf{Q}\mathbf{T}\mathbf{A}\mathbf{Q}$ with the property that each new \mathbf{A}_k although full, is "more diagonal" than its predecessor. Eventually, the off-diagonal entries are small enough to be declared zero.

After surveying the basic idea behind the Jacobi approach we develop a parallel Jacobi procedure.

MAAA T cA P} iAGA PlqA

The idea behind Jacobi's method is to systematically reduce the quantity

$$d(\cdot) = \sqrt{\sum_{i=1}^n \sum_{\substack{j=1 \\ j \neq i}}^n a_{ij}},$$

i.e., the Frobenius norm of the off-diagonal elements. The tools for doing this are rotations of the form

$$J(p, q, f) = \begin{bmatrix} 1 & \dots & 0 & \dots & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \dots & c & \dots & s & \dots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \dots & -s & \dots & -c & \dots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \dots & 0 & \dots & 1 \end{bmatrix} \begin{matrix} p \\ & q \\ p & q \end{matrix}$$

which we call Jacobi rotations. Jacobi rotations are different from Givens rotations; see §5.18. We submit to the name change in this section to honor the inventor.

The basic step in a Jacobi eigenvalue procedure involves (i) choosing an index pair (p, q) that satisfies $1 \leq p, q \leq n$, (ii) computing a cosine-sine pair (c, s) such that

$$\begin{bmatrix} b_p & b_q \\ b_p & b_q \end{bmatrix} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a_{pp} & a_{pq} \\ a_{qp} & a_{qq} \end{bmatrix} \begin{bmatrix} c & \\ -s & iJ \end{bmatrix} \quad (85.1)$$

is diagonal, and (iii) overwriting \mathbf{u} with $\mathbf{B} = \mathbf{J} \mathbf{T}_{(q)}^{-1}$, where $\mathbf{J} = J(p, q, f)$. Observe that the matrix \mathbf{B} agrees with \mathbf{u} except in rows and columns p and q . Moreover, since the Frobenius norm is preserved by orthogonal transformations, we find that

$$a_p^2 + a_q^2 + 2a_{pq}^2 = b_p^2 + b_q^2 + 2a_{pq}^2 = b_p^2 + b_q^2$$

It follows that

$$\begin{aligned} d(\mathbf{B})^2 &= \|\mathbf{B}\|_F^2 = \sum_{i=1}^n b_{ii}^2 = \|\mathbf{d}\|_F^2 = \sum_{i=1}^n a_{ii}^2 + (a_p + a_q)(b_p - b_q) \\ &= d(\cdot)^2 - 2a_{pq} \end{aligned} \quad (85.2)$$

It is in this sense that \mathbf{u} moves closer to diagonal form with each Jacobi step.

Before we discuss how the index pair (p, q) can be chosen, let us look at the actual computations associated with the (p, q) subproblem.

$\text{q} = \text{m} \cdot \text{A} - \text{A} \cdot \text{m}$ $\text{L} = \text{I} - \text{B}^{-1} \cdot \text{A}$ $\text{Li} = \text{E} = \text{A} \cdot \text{G}$ $\text{N} = \text{A} \cdot \text{G} \cdot \text{D} \cdot \text{A}$

To say that we diagonalize in (851) is to say that

$$\text{O} = b_{pq} = a_{pq}(c^2 - s^2) + (a_{pp} - a_{qq})cs. \quad (853)$$

If $a_{pq} = 0$ then we just set $c \neq 1$ and $s = 0$. Otherwise, define

$$\tau = \frac{a_{qq} - a_{pp}}{2a_{pq}} \quad \text{and} \quad t = s/c$$

and conclude from (853) that $t = \tan(\theta)$ solves the quadratic

$$t^2 + 2\tau t - 1 = 0.$$

It turns out to be important to select the smaller of the two roots:

$$t_{\min} = \begin{cases} 1/(\tau + \sqrt{1 + \tau^2}) & \text{if } \tau \geq 0 \\ 1/(\tau - \sqrt{1 + \tau^2}) & \text{if } \tau < 0 \end{cases}$$

This implies that the rotation angle satisfies 81: $\pi/4$ and has the effect of maximizing c :

$$c = 1/\sqrt{1 + t_{\min}^2}, \quad s = t_{\min} c.$$

This in turn minimizes the difference between A and the update B :

$$\| \text{B} - \text{A} \|_F^2 = 4(1 - c)^2 + (a_{ip}^2 + a_{iq}^2) + 2a_{pq}^2/c^2.$$

We summarize the 2by2 computations as follows:

Algorithm 8.5.1 Given an n by n symmetric A and integers p and q that satisfy $1 \leq p < q \leq n$, this algorithm computes a cosine-sine pair $\{c, s\}$ such that if $\text{B} = \text{J}(\text{p}, \text{q}) \text{G}(\text{p}, \text{q}) \text{A}(\text{p}, \text{q}) \text{G}(\text{p}, \text{q}) \text{J}(\text{p}, \text{q})$, then $b_{pq} = b_{qp} = 0$.

```

function [c, s] = synSchur2(A, p, q)
    if A(p, q) == 0
        t = (A(q, q) - A(p, p)) / (2 * A(p, q))
        if t == 0
            t = 1 / (sqrt(1 + t^2))
        else
            t = 1 / (sqrt(1 - t^2))
        end
        c = 1 / sqrt(1 + t^2), s = tc
    else
        c = 1, s = 0
    end
end

```

0t4mlA T- cA .2 }+G} 2P; iAGA r2. .7 G - 1A

As we mentioned above, only rows and columns p and q are altered when the (p, q) subproblem is solved. Once $\text{symch}u2$ determines the 2 by 2 rotation, then the update $J(p, q)^\top T A J(p, q)$ can be implemented in 6n fops if symmetry is exploited.

How do we choose the indices p and q? From the standpoint of maximizing the reduction of $\|A\|_F$ in (852), it makes sense to choose (p, q) so that a_{pq} is maximal. This is the basis of the classical Jacobi algorithm.

pe Is hli- 8.5.2 el 5T.11T ap2T16s1pp Given a symmetric $A \in \mathbb{R}^{n \times n}$ and a positive tolerance td , this algorithm overwrites A with VTAV where V is orthogonal and $\|VTAV\|_F \leq \|A\|_F$

$$V = I_n O \quad td \cdot \|A\|_F$$

while $\|A\|_F > x$

Choose (p, q) so $|a_{pq}| = \max_{i \neq j} |a_{ij}|$

$$[c, s] = \text{symch}u2(A, p, q)$$

$$A = J(p, q)^\top T A J(p, q)$$

$$V = V J(p, q)$$

end

Since $|a_{pq}|$ is the largest off-diagonal entry,

$$\|A\|_F^2 \leq N(a_{pq}, a_p)$$

where

$$N = \frac{n(n-1)}{2}.$$

From (852) it follows that

$$\|A\|_F^2 \leq (1 - \frac{1}{N}) \|A\|_F^2.$$

By induction, if $A^{(k)}$ denotes the matrix A after k Jacobi updates, then

$$\|A^{(k)}\|_F^2 \leq \left(1 - \frac{1}{N}\right)^k \|A^{(0)}\|_F^2.$$

This implies that the classical Jacobi procedure converges at a linear rate.

However, the asymptotic convergence rate of the method is considerably better than linear. Schonhage (1964) and van Kempen (1966) show that for k large enough there is a constant c such that

$$\|A^{(k)}\|_F^2 \leq \left(1 - \frac{1}{N}\right)^k \|A^{(0)}\|_F^2$$

i.e., quadratic convergence. An earlier paper by Henrici (1958) established the same result for the special case when A has distinct eigenvalues. In the convergence theory for the Jacobi iteration, it is critical that $|Q| \geq 7/4$. Among other things this precludes the possibility of interchanging nearly converged diagonal entries. This follows from

the formulae $b_{pq} = a_{pq} - t a_{pp} a_{qq}$ and $b_{qq} = a_{qq} - t a_{pp}$ which can be derived from Equation (85.1) and the definition $t = \sin(\theta)/\cos(\theta)$.

It is customary to refer to N Jacobi updates as a sweep. Thus, after a sufficient number of iterations, quadratic convergence is observed when examining of $\{\Lambda\}$ after every sweep.

There is no rigorous theory that enables one to predict the number of sweeps that are required to achieve a specified reduction in of $\{\Lambda\}$. However, Brent and Luk {1985} have argued heuristically that the number of sweeps is proportional to $\log(n)$ and this seems to be the case in practice.

8.5.4 $r^{1/2} \|P\|_F \leq \|\Lambda\| \leq \|P\|_F \sqrt{r}$

The trouble with the classical Jacobi method is that the updates involve $O(n)$ flops while the search for the optimal $\{p, q\}$ is $O(n^2)$. One way to address this imbalance is to fix the sequence of subproblems to be solved in advance. A reasonable possibility is to step through all the subproblems in row-by-row fashion. For example, if $n = 4$ we cycle as follows:

$$\{p, q\} = (1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4), (1, 2), \dots$$

This ordering scheme is referred to as cyclic by row and it results in the following procedure:

Algorithm 8.5.3 Given a symmetric matrix $A \in \mathbb{R}^{n \times n}$ and a positive tolerance tol , this algorithm overwrites A with $V^T A V$ where V is orthogonal and $\|(V^T A V)\| \leq tol \cdot \|A\|$.

```

V = I_n
S = tol * \|A\|
while |A| > S
    for p = 1:n - 1
        for q = p+1:n
            [c, s] = symdu2(A, p, q)
            A = J(p, q) T A J(p, q)
            v = v J(p, q)
        end
    end
end

```

The cyclic Jacobi algorithm also converges quadratically. (See Wilkinson {1962} and van Kempen {1986}.) However, since it does not require off-diagonal search, it is considerably faster than Jacobi's original algorithm.

8.5.5 Use $B \hat{\otimes} P$ as $\hat{\Lambda}$ in P

Using Wilkinson's error analysis it is possible to show that if r sweeps are required via Algorithm 8.5.3 and d_1, \dots, d_r specify the diagonal entries of the final, computed in

matrix, then

$$\sum_{i=1}^n (d_i - \lambda_i)^2 \leq (\text{tol} + k_r u) \|A\|_F$$

for some ordering of A's eigenvalues λ_i . The parameter k_r depends mildly on r .

Although the cyclic Jacobi method converges quadratically, it is not generally competitive with the symmetric QR algorithm. For example, if we just count flops, then two sweeps of Jacobi are roughly equivalent to a complete QR reduction to diagonal form with accumulation of transformations. However, for small n this liability is not very dramatic. Moreover, if an approximate eigenvector matrix V is known, then $V^{-1}AV$ is almost diagonal, a situation that Jacobi can exploit but not QR.

Another interesting feature of the Jacobi method is that it can compute the eigenvalues with small relative error if A is positive definite. To appreciate this point, note that the Wilkinson analysis cited above coupled with the §8.1 perturbation theory ensures that the computed eigenvalues $\hat{\lambda}_i$ satisfy

$$\frac{|\hat{\lambda}_i - \lambda_i(A)|}{\lambda_i(A)} \approx u \frac{\|A\|_2}{\lambda_i(A)} \leq u \kappa_2(A).$$

However, a refined, componentwise error analysis by Demmel and Veselic (1992) shows that in the positive definite case

$$\frac{|\hat{\lambda}_i - \lambda_i(A)|}{\lambda_i(A)} \approx u \kappa_2(D^{-1}AD^{-1}) \quad (854)$$

where $D = \text{diag}(f_{11}, \dots, f_{nn})$ and this is generally a much smaller approximating bound. The key to establishing this result is some new perturbation theory and a demonstration that if A_+ is a computed Jacobi update obtained from the current matrix A_c , then the eigenvalues of A_+ are relatively close to the eigenvalues of A_c in the sense of (854). To make the whole thing work in practice, the termination criterion is not based upon the comparison of $\|A\|_F$ with $u\|A\|_F$ but rather on the size of each $|a_{ij}|$ compared to $u\sqrt{a_{ii}a_{jj}}$.

8.5.6 LBB If $B \approx A$ è è

It is usually the case when solving the symmetric eigenvalue problem on a processor machine that $n \gg p$. In this case a block version of the Jacobi algorithm may be appropriate. Block versions of the above procedures are straightforward. Suppose that $n = rN$ and that we partition the n -by- n matrix A as follows:

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & & \vdots \\ A_{N1} & \cdots & A_{NN} \end{bmatrix}.$$

Here, each A_{ij} is r -by- r . In a block Jacobi procedure the (p, q) subproblem involves computing the $2r$ -by- $2r$ Schur decomposition

$$\begin{bmatrix} V_{pp} & V_{pq} \\ V_{qp} & V_{qq} \end{bmatrix}^T \begin{bmatrix} A_{pp} & A_{pq} \\ A_{qp} & A_{qq} \end{bmatrix} \begin{bmatrix} V_{pp} & V_{pq} \\ V_{qp} & V_{qq} \end{bmatrix} = \begin{bmatrix} D_{pp} & 0 \\ 0 & D_{qq} \end{bmatrix}$$

and then applying to A the block Jacobi rotation map V up of the V . If we call this block rotation V , then it is easy to show that

$$\operatorname{cf}(V^T A V)^2 = \operatorname{cf}(A)^2 \left(2\|A_{pq}\|_F^2 + \operatorname{cf}(A_p)^2 + \operatorname{cf}(A_q)^2 \right).$$

Block Jacobi procedures have many interesting computational aspects. For example there are several ways to solve the subproblems, and the choice appears to be critical. See Bischof (1987).

8.5.7 $\|\cdot\|_F$ is $\|\cdot\|_F$ $\|\cdot\|_F$ $\|\cdot\|_F$ $\|\cdot\|_F$ $\|\cdot\|_F$

The Block Jacobi approach to the symmetric eigenvalue problem has an inherent parallelism that has attracted significant attention. The key observation is that the $(1, 1)$ subproblem is independent of the $(2, 2)$ subproblem if the four indices i_1, i_2 and j_2 are distinct. Moreover, if we regard the A as a $2m$ -by- $2m$ block matrix, then it is possible to partition the set of off-diagonal index pairs into a collection of $2m - 1$ rotation sets, each of which identifies m nonconflicting subproblems.

A good way to visualize this is to imagine a chess tournament with $2m$ players in which everybody must play everybody else exactly once. Suppose $m = 4$. In "round 1" we have Player 1 versus Player 2, Player 3 versus Player 4, Player 5 versus Player 6 and Player 7 versus Player 8. Thus, there are four tables of action:

1	3	5	7
2	4	6	8

This corresponds to the first rotation set:

$$\text{rot.set}(1) = \{ (1, 2), (3, 4), (5, 6), (7, 8) \} \quad g \quad s \quad a$$

1	2	3	5
4	6	8	7

=

1	4	2	3
6	8	7	5

1	6	4	2
8	7	5	3

1	8	6	4
7	5	3	2

1	7	8	6
5	3	2	4

1	5	7	8
3	2	4	6

=

To align in order, the seven rotation sets define the parallel ordering of the 28 possible off-diagonal index pairs.

For general or a multiprocessor implementation would involve solving the subproblems within each rotation set in parallel. Although the generation of the subproblem rotations is independent, some synchronization is required to carry out the block similarity transformation updates.

Problems

P8.5.1 589968.ot.. .8 d..8. .ot86D6968.9.b

[)]

1 (:xk 1 Ifx bywDob 1 Ifc

$$J = \begin{bmatrix} C \\ -S \end{bmatrix}$$

$$\frac{w - \gamma}{\tau^2 - 2x\tau + z - \gamma} = 2a$$

• OhTheta class = AET UCRUUCT-R, ERUUTk FARESUT, =RUT=cA= . 1, LCAEA1R), . 2

P8.5.2 589 ∈ F xp.A = 1TAUET x22AR)R = 1ET1OTRHT1UA = 1xARx1JETxRUT

$$c_1 = \gamma I + E$$

z = M5p. **F.1** @2 kAQ mAk tkjk Vn

P8.5.3 leot.98: 8....1.8fe.8. N. R.5. 6 ...2892.9...8. .f R.51..ot8bP8.2.2...

P8.5.4 3..9.9.8.96 1.. .8.pt2289... 2.9.b -Notate86

$$\left[\begin{array}{cc} 7 & 1 \\ A_1 & n^1 \end{array} \right] _1$$

L J n0 3= .00• C~~X~~₀⁰=0d o: EB = L~~J~~₀⁰= zB(3₀, 1) t - 8AU = J(l, 2, :M
 $\tau N, C -)N$ t, - G = JTBJ, UyA)c₀² = 2) = G_n x ~~22~~ + C(L_{en} x x + Ir 12 8R14,A
 (196) ET XRUAR) R = EEUUYE UCA₁ ITBAUETATA) 2RXAT ,kA, 40) FAIAUTUT) UyE
 S, IAC (= AE3U(T aT MURUK

P8.5.5 f.8. .2..8.8.9..d 968.ot..k 8..otd1.9. .9..8....8 98 ..968..... 9.8.8>
 . .2 28f2.e... 8.. 9.. .828.2..ot688. f8.8.f8.2..898. .8..e.8 2.8.89.8fe.9.8.. 1 R5
 . .19 fog.9.9.8.8.911... .8.f. 986.98f 9.8thr shdd Jacdi method iAURTa(s)aAEf)l
 VT. 2EIR(UURa(Tr-RX|ETUyIR1,A 14), T)IIX.T)=0 6(,P 1277). 6t(n,(,-. N
 τI .(: .+) ... (N, N.N

P8.5.6 N.8. . .8.9.8 .98d8_m, :M Mt (2m - 1)m. N:M (.) (.: (- .1 ,1.P -) -) M M,

No and References for §8.5

1.. 8..d..ot..8. .. 8.88b968..ot.8.9 .8..8.Ne.f .. 98.e28..... .,..... ..98..9e.8

k..8.. (1846. ®MAAT8ATACUAARCEATAT) AE,yA(ETA,AE+Ra,XRE=UE(4)ASESb
TA(Ar))=AtaC,)IA),baET=a_{10x4}S=AP Cr lle's J. 90 51-94

IS CAT- R=IETIPUCARaS,IACT)- Alk UCAUR),RBAUy(IE=(-2)IA)=A= ITTAhTa
)=AE(XAT=,(REXFAAFa)aAbfa=AA

k1 Tr=r4.y, rEdBb1kmt=er(=)p 5R(r20)=rou uo=rip(5 Quart. Appl. Math. 14, I 7F ITx E:3; hLSAi83 A: ll.S qcatC. TSMnib.c.n8r f 8A2d8t. ll2i2c18289hSAMcA82213Uj. sSAAB Ecir l8i,c12d8st.A2McA12McA1Atna9iASciIA2213Uj. iACM 4 ..r ,7 7x A:E ni w,33ACN7,SVL AztAtl. 3zr IMc22Muc83ca22A9cr AaiiziAt 9i83A8iAM. lAi, st.A2McA12MnA At ACM 11, .5r .dx .: ll2c2iItA1 772'12A9iA1sA2213MIAizst.A2MnA12McA1At Numer. Math. 9, N JN sAAztl anzq818 = 39hS:I7.Si838

9... an,q818 CN7dM3zlt2 Ecir l8i,i2McA122 n z8,AM A,AM Ai8i tU6,in. Alg Applic. 1 Nr ,IM

hiSAM2212MA18AAM8A23,i3Mi2cAl8iAMr A8AA13A8

h A8MnA At dSM",8 22AsSAAB Al8iAMr A8AA1Azd83 V,cAAzC A r iA182218 M Al,S,2c8r 22A nr A8iizlAtAM,n2c182McA18iAM J. Appl. Math. 6, N,r N71: 91: i8tA8 C. 7ls ",8 V,cAtAzc9iAIShA22BtU6. ACM 9, NNN5.M 9.: ac,q818 CN.7f. 212A18 22A,i3Mi2cAl8iAMr A8AA1AtAzc9iAIShMIAAtNumer. Math 6 I.7r 5Jx

91. i8tA8 CN75x,a AtAzc9iAIShA2213U6 SIAM J. Appl. Math. 11, „d „M

3 sAml8mriNv,s „8 22A,i3Mi2cAl8iAMr A8AA1AisA1ShMIAAtNumer. Math 6 NJ,,IM

.:h1: iia 5A.SA8N0x ",8 VIi3Mi2A8iAMrA8 2mASAAmA,cA r iA182218 Numer. Math 9 Nr IIM

h A8MnA865: -c.AM,i8 C. 7dx.3a 9t2c.i2AM22,2IM.tl. AAM21A1Az9iAISc ,SAMi 2IM6,in. Alg. Applic. 1,d „NM

5:a: wMI3i831:9.E.hA C. Tsx.,a 22A18iAMr AaAA1Az9iAIShA2213Uj. Inst. Math Applic. 15, IT.TdT.

12A1M3AMc22A1SSMISzA22ci tAA8t.c.SIM2i82

a:u: I.AiMA82 CN. S M 22A18iAMr A8AA1A A 1218c,M 3MS c2M1M1AMcar.uT SIAM J. Matr. Anal. Applic. 16 N, 0'NI.M

-: EMi,i CN.7SM",8 22A183n2d8A2iinl,M822)9iAISd,2213U SIAM J. Matr. Anal. Applic. 17, J. N.;

B: iIMcdJJ5MA18iAMr A8AAwzlAq,MnH82A3tAzc9iAISmA2213U SIAM J. Matr. Anal. Applic. 29, 5,r 57x

z. := Xa x 20 Dyn < = ea 8a> 0= X=Q8eEdeEa (Fa EHv(J= 3(z))A= Ej= za z0= 8 r SIAM J. Matr. Anal. Applic. 31, N5r N5

EA2ic,A3Mn8MtA122i212iSzct2, 2m i2A,MiAt9iASni.A2mE88Az,3A8

9xiMU1839:EA..Az CN.Jsx.ALS,2n8r3AA,Mi2A A8ttt2A tsAi,A,ENir l8i,E1.n8ap ni2McA18iAM J. Numer. Anal. 27UT71r N x

9:a: EA..Az i13 5: BAAzC. J sx.9iAISciIA2213nhtlM8AAIM228VIU SIAM J. Matr. Anal. Applic. 13 NJ, NI,M

au: I.AiMA82 CN. SM3212A8 9iAISwAc81M8AA M228VIU siAM J. Matr. Anal. Applic. 15, INr Ndx

l: i122n. CN. S .3AA Mi2A A8ttt2A ls 22dnSt9iAIShA2213U SIAM J. Matr x Anal. Applic. 16, TT N5x

5: BAA,c,C. 7SM'3 212A18 22A3AA,MiAt st.)2McA9nr A8M,3A2d8M12i ITNA 4, 5T, M

u:l: EIScA19in: II,AMii1839:IIIMcdJJ5x"38,M22lr l8izr 2A212c18A Mi8r1Mnpmp M22At A2M,9r A8M,i \$IAM J. Matrix Anal. Applic. 25

... Numer. Math. 113, r 5

322,S2t2iA8A8 i2AAb2A82A9iAISc2AMi221822HMtAt l. i2McA18321 SI: 22M11rA1MMAtSI838AMr A8AA12A AitA. 81M.iz.i2McA,nt3tAI tt,388

... dz3t2n83n.h.lMI.c2. CN. Sm3hMIAA3M A 22A18i12E2M.iZi2Mc u. J. ACM 6 NTN.;

c: nln.II CNTM,,8 22A1i3Mi2A8iAMr A8AA1A 9iA182213M 2IM.iz i2MnA. Comput. J. 15, IT,IT7x

- at 11aPonTemten - 8iyi2 b4 li+e1dWe.W+ri=k1 iGme3Gillesip ussingG5 Numer.
 Math. 17, NgJJI,
 A. j.: 0.0+ p18zJO a : =ea -z(0 = 100E0E HAz J-e-' = = Xa a HaT=7= OEC
 5JG5151
 o. 3a 8.- epoeEn(1+0z J-e :=0c 03'0eXX0Ha= fEl HAM J. Matrix Anal. Applic.
 14, NGVd:
 d Aa211hp2A8i1t,c ia3i1r IMcp2MAA1S.A8(SM,)82A8p2ApAiMiMAM122AlM.i,
 A Acp2.calM13: Ai2cd8r IMla8IM.i1.ipMcAaRAc.ipnla,Ala,c3HMiSMA3c1A,,2
 99MAA82iepN , .3 nA22l3Mr caocdp, l. 3MScpMiM1A8 Math. Tables and Other
 Aids to Comp. 9 ,TGM
 - .9, ISAMg7, .8 hciar ,iMchi2hlaAlS 1AbniIMcAS, l.l NcAa,cl8i 18c2Mhi82
 ,M.i pcd8k6T 5 I,JGI,,
 9.fllp2Mlt683h449SAMzActg .sl12cla pl p29r r A8SMIS,A2IMtlaQoAn8r3AlSn2
 1tSAnAp2l3oi 8S3llq.k6Numer. Math. 11, NGNI:
 3, llm+ Ndx.,8 22A,33MipA8iAMr A8AA8AM3lbipl22AAIScnAp2l3pl3MScpMiMt
 IipMcAaRk6 8, INGI,,
 A. j.: • Q. p.pnD.JO = = XAHe= ,lad0,0(X (= (ze 8 + e-z)KMT 9 .5G.d,
 a.9, 9SAM1ANgJu .sl1pn18pl p2AAIS 1Ab9cr)8SMIS 1t i :IM lAaQoAn8rAlSc.2tSA
 nA22l6Numer. Math. 14, 15I Ge.
 - xh,.i8r +gT, .3 .iAlSr ltSA22lqMhc38r ,Mcbc883MSc2MiyfMch16
 Numer. Anal. 12, 77GTJu
 xMc+ Ng, .8 22A1S11AlaiAMr)8Ap2)9 AM1A22lqM1Ai 1nipMcA1,Numer.
 Math. 39, 7μGT,
 ..a, spAiM2gd x.3 .iAlSc,cq) 31r M2 ,MAIS,2c8rp2AA2,NAAlSl,pcplB i 2la2
 2AM,n2i8pMk6 SIAM J. Sci. Stat. Comput. 6 d,GdTM
 - :IA2zIJJdk.,8 3tSpl2cAAl8iMr BM)218,t.AIMr AAISn81r IMcp26 SIAM J. Matr
 Anal. Applic. 30 IgNEN μ,
 9-1Sc A22l3 ,MAIS 1Abt.HpMn8pMcA2iA31l SAA2AiA 1SA,4T
 9.9sAip18μggi .Ncp l8izcbi2t8AlS 1Abst.ApMcA13pMcAA,8i nl3n,Ao iAlS2pm13ki i
 Comput. J. 12, μ.T.N.T.
 a2, 9SAM1Ag8μ ,8 2yAncir l8iznbiph1Al. 1Abst..A2McAni2McA6 J. I st. Math
 Applic. 7, T,8CM
 a, 3a3AM,1883, ,lcb, ,NgT, .8 22A,icMipcaAlaiAMr A8AA31r Mc22a2cA2Nci 12
 8i inBA, AlS 1b st.ApMcAa12Mk6b i 1 st. Math. Applic. 12, I 7μGT μ,
 h 3a3AM,1883, ,lr bII,,gT x.3 .iAlSc,ltSHn)p7In, MAIS 1abit.ApMnAi2McAAi832
 SIIq.k6numer. Math. 25, „TG, 7,
 pmA8rcd8A13Ae
 ,4 niAqAt,μg , .i.c1pl8 >8a>AlSc nA22eVIi2AM8d183p29cr Aaii1AMIS 1Ai
 SIAM J. Matrix Anal. Applic. 16kI NG .x
 9aflei8Abtq IJJ .38 8S4cp-AlSc,zcqA22lqMAlS,2c8r.A8AMi 1dbASAMS11A
 sBNk6n. Alg. Applic. 358, Ig, JT,
 JMi ,i.S ln8d. S3SAM18AAM8A2i3Md,CSAA2, SiM31zA1Sr,AAe
 3: si.A2 ,NgT, .8 .iAlSci83.iAIi n.1c141Mcp2.Mi iMiZ4LSSpA16 Math. Comput.
 25 .Tg,gJx
 4xs,Alppk,IC.Ai 221831:A, aiMn,Ng7, .hiMi,zA1AqiAlSc9cr A8ii1Ar Mcp2a,c8r
 st.2,na3MM3t,in6Alg. Applic. 77, „ ,7x
 a4 2SAM1ArNg, .8 a,car p21ScnA22ld8i otSAMA,S)18 hypercube Multipr ccessors,
 1,1 ,oApip2A3.ks83nh,S 1r Ai2cl2k3zS2niM
 g. OJ= -aa=jkQz= 0(ihAz,p Di=z JQ= 8 >0= = 108E0E Da (eE= hQ= Je= Xa = a a"
 ,A= Ec i=LTA(N8J. Matrix Anal. Applic. 10, I,7G5,7
 .. ,A,hiuM3AqllS)MEN .3 VlioMipcaAl8iAMr AiiMi,zA1Sc hMIAM,Ncir lai 1zt
 N.c818p nipMcA22 2la3,2c6crA283k6.in. Alg. Applic. 145, TNddM
 T. & = = a = = q = = On JJA .+20 pDnXO= (E= q = nOh (= x=0= a=aa E=0104(J, N) SIAM J.
 Matr x Anal. Applic. 26 gd. NJJ:

8.6 Computing the SVD

If $U\Lambda V = B$ is the bidiagonal decomposition of $A \in \mathbb{R}^{mn}$, then $V^T(\Lambda\Lambda)V = B^TB$ is the tridiagonal decomposition of the symmetric matrix $\Lambda\Lambda \in \mathbb{R}^{mn}$. Thus, there is an intimate connection between Algorithm 542 (Householder bidiagonalization) and Algorithm 831 (Householder tridiagonalization). In this section we carry this a step further and show that there is a bidiagonal SVD procedure that corresponds to the symmetric tridiagonal QR iteration. Before we get into the details, we catalog some important SVD properties that have algorithmic ramifications.

ibnmrb .kl\\$ ff Tktebf kbf esbuK. \\$ f .TfbbT" \\$f t PDSb kPSb

There are important relationships between the singular value decomposition of a matrix A and the Schur decompositions of the symmetric matrices

$$S_1 = \Lambda\Lambda, \quad S_2 = AA^T, \quad S_3 = \begin{bmatrix} 0 & A^T \\ A & 0 \end{bmatrix}.$$

Indeed, if

$$U\Lambda V = \text{diag}(a_1, \dots, a_p)$$

is the SVD of $A \in \mathbb{R}^{mn}_{(r=1)p}$, then

$$V^T(\Lambda\Lambda)V = \text{diag}(a_r, \dots, a_s, 0, \dots, 0) \in \mathbb{R}^{mn} \quad (861)$$

and

$$U^T(AA^T)U = \text{diag}(a_r, \dots, a_s, 0, \dots, 0) \in \mathbb{R}^{mn} \quad (862)$$

Moreover, if

$$U = \begin{bmatrix} U_1 & U_2 \end{bmatrix}_{n \times mn}$$

and we define the orthogonal matrix $Q \in \mathbb{R}^{(mn) \times (mn)}$ by

$$Q = \frac{1}{\sqrt{2}} \begin{bmatrix} V & V & 0 \\ U_1 & -U_1 & \sqrt{2}U_2 \end{bmatrix},$$

then

$$QT \begin{bmatrix} & T \end{bmatrix}_Q = \text{diag}(a_1, \dots, a_n, -a_1, \dots, -a_n L_{mn}). \quad (863)$$

These connections to the symmetric eigenproblem allow us to adapt the mathematical and algorithmic developments of the previous sections to the singular value problem. Good references for this section include Lawson and Hanson (SL) and Stewart and Sun (MPT).

8.6.2 1ç i è à Ä B/A Ü ß ç Ü ppblè B á iç Ä è þ

We first establish perturbation results for the SVD based on the theorems of §§1. Recall that $a_i(A)$ denotes the i th largest singular value of A .

Theorem 8.6.1. If $A \in \mathbb{R}^{m \times n}$, then for $k = 1 : \min\{m, n\}$

$$\sigma_k(A) = \min_{\dim(S)=n-k+1} \max_{\substack{x \in S \\ y \in \mathbb{R}^m}} \frac{y^T A x}{\|y\|_2 \|x\|_2} = \max_{\dim(S)=k} \min_{x \in S} \frac{\|Ax\|_2}{\|x\|_2}.$$

In this expression, S is a subspace of \mathbb{R}^n .

Proof The rightmost characterization follows by applying Theorem 8.1.2 to ATA . For the remainder of the proof see Xiang (2006). \square

Corollary 8.6.2. If A and $A+E$ are in $\mathbb{R}^{m \times n}$ with $m \leq n$, then for $k = 1 : n$

$$|\sigma_k(A+E) - \sigma_k(A)| \leq \sigma_1(E) = \|E\|_2.$$

Proof Define A and E by

$$A = \begin{bmatrix} 0 & AT \\ A & 0 \end{bmatrix}, \quad A+E = \begin{bmatrix} 0 & (A+E)T \\ A+E & 0 \end{bmatrix}. \quad (864)$$

The corollary follows by applying Corollary 8.1.6 with A replaced by A and $A+E$ replaced by $A+E$. \square

Corollary 8.6.3. Let $A = [a_1 | \cdots | a_n] \in \mathbb{R}^{m \times n}$ be a column partitioning with n . If $A_r = [a_1 | \cdots | a_r]$, then for $r = 1 : n-1$

$$a_1(A_{r+1}) \geq a_1(A_r) \geq a_2(A_{r+1}) \geq \cdots \geq a_r(A_{r+1}) \geq a_r(A_r) \geq Q_{r+1}(A_{r+1}).$$

Proof Apply Corollary 8.1.7 to ATA . \square

The next result is a Widder-Hoffman theorem for singular values.

Theorem 8.6.4. If A and $A+E$ are in $\mathbb{R}^{m \times n}$ with $m \leq n$, then

$$\sum_{k=1}^n (a_k(A+E) - a_k(A))^2 \leq \|E\|_F^2.$$

Proof Apply Theorem 8.1.4 with A and E replaced by the matrices A and E defined by (864). \square

For AER $\mathbb{R}^{n \times n}$, we say that the k-dimensional subspaces S $\subset \mathbb{R}^n$ and T $\subset \mathbb{R}^n$ form a singular subspace pair if $x \in S$ and $y \in T$ imply $Ax \in T$ and $ATy \in S$. The following result is concerned with the perturbation of singular subspace pairs.

Theorem 8.6.5. Let A $\in \mathbb{R}^{n \times n}$ with $n = r + m$ be given and suppose that VER $\mathbb{R}^{n \times n}$ and UER $\mathbb{R}^{n \times n}$ are orthogonal. Assume that

$$V = \begin{bmatrix} V_1 & V_2 \end{bmatrix}_{r \times nr}, \quad U = \begin{bmatrix} U_1 & U_2 \end{bmatrix}_{r \times mr},$$

and that $\text{ran}(V)$ and $\text{ran}(U)$ form a singular subspace pair for A. Let

$$\text{UTAV} = \begin{bmatrix} A_{11} & 0 \\ r & A_{22} \end{bmatrix}_{mr \times nr}, \quad \text{UREV} = \begin{bmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{bmatrix}_{mr \times nr},$$

and assume that

$$8 = \min_{\substack{\sigma \in \sigma(A_{11}) \\ \gamma \in \sigma(A_{22})}} \|A - \gamma I\| > 0.$$

If

$$\|E\| \leq \frac{8}{5}$$

then there exist matrices PER $(nr) \times r$ and QER $(mr) \times r$ satisfying

$$\left[\begin{array}{c} P \\ Q \end{array} \right] L \in \mathbb{R}^{(nr+mr) \times r} \quad \text{such that } \|L\| \leq 4 \frac{\|E\|_F}{\delta}$$

such that $\text{ran}(U_1 + VQ)$ and $\text{ran}(U_2 + VP)$ is a singular subspace pair for $A + E$.

Proof. See Stewart (1973, Theorem 64). \square

Roughly speaking, the theorem says that $O(\epsilon)$ changes in A can alter a singular subspace by an amount $/8$ where 8 measures the separation of the associated singular values.

8.6.3 $\|x\|_{\text{F}} \leq \|Ax\| \leq \|A\|_{\text{F}}$

We now show how a variant of the QR algorithm can be used to compute the SVD of an AER $\mathbb{R}^{n \times n}$ with $n = r + m$. At first glance, this appears straightforward. Equation (8.6.1) suggests that we proceed as follows:

Step 1 Form $C = ATA$,

Step 2. Use the symmetric QR algorithm to compute $VtCV = \text{diag}(\lambda)$,

Step 3 Apply QR with column pivoting to AV obtaining $U\tilde{V}(AV)I = R$

Since R has orthogonal columns, it follows that $U^T A(V_1 \Pi)$ is diagonal. However, as we saw in §5.3.2, the formation of $A^T A$ can lead to a loss of information. The situation is not quite so bad here, since the original A is used to compute U .

A preferable method for computing the SVD is described by Golub and Kahan (1965). Their technique finds U and V simultaneously by implicitly applying the symmetric QR algorithm to $A^T A$. The first step is to reduce A to upper bidiagonal form using Algorithm 5.4.2:

$$U_B^T A V_B = \begin{bmatrix} & \end{bmatrix}' \quad B = \begin{bmatrix} d_1 & J & & \cdots & 0 \\ 0 & d_2 & \ddots & & \vdots \\ & \ddots & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & f_n \\ 0 & \cdots & 0 & d_n & 1 \end{bmatrix} \in \mathbb{J}' \times \mathbb{n}$$

The remaining problem is thus to compute the SVD of B . To this end, consider applying an implicit-shift QR step (Algorithm 8.3.2) to the tridiagonal matrix $T = B^T B$:

Step 1. Compute the eigenvalue λ of

$$T(m:n, m:n) = \begin{bmatrix} d_m^2 + f_{m-1}^2 & d_m f_m \\ d_m f_m & d_n^2 + f_m^2 \end{bmatrix}, \quad m = n-1,$$

that is closer to $d_n^2 + f_m^2$.

Step 2 Compute $c_1 = \cos(\theta)$ and $s_1 = \sin(\theta)$ such that

$$\begin{bmatrix} c_1 & s_1 \\ -s_1 & c_1 \end{bmatrix}^T \begin{bmatrix} d_1^2 - \lambda \\ d_1 f_1 \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}$$

and set $G_1 = G(1, 2)$.

Step 3. Compute Givens rotations G_2, \dots, G_{n-1} so that if $Q = G_1 \cdots G_{n-1}$ then $Q^T T Q$ is tridiagonal and $Q e_1 = G_1 e_1$.

Note that these calculations require the explicit formation of $B^T B$, which, as we have seen, is unwise from the numerical standpoint.

Suppose instead that we apply the Givens rotation G_1 above to B directly. Illustrating with the $n = 6$ case, we have

$$B_{\cdot BG_1} = \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ + & x & x & 0 & 0 & 0 \\ 0 & 0 & x & x & 0 & 0 \\ 0 & 0 & 0 & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix}.$$

We then can determine Givens rotations $U_1, V_1, U_2, \dots, V_{n-1}$, and U_{n-1} to chase the unwanted nonzero element down the bidiagonal:

$$B + U_1^T B = \begin{bmatrix} x & x & + & 0 & 0 & 0 \\ 0 & x & x & 0 & 0 & 0 \\ 0 & 0 & x & x & 0 & 0 \\ 0 & 0 & 0 & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix}, \quad B \leftarrow BV_2 = \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ 0 & x & x & 0 & 0 & 0 \\ 0 & + & x & x & 0 & 0 \\ 0 & 0 & 0 & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix},$$

$$B + U_2^T B = \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ 0 & x & x & + & 0 & 0 \\ 0 & 0 & x & x & 0 & 0 \\ 0 & 0 & 0 & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix}, \quad B + BV_3 = \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ 0 & x & x & 0 & 0 & 0 \\ 0 & 0 & x & x & 0 & 0 \\ 0 & 0 & + & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix},$$

and so on. The process terminates with a new bidiagonal f that is related to B as follows

$$B = (U_1^T \cdots U_k^T) B (G_1 V \cdots V_{k-1} d + U^T B V).$$

Since each V_i has the form $V_i = G(i, i+1, \theta_i)$ where $i \leq 2n-1$, it follows that $V_{k-1} = Q_{k-1}$. By the Implicit Q theorem we can assert that d and Q are essentially the same. Thus we can implicitly effect the transition from T to $d + f T_{11}$ by working directly on the bidiagonal matrix B .

Of course, for these claims to hold it is necessary that the underlying tridiagonal matrices be unreduced. Since the subdiagonal entries of $B^T B$ are of the form $d_i f_i$, it is clear that we must search the bidiagonal band for zeros. If $f_k = 0$ for some k , then

$$B = \begin{bmatrix} B_1 & 0 & & k \\ 0 & B_2 & & \\ & & \ddots & k \\ & & & \ddots & k \end{bmatrix}$$

and the original SVD problem decouples into two smaller problems involving the matrices B_1 and B_2 . If $d_k \neq 0$ for some $k \leq n$, then premultiplication by a sequence of Givens transformations can zero f_k . For example, if $n=6$ and $k=3$ then by rotating in row planes (34), (35), and (36) we can zero the entire third row

$$B = \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ 0 & x & x & 0 & 0 & 0 \\ 0 & 0 & 0 & x & 0 & 0 \\ 0 & 0 & 0 & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix} \quad \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ 0 & x & x & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & + & 0 \\ 0 & 0 & 0 & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix}$$

$$\begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ 0 & x & x & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & + \\ 0 & 0 & 0 & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix} \xrightarrow{(3,6)} \begin{bmatrix} x & x & 0 & 0 & 0 & 0 \\ 0 & x & x & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & x & 0 \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{bmatrix}.$$

If $d_h = 0$ then the last column can be zeroed with a series of column rotations in planes $(n-1, n), (n-2, n), \dots, (1, n)$. Thus, we can decouple if $f_1 \cdots f_{h-1} = 0$ or $d \cdots d_h = 0$. Putting it all together we obtain the following SVD analogue of Algorithm 832.

Algorithm 8.6.1 Given a bidiagonal matrix $B \in \mathbb{R}^{m \times n}$ having no zeros on its diagonal or superdiagonal, the following algorithm overwrites B with the bidiagonal matrix $d = (TBV)$ where $($ and V are orthogonal and V is essentially the orthogonal matrix that would be obtained by applying Algorithm 832 to $T = BTB$.

Let μ be the eigenvalue of the trailing 2by2 submatrix of $T = BTB$ that is closer to t_m

$$y = tu\mu$$

$$z = ti2$$

for $k = 1:n-1$

Determine $c = \cos(\theta)$ and $s = \sin(\theta)$ such that

$$\begin{bmatrix} y & z \end{bmatrix} \begin{bmatrix} c & s \\ -s & c \end{bmatrix} = \begin{bmatrix} a & 0 \end{bmatrix}.$$

$$B = B \cdot G(k, k+1, \theta)$$

$$y = h_{k,k}$$

$$z = h_{k+1,k}$$

Determine $c = \cos(\theta)$ and $s = \sin(\theta)$ such that

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} * \\ 0 \end{bmatrix}.$$

$$B = G(k, k+1, \theta) \cdot B$$

if $k < n-1$

$$y = h_{k+1,k}$$

$$z = h_{k+2,k}$$

end

end

An efficient implementation of this algorithm would store B 's diagonal and superdiagonal in vectors $d(1:n)$ and $/ (1:n-1)$, respectively, and would require $3n$ fops and $2n$ square roots. Accumulating U requires $6nm$ fops. Accumulating V requires $6n^2$ fops.

Typically, after a few of the above SVD iterations, the superdiagonal entry f_{n-1} becomes negligible. Criteria for smallness within B 's band are usually of the form

$$|f_i| : \text{td} \cdot (|d| + |d+1|),$$

$$|d| : \text{td} \cdot \|B\|$$

where td is a small multiple of the unit roundoff and $\|\cdot\|$ is some computationally convenient norm. Combining Algorithm 542 (bidiagonalization), Algorithm 861, and the decoupling calculations mentioned earlier gives the following procedure.

PROBLEM 8.6.2 T619pc(kp ha)81. 19rp Given $A \in \mathbb{R}^{n \times n}$ and t , a small multiple of the unit roundoff, the following algorithm overwrites A with the AV = D+E, where $U \in \mathbb{R}^{n \times n}$ is orthogonal, $V \in \mathbb{R}^{n \times n}$ is orthogonal, $D \in \mathbb{R}^{n \times n}$ is diagonal, and E satisfies $\|E\|_F \leq t\|A\|_F$.

Use Algorithm 542 to compute the bidiagonalization

$$\begin{bmatrix} B \\ 0 \end{bmatrix} \leftarrow (U_1 \cdots U_n)^T A (V_1 \cdots V_{n-2}).$$

until $q = n$

For $i = 1:n - 1$, set $b_{i+1,i}$ to zero if $|b_{i+1,i}| \geq t(|b_{i,i}| + |b_{i+1,i+1}|)$.

Find the largest q and the smallest p such that if

$$B = \begin{bmatrix} B_{11} & 0 & 0 \\ 0 & B_{22} & 0 \\ 0 & 0 & B_{33} \end{bmatrix} \quad \begin{matrix} p \\ n-p \\ q \end{matrix}$$

$p \quad n-p \quad q$

then B_{33} is diagonal and B_{22} has a nonzero superdiagonal.

if $q \neq n$

if any diagonal entry in B_{22} is zero, then zero the superdiagonal entry in the same row

else

Apply Algorithm 8.6.1 to B_{22}

$$B = \text{diag}(p, U, I_q, m_n)^T B \text{diag}(q, V, I_q)$$

end

end

end

The amount of work required by this algorithm depends on how much of the SVD is required. For example, when solving the LS problem, U need never be explicitly formed but merely applied to b . It is developed. In other applications, only the matrix $U_1 = U(:, 1:n)$ is required. Another variable that affects the volume of work in Algorithm 8.6.2 concerns the R-bidiagonalization idea that we discussed in §549. Recall that unless A is "almost square," it pays to reduce A to triangular form via QR and before bidiagonalizing. If R-bidiagonalization is used in the SVD context, then we refer to the overall process as the R-SVD. Figure 8.6.1 summarizes the work associated with the various possibilities. By comparing the entries in this table (which are meant only as approximate estimates of work), we conclude that the R-SVD approach is more efficient unless $n \ll n$.

8.6.4 f. $\tilde{\lambda}_k \tilde{v}_k$ è $\tilde{\lambda}_k \tilde{v}_k$

It is straightforward to adapt the Jacobi procedures of §§5 to the SVD problem. Instead of solving a sequence of 2by-2 symmetric eigenproblems, we solve a sequence

Required	Golub-Reinsch SVD	R-SVD
E	$4mn^2 - 4n^3/3$	$2mn^2 + 2n^3$
E, V	$4mn^2 + 8n^3$	$2mn^2 + 11n^3$
E, U	$4m^2n - 8mn^2$	$4m^2n + 13n^3$
E, U ₁	$14mn^2 - 2n^3$	$6mn^2 + 11n^3$
E, U ₁ V	$4m^2n + 8mn^2 + 9n^3$	$4m^2n + 22n^3$
E, U ₁ V	$14mn^2 + 8n^3$	$6mn^2 + 20n^3$

Figure 8.6.1. Work associated with various SVD-related calculations

of 2 by 2 SVD problems. Thus, for a given index pair (p, q) we compute a pair of rotations such that

$$\left[\begin{array}{cc} & \\ \bullet & \bullet \\ \hline \bullet & \bullet \end{array} \right]^T \left[\begin{array}{cc} a_p & a_{pq} \\ a_q & a_{qq} \end{array} \right] \left[\begin{array}{cc} c_2 & s_2 \\ -s_2 & c_2 \end{array} \right] = \left[\begin{array}{cc} d_p & 0 \\ 0 & d_q \end{array} \right].$$

See P8.6.5. The resulting algorithm is referred to as **two-sided** because each update involves a pre- and a post-multiplication.

A one-sided Jacobi algorithm involves a sequence of pairwise column orthogonalizations. For a given index pair (p, q) , a Jacobi rotation (p, q, θ) is determined so that columns p and q of A ((p, q, θ)) are orthogonal to each other. See P8.6.8. Note that this corresponds to zeroing the (p, q) and (q, p) entries in ATA . Once AV has sufficiently orthogonal columns, the rest of the SVD (U and E) follows from column scaling $AV = UE$.

Problems

P8.6.1 N.. N.2eot. N. 96 .d .. .9b.

$$.9b.$$

$$S = \begin{bmatrix} \cdot & \cdot \\ \cdot & \cdot \end{bmatrix} \quad \begin{matrix} A \\ J \\ O \end{matrix}$$

(W, B, M, 100, A_B^{m×r})

P8.6.2 7 ot.6 ..deot . 8 ..n : . sub A (A + jC): CER^{m × n}

$$A = \begin{bmatrix} & - \\ & \end{bmatrix}$$

$$P8.6.3 \cdot e_8 = p_{n \times n} \cdot \begin{pmatrix} \in^n \\ |+^n \\ |+^n \\ +^n \end{pmatrix} \cdot d(1:n) \cdot (k - 1) \cdot \begin{pmatrix} (kX^n) \end{pmatrix} \quad (4, \quad)$$

P8.6.4 1.e2 96. $\begin{pmatrix} z_m \\ \vdots \\ z_1 \end{pmatrix} = \begin{pmatrix} \lambda & & & \\ & \ddots & & \\ & & \ddots & \\ & & & \lambda \end{pmatrix} \begin{pmatrix} x_m \\ \vdots \\ x_1 \end{pmatrix}$

[O-B]_K

$$w = K - R^{m \times m} \cdot \lambda \cdot n \cdot [R^T \lambda]_+ + [I_m - R^T \lambda]_+ \cdot A \in \mathbb{R}^{n \times n},$$

P8.6.5 R.558.

$$C = \begin{bmatrix} w & x \\ y & z \end{bmatrix}$$

.8 .8..N .fe.8 ..d8.... N. .8..e.fe6d_co83t.nppc²⁺ s²= rt, ramop

$$B = \begin{bmatrix} & c & s \\ & -s & c \end{bmatrix}$$

ntttCCAInA-SAlCSnaA.npn8zr lMnpMolx.xl IS2onp tpoSgA lMnpMCAlCS.pn
 2mA dEC. 6aXREn6XaR), A 4-AjU(jA2A=(1 UR(T3XTARX)EThCIE a(b4nID n(A)-i
 (, A E R^{n × n}. " ,) p₀.1^σ(p, q) kn2p < q 9-lSn pMoat, MC opnl8tq, 1h) oa3J(p, q, 92) ar
 3A2AMCt, SA8moc2

$$B = J(p, q, 91)^T AJ(p, q, 92),$$

$$\text{pmA}_{\text{aq}} = \text{bcp} = \mathbf{J} \times \mathbf{smI}$$

$$\text{of } (B)^2 = \text{ of } (A)^2 - a \cdot g \cdot a \cdot p$$

$$= 3\text{Al} + 18\text{tr } \overset{\text{Mg}}{\text{Mg}}\text{At} + \text{AAsI}_3 + \text{oAt} + \text{Zn} + \text{St} + \overset{\text{O}}{\text{O}} + \text{Al} + \text{Sn} = \overset{\text{SE}}{\text{Mg}}\text{IAA} + \overset{\text{Mg}}{\text{Mg}}_n + \overset{\text{O}}{\text{O}}\text{BA} + \text{E} \in \mathbb{R}^{n \times n}.$$

for p = 1:n 9 N

for q = p + 1:n

$$A = J(p, q, 91)^T A J(p, q, 62)$$

end

end

yoo.xθ e)z-• 10E= n0 = = =)= a= 8 x0z-• 0f0 aq)= OJ oFAMn= 108 ,SEpA9
spl. 2m02A nt,SSAMz1AMn08r qmMAMAr n88r 2mAAASymAn ftzLAM, SSAM
2Mr car ,zolvMAASyMAICS,ApA95Ar SApr (a(•)A- d(•) O)R)kNO= 00 r lP)o
r)00 k+O + ((O (N0) • mEOOn)RnE nElna

P8.6.6 589 .6n y SAn8R^m)¹n n₁ - 1 2)!, (1)"p"¹)y)E, (Q St

$$Q = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}.$$

cniA o tpsS,AMnpMAlCShpnao83s tAm2poppMAl,C8t1. IyJQ dMMpmlr lao
A-p lpmAM9

Notes and References for §8.6

8. d8.8... .8...8..fe.8..n 8.8...86 8b.8 . 868.8.8..86n

0fN86.. (o(((A)x) (NOnE l=(• El • r(NOf)(- mE)m)O)) + (• = f (f s)SIAM Rev
 35 ..Nr.77x
 3259A,n8A for xpz18((r A)) + (m k f • E • r (NOf E) - m f a (O)) + (• = f k l • E SIAM Rev
 of Linear Algebra , nxlr SA&AyA2oSc0883.oz, ynl838y7., N9
 3 SAMp,MS 22M12mAE nt3AiA,lSAa8spA,dM83p,8 = I hl. xsAAoxtl8
 h8xaAen8(oD(A)(El)Er nklp)))= 1) o) O k l •) f k = (NOfE(RnE)nR)O)) + (• = lk l • E -
 BIT 12 .. NN N X
 cxaxspA.M12(D(A)xEE• E (O(E)Er n(f(p))))= A E)r = (n(O = • O m)(N)OE(m) xl(1 -
 m nR)O(E• r RSIAM Review 15, TITr T7,x
 3xl,pA (1(D(A)x) KNO)R = OOf(OmrOmr))f E)- En)m)OER+ • = k • E+ nAnkELO -
 Lin. Alg. Applic. 11, dT .x
 cxa9s2A.dMq(D(A)) x^ (O•) kN(OE)Er nkl•))f)l - nER = s Nin. Alg. Applic. 28
 + Optpn
 n nL02 03=(e(a(A)))O)) xEPCH(OE)Er nklx(mE=f • AE)+ nER = En)O=s Nin. r
 Appl. 56 I 5Nr57x
 sxAmoArl 'oM830xAxfStAa1(((A)qnQm) xEE• EAExf (QmrR)O)) (f)l - nE)l -
)O)) + (• = f k l • E SIAM Rev. Math. 68, I.r.r II 5x
 lx9BoAAd(((A)))O)) P(xE)E(E(Fr nklx)(mE=f • AE(N))s N SIAM J. Matr r
 Appl. 15 77NfN

- . ipa 9y EalPensioHa OraAu() .20a+VRplich(at qe oHa Cvich(65Numer. Math
 $\tau_0 + \frac{1}{2} \text{C} + p$.
- n Cf oM22ODyn = £VOO = .• OC = OXOO = - O: anof100a z 10 5. r iJ54
 Gr xn2ae cxa4s2A.iMpJJ.43 2A. lAzipniiAMplMSipnAMMsar IzisVStSiAAit1
 Lin. Alg. Applic. 919, iN N4
- c.-K. S a j 0Aa = M2pp DW= On := a0a p a a 0 a = D6EE(a = - Aaaa = aE00
 J. Matrix Anal. Applic. 24, i7r 5N4
 x4xE, ScAiae f 4M, = JJI, hAMplMSi2nA, MlMs., ,piaA, ItwAt, , s2ar IzisM
 tSiAAitib 42 dir(J.4
- K.A. i 1n 002PpBe= 0=XeEa= az eO(Cl : a a a 0ECC x fo = - 0118 M. J. Matrix Anal.
 Applic. 27i,r iT54
- M. 2=30 M2ppDXO= = p= Gta • 000G= • X0 0o0EEX a 0A pAa A= ND0 Alg
 Applic. 419, 5r TT4
- H. n0a M2ppDyn C=C0 AO= 00x0f= 0o08 eaz = 0Grafa(dE0a a E00=Aa :=
 = a 0pNn Alg Applic. 414, iTJr iT54
- , rn2ae ax pa = JJT4 A,CScAAbAMp,MSip2aet809nr AattppAate par I,iBiziA
 EAA,S,tcp2atisiAM J. Matr x Anal. Applic. 29, 7i57..
- f 4pAe iae BxiMc= JJd.4lAzi2ci9nr AaiiziAe s2arizMBiz,AhAM2,MSipn,aizAe
 E2r ,ai,te,CcaiplipM2AAut 483 ut,IyM.
- ,1Ar AIAapMetIe lot tJar madJr AII. el,lpStJ Mn,xmj -
- G.H. Capa n= L0T a • M2pp Dea aE: A a00 -0-=aa aa OXO0: Q,W0-Cbaa ==Nb
 SIAM J. Numer. Anal. 2 iJ, -IIi2
- P.A.)p0O= a 0013ACAM02DyA= oE• OXO0= pAa a0ECC x fCoC10GeOx faOd
 ,a zod eNmum. ACM 12, 7i-v7.4
- G.H. nCapa n= ej 000 oM2p D20=pAa a0ECC x fCoO= 80 80p a 0CaP z OC= o-
 Numer. Math. 14, iJ5-iJ-
- ONMA,ipAe ,Mn2eAA,,S.Aa21ae iaizttntu tAA8
- xOMia= NI 4.3a OSMAe 3zr ,M2pMr,SIpnar 2mAr I,iBiziAeAA,,S,tn2au6CM
 ' n. Math. Sof w. 8 INSE.
- C.C.,,)oA2iE,dkJahn,xleqelxa.aguAgAr AM2p gmx.tigp13TSIAM J. Sci. Stat.
 Comput. 4 i(II4
- f 9E,8r iM2p54.0CSM,in2AAIM-t A,S,pAe scar ,ziBiziA/ u6AM J. Sci. Stat.
 Comput. 4, T(II)N,4
- pBia I 1Az BiaeAizzAiae 3x.iAr ACiat NTx .3a 91AcAa2e l)zniSzA,r IMnpMC
 r,CSIpcapmscar ,iMStSiAA i nipMr8t,AnipAepm s2At2s2r Iz,BI,AtuiJ.
 Comp. Appl. Math. 19, 5Nr 55J4
- P. Oo= n, Ox xOaSn - Sea 80XO(M2pp D. • 000oa - 0C= - a= Aa a0E= XfCoo= (=
 a = 30 x o 02 AOE a = 80A N Numer. Anal. 28 Mulf Mr iu6
- 1kretJS1ennAbu. hAa8ep7MEd c4 9igrp5 4l,gMAJ1 tehahMcep Ielxa.aguAgAr ,
 tMgLin. Alg. Applic. ,tZu(r (J)4
- kAmM.iaa iaeli 2m= N5.4.2. AMcM2pp,etM2AA,CSI2ipnla 3ai, tp2AsnarziM
 Bi.,AEAA,,S,tnp2atTNA 1 T1rd4
- n we MacMcCAiae hxBiaE,MA8 (i .4.3a OCSz2Appw2enir ,ai,n32zr ,M2p
 0,LhtAhM,S,AGibIT 94. (Jyv5i
 xBx uAl8iaexvhM,ApN.4.3AA,Mi2ar „iBizi,Atiae E21AMAa2s2zr ,M2p
 Numer. Math. 67, NNI 5J4
- pxmiaemACMiae xAOstAa N..4.3aizt2.. i VI 3zr ,McpMA,,S,pnar pna „iM
 ilAtuiSIAM J. Matr x Anal. Applic. 16, IJ-.5.4
- fa Iip2= NT.4.lmA,Mpm,r ,ai 3rMn2iSiAM J. Sci. Comput. 18 ((75NN79
 xBDAmaiae,Nd 4.3AA,MipA2aparsiar I,iBiziAt vneA,ai,IipMnAAt 9cr Aaii,IAt
 , pqA,Lpt,CApM2air ,aiZipMnAAt SiAM J. Matr x Anal. Applic. 20 5T5-5.4
 f 4ir miG IJU4.VI i A2IM232nAlS,ApAhal2nraiae 3AAIMipA,S,pipn,81. pmA
 pBE iuLin. Alg. Applic. 909, N5Nr4
- E22A ea A,asIA,Mpm,eM pnenir ,aESM,SzAgi/SAaAiAz,SApmijMai,,r ,It
 r pApMnenir AaiizpMipAllp2Aena7d4i4iC
- J.W. ,Ox x OA = L0T a • M2pp DyEEp -0o= a0pa a6Do)0OCaAa a = 0EOo J.
 Sci. Stat. Comput. 11, dT5 NI4

- 12.2 $\text{t}(\text{c naT T212 Gne4e4 typZ2B b0 PnmnAA0Ahr)=A kn 1rd=4h =AeQdAnm5A x Tpxf(=)ra i2m/H)jS SIAM J. Matrix Anal. Applic. 15, J'.d.$
- I. I. o83s.A.9nrA8t2o2 JM3 Ncic3A, o835Al8sP4lMn22012mAn3nor18sB EkiSIAM J. Matrix Anal. Applic. 16 T.GI.
- hl. andH.tkw.,o8rk083A. BC.AqIJJx AlCS,2n822Awc3nl8q, sBNr tnaI .sa IT,o2niA,t $\text{IIM2MIa}2\text{18k}$ SIAM J. Matrix Anal. Applic. 28, JTrIC.
- l, l83o, o3. 2oqoCIMbJ0x.3 2A. 3qrMc22Msc8rI, oBq,ANAlSltc2cla i83 p. h0mA,c3nl8kPar lld Comput. 35, „mr.” .
- ulM2MI A21 ~~SAIS MISqA21~~ MAM22AMAAt23A4, n3or18sB rAAe
- s. BoI II o83o. h0Mtq., J. . oModA5pa3wc3norlaqc3o2cl8lM3AMASorladq 2McAAtklll Comput. 20, $\mu\Gamma Nld$. EACCAa3h, lAi iIJJ,I. 3AAIMstAnl. aAqqNnorl8odqN Cn8i825.o 2MdA1tum Math. 98, .. r NJ,x
2. I.2M18oM8k. Bo8woMk83l. Bc83ASMldJd 3.2. 3,rlMn2nM22AAIMtrAAAL ,o2nlA, NlCnao88rs, oMStSoAAtk6comp. Appl. Math. 218, I,dr J7x
- 9oAlSIA2ml3,M 22AsBE. oqn82P1 Ao2Ar1MdfA215tn3A3, lSc orlMn22MASA o2A3 SAM,M. 22H A,S3o2Av M13 Ao8Hc,Aa4An2HM22A2MeaAMAitc8npl8q
- 9u. ,lrSA2,Ro8230.JMsI, l2nlal. ,c8Tod9l o2nl8st E Rorl8q, c3o2hl81H1AnA8pMinh Quart. Appl. Math. 13, $\mu\Gamma Gp,IM$
- ..9. ulMtt2o23 hA8MnA0UJMm AtA, r9oAlSIA2ml3MA1.SI2c8 22lMnai, ABSI l, oAl.Sqb - o2Mcbki ris. AMS 94, $\mu\Gamma IM$
- A.A.honrAa3 BcE1IMA8nd7M,8 2mAl o3Mo2A1iAMrA8AIrSA2,n23 3rlMn2pP ,MAICS,2o822An8qN8cqNAAICs2dnl8kni Alg. Applic. 77, J.rup.,
- 9,h AmoM,n2M BoaElIMA8nD. ,a,lrSA2,n2b sBN3,rlMn22822A1MATAaAA A,1t2AMtin. Alg. Applic. 95, $\mu,r N7J$.
- vvonju6dx.212Ala 2mN,o3Mota18AMrA8AIrSA2,no823qrMn22MAl.SI2n8pp2 snal, oBq,AEAAISltc2nlal. Lin. Alg. Applic. 104, $\mu,\mu r N,JM$
- 9.hA2oM,cAM18SFc8k83h. Bo8NlIMA8dd. ,8 91'AcA82S qACA82o2dBrSA2,noap3 3rlMn22MAICSI2c8mAsrI, oBq,ANAAICs2dnl8kni Numer. Math. 52, IT,r, JJ.
- 5.BvuAM8caIld. Jmn8AoM81AMrA82mA, AtA,nA r o888,lrSA2qnoa282plEMT Numer. Math. 56, T,r,l.
- vEM.oocA3. BAA,cijJJjM2H. u2 4833AAIM82AISsBN 3qrM08200 AM J. Matr Anal. Applic. 29, $\mu,IIr \mu,IM$
- l2A laA tc3A1SsBNSMIAA3MAr MSAM2215 3o2H + -AV SMI3 Ac8AsIAaA2 l. c2AMo221Alq.8t 2mo2M8AMh.c8Bf22lrl8qAe
- 9.A12.m iNDJC.3 ,aA5sn3A6o,MC42nlBA22l3M2mA8r,,oBq,IA EAAlSlt22nla 3rASM6A8MSlSgAfcomput. J. 18, Tr, T7M
- hA. oo8tHaN.ldM.lA3 Ac8r2r 21CSAM sAAStn8.t2AaAt- A22lji c8Singular Value Decomposition and Signal Processing 9.u. NASMA22AM2H22l, o83k3,t2AM3o , BAt4c o83B. o4M60M.3 2lta l8 o,8A5sn3A9qplS&rlM2mChumer. Math. 56, p+ ecp U
- e)= r - $\text{AXfA0xJ} = \lambda z)0 = 8eo(ao o = 23)z)E = n0 , J: = o0x) = c)Efp = a Eer$
- kUOxX OAAI ipL On -OA06s(E)lB(0,1)8 :: le x)I>222UD= xfpz0=y 0(8 ypAA: DE= x f= o(z0= (8,j)A0z)0EE: = Lel,Alg. Applic. 299, $\mu\Gamma NM$
- NM,oiN. J. .3 hlr2AMAlM2o2nlal. 22An8l qoMA21M8o MAA183n218A82 3rlMn2k6MA J. Numer. Anal. 19, $\mu\mu Gp$
- z. ,= x). >22o8 DWxf10X08-ZlZ-On(8,z)z (e-8yDE: = 00pA)A: e= Xfpz= 60 yA=)zX08(8-z)X0zDE JAM. J. Sci. Comput. 18, NJJ, INIM
- u.I. EI8nAl 9o8M1iJJ,x.3 2l2Tla I,2nS,nAo2nAqoM9MMIM8AAIM 3rlMc22k SIAM J. Matrix Anal. Applic. 25, $\mu\mu JI\mu r, \mu\mu$
- l2ASoMi,ACS,ACA82d228I oAlSBNmito,l8r o83n82AMAt2o82lMtT
- u.l. ,Iq iNdJMAICSI2n8mHari qoMo,IAEAAISltn2nl8 22A0, 3A oBkMi ns. Math. Sof w. 6 J,r ,0.

ú h) 1 - 1A f .E h71f T) .3,1 ,C..1E,μ ,I- μ H1A...,,.. 7E1- μ ,0,μ).
 , 15H3, „oE „ „ „ 9 SIAM J. Sci. Stat. Comput. 6 uNyGn
 r x0pa27s.k xor eZ-) = Ie2 xle2 7Nrydh) l1AoAeAS12, alh2,xeepIeexa9a.l1AlrStMl2
 \$2rare)l22ata- (pl.arrlpr7S VLSI Computer Systems 1, 242-270
 F.T. & cMOZpp ZM(a 8 -X-p=1E0+-+a=e Xf p z08 - p in 0+ Lin. Alg. Applic.
 7 + 0.+ o1
 M.) 0-e-a 8yLoa XMZpp z H: fz(- E0ea EYh(= (e=qXl080NOX0= = (E 08 ,
 lp 0 8B - p in 10E= X.=(+Z> . Pr(8eedings International Conference on Parallel
 Processing 433-440
 R. OE= O(nM)pp z 0= A > (2- J8a> 80p0 :=Aa K0 = n 10x8- 8 8 : 0-0e+ (=0: AOP
 y= e-SIAM J. Sci. Stat. Comput. 7, „NG.,
 C.H.) (+ EJa R-L a &= M8ppz e= Xf: zJ8= l= 8j (8 = j= X=e E0+ (8lear=e+ > .
 Scale Eigenvalue Problems, CA, ..., i83lMacq,r 2St= AdU2Mpl,i ao 3.pAM3.UNG
 77X
 C.H.) (+ EJM10 puz. JQ3= - 0(LA: Eoa E= n(A83n0= EpynOEq(= OE-(81,0+ 2a a
 cube Multiprocessors, I: ..Ip. = A.. s83nhISapcl8,hcqinA,S2n13,
 C.H.) O+ EJM10 peze= Xf: z(808 - p in 10E= Xf= + 784(8 = = (0e+ 0:OKOE= ==
 == E0+>Parallel Comput. 11, ,TG,dM
 M.) O ala2c ,a 120= az 0= ,E28&12(- =M20 2pz i8r= 0= aj =X> 01X= = E8N4(8 -
 a = a) A91Ec 100=Ry 1 (= JXPparallel Comput. 36, 297-307.

8.7 Generalized Eigenvalue Problems with Symmetry

This section is mostly about a pair of symmetrically structured versions of the generalized eigenvalue problem that we considered in §7.7. In the symmetric definite problem we seek nontrivial solutions to the problem

$$Ax = \lambda Bx \quad (87.1)$$

where $A \in \mathbb{R}^{n \times n}$ is symmetric and $B \in \mathbb{R}^{n \times n}$ is symmetric positive definite. The generalized singular value problem has the form

$$A^T Ax = \lambda B^T Bx \quad (87.2)$$

where $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{m \times m}$. By setting $B = I_n$ we see that these problems are (respectively) generalizations of the symmetric eigenvalue problem and the singular value problem.

klwhhb 7Sb uKqqS f .Tf pS0et Tf S0S SetP TfSib bTSEe k dSb

The as02 rr

For a symmetric definite pair $\{A, B\}$, it is possible to choose a real nonsingular X so that X^TAX and X^TBX are diagonal. This follows from the next result.

Theorem 8.7.1. Suppose A and B are n -by- n symmetric matrices, and define $C(\mu)$ by

$$C(\mu) := \mu A + (1 - \mu)B - \mu E J. \quad (8.7.3)$$

If there exists a $\mu \in [0, 1]$ such that $C(\mu)$ is nonnegative definite and

$$\text{null}(C(\mu)) = \text{null}(A) \cap \text{null}(B)$$

then there exists a nonsingular X such that both X^TAX and X^TBX are diagonal.

Proof. Let $\mu \in [0, 1]$ be chosen so that $C(\mu)$ is nonnegative definite with the property that $\text{null}(C(\mu)) = \text{null}(A) \cap \text{null}(B)$. Let

$$Q^T C(\mu) Q = \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix}, \quad D = \text{diag}(d_1, \dots, d_k), \quad d_i > 0,$$

be the Schur decomposition of $C(\mu)$ and define $X_i = Q_i \cdot \text{diag}(n^{-1/2}, I_{n-k})$. If

$$A_1 = X_1^T A X_1, \quad B_1 = X_1^T B X_1, \quad C_1 = X_1^T C(\mu) X_1,$$

then

$$C_1 = \begin{bmatrix} I_k & 0 \\ 0 & 0 \end{bmatrix} = \mu A_1 + (1 - \mu)B_1.$$

Since

$$\text{span}\{e_k, \dots, e_n\} \subseteq \text{null}(C_1) \subseteq \text{null}(A_1) \cap \text{null}(B_1)$$

it follows that A_1 and B_1 have the following block structure:

$$A_1 = \begin{bmatrix} Au & 0 \\ 0 & 0 \end{bmatrix}_{k \times n-k}, \quad B_1 = \begin{bmatrix} Bu & 0 \\ 0 & 0 \end{bmatrix}_{k \times n-k}.$$

Moreover $I_k = \mu Au + (1 - \mu)Bu$.

Suppose $\mu = 0$. It then follows that if $Z^T B u Z = \text{diag}(b_1, \dots, b_k)$ is the Schur decomposition of B_{11} and we set

$$X = X_1 \cdot \text{diag}(Z, I_{n-k})$$

then

$$X^T B X = \text{diag}(b_1, \dots, b_k, 0, \dots, 0) \equiv D_B$$

and

$$X^T A X = \frac{1}{\mu} X^T (C(\mu) - (1 - \mu)B) X \stackrel{\text{ta}}{=} \frac{1}{\mu} \left(\begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} - (1 - \mu)DB \right) = DA$$

On the other hand, if $\mu = Q$ then let $Z^T A Z = \text{diag}(a_1, \dots, a_k)$ be the Schur decomposition of A , and set $X = X_1 \text{diag}(Z, I_{n-k})$. It is easy to verify that in this case as well, both $X^T A X$ and $X^T B X$ are diagonal.

Frequently, the conditions in Theorem 8.7.1 are satisfied because either A or B is positive definite.

Corollary 8.7.2. If $A - AB^{-1}B$ is symmetric definite, then there exists a nonsingular

$$X = [x_1 | \cdots | x_n]$$

such that

$$X^T A X = \text{diag}(a_1, \dots, a_n)$$

and

$$X^T B X = \text{diag}(b_1, \dots, b_n).$$

Moreover, $A x_i = B x_i$ for $i = 1, \dots, n$ where $A = a_i b_i$.

Proof. By setting $\mu = 0$ in Theorem 8.7.1 we see that symmetric definite pencils can be simultaneously diagonalized. The rest of the corollary is easily verified. \square

Stewart (1979) has worked out a perturbation theory for symmetric pencils $A - AB$ that satisfy

$$c(A, B) = \min_{\|x\|_2=1} (x^T A x)^2 + (x^T B x)^2 > 0. \quad (87.4)$$

The scalar $c(A, B)$ is called the **Cr wfor number** of the pencil $A - AB$.

Theorem 8.7.3. Suppose $A - AB$ is an n -by- n symmetric definite pencil with eigenvalues

$$\lambda_1, \lambda_2, \dots, \lambda_n$$

Suppose E_A and E_B are symmetric n -by- n matrices that satisfy

$$E^2 = \|E_A\|_2^2 + \|E_B\|_2^2 < c(A, B).$$

Then $(A + E_A) - >(B + E_B)$ is symmetric definite with eigenvalues

$$\mu_1 \geq \cdots \geq \mu_n$$

that satisfy

$$|\arctan(\mu_i) - \arctan(\mu_j)| \leq \arctan(E/c(A, B))$$

for $i = 1, \dots, n$.

Proof. See Stewart (1979). \square

qA AA ImN F}1N+A .L Ni-GA jAA A 1%7)

Turning to algorithmic matters, we first present a method for solving the symmetric-definite problem that utilizes both the Cholesky factorization and the symmetric QR algorithm.

Algorithm 8.7.1 Given $A = ATEI^{n \times n}$ and $B = BTEI^{n \times n}$ with B positive definite, the following algorithm computes a nonsingular X such that $X^TAX = \text{diag}(a_1, \dots, a_n)$ and $X^TBX = I_n$.

Compute the Cholesky factorization $B = GGT$ using Algorithm 4.2.2.

Compute $C = c^{-1}Ac^{-T}$.

Use the symmetric QR algorithm to compute the Schur decomposition

$$QTCQ = \text{diag}(a_1, \dots, a_n).$$

Set $X = c^{-1}TQ$.

This algorithm requires about $14n^3$ fops. In a practical implementation, A can be overwritten by the matrix C . See Martin and Wilkinson (1969) for details. Note that

$$\lambda(A, B) = \lambda(A, GG^T) = \lambda(G^{-1}AG^{-T}, I) = \lambda(C) = \{a_1, \dots, a_n\}.$$

If \hat{a}_i is a computed eigenvalue obtained by Algorithm 8.7.1, then it can be shown that

$$\hat{a}_i \in \lambda(G^{-1}AG^{-T} + E_i)$$

where

$$\|E_i\|_2 \approx \|u\|_2 \|A\|_2 \|B^{-1}\|_2.$$

Thus, if B is ill-conditioned, then \hat{a}_i may be severely contaminated with roundoff error even if a_i is a well-conditioned generalized eigenvalue. The problem, of course, is that in this case, the matrix $C = c^{-1}Ac^{-T}$ can have some very large entries if B , and hence G , is ill-conditioned. This difficulty can sometimes be overcome by replacing the matrix G in Algorithm 8.7.1 with $Vn - \frac{1}{2}V^2$, where $VIBV = D$ is the Schur decomposition of B . If the diagonal entries of D are ordered from smallest to largest, then the large entries in C are concentrated in the upper left-hand corner. The small eigenvalues of C can then be computed without excessive roundoff error contamination (or so the heuristic goes). For further discussion, consult Wilkinson (AE, pp. 337–339).

The condition of the matrix X in Algorithm 8.7.1 can sometimes be improved by replacing B with a suitable convex combination of A and B . The connection between the eigenvalues of the modified pencil and those of the original are detailed in the proof of Theorem 8.7.1.

Other difficulties concerning Algorithm 8.7.1 relate to the fact that $c^{-1}Ac^{-T}$ is generally full even when A and B are sparse. This is a serious problem, since many of the symmetric-definite problems arising in practice are large and sparse. Crawford (1973) has shown how to implement Algorithm 8.7.1 effectively when A and B are banded. Aside from this case, however, the simultaneous diagonalization approach is impractical for the large, sparse symmetric-definite problem. Alternate strategies are discussed in Chapter 10.

8.7.3 $\lambda^{1/2} \in \mathbb{R}$ $\in \frac{1}{2}\mathbb{B} \in \mathbb{R}$

Many of the symmetric eigenvalue methods presented in earlier sections have symmetric definite generalizations. For example, the Rayleigh quotient iteration (8.26) can be extended as follows.

x_0 given with $\|x_0\|_2 = 1$

for $k = 0, 1, \dots$

$$\mu_k \leftarrow x_k^T A x_k / x_k^T B x_k \quad (8.75)$$

$$\text{Solve } (A - \mu_k B) z_{k+1} = B x_k \text{ for } z_{k+1}.$$

$$x_{k+1} \leftarrow z_{k+1} / \|z_{k+1}\|_2$$

end

The main idea behind this iteration is that

$$A \text{ ta } \frac{x_k^T A x_k}{x_k^T B x_k} \quad (8.76)$$

minimizes

$$f(\cdot) = \frac{\|\cdot\|_2^2}{\|\cdot\|_B^2} \quad (8.77)$$

where $\|\cdot\|_B$ is defined by $\|\cdot\|_B = \sqrt{\lambda_B}$. The mathematical properties of (8.75) are similar to those of (8.26). Its applicability depends on whether or not systems of the form $(A - \mu_k B)z = x$ can be readily solved. Likewise, the same comment pertains to the generalized orthogonal iteration:

$Q_0 E R_0 x_0$ given with $Q_0 Q_0^T = I_p$

for $k = 1, 2, \dots$ $\quad (8.78)$

$$\text{Solve } B z_k \leftarrow A Q_k \text{ if } r z_k$$

$$z_k \leftarrow Q R_k \text{ (QR factorization, } Q_0 E R_0 x_0, R_k E_w x_k)$$

end

This is mathematically equivalent to (7.36) with A replaced by $B^{-1}A$. Its practicality strongly depends on how easily it is to solve linear systems of the form $Bz = y$.

8.7.4 $r^{1/2} \in \mathbb{R} \hat{X} \in \mathbb{R}^{n \times n} \in \mathbb{R}^{m \times n} \in \mathbb{R}^{m \times m}$

We now turn our attention to the generalized singular value decomposition introduced in §6.16. This decomposition is concerned with the simultaneous diagonalization of two rectangular matrices A and B that are assumed to have the same number of columns. We restate the decomposition here with a simplification that both A and B have at least as many rows as columns. This assumption is not necessary, but it serves to undistract our presentation of the GSVD algorithm.

Theorem 8.7.4 (Tall Rectangular Version). If $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{m \times n}$ have at least as many rows as columns, then there exists an orthogonal matrix $U_1 \in \mathbb{R}^{m \times m}$, an orthogonal matrix $U_2 \in \mathbb{R}^{n \times n}$, and a nonsingular matrix $X \in \mathbb{R}^{n \times m}$ such that

$$U_1 A X = \text{diag}(\alpha_1, \dots, \alpha_n),$$

$$U_2 B X = \text{diag}(1, \dots, n).$$

Pr of. See Theorem 6.1 . 1

The generalized singular values of the matrix pair $\{A, B\}$ are defined by

$$\sigma(A, B) = \{\alpha_1/\beta_1, \dots, \alpha_n/\beta_n\}.$$

We give names to the columns of X , U_1 and U_2 . The columns of X are the right generalized singular vectors, the columns of U_1 are the left-A generalized singular vectors, and the columns of U_2 are the left-B generalized singular vectors. Note that

$$AX(:, k) = \alpha_k U_1(:, k),$$

$$BX(:, k) = \beta_k U_2(:, k),$$

for $k = 1:n$.

There is a connection between the GSVD of the matrix pair $\{A, B\}$ and the "symmetric-definite-definite" pencil $ATA - \lambda BTB$. Since

$$X^T(ATA - \lambda BTB)X = D_A^T D_A - \lambda D_B^T D_B = \text{diag}(\alpha_k^2 - \lambda \beta_k^2),$$

it follows that the right generalized singular vectors of $\{A, B\}$ are the generalized eigenvectors for $ATA - \lambda BTB$ and the eigenvalues of the pencil $ATA - \lambda BTB$ are squares of the generalized singular values of $\{A, B\}$.

All these GSVD factors revert to familiar SVD factors by setting $B = I_n$. For example, if $B = I_n$, then we can set $X = U_2$ and $U_1 AX = DA$ is the SVD.

We mention that the generalized singular values of $\{A, B\}$ are the stationary values of

$$\langle ABx, x \rangle = \frac{\|Ax\|^2}{\|Bx\|^2}$$

and the right generalized singular vectors are the associated stationary vectors. The left-A and left-B generalized singular vectors are stationary vectors associated with the quotient $\|y\|_2^2/\|x\|_2^2$ subject to the constraints

$$A^T x = B^T y, \quad x \perp \text{null}(A^T), \quad y \perp \text{null}(B^T).$$

See Chu, Fiedler, and Golub (1997).

A GSVD perturbation theory has been developed by Sun (1983, 1988, 2000), Paige (1984), and Li (1990).

~~qgAA A J MN-GfA -- oAHlD9A A+GfA -- oAIA 9il M+G-GIA~~

Our proof of the GSVD in Theorem 6.1.1 is constructive and makes use of the SVD decomposition. In practice, computing the GSVD via the CS decomposition is a viable strategy.

~~mash5li- 8.7.2 (GSVD (Tall, Fiedler-Rank Version)) Assume that $A \in \mathbb{R}^{m_1 \times n}$, $B \in \mathbb{R}^{m_2 \times n}$, with $m_1 \leq n$, $m_2 \leq n$, and $\text{null}(A) = \text{null}(B) = n$. The following algorithm computes an orthogonal matrix $U_1 \in \mathbb{R}^{m_1 \times m_1}$, an orthogonal matrix $U_2 \in \mathbb{R}^{m_2 \times m_2}$, a nonsingular matrix $X \in \mathbb{R}^{n \times n}$, and diagonal matrices $D_A \in \mathbb{R}^{m_1 \times m_1}$ and $D_B \in \mathbb{R}^{m_2 \times m_2}$ such that $U_1^T A X = D_A$ and $U_2^T B X = D_B$.~~

Compute the QR factorization

$$\begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} R.$$

Compute the CS decomposition

$$\begin{aligned} U_1 Q_1 V &= DA = \text{diag}(a_1, \dots, a_n), \\ U_2 Q_2 V &= DB = \text{diag}(s_1, \dots, s_n). \end{aligned}$$

Solve $RX = V f_r X$.

The assumption that $\text{null}(A) \cap \text{null}(B) = \{0\}$ is not essential. See Van Loan (1985). Regardless, the condition of the matrix X is an issue that affects accuracy. However, we point out that it is possible to compute designated right generalized singular vector subspaces without having to compute explicitly selected columns of the matrix $X = R^{-1}$. For example, suppose that we wish to compute an orthonormal basis for the subspace $S = \text{span}\{x_1, \dots, x_k\}$ where $X = X(:, i)$. If we compute an orthogonal Z and upper triangular T so $TZ^T = V^T R$, then

$$ZT^{-1} = R^{-1}V = X$$

and $S = \text{span}\{z_1, \dots, z_k\}$ where $z_i = Z(:, i)$. See P522 concerning the computation of Z and T .

8.7.6 P3 $\hat{Q}_1 \hat{Q}_2 \hat{U}_1 \hat{U}_2 \hat{F}_1 \hat{F}_2 \hat{R} = \hat{S} \hat{Q}_1 \hat{Q}_2 \hat{U}_1 \hat{U}_2 \hat{V}$

At first glance, the computation of the CS decomposition looks easy. After all, it is just a collection of SVDs. However, there are some complicating numerical issues that are best to be addressed. To build an appreciation for this, we step through the "thin" version of the algorithm developed by Van Loan (1985) for the case

$$Q = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} = \left[\begin{array}{cccc|cc} x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ \hline x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \\ x & x & x & x & x & x \end{array} \right].$$

In exact arithmetic, the goal is to compute 5 by 5 orthogonal matrices U_1, U_2 and V such that

$$\begin{aligned} U_1 Q_1 V &= C = \text{diag}(c_1, c_2, c_3, c_4, c_5), \\ U_2 Q_2 V &= S = \text{diag}(s_1, s_2, s_3, s_4, s_5). \end{aligned}$$

In floating point, we strive to compute matrices $\hat{U}_1^T Q_1 \hat{V}$ and $\hat{U}_2^T Q_2 \hat{V}$ that are orthogonal to working precision and which transform Q_1 and Q_2 into nearly diagonal from Q .

$$\text{fl}(\hat{U}_1^T Q_1 \hat{V}) = \text{diag}(\hat{c}_k) + E_1, \quad \|E_1\| \approx u, \quad (87.9)$$

$$\text{fl}(\hat{U}_2^T Q_2 \hat{V}) = \text{diag}(\hat{s}_k) + E_2, \quad \|E_2\| \approx u. \quad (87.10)$$

In what follows, it will be obvious that the computed versions of U_1 , U_2 and V are orthogonal to working precision, as they will be "put together" from numerically sound QR factorizations and SVDs. The challenge is to do this (87.9) and (87.10).

We start by computing the SYD

$$U_2^T Q_1 V = S$$

followed by the QR factorization

$$U_1 R = Q_1 V.$$

Overwriting Q_2 with S and Q_1 with R gives

$$Q = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} \\ r_{21} & r_{22} & r_{23} & r_{24} & r_{25} \\ r_{31} & r_{32} & r_{33} & r_{34} & r_{35} \\ r_{41} & r_{42} & r_{43} & r_{44} & r_{45} \\ r_{51} & r_{52} & r_{53} & r_{54} & r_{55} \\ \hline s_1 & \delta_{12} & \delta_{13} & \delta_{14} & \delta_{25} \\ \delta_{21} & s_2 & \delta_{23} & \delta_{24} & \delta_{25} \\ \delta_{31} & \delta_{32} & s_3 & \delta_{34} & \delta_{35} \\ \delta_{41} & \delta_{42} & \delta_{43} & s_4 & \delta_{45} \\ \delta_{51} & \delta_{52} & \delta_{53} & \delta_{54} & s_5 \end{bmatrix} \quad \epsilon_{ij} = O(u),$$

$$\delta_{ij} = O(u),$$

Since the columns of this matrix are orthonormal to machine precision, it follows that

$$|r_{11} r_{1j}| \approx u, \quad j = 2:5.$$

Note that if $\|u\| = O(1)$, then we may conclude that $\|r_{11}\| \approx u$ for $j = 2:5$. This will be the case if (for example) $u \approx 1/\sqrt{2}$. For then

$$|r_{11}| \approx \sqrt{1 - s_1^2} \geq \frac{1}{\sqrt{2}}.$$

With this in mind, let us assume that the singular values s_1, \dots, s_5 are ordered from little to big and that

$$0 \leq s_1 \leq s_2 \leq \frac{1}{\sqrt{2}} < s_3 \leq s_4 \leq s_5. \quad (87.11)$$

Working with the near-orthonormality of the columns of \mathbf{Q} we conclude that

$$\mathbf{Q} = \left[\begin{array}{cc|ccc} c_1 & \epsilon_{12} & \epsilon_{13} & \epsilon_{14} & \epsilon_{15} \\ \epsilon_{21} & c_2 & \epsilon_{23} & \epsilon_{24} & \epsilon_{25} \\ \hline \epsilon_{31} & \epsilon_{32} & r_{33} & r_{34} & r_{35} \\ \epsilon_{41} & \epsilon_{42} & \epsilon_{43} & r_{44} & r_{45} \\ \epsilon_{51} & \epsilon_{52} & \epsilon_{53} & \epsilon_{54} & r_{55} \\ \hline s_1 & \delta_{12} & \delta_{13} & \delta_{14} & \delta_{25} \\ \delta_{21} & s_2 & \delta_{23} & \delta_{24} & \delta_{25} \\ \hline \delta_{31} & \delta_{32} & s_3 & \delta_{34} & \delta_{35} \\ \delta_{41} & \delta_{42} & \delta_{43} & s_4 & \delta_{45} \\ \delta_{51} & \delta_{52} & \delta_{53} & \delta_{54} & s_5 \end{array} \right]$$

$\epsilon_{ij} = O(\mathbf{u}),$
 $\delta_{ij} = O(\mathbf{u}).$

Note that

$$|r_{34}| \approx \frac{\mathbf{u}}{|r_{33}|} \approx \frac{\mathbf{u}}{\sqrt{1 - s_3^2}}.$$

Since s_3 can be close to 1, we cannot guarantee that r_{34} is sufficiently small. Similar comments apply to r_{35} and r_{45} .

To rectify this we compute the SVD of $\mathbf{Q}(35|35)$, taking care to apply the U-matrix across rows 3 to 5 and the V matrix across columns 3 to 5. This gives

$$\mathbf{Q} = \left[\begin{array}{cc|ccc} c_1 & \epsilon_{12} & \epsilon_{13} & \epsilon_{14} & \epsilon_{15} \\ \epsilon_{21} & c_2 & \epsilon_{23} & \epsilon_{24} & \epsilon_{25} \\ \hline \epsilon_{31} & \epsilon_{32} & c_3 & \epsilon_{34} & \epsilon_{35} \\ \epsilon_{41} & \epsilon_{42} & \epsilon_{43} & c_4 & \epsilon_{45} \\ \epsilon_{51} & \epsilon_{52} & \epsilon_{53} & \epsilon_{54} & c_5 \\ \hline s_1 & \delta_{12} & \delta_{13} & \delta_{14} & \delta_{25} \\ \delta_{21} & s_2 & \delta_{23} & \delta_{24} & \delta_{25} \\ \hline \delta_{31} & \delta_{32} & t_{33} & t_{34} & t_{35} \\ \delta_{41} & \delta_{42} & t_{43} & t_{44} & t_{45} \\ \delta_{51} & \delta_{52} & t_{53} & t_{54} & t_{55} \end{array} \right]$$

$\epsilon_{ij} = O(\mathbf{u}),$
 $\delta_{ij} = O(\mathbf{u}).$

Thus, by diagonalizing the $(2,2)$ block of \mathbf{Q}_1 we fill the $(2,2)$ block of \mathbf{Q}_2 . However, if we compute the QR factorization of $\mathbf{Q}(8|10|35)$ and apply the orthogonal factor across

rows 8.10 then we obtain

$$Q = \left[\begin{array}{cc|ccc} c_1 & \epsilon_{12} & \epsilon_{13} & \epsilon_{14} & \epsilon_{15} \\ \epsilon_{21} & c_2 & \epsilon_{23} & \epsilon_{24} & \epsilon_{25} \\ \hline \epsilon_{31} & \epsilon_{32} & c_3 & \epsilon_{34} & \epsilon_{35} \\ \epsilon_{41} & \epsilon_{42} & \epsilon_{43} & c_4 & \epsilon_{45} \\ \epsilon_{51} & \epsilon_{52} & \epsilon_{53} & \epsilon_{54} & c_5 \\ \hline \mathbf{81} & \delta_{12} & \delta_{13} & \delta_{14} & \delta_{25} \\ \delta_{21} & s_2 & \delta_{23} & \delta_{24} & \delta_{25} \\ \hline \delta_{31} & \delta_{32} & t_{33} & t_{34} & t_{35} \\ \delta_{41} & \delta_{42} & \delta_{43} & t_{44} & t_{45} \\ \delta_{51} & \delta_{52} & \delta_{53} & \delta_{54} & t_{55} \end{array} \right]$$

$\epsilon_{ij} = O(\mathbf{u}),$
 $\delta_{ij} = O(\mathbf{u}).$

Using the near-orthonormality of the columns of Q and the fact that c_3, c_4 , and c_5 are all less than $1/\sqrt{2}$, we can conclude (for example) that

$$|t_{34}| \approx O\left(\frac{\mathbf{u}}{|t_{33}|}\right) \approx O\left(\frac{\mathbf{u}}{\sqrt{1 - c_3^2}}\right) = O(\mathbf{u}).$$

Using similar arguments we may conclude that both t_{35} and t_{45} are $O(\mathbf{u})$. It follows that the updated Q_1 and Q_2 are diagonal to within the required tolerance and that (8.7.9) and (8.7.10) are achieved as a result.

in **infob** 7.8. tk:§ f Plif fbeEE k tfeb

Paige (1986) developed a method for computing the GSVD based on the Kogbetliantz-Jacobi SVD procedure. At each step a 2by-2 GSVD problem is solved, a calculation that we briefly examine. Suppose F and G are 2by-2 and that G is nonsingular. If

$$U_1^T (FG^{-1}) U_2 = \Sigma = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix}$$

is the SVD of FG^{-1} , then $\sigma(F, G) \in \{\sigma_1, \sigma_2\}$ and

$$U_1^T F = (U_2^T G) E.$$

This says that the rows of $U_1^T F$ are parallel to the corresponding rows of $U_2^T G$. Thus, if Z is orthogonal so that $U_2^T G Z = G$ is upper triangular, then $U_1^T F Z = F$ is also upper triangular. In the Paige algorithm, these 2by-2 calculations resonate with the preservation of the triangular form that is key to the Kogbetliantz procedure. Moreover, the A and B input matrices are separately updated and the updates only involve orthogonal transformations. Although some of the calculations are very delicate, the overall procedure is tantamount to applying Kogbetliantz implicitly to the matrix AB^{-1} .

8.7.8 $\hat{A} \hat{B} \hat{U} \hat{C}$ è $\hat{A} \hat{B} \hat{U} \hat{C} \hat{R}$

What we have been calling the "generalized singular value decomposition" is sometimes referred to as the quotient singular value decomposition or QSVD. A key feature of the decomposition is that it separately transforms the input matrices A and B in such a way that the generalized singular values and vectors are exposed, sometimes implicitly.

It turns out that there are other ways to generalize the SVD. In the product singular value decomposition problem we are given $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{n \times p}$ and require the SVD of ATB . The challenge is to compute $UT(ATB)V = E$ without actually forming ATB ; that operation can result in a significant loss of information. See DeMoor (1998, 2000).

The restricted singular value decomposition involves three matrices and is best motivated from a variational point of view. If $A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times q}$, and $C \in \mathbb{R}^{q \times p}$, then the restricted singular values of the triplet $\{A, B, C\}$ are the stationary values of

$$\psi_{A,B,C}(x, y) = \frac{y^T Ax}{\|By\|_2 \|Cx\|_2}.$$

See Zha (1991), De Moor and Golub (1991), and Chu, De Lathouwer, and De Moor (2000). As with the product SVD, the challenge is to compute the required quantities without forming inverses and products.

All these ideas can be extended to chains of matrices, e.g., the computation of the SVD of a matrix product $A = A_1 A_2 \cdots A_k$ without explicitly forming A . See De Moor and Zha (1991) and De Moor and Van Dooren (1992).

8.7.9 $\hat{A} \hat{B} \hat{U} \hat{C}$ è $\hat{A} \hat{B} \hat{U} \hat{C} \hat{D}$

We build on our §7.7.9 discussion of the polynomial eigenvalue problem and briefly consider some structured versions of the quadratic case.

$$(\lambda^2 M + \lambda C + K) x = 0, \quad M, C, K \in \mathbb{R}^{m \times m}. \quad (87.12)$$

We recommend the excellent survey by Tisseur and Meerbergen (2001) for more detail.

Note that the eigenvalue in (87.12) solves the quadratic equation

$$(x^H M x) \lambda^2 + (x^H C x) \lambda + (x^H K x) = 0. \quad (87.13)$$

and thus

$$\lambda = \frac{-(x^H C x) \pm \sqrt{(x^H C x)^2 - 4(x^H M x)(x^H K x)}}{2(x^H M x)}, \quad (87.14)$$

assuming that $x^H M x \neq 0$. Linearized versions of (87.12) include

$$\begin{bmatrix} 0 & N \\ K & C \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} = \lambda \begin{bmatrix} N & 0 \\ 0 & -M \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \quad (87.15)$$

and

$$\begin{bmatrix} -K & 0 \\ 0 & N \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} = \lambda \begin{bmatrix} C & M \\ N & 0 \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} \quad (87.16)$$

where $N \in \mathbb{R}^{n \times n}$ is nonsingular.

In many applications, the matrices M and C are symmetric and positive definite and K is symmetric and positive semidefinite. It follows from (87.14) that in this case the eigenvalues have nonpositive real part. If we set $N = K$ in (87.15), then we obtain the following generalized eigenvalue problem

$$\begin{bmatrix} & \\ ; & \end{bmatrix} \begin{bmatrix} : \\ : \end{bmatrix} = A \begin{bmatrix} & - \\ - & \end{bmatrix} \begin{bmatrix} : \\ : \end{bmatrix}.$$

This is not a symmetric-definite problem. However, if the overdamping condition

$$\min_{x^T x=1} (x^T C x)^2 - 4(x^T M x)(x^T K x) = 1^2 > 0$$

holds, then it can be shown that there is a scalar $\mu > 0$ so that

$$A(\mu) = \begin{bmatrix} \mu K & K \\ K & C - \mu M \end{bmatrix}$$

is positive definite. It follows from Theorem 87.1 that (87.16) can be diagonalized by congruence. See Veselic (1999).

A quadratic eigenvalue problem that arises in the analysis of gyroscopic systems has the property that $M = M^T$ (positive definite), $K = K^T$, and $C = -CT$. It is easy to see from (87.14) that the eigenvalues are all purely imaginary. For this problem we have the structured linearization

$$\begin{bmatrix} A & - \\ : & \end{bmatrix} \begin{bmatrix} : \\ : \end{bmatrix} = A \begin{bmatrix} & \\ & \end{bmatrix} \begin{bmatrix} : \\ : \end{bmatrix}.$$

Notice that this is a Hamiltonian/skew-Hamiltonian generalized eigenvalue problem.

In the quadratic palindromic problem, $K = M^T$ and $C = -C^T$ and the eigenvalues come in reciprocal pairs, i.e., if $Q(A)$ is singular then so is $Q(1/A)$. In addition, we have the linearization

$$\begin{bmatrix} M^T & M^T \\ c - M & M^T \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} = A \begin{bmatrix} -M & M^T - C \\ -M & -M \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix}. \quad (87.17)$$

Note that if this equation holds, then

$$(\lambda^2 M + \lambda C + M^T)(y + z) = 0. \quad (87.18)$$

For a systematic treatment of linearizations of structured polynomial eigenvalue problems, see Mackey, Mackey, Mehl, and Mehrmann (2006).

Problems

P8.7.1 ... 8. A E R n k k R V X T w G R o n g R v w A w w w Q) . x R = a E Q = = h u E = X f : z 0 8 c - A c - T

P8.7.2 ... 8. A E R n k k R V X T w B E C k k R V X T W A R R w A R B R q) w x) T V A V R R A R P k 2 AB U C R U == , C (X U C I h a U (E T Y R U) () U C . T = c . n

dAhrsA16

P8.7.3 7mot..88ot..dot8..n .8.8.. .8.688. AT3a3 = ip1 p2Acr 3ai.zAa3
 rcr ap1M1t pmA3M,c Aa3Aai,zSMISzA.

$$\begin{bmatrix} 0 & A^T B \\ B^T A & 0 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} = \sigma \begin{bmatrix} A^T A & 0 \\ 0 & B^T B \end{bmatrix} \begin{bmatrix} \vdots \\ \vdots \end{bmatrix}.$$

P8.7.4 ..86 ... feb .. 8.ot.n nfe.dg..otfe .68t86.8.8b... 2 ...8.A.a3 r u r
 .9tar,z.MsAnpmmp = AB⁻¹.lmctml, pmpt..ApMcSAAet,r MAttAapcztAaAM,v

P8.7.5 ..86 .86 .8 .8.8.. ..8.82 ..8 d8.8..otfe.fegneot.e8 ..8 ot8A.a3
 B (= n= oexXXO= (E (= : O: (Q' a> O = (= Oa

P8.7.6 c.8YER^{n×n}[λ] λ[λ] λ[λ] ∈ ,)] .)] .R^m, λ[λ]) I2..,H₂A=AyH.. H₂
 kAJWMA(λ) ZA H₂R₂A An A 2

P8.7.7 e.e. .8.8

$$\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} = \lambda \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix}$$

.8A ER^{m×n}, ER^{m×m}, R^{m+1}R^{n×n}, R^{n×j}, λR^m, R^{m+1}R^{m+1}, ..., n)) + ,)
 R^j R^{m+1} 1 A3 1 MAtSAApA ApApmA AaAM.zAa3Aai,zSMISzAlpmA
 2r,zM,IAtl. G AG T

P8.7.8 .e.e. 8.8 A.a3 r M3 SApMrSAr pr3AaApAml. ml. plAl.S,pA>,A, r ..a3 pmA
 9M14r acr AaiAApApmApmzAqApMn3pA' 3AAI.Slr pclav

P8.7.9 386..n8..68..8 otm2

$$\begin{array}{c} 2. \\ \frac{Bx=\beta^2}{x^T Cx=\gamma^2} \end{array} \|Ax - b\|_2, \quad A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, B, C \in \mathbb{R}^{n \times n}.$$

X , ≠ X, X, X-Bγ.β,βIn X, , ≠ Z E R^{n×n}: Rⁿ : + R^mR^m, λλ ∈],), R'
 λRⁿTBZ = 3c.r, = nM.vn) R^mz Tc z = In. 3t,A pmpf > ... ≥ λ_n. Rⁿ(λ), λR^m, λλ))
 s = 10 (n λk λk λk λ_n. Z, = C(LC(LrCA(Lc)=nEQnE(XAT
 c r x = x rxS = xxrs - x a == x rz Sx 4A · x x

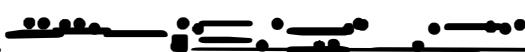
$$\begin{array}{c} .r a \\ y^T W y = \beta^2 - \lambda_n \gamma^2 \end{array} \text{IAx H2}$$

= W= x -a(M1 Maap_nI,

P8.7.10 ..86 ... R aH2.0t8. 7 d7

Notes and References for §8.7

k. .86 b.. 8.8.. .fe2.0tfeb228..fe. .8..otA - -



P. S (= E= (O= : S a = : X(122PI. De(= = yGE(Xe30= Xa= A(= = X(doco= : OO= aE= - n: (> (AO= Eθ - == NAMO review 41, iJTr ii5M

:A tAatr ptiippA ApMcA.3A: 88cpAai.zSMIS,ActAliAM23spA.Mpa3 si a = I hlm
 rmSvT5AAztIT

- 9vAM..M3.(T7Tv3 sp.zAAaAM,c23 Aai.zH hMSzA J. Numer. Anal. 13
 dir dJM

- v v5c.3 lvI.p2r .f (I .. .cAaAM.zc9A8i3z,A. E3: apAM.cpaI.pMrHr Mu6
 Alg. Applic. 211, 5J05I,

r A2A8ra3 2v9vr r 2. f l2A 2.MAtPEA: apAM.cpaI.pMrHr Mu6
 9r rAai,AMISzA6in. Alg. Applic. 302 3, 75 T7M

rv 5wr.a3 lvI.p2r .tIJJT Mctpar Ml. AM.npr hr MEcr la,r 3Sa2 2la32r
 .z23SzAAM.npr hcM16 JAM J. Matr. Anal. Applic. 28, 5JN5y4

y. {c (= o: p4 20 2Dxo= :)θ·nO: (= A: A=QfAa- .= MAOθO= (Aa= No= ((
 - ayO= XfaOθo= N 50 (JQNIN,

akA Tq.a 4t₁ t d₁ 4a M₁ dq dt G₁ 92 synalPstidHa .8 R₁Q₁s₁t t e H
q₁=e Q₁t₁H₁t₁d₁A₁e(q₁SIAM J. Matr. Anal. Applic. 32, 71 Ir 754

30Lp,I₁pInS,ttsS,A Enr,a,nA ttCCApunA.eA-S₁A₁AMc,H₁CAurA₁tAtint₁L
A,arulAaA₁at,uC.Ln,₁tn.,A,aenp₂A₁E₁2ItCCApp,E₁ A.apu,,n₁rp₁C.rA6p. A
Su,S,tAEhA,A,tenare

14s4I.u.na.aE .4.4 ac,+at,a = M₁tE J.Ae,A.c,a .. stCCA.unAaSu,S,AC = >Bx a
j.A(= XE CnA₁O₁₁ (= G(O₁ Numer. Math. 11, .. NNJ 4

.4 ucbaE 14.AnSAurAM₁T.3a 3,r,un.2C.ML280A,aEn,c,aAAaAM.,n.AfrAai,I₁ua
.21 y₁SIAM J. Numer. Anal. 9. Tdr dd4

34w₁atAr Aut₁M₁43a 3,r,unppG ppAtCCA.unAaAM.,n.Ae 9crAhi,FAiyG.in
Alg. Applic. 58. i5N₁d4

s4Ap,af₁Q₁uA=JJ₁J₁3a 91AnAaE s.S,A30r,unLpM₁ppAtCC3pMnAMEAaAm₁
9nrAai,O₁AM,S,AC SIAM J. Matrix Anal. Applic. 21, N₁Ir (Id4

h4 E..nAty244nr2Cy.aE u4anttAI₁M₁JNT₁3a,ttnt .. p3Ap,OA I₁Alp,EnLp₁fL2u n
IA=aACAaps,icar.pAst' CA₁EAA 8epAu.0cA₁erAaSM₁AG SIAM J. Matr.
Applic. 23, iT₁Ir i.4

.4 anttA,uf I₁J₁4nen,r,aQEn,r,a, IAeApn,a stCC' 53E₁A=ah₁M₁ SAM J. Matr
Anal. Applic. 26, INr I₁I₁4

9bS,,nLnar.aEAeaAtA.), B A.a S₁CSMp.8InAe

.4 hApA₁E .4.4 a202at,a = N.T4.9nrAa.OI At Ax = >Bx .np2w7aestCCApufAaE B,"
Comput. J. 12, 5d,iJ₁4

A44 AM.,uEN₁5₁AEI ALn,a . wa₁StCCA.unAaAu,n.AfrAai,I₁Au,SACy6₁commun.
ACM 16 iNii4

n45IC.a = N5T4.3a 3,r,unppGMLAwaeA₁StCCA.unAaAu,n.AELunb 9nrAaD₁2
,AG SIAM J. Matrix Anal. Applic. 14, 5T₁Ir 3d

54n₂a, 4 nnyAE -4-Aar = N.T4.3a 3,r,un.2C,u p2AAaAu,n.AECCApMoA .0
9nr2al O₁Au,SACy6₁umer. Algorithms 8, 17 N₁4

apAAbctLAaAAS,tcLnit₁GEAQ8mAAM₁A,CSnaLn,aA.Q B.. AA8M,0a2A,u2id4T₁4 4
f aLAuAtL₁lq₁5yAL2M,CSIL..n,a tlA2 A,,Sna.n,apctSAA₁3t₁AEAAe

A44Au..ME = .d7T₁Br,ucLpC₁7hE uf 2E3,l,pnaAp, unaE htn.niAE A= amp2a2,uaA₁P
Snap2a .. IA, st' CA₁Ap₁MC₁ACM T rs. Math. Sof w. 12, ITd MI dI
A4M₁h₁y 244nrpi.y ae u4 a2tAI₁M₁T₁3a ,iS uiaE3MA₁Or,M₁p₁u EALAAIE2₁e.n
.AuCnlahrhu₁6 SIAM J. Matrix Anal. Applic. 31, N₁N₁NNN 4

3t A CAan,8AEyat AA28ns,AM,pAttCCA.MAaAai,ISM,S,A₁GA a.IuOAhLAatnaas
m = n x V 1x m + q*xVx1x x wxxC2x n r 8 = n x 1xn S r C2xnxrx1s x x

n ' -S x .00= wq *Cu n +S₁x n iV2x 1qx + f Gxu x x S₁S x +. V @ x mazmazx
XO=A₁inN₁Alg Applic. 132, 7,r.NM

l, A4nr= N.T4a 9nrAaici,IA₁McLn,atr.t,Anr2VI,nAaI,pMnh3aAnOt EA= ar₁AS.
Lin. Alg. Applic. 208/209, iT₁Nd54

apAMMA .0AaAM.,n..c,at.2A.A,SnCA.2e

54BAAO₁N5T4 .A,Sn9crAaMAe,A₁2c,zMn.2G EA= lnIA.Mrlb₁M₁6 Numer. Math. 64
+ iO₁oZ.

e. AOJrM₁21. Do(ECn₁9A₁y0800E = JDG0- = 10v0= (A31h000v0((y0= -3M₁p0
rOx₁ S₁INM J. Matr. Anal. Applic. 25, 71 .d.4

.C,LSt CApp,E₁in .t, , aESS,R₁an,ae

54nnaE aM 4n= N5T4 .3 ..C.,St 3,r,MnppG . stCCA.unAaAu,2AEnrAaSu ..
Numer. Algorithms 4, N₁7T₁4

a4par.aE 54.4 n.y.4aMIS = N5T4a .pA,C.,St A.2,eMhAupIuSSAC
AaAu,2A0nrAa .IM,S,AC SIAM J. Sci. Comput. 19, N₁Ir N₁i.4

aA t2., piA,C,uAp, tt S,IL ttCCApunAr ES₁S₁6.h p rAaAM₁MtnpA₁ApSL2u If
= &0= E(OE0on(= GOG f C000onGOf A0x(O@a OE(y@= OE (A0bex0AP(Ir
OvOE66.9OT

- H. kJ y (= G.a (ACII) x 2e Dx10-XY E (r)f AGnG n (Elvln 0CA>0.11 = n E (rObn C
 tOyH > ExPCnA Nal lle Comput. 20 (Jd.vN) 7,
 usnarI u...lana, .21.sBE On.1u.Iu.21.unma.01.lulaA#8A,,E1e
 c.F. (= C (= X2ope6 D 1h = A E160Jn008 p((FpmECx f C0Cn) SIAM J. Numer. Anal.
 19 T7d5v
 99hr1.aE I 4s.IaE1uTN.d19.a..uET 3.18luOn.1esna,O B101E 1A,CSTn.n,6SIAM
 J. Numer. Anal. 18, 5dvJ.,
 mp 2sTmmt. .21.sBE nE1.n0fha s.1.uaE sIa,I haTa a r a x x sSs s n r xSx,ua +
 a,2 a 2x0nu w a a + ax Ss s Sx12an xx a ax S2s S62a1a r SIAM J.
 Numer. Anal. 20 7IN 7I.
 99hr1. Ndpv.3 :.1 ,a. 1ITIO., s.a ... areslaTnxnint.21As.aE.sB E 1A,CS
 tn.nTn6SIAM J. Numer. Anal. 21, Nd7 ..(1
 r .,n.N.5pv .w,aET ,8 hluxI uS.xn,Ila luOn.1EsnarI0B1 1/aE .. 3TT,An.1E sIS
 tSAy6SIAM J. Matrix Anal. Applic. 14 N.v 5iv
 f9M1sIa (.dpv .hluIuS..n,a3a.0TnT.1a lu,n.1E snarI0s1STS.A' NTn6 Math. 79
 7N:7In 1
 fN4 sIa.IJJp A,aEn,n,aCS lu.aE wA+ .uE9uu,uu .21la3u 1n.rEsnarI,,uB.,I1 E 1
 A,CS,Tn.n,au6SIAM J. Matrix Anal. Applic. 22, 51 5v5N.
 ,sN2la.aE av,n.IJJp .3 :.1 ,8 wA+ .uE9uu,uu8a,t/nT .23.' alu.OnsE1,uB.OI 1
 E 1A,CSTn.n,SIAM J. Matrix Anal. Applic. 90 N.5 5TJ
 p2 i un.n,a2uAx3un.xn,x21sBE nT8,t.1E nae
 M.T. eJp-j B R = Gn H A 0E 1(3x xC22aDi=(= (EO(v0CE(pA(vOChn = n E (AObn C
 00iy pA(FpmECx f C10tStSIAM J. Matrix Anal. Applic. 18 NJdIvNj.I4
 - ass1An,a\$1.118 sBE .aE.21S 1aAa ->B a c 1x a • axax,a 1
 b. T ylvExZzD DpJin = n E " 100y pA(FpmECx f C10v0C= vJn A+>nE(S
 ,1Cu6IT 24, .7d d51
 s. S,212,ET,u A,CSI,na21As.aEr 18luOn.1En8rI,i01 1E1A,CS,Tn.n,aTE1AunS1E
 g.w. Ovn 3 102za DeCx f p wOhey ,a ECx f C10CQO(EvOvQE pJCG CEA vAOd = N
 Numer. Math. 40 L.Tv5J7V
 g.w. Ovn 3 102zaDyAn vJGCEeCx f p r0Jn = n E (A00pA(FpmECx f C10v0C= =
 Matr x Pencils , we5SrT.uCGE 3, IIpl ,1ET4SundMB lu,lu .u+ljTMJ,
 9Ba „a .N.dpr A,CSI,narp1As.aE .1a lu,n.1E sn8lQuB' E 1A,CS,Tmau6Numer.
 Math. 46, IT,vi.Iv
 f,EsI „a .1J(lp .s.S0 A,CSI...n,a,, .21AsE 1A,CS,Tn.n,ac..a1I wnEn,r,a.0m,au6
 SIAM J. Matrix Anal. Applic. II NM(1
 p1 nE1. ITna215rS 1x,naxSu,A1E1u1 .21.sBE Su,SO16TEli1QS 1fnae
 NhnR3Nd7 ,A,CSI xna2r:1a luOn.1EsnarIQuB.OI 1E 1A,CSTn.n,SIAM J. Sci. Stat.
 Comput. 7, N(T Ni74
 z.) (O (= KJ(X22eODy n 3XE n f ECx f p y0Jn = n E (A00pA(FpmECx f C10
 00= y pA(FpmECx f C10v0C SIAM J. Sci. Comp. 14 (J.Tv(JNI,
 z.) (O (eaG)JrOx xna22eOD C.f- wD8 y n E (A00y pG(FpmECx f C10v0C SIAM
 J. Sci. Comput. 14, N7ivNd7,
 .p16 1x2,E/u A,CSI xna21.sBE naAOI ele
 z. ,Ex (n02ze Dyn(= ysyAy0CEeEc x f p y0Jn = n E (A00pA(FpmECx f C10
 vOc SIAM J. Numer. Anal. 1 .(dJiv(dJiv
 z. ,Ex (n = G e0on loxf 22Di=yEEp E.(pCvOr0OvA(E(ApCix f p v(vOcC(vOiy,
 sN n1E,aEJJpv .3 :1 3SSu.A2x .laru,n.1EsnarIQuB.OI 1E 1A,' Tn.n,6SIAM J.
 Matrix Anal. Applic. 27, i5Gii11
 s S,1C12,ET,u A,' SI,n1fSu,EA.8e uIT.unAs1ETu1EnTAITf21,nar S.S 1uTT

- k 9ntedSMQQQt6 lnt Rk6 dt mmEdt By TEs blds a hsMZ4ldh 5le qpp
 c(sia 8 dPhip 8 9B kdt SIAM J. Sci. Stat. Comput. 7, (NT)Nv
 51B, uAMaaE s.CC.M,carMNddi .3 hM,ElApfaEl nAONB.OIAEAA,CS,tcpCM
 a, I.pMcAAE w,aAAAOn.pna 168 Linear Algebr in Systems and Contr 1, w12.E.LL.
 ApO,AETef 3I hISOnA.pc,Bm,EASpcd31
 w,EAI,,M.aE.1 -p.MN..N.3 h3A., AaAM.0c.pc,8tppAMEn8Mn8r1OBMOI3EAA,i
 S,tnpn6in Alg. Applic. 147, i7.v.JV
 .1 -p. M.(uv.apAlAtpMnApAH,,MB.OIAEAA,CS,tnpc,al.pMcfinSOApXSIAM J. Matrix
 Anal. Applic. 12 (II N.iZ
 w,EAI,,M.aE.1v ..,MS..(u .apAlAtpMnApAH,,MB.OIAEAA,CS,tnpr,MC,SAM,ppAt
 3SSOnA.pcq SIAM J. Matrix Anal. Applic. 12, iJNv.u
 w,EAI,,M.aE BAI E,MA(OI uMraAM.On.nppAcarIONB.OIA8EV1 EAA,CS,tnpn6
 SIAM J. Matrix Anal. Applic. 13 ..5 .N.i
 .1 -p. MN..Ju.3 2ICAMCa, 3r,MnAACSIPnappA tpMcApAHM BEA,CS,tnpn,a
 I.pMnHs,ApXin Alg. Applic. 168, NvI.i
 .19 3ECt8vaIw,e8A.t+ u8E u la hI+MN.i uv.A,CSIpn8ppAsBE., a, I InarI,M
 I.pMnA SIAM J. Matrix Anal. Applic. 15, 577v5dM
 -vEMC.M.du.3AAIM.PACSI p,pc,a ppdM,elApr ,aECS,MB.OIAEAA,CS,tcpnpp
 3SSOln,8tXSIAM J. Numer. Anal. 35, (7. .iv
 EApIuIEAnpp.IAMaE AI,,M MUJJu.8 ppAA,CSI p,pc,a pp2 AtpMnA
 B,IAEAA,CS,tnpnA ppAA,tcaA snEAA,CS,tnpuXSIAM J. Matrix Anal. Applic. 22,
 .dJv7Nv
 E IApI.aE wIAI,,M MUJJu.8 . i.Mn.pn,a,MCIOpc,8 pp3VsBE.8E ppAs BEu6in
 Alg. Applic. 311, 7N TdM
 uMA,iAM.r,AtpMI ApisMAI.pnArAai 7SIAM,S,ACAAAC
 h ,.aA.pAMN..NZ.VI.EM.pc9nr2aiE,IAM,SOApXin Alg. Applic. 150 i.0v.J7i
 uvantAI,ME 2f .thr p,C MU Nv.spMI ApI MAE htAI EM,SOApX, 9crAai.,IAIM,SOApX
 .npp3SSOA,pn,at SIAM J. Matrix Anal. Applic. 23, NtvI.Jd
 u1 anttAE, IAAAMSAnhAai .apAVI: M.pc9nrA8i.OIAM,SOApX SIAM Review 43, 15v
 Id7v
 BVApMC.aa ISOp + naUJJu.hqt8Cc09crA8i.OIAM,SOApX CnOp,ara spMI ApI MAu
 Electr. T ns. Numer. Anal. 13 (JAN(di
 lvw20. u1v ,.OI Su8e 5v, .. MUJJu.3 LI tS.88SM,T cCIpp,EMppAVI.EM.LnA
 9nrAai.,IAIM,SOApX SIAM J. Matrix Anal. Applic. 26 /tyr NMw
 .vh rC.aa3avrC.ea.)v ra.,3 Cn.I vra puC2nwN116vchtpatopa(g.nguMCRMian,oa
 (pg,auAigg ISOpCtMgAiggmlS2aCpSTCtg2AG7. Matrix Anal. Applic. 28 NJI.N
 N,Ni
 whAtpAaeBjJw.2ICAMCaApp,EtMppAnEcr,8. tSAMS,OIAQEM,spAAai.,IAIM,
 QAC SIAM J. Matrix Anal. Applic. 28 NNTN,IV
 9Z 51aApIa1r Laru av.a1,c8.8E AvAI MUJJu.BnSM, pn,8p h,cat' h,naEMIA
 9nrAai.,rhM,S,ACAA spMI ApI MA.hMAtASm880r,MnppCtuFp. Appl. Math. 219,
 15T v II

Chapter 9

Functions of Matrices

P_e

P_I

P_n

P_O

G

G

Computing a function $f(A)$ of an n -by- n matrix A is a common problem in many application areas. Roughly speaking, if the scalar function $f(z)$ is defined on (A) , then $f(A)$ is defined by substituting " A for " z " in the "formula" f for $f(z)$. For example, if $f(z) = (1+z)/(1-z)$ and $1 \in (A)$, then $f(A) = (I+A)(I-A)^{-1}$.

The computations get particularly interesting when the function f is transcendental. One approach in this more complicated situation is to compute an eigenvalue decomposition $A = YBY^{-1}$ and use the formula $f(A) = Yf(B)Y^{-1}$. If B is sufficiently simple, then it is often possible to calculate $f(B)$ directly. This is illustrated in §91 for the Jordan and Schur decompositions.

Another class of methods involves the approximation of the desired function $f(A)$ with an easy-to-calculate function $g(A)$. For example, g might be a truncated Taylor series approximation to f . Error bounds associated with the approximation of matrix functions are given in §92.

In §93 we discuss the special and very important problem of computing the matrix exponential e^A . The matrix sign, square root, and logarithm functions and connections to the polar decomposition are treated in §94.

Reading Notes

Knowledge of Chapters 3 and 7 is assumed. Within this chapter there are the following dependencies:

§91 . . . §92 . . . §93

§94

Complementary references include Horn and Johnson (TMA) and the definitive text by Higham (FOM). We mention that aspects of the $f(A)$ -times-a-vector problem are treated in §102.

9.1 Eigenvalue Methods

Here are some examples of matrix functions:

$$p(A) = I + A,$$

$$r(A) = \begin{pmatrix} 1 & \\ & -1 \end{pmatrix}, \quad 2 \notin \sigma(A),$$

$$e^A = \sum_{k=0}^{\infty} \frac{A^k}{k!}.$$

Obviously, these are the matrix versions of the scalar-valued functions

$$p(z) = 1 + z,$$

$$r(z) = (1 - (z/2))^{-1}(1 + (z/2)), \quad z'' z,$$

$$e^z = \sum_{k=0}^{\infty} \frac{z^k}{k!}.$$

Given an n -by- n matrix A , it appears that all we have to do to define $f(A)$ is to substitute A into the formula for f . However, to make subsequent algorithmic developments precise, we need to be a little more formal. It turns out that there are several equivalent ways to define a function of a matrix. See Higham (FOM, §12). Because of its prominence in the literature and its simplicity, we take our "base" definition one that involves the Jordan canonical form (JCF).

,mhmhb eb 4kxtt ti hteSb p8OetTF Tkb

Suppose $A \in \mathbb{C}^{n \times n}$ and let

$$A = X \cdot \text{diag}(J_1, \dots, J_q) \cdot X^{-1} \tag{9.1.1}$$

be its JCF with

$$J_i = \begin{bmatrix} A & 1 & \cdots & \cdots & 0 \\ 0 & A & 1 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 1 \\ 0 & \cdots & \cdots & 0 & A \end{bmatrix} \in \mathbb{C}^{n_i \times n_i}, \quad i = 1:q. \tag{9.1.2}$$

The matrix function $f(A)$ is defined by

$$f(A) = X \cdot \text{diag}(F_1, \dots, F_q) \cdot X^{-1} \quad (913)$$

where

$$F_i = \begin{bmatrix} f(i) & J^{(1)}(i) & \dots & \dots & \frac{J^{(n-1)}(i)}{(n-1)!} \\ 0 & f(i) & \ddots & \dots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & f^{(1)}(\lambda_i) \\ 0 & \dots & \dots & \dots & f(\lambda_i) \end{bmatrix}, \quad i = 1:q, \quad (914)$$

assuming that all the required derivative evaluations exist.

,mhmib 7\\$b Ct KPlkbus\\$eb 9\\$3 x\\$\\$ tQtQ1kb

If f can be represented by a Taylor series on A 's spectrum then $f(A)$ can be represented by the same Taylor series in A . To fix ideas, assume that f is analytic in a neighborhood of $z_0 \in E$ and that f is r -times differentiable at z_0 . We have

$$f(z) = \sum_{k=0}^{\infty} \frac{J^{(k)}(z_0)}{k!} (z - z_0)^k, \quad |z - z_0| < r. \quad (915)$$

Our first result applies to a single Jordan block.

Lemma 9.1.1. Suppose $B \in \mathbb{C}^{m \times m}$ is a Jordan block and write $B = \lambda I_m + E$ where E is its strictly upper bidiagonal part. Given (915), if $|z - z_0| < r$, then

$$f(B) = \sum_{k=0}^{\infty} \frac{J^{(k)}(z_0)}{k!} (B - z_0 I_m)^k.$$

Pr of. Note that powers of E are highly structured, e.g.,

$$E = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad E^2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad E^3 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

In terms of the Kronecker delta, if $0 \leq p \leq n-1$, then $E^p_{ij} = (\delta_{ij} - p)$. It follows from (914) that

$$f(B) = \sum_{p=0}^{n-1} f^{(p)}(\lambda) \frac{E^p}{p!}. \quad (916)$$

On the other hand, if $p > m$, then $E^p = 0$. Thus, for any $k \leq j$ we have

$$\begin{aligned}(B - z_0 I)^k &= ((\lambda - z_0)I + E)^k = \sum_{p=0}^k \frac{k(k-1)\cdots(k-p+1)}{p!} \cdot (\lambda - z_0)^{k-p} \cdot E^p \\ &= \sum_{p=0}^{\min\{k,m-1\}} \left[\frac{d^p}{d\lambda^p} (\lambda - z_0)^k \right] \frac{E^p}{p!}.\end{aligned}$$

If N is a nonnegative integer, then

$$\sum_{k=0}^N \frac{f^{(k)}(z)}{k!} (B - zD)^k = \sum_{p=0}^N \frac{f^{(k)}(z)}{d^p} \left(\sum_{k=0}^N \frac{f^{(k)}(z)}{k!} (A - zD)^k \right) \frac{E^p}{p!}.$$

The lemma follows by taking limits with respect to N and using both (9.1.6) and the Taylor series representation of $f(z)$. D

A similar result holds for general matrices.

Theorem 9.1.2. If f has the Taylor series representation (9.1.5) and $D - zD$ is invertible for all $z \in \mathbb{C} \setminus \text{spec}(A)$, where $\text{spec}(A) \subset \mathbb{C}$, then

$$f(A) = \sum_{k=0}^{\infty} \frac{f^{(k)}(z)}{k!} (A - zD)^k.$$

Proof. Let the JCF of A be given by (9.1.1) and (9.1.2). From Lemma 9.1.1 we have

$$f(J) = \sum_{k=0}^{\infty} \alpha_k (J - zD)^k, \quad \alpha_k = \frac{f^{(k)}(z)}{k!},$$

for $i = 1, q$. Using the definition (9.1.3) and (9.1.4) we see that

$$\begin{aligned}f(A) &= X \cdot \text{diag} \left(\sum_{k=0}^{\infty} \alpha_k (J_1 - z\omega_1)^k, \dots, \sum_{k=0}^{\infty} \alpha_k (J_q - z\omega_q)^k \right) \cdot X^{-1} \\ &= X \cdot \left(\sum_{k=0}^{\infty} \alpha_k (J - zD)^k \right) \cdot X^{-1} \\ &= \sum_{k=0}^{\infty} \alpha_k X (A - zD) X^{-1} = \sum_{k=0}^{\infty} \alpha_k (A - zD)^k,\end{aligned}$$

completing the proof of the theorem. D

Important matrix functions that have simple Taylor series definitions include

$$\exp(A) = \sum_{k=0}^{\infty} \frac{A^k}{k!},$$

$$\log(J - A) = \sum_{k=1}^{\infty} \frac{A^k}{k}, \quad \text{if } A \neq J, A \in \mathbb{C}(A),$$

$$\sin(A) = \sum_{k=0}^{\infty} (-1)^k \frac{A^{2k+1}}{(2k+1)!},$$

$$\cos(A) = \sum_{k=0}^{\infty} (-1)^k \frac{A^{2k}}{(2k)!}.$$

For clarity in this section and the next, we consider only matrix functions that have a Taylor series representation. In that case it is easy to verify that

$$A \cdot f(A) = f(A) \cdot A \quad (917)$$

and

$$f(X^{-1}AX) = X \cdot f(A) \cdot X^{-1}. \quad (918)$$

,mhhrb etb AT\\$ f Sff kb eEE kt feb

If $A \in \mathbb{C}(A)$ is diagonalizable, then it is particularly easy to specify $f(A)$ in terms of A 's eigenvalues and eigenvectors.

Corollary 9.1.3. If $A \in \mathbb{C}(A)$, $A = X \cdot \text{diag}(A_1, \dots, A_n) \cdot X^{-1}$, and $f(A)$ is defined then

$$f(A) = X \cdot \text{diag}(f(\lambda_1), \dots, f(\lambda_n)) \cdot X^{-1}. \quad (919)$$

Pr 6 This result is a easy consequence of Theorem 9.1.2 since all the Jordan blocks are 1-by-1. \square

Unfortunately, if the matrix of eigenvectors is ill-conditioned, then computing $f(A)$ via (918) is likely to introduce errors of order $u_1(X)$ because of the required solution of a linear system that involves the eigenvector matrix X . For example, if

$$A = \begin{bmatrix} 1+10s & 1 \\ 0 & 1-10s \end{bmatrix}.$$

then any matrix of eigenvectors is a column-scaled version of

$$X = \begin{bmatrix} 1 & -1 \\ 0 & 2(1-10s) \end{bmatrix}$$

and has a 2-norm condition number of order 10^5 . Using a computer with machine precision $\approx 10^{-7}$, we find

$$f(x^{-1} \text{diag}(\exp\{l + 10 s\}, \exp\{l - 10 s\})x) = \begin{bmatrix} 2718307 & 2750000 \\ 000000 & 2718254 \end{bmatrix}$$

while

$$eA = \begin{bmatrix} 2718309 & 2718282 \\ 000000 & 2718255 \end{bmatrix}.$$

The example suggests that ill-conditioned similarity transformations should be avoided when computing a function of a matrix. On the other hand, if A is a normal matrix, then it has a perfectly conditioned matrix of eigenvectors. In this situation, computation of $f(A)$ via diagonalization is a recommended strategy.

ghgsb eb ufeDb p\\$ fk3k eTf Tk1a33 lk tsfb

Some of the difficulties associated with the Jordan approach to the matrix function problem can be circumvented by relying upon the Schur decomposition. If $A = QTQH$ is the Schur decomposition of A , then by (9.1.8),

$$f(A) = Qf(T)QH$$

For this to be effective, we need an algorithm for computing functions of upper triangular matrices. Unfortunately, an explicit expression for $f(T)$ is very complicated.

Theorem 9.1.4. Let $T = (t_{ij})$ be an n -by- n upper triangular matrix with $A_{ii} = t_{ii}$ and assume $f(T)$ is defined. If $f(T) = (f_{ij})$, then $f_{ij} = 0$ if $i > j$, $f_{ij} = f(A_i)$ for $i = j$, and for all $i \neq j$ we have

$$f_{ij} = \sum_{(s_0, \dots, s_k) \in S_{ij}} t_{s_0, s_1} t_{s_1, s_2} \cdots t_{s_{k-1}, s_k} f[\lambda_{s_0}, \dots, \lambda_{s_k}], \quad (9.1.10)$$

where S_{ij} is the set of all strictly increasing sequences of integers that start at i and end at j , and $[\lambda_{s_0}, \dots, \lambda_{s_k}]$ is the k th corner divided difference of f at $\{\lambda_{s_0}, \dots, \lambda_{s_k}\}$.

Pr. See Desdaux (1963), Davis (1973), or Van Loan (1975). \square

To illustrate the theorem, if

$$T = \begin{bmatrix} \lambda_1 & t_{12} & \bullet \\ 0 & \lambda_2 & t_{23} \\ 0 & 0 & \lambda_3 \end{bmatrix}$$

then

$$f(T) = \begin{bmatrix} f(A_1) & t_{12} \frac{f(A_2) - f(A_1)}{A_2 - A_1} & F_{13} \\ 0 & f(A_2) & t_{23} \frac{f(A_3) - f(A_2)}{A_3 - A_2} \\ 0 & 0 & f(A_3) \end{bmatrix},$$

where

$$F_{13} = \frac{1}{\lambda_3 - \lambda_1} \left[\frac{(\lambda_3 - f(\lambda_1))}{\lambda_3 - \lambda_2} + t_{12}t_{23} \cdot \frac{\frac{J(\lambda_3) - J(\lambda_2)}{\lambda_3 - \lambda_2} - \frac{f(\lambda_2) - J(\lambda_1)}{\lambda_2 - \lambda_1}}{\lambda_3 - \lambda_1} \right].$$

The recipes for the upper triangular entries get increasingly complicated as we move away from the diagonal. Indeed, if we explicitly use (9.1.10) to evaluate $f(T)$, then $O(2^j)$ fops are required. However, Parlett (1974) has derived an elegant recursive method for determining the strictly upper triangular portion of the matrix $F = f(T)$. It requires only $2n^2/3$ fops and can be derived from the commutativity equation $FT = TF$. Indeed, by comparing (i, j) entries in this equation, we find

$$\sum_{k=i}^{j-1} t_{ik} t_{kj} = \sum_{k=i}^{j-1} t_{ik} t_{kj} \quad j > i,$$

and thus, if t_{ii} and t_{11} are distinct,

$$f_{ij} = t_{ij} \frac{f_{jj} - f_{ii}}{t_{jj} - t_{ii}} + \sum_{k=i+1}^{j-1} \frac{t_{ik} f_{kj} - f_{ik} t_{kj}}{t_{jj} - t_{ii}}. \quad (9.1.11)$$

From this we conclude that f_{ij} is a linear combination of its neighbors in the matrix F that are to its left and below. For example, the entry f_{15} depends upon $f_{11}, f_{22}, f_{33}, f_{44}, f_{55}$ and f_{35} . Because of this, the entire upper triangular portion of F can be computed superdiagonal by superdiagonal beginning with $\text{diag}(f(t_0), \dots, f(t_m))$. The complete procedure is as follows.

m2sh5ni - svdiyI erwHsxnso55e This algorithm computes the matrix function $F = f(T)$ where T is upper triangular with distinct eigenvalues and f is defined on $\mathbb{R}(T)$.

```

for i = 1:n
    fii = f(tii)
end

for p = 1:n - 1
    for i = 1:n - p
        j = i + p
        s = tij(fjj - 1)
        for k = i + 1:j - 1
            u = u + tiklji - fkklji
        end
        fjj = s/(tll - tii)
    end
end

```

This algorithm requires $2n^2/3$ fops. Assuming that $A = QTQ^H$ is the Schur decomposition of A , $f(A) = QFQ^H$ where $F = f(T)$. Clearly, most of the work in computing $f(A)$ by this approach is in the computation of the Schur decomposition, unless f is extremely expensive to evaluate.

$$\det(A - \lambda I) = 0 \Rightarrow \text{Eigenvalues of } A$$

If A has multiple or nearly multiple eigenvalues, then the divided differences associated with Algorithm 9.1.1 become problematic and it is advisable to use a block version of the method. We outline such a procedure due to Parlett (1974). The first step is to choose Q in the Schur decomposition so that we have a partitioning

$$T = \begin{bmatrix} T_{11} & & & \\ 0 & \ddots & & \\ \vdots & & \ddots & \\ 0 & 0 & \cdots & T_{pp} \end{bmatrix}$$

where $(T_{ii})_{n_i}, (T_{jj})_{n_j} = \dots$ and each diagonal block is associated with an eigenvalue cluster. The methods of §7.6 are applicable for this stage of the calculation.

Partition $F = f(T)$ accordingly

$$F = \begin{bmatrix} 1 & & & \\ \vdots & \ddots & & \\ 0 & 0 & \ddots & F_{pp} \end{bmatrix}$$

and notice that

$$F_{ii} = f(T_{ii}), \quad i = 1:p$$

Since the eigenvalues of T_{ii} are clustered, these calculations require special methods. Some possibilities are discussed in the next section.

Once the diagonal blocks of F are known, the blocks in the strict upper triangle of F can be found recursively, as in the scalar case. To derive the governing equations, we equate (i,j) blocks in $FT = TF$ for $i \neq j$ and obtain the following generalization of (9.1.1):

$$F_{ij}T_{jj} - T_{ii}F_{ij} = T_{ij}F_{jj} - F_{ii}T_{ij} + \sum_{k=i+1}^{j-1} (T_{ik}F_{kj} - F_{ik}T_{kj}). \quad (9.1.12)$$

This is a Sylvester system whose unknowns are the elements of the block F_{ij} and whose right-hand side is "known" if we compute the F_{ij} one block superdiagonal at a time. We can solve (9.1.12) using the Bartels-Stewart algorithm (Algorithm 7.6.2). For more details see Higham (FOM, Chap. 9).

Q113r - ir lzs2Rs2rwAlo sFz-rE-l8 sRwrlr

Does the Schur-Parlett algorithm avoid the pitfalls associated with the diagonalization approach when the matrix of eigenvectors is ill-conditioned? The proper comparison of the two solution frameworks requires an appreciation for the notion of condition applied to the $f(A)$ problem. Toward that end we define the relative condition of f at matrix $A \in \mathbb{C}^{N \times N}$ is given as

$$\text{cond}_f(f, A) = \lim_{\epsilon \rightarrow 0} \sup_{\|E\| : \|E\| \leq \epsilon \|A\|} \frac{\|f(A+E) - f(A)\|}{\|f(A)\|}$$

This quantity is essentially a normalized F_{chet} derivative of the mapping A → f(A) and various heuristic methods have been developed for estimating its value.

It turns out that the careful implementation of the block Schur-Parlett algorithm is usually forward stable in the sense that

$$\frac{\|P - J(A)\|}{\|f(A)\|} \leq \epsilon$$

where P is the computed version of $f(A)$. The same cannot be said of the diagonalization framework when the matrix of eigenvectors is ill-conditioned. For more details, see Higham (FOM Chap. 3).

Problems

P9.1.1 J..8 .8

$$A = \begin{bmatrix} \lambda & \mu_1 \\ \mu_2 & \lambda \end{bmatrix}, \quad \mu_1\mu_2 < 0.$$

ro.8. 8868. .8..8. .8 ...86,8. .888.88.8. N.2 8bd.8...86N. 8b. A),

$$P9.1.2 \quad 86..88.8..62. \quad 88.6. \quad f(T) = +(-1)^{\sum_{i=1}^n T_i}$$

$$\text{P9.1.3 J.1.8 } A = [x_1 | \cdots | x_n]^T = [y_1 | \cdots | y_n]^H, \quad p_1$$

$$f(A) = \sum_{k=1}^n f(\lambda_i) x_i y_i^H.$$

P9-1-4 .686 6.

$$T = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix}; \quad f(T) = \begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix} \mathbf{J}$$

$$r \models a \wedge F_{11} \wedge f(T_{11}), \quad F_{22} \equiv f(T_{22}), \quad \dots \quad (f(T) \models \phi)$$

Notes and References for §9.1

$\bullet = r \times \bullet = \bullet a \Rightarrow x, \bullet x * a = w \times \bullet x \bullet a$ $\sum A = (\sum_{i=1}^{n+1} H_i f_i)(\sum_{i=1}^{n+1} H_i f_i)$

62 $\frac{1}{5} \cdot \frac{v_i}{v_j} = \frac{A^b}{A^H}$, $(0)(0) = 1(0), 1 + (\sum_{i=1}^{H-1} A_{11}) = 1 + 1 = 2$ Amer. Math. Monthly

aA „lnar SiSAutAAI8AAua'cp. ep2A A2 ueAAI.Sltc5clae np/AOipdat2ms2A (A)
=AI (

.. NAtAlo, h, i5TIV .wl,8et ,u p23SAApn20. l. 8Apclst I7pucAAt Numer. Math 5, (d-190.

A,OB8.lia .(TTi.3 sp,et l. p2Aipuch9bSl8A8c,u2,AunA₁₉S1Ottnt rAS₂lupNJu
NAS.Mp.A₁₈lip.AipnAtur8ciAutdpdli8A2AtpAuu 9arSi₁₅sSOA ·x = -3+.xu Ma
C = "4x = "x = + T4x4a ·a · T

$$\bullet s u = \Theta Q H w x n S + x = 4 = 2x = xxSx + xSx + xSx + xSx = x + 4x = 5x$$

c. $\text{OA} = +\alpha_{\text{max}} \cdot x \cdot 2 + 4 \text{ex, max } x = 4x, x = xx = zw = w \text{ & } 2s = \underline{\underline{xx}} = \underline{\underline{w}}$
 Alg. Applic. 14, NCFNIr,
 h8NiCiAt8e 2v., onr2i, .IJ.5T.3 sA,ur hiuAppc., u Al.S,pc8r Iipunl6I 8Aq8u11
 SIAM J. Matr. Anal. Applic. 25, i7vid,

0 point i.e. $\text{e}^A = \text{I} + A + \frac{A^2}{2!} + \dots + \frac{A^q}{q!}$

$\text{w}^5 \text{ rLNTCE.21CAucAACSILC}_{\text{a}} \text{ I.Luchr aALc, EAS.uLCAlfa,uCL2a}$
 $\text{hu,AAAtt2dAS,ulr If 2uM,dTn acAutcl, CA skAEAv}$
 $\text{9vWE..2At,IJITv.3SSu,bcCLAc,r,a.1c. Lc,au,9AM J. Matr Anal. Applic. 29, NJyN7,9NJ}$
 $\text{apAtAatC2cLc,Luchr aAL2At,SAuLIuS.IctcAIttAEc}$
 $\text{A25AaaAtaE 3,vn,IS ,Nd..1 A,aEcLc,9LcCLAtu I.Luchr aAL2At,9AM J. Matr Anal. Applic. 10, NJy,M}$
 $\text{A25AaaAtaE 3,vn,IS ,N., 2sC.10s,CSQAL,LctLcA,0EcLc,9L2C.LAt, AaAu,0Lucr aAL2At,9AM J. Sci. Comput. 15, 57-7NM}$
 $\text{11,Lp2 ,N y,MA,aE2c,9Lc,L2au I.Lu2fr aAL2At LpAApI EAA,CS,t2ay9AM J. Matr Anal. Applic. 16, 7,Td,}$

9.2 Approximation Methods

We now consider a class of methods for computing matrix functions which at first glance do not appear to involve eigenvalues. These techniques are based on the idea that, if $g(z)$ approximates $f(z)$ near $A(A)$, then $f(A)$ approximates $g(A)$, e.g.

$$e^A = \text{I} + A + \frac{A^2}{2!} + \dots + \frac{A^q}{q!}.$$

We begin by bounding $\|f(A) - g(A)\|$ using the Jordan and Schur matrix function representations. We follow this discussion with some comments on the evaluation of matrix polynomials.

,hhhb eb 4k.ttb ett PKeTeb

The Jordan representation of matrix functions (Theorem 9.1.2) can be used to bound the error in an approximant $g(A)$ of $f(A)$.

Theorem 9.2.1. Assume that

$$A = X \cdot \text{diag}(J_1, \dots, J_q) \cdot X^{-1}$$

is the JCF of $A_E \in \mathbb{K}^{n \times n}$ with

$$J_i = \begin{bmatrix} \lambda_i & 1 & \cdots & \cdots & 0 \\ 0 & A & 1 & \vdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & \cdots & \cdots & \cdots & A \end{bmatrix}, \quad \text{n-by-n,}$$

for $i = 1:q$. If $f(z)$ and $g(z)$ are analytic on an open set containing $A(A)$, then

$$\|f(A) - g(A)\| \leq \max_{\substack{1 \leq i \leq p \\ 0 \leq r \leq n_i - 1}} n_i \frac{|f^{(r)}(\lambda_i) - g^{(r)}(\lambda_i)|}{r!}.$$

Pr of. Defning $h(z) = f(z) - g(z)$ we have

$$\|f(A) - g(A)\|_2 = \|X \text{diag}(h(J_1), \dots, h(J_q)) X^{-1}\|_2 = \|Z(X) \max_{1 \leq i \leq q} \|h(J_i)\|_2\|_2$$

Using Theorem 9.12 and equation (238) we conclude that

$$\|h(J_i)\|_2 \leq n_i \max_{0 \leq r \leq n_i-1} \frac{|h^{(r)}(\lambda_i)|}{r!}$$

thereby proving the theorem D

ihimib eb ufeDxb ett PKeTeb

If we use the Schur decomposition $A = QTQ^H$ instead of the Jordan decomposition, then the norm of T's strictly upper triangular portion is involved in the discrepancy between $f(A)$ and $g(A)$.

Theorem 9.22 Let $QHAQ = T = \text{diag}(i) + N$ be the Schur decomposition of $A \in \mathbb{M}_n$, with N being the strictly upper triangular portion of T . If $f(z)$ and $g(z)$ are analytic on a closed convex set Ω whose interior contains (A) , then

$$\|f(A) - g(A)\|_F : \sum_{r=0}^n \delta_r \frac{\|N^r\|_F}{r!}$$

where

$$\delta_r = \sup_{z \in \Omega} |f^{(r)}(z) - g^{(r)}(z)|.$$

Pr of. Let $h(z) = f(z) - g(z)$ and set $H = (h_{ij}) = h(A)$. Let \mathcal{E} denote the set of strictly increasing integer sequences (s_0, s_1, \dots, s_r) with the property that $s_0 = i$ and $s_r = j$. Notice that

$$h_{ij} = \sum_{r=1}^{j-i} s_r$$

and so from Theorem 9.13 we obtain the following for all i, j :

$$h_{ij} = \sum_{\substack{r=1 \\ s \in S}}^{j-i} s_{s_0, s_1, s_2, \dots, s_{r-1}, s_r} h[s_0, \dots, s_r] = \sum_{\substack{r=1 \\ s \in S}}^{j-i} s_{s_0, s_1, s_2, \dots, s_{r-1}, s_r} h[\lambda_{s_0}, \dots, \lambda_{s_r}]$$

Now since H is convex and h analytic, we have

$$|h[\lambda_{s_0}, \dots, \lambda_{s_r}]| \leq \sup_{z \in \Omega} \frac{|h^{(r)}(z)|}{r!} = \frac{\delta_r}{r!}. \quad (921)$$

Furthermore if $|N|^r = (n_{ij}^{(r)})$ for $r \geq 1$, then it can be shown that

$$n_{ij}^{(r)} = \begin{cases} 0, & j \neq i + r, \\ \sum_{s \in S_{ij}^{(r)}} |n_{s_0, s_1} n_{s_1, s_2} \cdots n_{s_{r-1}, s_r}|, & j = i + r. \end{cases} \quad (922)$$

The theorem now follows by taking absolute values in the expression for h_{ij} and then using (921) and (922).

There can be a pronounced discrepancy between the Jordan and Schur error bounds. For example, if

$$A = \begin{bmatrix} -0.01 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0.01 \end{bmatrix}.$$

If $f(z) = e^z$ and $g(z) = 1 + z + z^2/2$, then $\|f(A) - g(A)\| \approx 10.5$ in either the Frobenius norm or the 2-norm. Since $\text{L}(X) \approx 10^7$, the error predicted by Theorem 9.2.1 is $O(1)$, rather pessimistic. On the other hand, the error predicted by the Schur decomposition approach is $O(10^3)$.

Theorems 9.2.1 and 9.2.2 remind us that approximating a function of a normal matrix is more complicated than approximating a function of a scalar. In particular, we see that if the eigensystem of A is ill-conditioned and/or A 's departure from normality is large, then the discrepancy between $f(A)$ and $g(A)$ may be considerably larger than the maximum of $|f(z) - g(z)|$ on $\lambda(A)$. Thus, even though approximation methods avoid eigenvalue computations, they evidently appear to be influenced by the structure of A 's eigensystem. It is a perfect venue for pseudospectral analysis.

gjhrb Gt KPKbe33 .k eT.t tf eb

A common way to approximate a matrix function such as e^A is by truncating its Taylor series. The following theorem bounds the errors that arise when matrix function such as these are approximated via truncated Taylor series.

Theorem 9.2.3. If $f(z)$ has the Taylor series

$$f(z) = \sum_{k=0}^{\infty} \alpha_k z^k$$

on an open disk containing the eigenvalues of $A \in \mathbb{C}^{n \times n}$, then

$$\left\| f(A) - \sum_{k=0}^q \alpha_k A^k \right\|_2 \leq \frac{n}{(q+1)!} \max_{0 \leq s \leq 1} \|A^{q+1} f^{(q+1)}(As)\|_2.$$

Pr 6 Define the matrix $E(s)$ by

$$f(As) = \sum_{k=0}^q \alpha_k (As)^k + E(s), \quad 0 \leq s \leq 1. \quad (923)$$

If $f_{ij}(s)$ is the (i,j) entry of $f(A_s)$, then it is necessarily analytic and so

$$f_{ij}(s) = \left(\sum_{k=0}^q \frac{f_{ij}^{(k)}(0)}{k!} s^k \right) + \frac{f_{ij}^{(q+1)}(\varepsilon_{ij})}{(q+1)!} s^{q+1} \quad (924)$$

where ε_{ij} satisfies $0 < \varepsilon_{ij} < 1$.

By comparing powers of s in (923) and (924) we conclude that $e_{ij}(s)$, the (i,j) entry of $E(s)$, has the form

$$e_{ij}(s) = \frac{f_{ij}^{(q+1)}(\varepsilon_{ij})}{(q+1)!} s^{q+1}.$$

Now $A^{q+1}(s)$ is the (i,j) entry of $A^{q+1}f(q+1)(A_s)$ and therefore

$$|e_{ij}(s)| \leq \max_{0 \leq s \leq 1} \frac{f_{ij}^{(q+1)}(s)}{(q+1)!} \leq \max_{0 \leq s \leq 1} \frac{\|A^{q+1}f(q+1)(A_s)\|_F}{(q+1)!}.$$

The theorem now follows by applying (238). \square

We mention that the factor of n in the upper bound can be removed with more careful analysis. See Mathias (1993).

In practice, it does not follow that greater accuracy results by taking a longer Taylor approximation. For example, if

$$A = \begin{bmatrix} -49 & 24 \\ -64 & 31 \end{bmatrix},$$

then it can be shown that

$$e^A = \begin{bmatrix} -0.735759 & 0.551819 \\ -1.471518 & 1.103633 \end{bmatrix}.$$

For $q = 59$ Theorem 923 predicts that

$$\left\| e^A - \sum_{k=0}^q \frac{A^k}{k!} \right\|_2 \leq \frac{n}{(q+1)!} \max_{0 \leq s \leq 1} \| A^{q+1} e^{As} \|_2 \leq 10^{-60}.$$

However, if $n = 10^{-1}$, then we find

$$f(A) = \left[\sum_{k=0}^{59} \frac{A^k}{k!} \right] = \begin{bmatrix} -2225880 & -14322766 \\ -6149981 & -3474280 \end{bmatrix}.$$

The problem is that some of the partial sums have large elements. For example, the matrix $I + A + \dots + A^{57}/17!$ has entries of order 10^{-7} . Since the machine precision is approximately 10^{-7} , rounding errors larger than the norm of the solution are sustained.

The example highlights the a well known shortcoming of truncated Taylor series approximation it tends to be effective only near the origin. The problem can sometimes be circumvented through a change of scale. For example, by repeatedly using the *double angle formulae*

$$\cos(2A) = 2\cos(A)^2 - I, \quad \sin(2A) = 2\sin(A)\cos(A),$$

the cosine and sine of a matrix can be built up from Taylor approximations to $\cos(A/2^k)$ and $\sin(A/2^k)$:

So= Taylor approximate to $\sin(A/2^k)$

Co= Taylor approximate to $\cos(A/2^k)$

for j = 1:k

$$Si = 2Sj - 1G - 1$$

$$G = 2Cj - 1I$$

end

Here k is a positive integer chosen so that, say, $|A| < 2^{-k}$. See Serbin and Blalock (1979), Higham and Smith (2008), and Hargreaves and Higham (2005).

,himsb bf t P Df Tt"bt f .TebPK tkTtPeb

Since the approximation of transcendental matrix functions usually involves the evaluation of polynomials, it is worthwhile to look at the details of computing

$$p(A) = b_0I + b_1A + \cdots + b_qA^q$$

where the scalars $b_0, \dots, b_q \in \mathbb{R}$ are given. The most obvious approach is to invoke Horner's scheme

Given a matrix A and $b(q)$, the following algorithm computes the polynomial $F = b_qA^q + \cdots + b_1A + b_0I$.

$$F = b_qA + b_{q-1}I$$

for k = q - 2 - 10

$$F = AF + b_kI$$

end

This requires $q - 1$ matrix multiplications. However, unlike the scalar case, this summation process is not optimal. To see why, suppose $q = 9$ and observe that

$$p(A) = A^3(A^3(b_9A^3 + (b_8A^2 + b_7A + b_6I)) + (b_5A^2 + b_4A + b_3I)) + b_2A^2 + b_1A + b_0I.$$

Thus, $F = p(A)$ can be evaluated with only four matrix multiplications

$$A2 = A^2,$$

$$A3 = AA2$$

$$F_1 = b_9A3 + b_8A2 + b_7A + b_6I,$$

$$F_2 = A3F_1 + b_5A2 + b_4A + b_3I,$$

$$F = A3F_2 + A2 + b_1A + b_0I.$$

In general, if s is any integer that satisfies $1 \leq s \leq J$, then

$$p(A) = \sum_{k=0}^r B_k \cdot (A^s)^k, \quad r = \text{floor}(q/s), \quad (925)$$

where

$$B_k = \begin{cases} b_{sk+s-1} A^{s-1} + \dots + b_{sk+1} A + b_{sk} I, & k = 0:r-1, \\ b_q A^{q-sr} + \dots + b_{sr+1} A + b_{sr} I, & k = r. \end{cases}$$

After A^2, \dots, A^s are computed, then Horner's rule can be applied to (9.2.5) and the net result is that $p(A)$ can be computed with $s+r-1$ matrix multiplications. By choosing $s = \text{floor}(\sqrt{q})$, the number of matrix multiplications is approximately minimized. This technique is discussed by Paterson and Stockmeyer (1973). Van Loan (1978) shows how the procedure can be implemented without storage arrays for A^2, \dots, A^s .

~~high level language~~ \rightarrow KEDf Tt'b LjSkeb Kb tb ht fxTeb

The problem of raising a matrix to a given power deserves special mention. Suppose it is required to compute A^B . Noting that $A^4 = (A^2)^2$, $A^8 = (A^4)^2$, and $A^B = ABA^4A$, we see that this can be accomplished with just five matrix multiplications. In general we have

~~matlab~~ 9.2.2 en14TFp61 i114.rp The following algorithm computes $F = A^s$ where s is a positive integer and $A \in \mathbb{R}^{n \times n}$.

```

Let  $s = \sum_{k=0}^t f_k 2^k$  be the binary expansion of  $s$  with  $f_t = 0$ 
Z = A, q = 0
while /q = 0
    z = z^2, q = q+1
end
F = Z
for k = q+1:t
    Z = Z^2
    if /k = 0
        F = FZ
    end
end

```

The algorithm requires at most $2\text{floor}(\log_2(s))$ matrix multiplications. If s is a power of 2, then only $\log_2(s)$ matrix multiplications are needed.

~~high level language~~ etfS'x tf Tt'bht ~~matlab~~ Tkeb

and that A is the solution of the integration of a parameterized differential equation $y'(t) = f(t)y(t) + g(t)$. We can write $y(t) = A(t)f(t) + B(t)g(t)$

approximate

$$\mathbf{F} = \int_0^b f(At)dt \Leftrightarrow [\mathbf{F}]_{ij} = \int_0^b [f(At)]_{ij} dt$$

by applying any suitable quadrature rule. For example, with Simpson's rule, we have

$$\mathbf{F} - \tilde{\mathbf{F}} = \frac{h}{3} \sum_{k=0}^m w_k f(A(a + kh)) \quad (926)$$

where m is even, $h = (b - a)/m$, and

$$w_k = \begin{cases} 1 & k = 0, m, \\ 4 & k \text{ odd}, \\ 2 & k \text{ even, } k \neq 0, m. \end{cases}$$

If $(d^4/dz^4)f(zt) = f^{(4)}(zt)$ is continuous on $[a, b]$ and if $f^{(4)}(At)$ is defined on this same interval, then it can be shown that $\mathbf{F} = \mathbf{F} + \mathbf{E}$ where

$$\|\mathbf{E}\|_2 \leq \frac{nh^4(b-a)}{180} \max_{a \leq t \leq b} \|f^{(4)}(At)\|_2. \quad (927)$$

Let f_{ij} and e_{ij} denote the (i, j) entries of \mathbf{F} and \mathbf{E} , respectively. Under the above assumptions we can apply the standard error bounds for Simpson's rule and obtain

$$|e_{ij}| \leq \frac{h^4(b-a)}{180} \max_{a \leq t \leq b} |e_i^T f^{(4)}(At) e_j|.$$

The inequality (927) now follows since $\|\mathbf{E}\|_2 \leq \max_{a \leq t \leq b} |e_{ij}|$ and

$$\max_{a \leq t \leq b} |e_i^T f^{(4)}(At) e_j| \leq \max_{a \leq t \leq b} \|f^{(4)}(At)\|_2.$$

Of course, in a practical application of (926), the function evaluations $f(A(a + kh))$ normally have to be approximated. Thus, the overall error involves the error in approximating $f(A(a + kh))$ as well as the Simpson rule error.

hahfb ebkk f \$kfb f)\$b .tDfeKb (tf \$'xt Plsxx Dff Tkb

Yet another way to define a function of a matrix $C E \in \mathbb{M}_n$ is through the Cauchy integral theorem. Suppose $f(z)$ is analytic inside and on a closed contour Γ which encloses $\lambda(A)$. We can define $f(A)$ to be the matrix

$$f(A) = \frac{1}{2\pi i} \oint_{\Gamma} f(z)(zI - A)^{-1} dz. \quad (928)$$

The integral is defined on an element-by-element basis

$$f(A) = (f_{kj}) \implies f_{kj} = \frac{1}{2\pi i} \oint_{\Gamma} f(z)e_k^T(zI - A)^{-1} e_j dz.$$

Notice that the entries of $(\mathbf{zI} - \mathbf{A})^{-1}$ are analytic on \mathbf{r} and that $f(\mathbf{A})$ is defined whenever $f(z)$ is analytic in a neighborhood of $A(\mathbf{A})$. Using quadrature and other tools, Hale, Higham, and Trefethen (2007) have shown how this characterization can be used in practice to compute certain types of matrix functions.

Problems

P9.2.1 e..fe>..101@

P9.2.2 .316 13.2fe A P O Venegip A) lbr ttAE t.r t= EpS, laE
II Or. + A P II A P I - II A P

P9.2.3 .fe13.1... 10 .9f6 2 .. 1. fN..168e....8bc..1fe1.or-

$$L'' + \frac{1}{k^2} L \left(\frac{1}{k^2} \frac{\partial^2}{\partial x^2} \right) - \frac{1}{k^2} a : a L \left(\frac{1}{k^2} \frac{\partial^2}{\partial x^2} \right)$$

P9.2.41.. A E Rxn_{k(F)} X₁ X₂ An J Shy I H

Ex 1: If $f(x) = x^2 - 4x + 3$, find $\int_1^3 f(x) dx$.
 Solution: $\int_1^3 (x^2 - 4x + 3) dx = \left[\frac{x^3}{3} - 4x^2 + 3x \right]_1^3 = \left(\frac{27}{3} - 4(9) + 9 \right) - \left(\frac{1}{3} - 4 + 3 \right) = 27 - 36 + 9 - \frac{1}{3} + 4 - 3 = -\frac{1}{3}$

P9.2.5 1.... A $\in \mathbb{R}^{3 \times 3}$ " + LW EQ E k $\alpha x, f_k y$ $A^4 = al + \{A\}$, STE li 1QS
 M1AIM1Ar, M1Ar, fki yamA Ω $+ fka, M - 3$

Notes and References for §9.2

13. 8.1fe..8fe1P8 .. 8. N. .8..1.c.82fe1, feffe.....f fe, or

N§. 3.1...1. .f 55.2бP9 2fo.23.c..... 1>c1..... N..1fe..f1fe1.c.....
21P.....1. 318.8 fe.87 SIAM J. Comput. 2 7Jr 774

E,925al.2.,dNT The Art of Computer Programming, Vol. 2. Seminumerical Algorithms, t1A,aE
1Er .r 3Er tla Itatm1Er ar UBM

ap1,,Ma1M ,I.rl,aCLM8,0talCr .0t.a0t.1E r ae

A2uB.a n.a ,NTdT.3 2.1 la LpBi.OI..r ,a., I..Mr bl,ta,Cr .Or IESE ' ns. Avtom.
Contr lAC-24 5JG6IN4

..prMS1A.t .. MrI bAr ,SSM,br C.LaEari.Ol..r ,aM'Er tAI tt1Eae

.. w,O .aE2 22CCrMNdT4a .p191,I.r ,8,, I..Mr b aAr ,atila,t h.1Ms1Mu1
SIAM J. Matr. Anal. Appl. 11(1) 1-4

I21..p2t ..N 5T 4SSMlbr C..r, h.Mr bB,IrE IaALr [htu6](#) J. Matr x Anal. Applic. 14,
NJ 7N75NJ

222.2rpC .aE h2325ar rpN .T. .I. MrHbMr r a24r1hM1A2r3Mr .pC As16 M J.
Matti Anal. Applic. 16, 5, 514

h2s1,r .ar ,NiT4 ,a .p1 E IM2.r i1t1.LMrhbMtu6SIAM J. Matrix Anal. Appl. 17, 7,Jr 7d4

E 251wMatL 14 FeAM E 22 n.l.a , IJTT L Lr laO1 LMrrb aA.r laE laQ,al 1SE. It u6
 SIAM J. Matr. Anal. Applic. 22, N,r N4

uIM Er tAIth. 161pEt MICSILr.21tr a1 Afr a1. C.Mrthe

s4sIMsraE s2,,AO,NT,T 43a 30r,Mr .pMA,CSI,r ap1ILMrAbt2alU SIAM J. Sci. Stat. Comput. 1 Ndr IJ9

222r rp.CaE I „MsCr ,LJ,5T A,CSI,r aEpI.Mr A,traLuGumer. Algorithms 34 N 5 I 74
.2 .MrMli ltaE 22 .rrp.C ,IJJT 491Ar 18,r,Mr .pMprI.LMrAhr aAE sr a 1Nnumer.
Algorithms 40 5d5 41

ap1AlCSI-r la / (A) = =i)laS)kS= E)kBEI= i=.)IMB. i)E

I. I.-BPL+, II kBp .), 84 FBdBk/B) B-M. ®Sb=k)1 Aa, „r, 3TuAE l2,L1E I..Mr b
r. aAr. tAlaJ. JML. JrM. OSM. J. Numer. Anal. 46 L. Jr. L. 54

9.3 The Matrix Exponential

One of the most frequently computed matrix functions is the exponential

$$e^{At} = \sum_{k=0}^{\infty} \frac{(At)^k}{k!}.$$

Numerous algorithms for computing e^{At} have been proposed, but most of them are of dubious numerical quality, as is pointed out in the survey articles by Moler and Van Loan (1978) and its update Moler and Van Loan (2003). In order to illustrate what the computational difficulties are, we present a "scaling and squaring" method based upon Padé approximation. A brief analysis of the method follows that involves some e^{At} perturbation theory and includes comments about the shortcomings of eigenanalysis in settings where nonnormality prevails.

,hrghb eb .ttrb e33 .k eTtf Tktb h\\$ f ekfb

Following the discussion in §9.2, if $g(Z) = e^Z$, then $g(A) = e^A$. A very useful class of approximants for this purpose are the Padé fractions defined by

$$R_{pq}(z) = D_{pq}(z)^{-1} N_{pq}(z),$$

where

$$N_{pq}(z) = \sum_{k=0}^p \frac{(p+q-k)! k!}{(p+q)! k! (p-k)!} z^k$$

and

$$D_{pq}(z) = \sum_{l=0}^q \frac{(p+q-l)! l!}{(p+q)! l! (q-l)!} (-z)^l.$$

Notice that

$$R_{pq}(z) = 1 + z + \dots + z^p$$

is the order- p Taylor polynomial.

Unfortunately, the Padé approximants are good only near the origin, as the following identity reveals

$$e^A = R_{pq}(A) + \frac{(-1)^q}{(p+q)!} A^{p+q+1} D_{pq}(A)^{-1} \int_0^1 u^p (1-u)^q e^{A(1-u)} du. \quad (93.1)$$

However, this problem can be overcome by exploiting the fact that

$$e^A = (e^{A/m})^m$$

In particular, we can scale A by m such that $F_{pq} = R_{pq}(A/m)$ is a suitably accurate approximation to $e^{A/m}$. We then compute F_j using Algorithm 9.2.2. If m is a power of two, then this amounts to repeated squaring and so is very efficient. The success of the overall procedure depends on the accuracy of the approximant

$$F_{pq} = \left(R_{pq} \left(\frac{A}{2^j} \right) \right)^{2^j}.$$

In Moler and Van Loan (1978) it is shown that, if

$$\frac{\|A\|_\infty}{2^j} \leq 2,$$

then there exists an $E \in \mathbb{I}^{n \times n}$ such that $F_{pq} = e^{A^T E} A E = EA$, and

$$\|E\|_\infty \leq \epsilon(p, q) \|A\|_\infty,$$

where

$$\epsilon(p, q) = 2^3(p+q) \frac{Pq!}{(p+q)!(p+q+1)!}.$$

Using these results it is easy to establish the inequality

$$\frac{\|e^{A^T} F_{pq}\|_\infty}{\|e^{A^T}\|_\infty} \leq \epsilon(p, q) \|A\|_\infty e^{\|pq\|_A \delta}.$$

The parameters p and q can be determined according to some relative error tolerance. Since F_{pq} requires about $j + \max\{p, q\}$ matrix multiplications, it makes sense to set $p = q$ as this choice minimizes $\epsilon(p, q)$ for a given amount of work. Overall we obtain

mpishel(- 9.3.1 (Scaling and Squaring) Given $\delta > 0$ and $A \in \mathbb{I}^{n \times n}$, the following algorithm computes $F = e^{A^T} E$ where $\|E\|_\infty \leq \delta \|A\|_\infty$.

$$j = \max\{0, 1 + \text{fcor}(\log_2 \|A\|_\infty)\}$$

$$A = A/2^j$$

Let q be the smallest nonnegative integer such that $\epsilon(q, q) \leq \delta$

$$D = I, N = I, X = I, c = 1$$

for $k = 1:q$

$$c = c \cdot (q - k + 1) / ((2q - k + 1)k)$$

$$X = AX, N = N + c \cdot X, D = D + (-1)^k c \cdot X$$

end

Solve $DF = N$ for F using Gaussian elimination

for $k = 1:j$

$$F = F^2$$

end

This algorithm requires about $2(q+j+1/3)n^3$ ops. Its roundoff error properties have been analyzed by Ward (1977). For further analysis and algorithmic improvements see Higham (2005) and Al-Mohy and Higham (2009).

The special Horner techniques of §9.2.4 can be applied to quicken the computation of $D = D_{qq}(A)$ and $N = N_{qq}(A)$. For example, if $q = 8$ we have $N_{qq}(A) = U + AV$ and $D_{qq}(A) = U - AV$ where

$$U = c_0 I + c_2 A^2 + (c_4 I + c_6 A^2 + c_8 A^4) A^4$$

and

$$V = c_1 I + c_3 A^2 + (c_5 I + c_7 A^2) A^4.$$

Clearly, N and D can be computed with five matrix multiplications instead of seven as required by Algorithm 9.3.1.

$\text{UAPAA} \rightarrow N = A - G A T \rightarrow A$

Is Algorithm 9.3.1 stable in the presence of roundoff error? To answer this question we need to understand the sensitivity of the matrix exponential to perturbations in A . The rich structure of this particular matrix function enables us to say more about the condition of the e^A problem than is typically the case for a general matrix function (See §9.1.6.)

The starting point in the discussion is the initial value problem

$$\dot{X}(t) = AX(t), \quad X(0) = I,$$

where $A, X(t) \in \mathbb{R}^{n \times n}$. This has the unique solution $X(t) = e^{At}$, a characterization of the matrix exponential that can be used to establish the identity

$$e^{(A+E)t} - e^{At} = \int_0^t e^{A(t-s)} E e^{(A+E)s} ds.$$

From this it follows that

$$\frac{\|e^{(A+E)t} - e^{At}\|_2}{\|e^{At}\|_2} \leq \frac{\|E\|_2}{\|e^{At}\|_2} \int_0^t \|e^{A(t-s)}\|_2 \|e^{(A+E)s}\|_2 ds.$$

Further simplifications result if we bound the norms of the exponentials that appear in the integrand. One way of doing this is through the Schur decomposition. If $Q^H A Q = \text{diag}(\lambda_i) + N$ is the Schur decomposition of $A \in \mathbb{R}^{n \times n}$, then it can be shown that

$$\|e^{At}\|_2 \leq e^{\alpha(A)t M_s(t)}, \quad (9.3.2)$$

where

$$\alpha(A) = \max\{\operatorname{Re}(\lambda) : \lambda \in \sigma(A)\} \quad (9.3.3)$$

is the spectral abscissa and

$$M_s(t) = \sum_{k=0}^{\infty} \frac{\|Nt\|_2^k}{k!}.$$

With a little manipulation it can be shown that

$$\frac{\|e^{(A+E)t} - e^{At}\|_2}{\|e^{At}\|_2} \leq t \|E\|_2 M_s(t)^2 \exp(t M_s(t) \|E\|_2).$$

Notice that $M_s(t) = 1$ if and only if A is normal, suggesting that the matrix exponential problem is 'well-behaved' if A is normal. This observation is confirmed by the behavior of the matrix exponential condition number $v(A, t)$, defined by

$$v(A, t) = \max_{\|E\|_2 \leq 1} \left\| \int_0^t e^{A(t-s)} E e^{As} ds \right\|_2 \frac{\|A\|_2}{\|e^{At}\|_2}.$$

This quantity, discussed by Van Loan (1977), measures the sensitivity of the map $A \rightarrow e^{At}$ in that for a given t , there is a matrix E for which

$$\frac{\|e^{(A+E)t} - e^{At}\|_2}{\|e^{At}\|_2} \approx v(A, t) \|E\|_2.$$

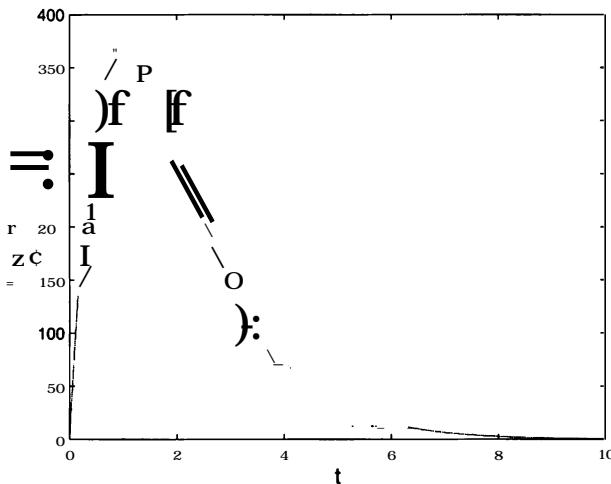


Figure 93.1. $\| e^{At} \|_2$ can grow even if $a(A) \neq 0$

Thus, if $v(A, t)$ is large, small changes in A can induce relatively large changes in e^{At} . Unfortunately, it is difficult to characterize precisely those A for which $v(A, t)$ is large. (This is in contrast to the linear equation problem $Ax = b$, where the ill-conditioned A are neatly described in terms of SVD.) One thing we can say, however, is that $v(A, t) \geq t \|A\|_2$ with equality holding for all nonnegative t if and only if the matrix A is normal.

hrmrbb .esDtk eESff xtb

Dwelling a little more on the effect of nonnormality, we know from the analysis of §9.2 that approximating e^{At} involves more than just approximating e^{zt} on $A(A)$. Another due that eigenvalues do not "tell the whole story" in the e^{At} problem has to do with the inability of the spectral abscissa (933) to predict the size of $\|e^{At}\|_2$ as a function of time. If A is normal, then

$$\|e^{At}\|_2 = e^{\alpha(A)t}. \quad (934)$$

Thus there is uniform decay if the eigenvalues of A are in the open left half plane. But if A is non-normal, then e^{At} can grow before decay sets in. The 2-by-2 example

$$A = \begin{bmatrix} -1 & 1000 \\ 0 & -1 \end{bmatrix} \Leftrightarrow e^{At} = e^{-t} \begin{bmatrix} 1 & 1000t \\ 0 & 1 \end{bmatrix} \quad (935)$$

plainly illustrates this point in Figure 93.1.

Pseudospectra can be used to shed light on the transient growth of $\|e^{At}\|$. For example, it can be shown that for every $f > 0$

$$\sup_{t>0} \|e^{At}\|_2 \leq \frac{\alpha(f)}{f} \quad (936)$$

where $a(E)$ is the pseudospectral abscissa introduced in (7.88):

$$aE(A) = \sup_{z \in \Lambda_e(A)} \operatorname{Re}(z).$$

For the 2by2 matrix in (935), it can be shown that $\text{a.o.(A)}/.01 = 216$, a value that is consistent with the growth curve in Figure 931. See Trefethen and Embree (SAP, Chap. 15) for more pseudospectral insights into the behavior of $\|e^{At}\|_2$.

$$\tilde{F}, |,| \quad s \quad s \quad \dots \quad \alpha$$

With this discussion we are ready to begin thinking about the stability of Algorithm 9.31. A potential difficulty arises during the squaring process if A is a matrix whose exponential grows before it decays. If

$$G = \text{R}_{\text{QH}}(\mathbf{z}_i), \dots, e^{A/2^j}$$

then it can be shown that rounding errors of order

$$\gamma = \mathbf{u} \| G^2 \|_2 \cdot \| G^4 \|_2 \cdot \| G^8 \|_2 \cdots \| G^{2^{j-1}} \|_2$$

can be expected to contaminate the computed G^{2^j} . If $\|e^{At}\|_2$ has a substantial initial growth, then it may be the case that

$$\gamma \gg \|G^{2^j}\|_2 \approx \|e^A\|_2,$$

thus ruling out the possibility of small relative errors.

If A is normal, then so is the matrix G and therefore $\|Cm\|_2 = \|G\|^m$ for all positive integers m . Thus, " $\|G^2\|_2 \leq \|e^A\|_2$ " and so the initial growth problems disappear. The algorithm can essentially be guaranteed to produce small relative error when A is normal. On the other hand, it is more difficult to draw conclusions about the method when A is nonnormal because the connection between $v(A, t)$ and the initial growth phenomena is unclear. However, numerical experiments suggest that Algorithm 931f is to produce a relatively accurate e^A only when $v(A, 1)$ is correspondingly large.

Problems

P9.3.1 .616 16.1 e_{-A+B}t = eAt eBt eE (rrc.8e ,80t nAB = BA IQQHnFA==,Sky =TxAk . rSLAx-AxTAQ .8e A,C).uA AAA1AcA8t.

P9.3.2 ..d d1.816.1A Q=oAl¹¹B₁ kQa+ySLky.k.Sky.eA (8 QJOIO aX'0:(LECf(x(= = R1(A) xA SxkS)= F¹ k/AxQ1 Sk/Ax2=m²O (8 GeF3IOER9A) T-SxkSSO=)

P9.3.3 .316 16.1> A T S E TO I X P k / A Q k / A w M T = k b F Q p == a k / k A = ex. f = p =) k . g

P9.3.4 .616 13.1.>

$$\exp \left(\begin{bmatrix} -A^T & P \\ 0 & A \end{bmatrix} z \right) = \begin{bmatrix} F_{11} & F_{12} \\ 0 & F_{22} \end{bmatrix}_n^n$$

1386

$\text{HF}_2 \rightarrow \text{Z}_{\text{eA}}^{\text{T}_{\text{tp}}} \text{eAtdt}$

P9.3.5 N. 8 ...1..13. N. 1.d.1... eA 3J08A = uvT, u,v E Rn,

P9.3.6 .I.8 .. ER^{n × n} - (3^N - 3 v ∈ R^{n × N} (p₃ ((β₁ U (.

.pIMR=r G B=r r.nI. AlaAsxEp..

$$\parallel eAt \quad P_{e\mu(A)t}$$

.21M1 Jy

Notes and References for §9.3

N..3 8>63.P..... c.P6c. ...1c.f. .. .p1..cc8....3. N1.f c.13.N..16..
!.... ..Ic....or

31PN 8... f 30fed. 58. (1L) 7 qpxH tHank rk P2 2M yA TRWKA z =
:= PR. SIAM Review 20 d (d57)
A 4w1gIMaE A9.4B n.l.a ext q qpxH tH rk y2 2M PA TRWPA z =
:= PR. 18A 1 Hk H SIAM Review 45.5r i.v

sA,2ar,aE ts,Mnam,p hEi .SSu,T 2C (OH +11) =Tl QAH A 1 y
L TAI 2M A I P (OH +11) H R Qk IH xk 2 v (M
Bx R E O ei>2k Wk 2 -1 =WIR H A 2 R H D R RWQyQ = I KHH
X

r. **H**A = **l** **ll** **(Mlv)**7 **q** **zWBRDk** **y2y** **MH2** **TRV** **BQ** Numer. Math.

A44Ba.nla (M.L.) q(H (k yP2 = 1 2MTP2 x - 1 2MTPA1 (y)PA : P.R
TRM yAQ x Pade and Rational Approximation, 94w41.aE lxs4B.ur. (k)ZKT
- k = u v

SIAM J. Numer. Anal. 14, 777-794 (1977)

Inst. Math. Applic. 11, 57. P.r

34aurr (1L7) ~~QWV2~~ 2 FA TRVIA 2 = YR -- L FXT ~~KAIPIK~~ J.
 Inst. Math. Applic. 15 IT5 ITdv
~~TE226 ECGS 60~~ ~~YHADP~~ ~~N~~ ~~E~~ ~~TIN KAO S~~ ~~COT~~ (C)

n4E2A2aE 3xhS2a2aw Jq qH AMRAH2 J PA TRWKAQ Z = S2T x(Q)
 HR : Lin. Alg. Applic. 308 (2005) JI

(AQ) : Lin. Alg. Applic. 240 (NN5V
 2142-nC GT Ed. I EON-1 L WY B P N v ER TRWxQ 1 AP SIAM

2441p.C J. Matr. Anal. Applic. 26 N 1 2005 : SIAM

39.43gr hpt .aE 242r2C (av) q =u LTx =l LVA =Qy II IA =yR
TRWIAQ : SIAM J. Matr Anal. Applic. 31, PjR .dv

3 Sul. l. 9s,x2a (+1) ll yA EQ T =Wk AL

RLE 61 (III) J. Math. Phys. 40, IIJr Inv LPHO MATR. Pk I) L 2B(-) H2T (-) AQT AT
 PACNAWAK J. Math. Phys. 40, IIJr Inv LPHO MATR. Pk I) L 2B(-) H2T (-) AQT AT

aplulu1C.at .SS,2A..n,a2aAlaxulg2lut A.gg2M.21AlCSI...nlal. xp1C.xu2IT Slala.2
f xplgnal1S.2C.gMr.,MMSgtu IT .CSgtu2t 2a.3r.M.gt naxtg1faxMnT IT
u1M ls nMltv

.4f 12aa. EA4n2ngSt(L1)7a QyA II R >MayA z y -Q y1LyA
 -x.Banla(1/)QyA IEEE T ns. Autom. Contr IAC-16 JInJ.1
 -x.Banla(1/)QyA IEEE T ns. Autom.

3a,aElutx.aEndr x21S r AbS r caE 2xtlatn.2nhtp3gSr.21a .tlttnar x2S 1M,uC.aA1
1 grlunx2OMAICs xnap1C MnTT Sla²xal uCax2T2uIA 2aaAgI Elte

rE t h₁ti ((yzz)Hlicatsl latl regogsldial licatsl 2g lnd,lgcl qpiatA oda BIT 17, 5r. T1
 A luB,8 n,8 ((yzz)nl ,Fneeeisdochdil onel'lnc1 qpiieo dfMellAM J. Numer. Anal. 14,
 T(.fr
 IMI. p2c. ((yH)nl,qlwdldil onel'numed ienhdlochelil onel'lnc1 qpiiae odf.aNumer. Math
 63, INs11
 f 4 - 2) Ae avu4AU ((yye)VlieghHochesidln,lgcl qpiieicdl latl Fredhiupod odaal
 Adv. Appl. Math. 16, 5IN5T.1
 34430r 1,28E 244r2C (tity)Gliupocaii onekmed ienhdlochelil onel'lnc1 qpiieald+1
 tdonlil appwddldialil iatddial causeg qslct IdazzSIAM J. Matr. Anal. Applic. 30, (7,r
 (7.T1

3 t,e.uA SAqrAu ACSIpc8C,0,EA8,18E QMrts.Mt&cpurAT S,RA8,2,0,puc8E
 MATLAB 06E 0o0=0= 0< C3E 0, E0= EOT

j =).1 OQ(y) ,qpi edoHleidtg1 mreid 2g iiupoodail ,lgcl qpiAacdfs a CM T ns.
 Math Sof w. 24, N5r(.7v

A,8tnEAu,Lc,8h,15II,8E h,5v,t2,t L2,pl2AT S,8ASpcSDMI ApI&VATa2i3 cC
 S,ML,8lu,SAuptAT

, PIA .SE1 Att(ed,qaogwtcselTe+ lldhe regogsldial licatsl 2g npiaTao,sl idqsseld,vwl
 ciaeaeildch ,loghesa J Numer. Math. 110, 5,5rJ51
 .2 A.uEt ,ae uN L4(ltt(AhpAiaaldws il eeteeuwTondh ,logrTsl latl eilgdns id
 kgoriiial ,logres.a J. Comput. Appl. Math. 233, Id7ThdT.M

9.4 The Sign, Square Root, and Log of a Matrix

The matrix logarithm problem is the inverse of the matrix exponential problem. Not surprisingly, there is an inverse of the scaling and squaring procedure given in §9.3.1 that involves repeated matrix square roots. Thus, before we can discuss $\log(A)$ we need to understand the J problem. This in turn has connections to the matrix sign function and the polar decomposition.

,hshhb 7Sb ht f xTebTtb 3Dtf f Tkb

For all $z \in \mathbb{C}$ that are not on the imaginary axis, we define the $\text{sign}(z)$ function by

$$\text{sign}(z) = \begin{cases} -1 & \text{if } \operatorname{Re}(z) \leq 0 \\ +1 & \text{if } \operatorname{Re}(z) > 0 \end{cases}$$

The sign of a matrix has a particularly simple form. Suppose $A \in \mathbb{C}^{n \times n}$ has no pure imaginary eigenvalues and that the blocks in its JCF $A = XJX^{-1}$ are ordered so that

$$J = \begin{bmatrix} J_1 & 0 \\ 0 & J_2 \end{bmatrix}_{m_1 \times m_2}^{m_1 \times m_2}$$

where the eigenvalues of $J_1 \in \mathbb{C}^{m_1 \times m_1}$ lie in the open left half plane and the eigenvalues of $J_2 \in \mathbb{C}^{m_2 \times m_2}$ lie in the open right half plane. Noting that all the derivatives of the sign function are zero, it follows from Theorem 9.1.1 that

$$\text{sign}(A) = X \begin{bmatrix} \text{sign}(J_1) & 0 \\ 0 & \text{sign}(J_2) \end{bmatrix} X^{-1} = X \begin{bmatrix} -I_{m_1} & 0 \\ 0 & I_{m_2} \end{bmatrix} X^{-1}.$$

With the partitions

$$X = \begin{bmatrix} X_1 & X_2 \\ m_1 & m_2 \end{bmatrix} \quad X^{-H} = \begin{bmatrix} Y_1 & Y_2 \\ m_1 & m_2 \end{bmatrix},$$

we have

$$\text{sign}(A) = X_2 Y_2^H - X_1 Y_1^H$$

$$I_n = X_1 Y_1^H + X_2 Y_2^H$$

and so

$$X_2 Y_2^H = \frac{1}{2} (I_n + \text{sign}(A)).$$

Suppose apply QR-with column pivoting to this rank-m2 matrix

$$\frac{1}{2} (I_n + \text{sign}(A)) I = QR.$$

It follows that $\text{ran}(Q(:, 1:m2)) = \text{ran}(X_2)$, the invariant subspace associated with A's right half-plane eigenvalues. Thus, an approximation of $\text{sign}(A)$ yields approximate invariant subspace information.

A number of iterative methods for computing $\text{sign}(A)$ have been proposed. The fact that $\text{sign}(z)$ is a zero of $g(z) = z^2 - 1$ suggests a matrix analogue of the Newton iteration

$$z_{k+1} = z_k - \frac{g(z_k)}{g'(z_k)} = \frac{1}{2} \left(z_k + \frac{1}{z_k} \right),$$

i.e.,

$$S_0 = A$$

for $k = 0, 1, \dots$

$$S_{k+1} = (S_k + S^{-1}) / 2$$

end

(94.1)

We proceed to show that this iteration is well-defined and converges to $\text{sign}(A)$, assuming that A has no eigenvalues on the imaginary axis.

Note that if $a + bi$ is an eigenvalue of S_k , then

$$\frac{1}{2} \left(a + bi + \frac{1}{a + bi} \right) = \frac{a}{2} \left(1 + \frac{1}{a^2 + b^2} \right) + \frac{b}{2} \left(i - \frac{1}{a^2 + b^2} \right)i.$$

is an eigenvalue of S_{k+1} . Thus, if S_k is nonsingular, then S_{k+1} is nonsingular. It follows by induction that (94.1) is defined. Moreover, $\text{sign}(S_k) = \text{sign}(A)$ because an eigenvalue cannot "jump" across the imaginary axis during the iteration.

To prove that s_k converges to $s = \text{sign}(A)$, we first observe that $ss_k = s_k s$ since both matrices are rational functions of A. Using this commutativity result and the identity $S^2 = s$, it is easy to show that

$$S_{k+1} - S = SJ^{-1}(S_k - S)^2 \quad (94.2)$$

and

$$s_{k+1} + s = \frac{1}{2} SJ^{-1}(S_k - S)^2. \quad (94.3)$$

If M is a matrix and $\text{sign}(M)$ is defined, then $M + \text{sign}(M)$ is nonsingular because its eigenvalues have the form $\lambda + \text{sign}(\lambda)$ which are clearly nonzero. Thus, the matrix

$$S_k + S = S_k + \text{sign}(A) = S_k + \text{sign}(S_k)$$

is nonsingular. By manipulating equations (942) and (943) we conclude that if

$$G_k = (S_k - S)(S_k + S)^{-1}, \quad (944)$$

then $G_{k+1} = G_k^2$. It follows by induction that $G_k = \mathbf{c}\mathbf{t}$. If $A \in \mathbb{A}(A)$, then

$$\mu = \frac{A - \text{sign}(A)}{A + \text{sign}(A)}$$

is an eigenvalue of $G_0 = (A - S)(A + S)^{-1}$. Since $|\mu| < 1$ it follows from Lemma 7.32 that $G_k \rightarrow 0$ and so

$$S_k = S(I + G_k)(I - G_k)^{-1} \rightarrow S.$$

Taking norms in (942) we conclude that the rate of convergence is quadratic:

$$\|S_{k+1} - S\| \leq \frac{1}{2} \|S_k^{-1}\| \cdot \|S_k - S\|^2.$$

The overall efficiency of the method in practice is a concern since $O(n^3)$ fops per iteration are required. To address this issue several enhancements of the basic iteration (941) have been proposed. One idea is to incorporate the Newton approximation

$$S_k^{-1} \approx S_k(2I - S_k^2).$$

(See P94.1) Using this estimate instead of the actual inverse in (941) gives update step

$$S_{k+1} = \frac{1}{2}(S_k + S_k(2I - S_k^2)) = \frac{1}{2}S_k(3I - S_k^2). \quad (945)$$

This is refined to a the Newton-Schultz iteration. Another idea is to introduce a scale factor:

$$S_{k+1} = \frac{1}{2}((\mu_k S_k) + (\mu_k S_k)^{-1}). \quad (946)$$

Interesting choices for μ_k include $|\det(S_k)|^{1/n}$, $\sqrt{\rho(S_k^{-1})/\rho(S_k)}$, and $\sqrt{\|S_k\|_F \cdot \|S_k^{-1}\|_F}$ where $\rho(\cdot)$ is the spectral radius. For insights into the effective computation of the matrix sign function and related stability issues see Kenney and Laub (1991, 1992), Higham (2007), and Higham (FOM, Chap. 5).

msmib C\\$b ht f.TebnDt x\\$b9kf b

Ambiguity arises in the $f(A)$ problem if the underlying function has branches. For example, if $f(x) = \sqrt{x}$ and

$$A = \begin{bmatrix} 4 & 10 \\ 0 & 9 \end{bmatrix},$$

then

$$A = \begin{bmatrix} 2 & 2 \\ 0 & 3 \end{bmatrix}^2 = \begin{bmatrix} -2 & 10 \\ 0 & 3 \end{bmatrix}^2 = \begin{bmatrix} -2 & -2 \\ 0 & -3 \end{bmatrix}^2 = \begin{bmatrix} 2 & -10 \\ 0 & -3 \end{bmatrix}^2,$$

which shows that there are at least four legitimate choices for V . To clarify the situation we say F is the *principal square root* of A if (a) $F^2 = A$ and (b) the eigenvalues of F have positive real part. We designate this matrix by $A^{1/2}$.

Analogous to the Newton iteration for scalar square roots, $X_{k+1} = (X_k + A/X_k)/2$ we have

$$X_0 = A$$

$$\text{for } k = 0, 1, \dots$$

$$x_{k+1} = (x_k + X_k^{-1}A)/2$$

end

(947)

Notice the similarity between this iteration and the Newton sign iteration (941). Indeed, by making the substitution $X_k = A^{1/2}S_k$ in (947) we obtain the Newton sign iteration for $A^{1/2}$. Global convergence and local quadratic convergence follow from what we know about (941).

Another connection between the matrix sign problem and the matrix square root problem is revealed by applying the Newton sign iteration to the matrix

$$\tilde{A} = \begin{bmatrix} 0 & A \\ I & 0 \end{bmatrix}.$$

Designate the iterates by \tilde{S}_k . We show by induction that \tilde{S}_k has the form

$$\tilde{S}_k = \begin{bmatrix} 0 & X_k \\ Y_k & 0 \end{bmatrix}.$$

This is true for $k = 0$ by setting $X_0 = A$ and $Y_0 = I$. To see that the result holds for $k > 0$ observe that

$$\tilde{S}_{k+1} = \frac{1}{2}(\tilde{S}_k + \tilde{S}_k^{-1}) = \frac{1}{2}\left(\begin{bmatrix} 0 & X_k \\ Y_k & 0 \end{bmatrix} + \begin{bmatrix} 0 & Y_k^{-1} \\ X_k^{-1} & 0 \end{bmatrix}\right)$$

and thus

$$x_{k+1} = (x_k + y_k)/2 \quad Y_{k+1} = (Y_k + X_k^{-1})/2. \quad (948)$$

Another induction argument shows that

$$X_k = AY_k, \quad k = 0, 1, \dots, \quad (949)$$

and so

$$X_{k+1} = (X_k + AX_k^{-1})/2, \quad Y_{k+1} = (Y_k + A^{-1}Y_k^{-1})/2. \quad (9410)$$

It follows that $X_k = A^{1/2}$ and $Y_k = A^{-1/2}$ and we have established the following identity:

$$\text{sign}\left(\begin{bmatrix} 0 & A \\ I & 0 \end{bmatrix}\right) = \begin{bmatrix} 0 & A^{1/2} \\ A^{-1/2} & 0 \end{bmatrix}.$$

Equation (94.8) defines the *Denman-Beavers iteration* which turns out to have better numerical properties than (94.7). See Meini (2004), Higham (FOM Chap. 6), and Higham (2008) for an analysis of these and other matrix square root algorithms.

,mshrb 7\\$b lkPt b p\\$f kEk eTf Tkb

If $Z = a + bi$ is a nonzero complex number, then its polar representation is a factorization of the form $Z = r e^{i\theta}$ where $r = \sqrt{a^2 + b^2}$ and $e^{i\theta} = \cos(\theta) + i \sin(\theta)$ is defined by $(\cos(\theta), \sin(\theta)) = (a/r, b/r)$. The *polar decomposition* of a matrix is similar.

Theorem 9.4.1 (Polar Decomposition). If $A \in \mathbb{R}^{n \times n}$ and $n \geq n$, then there exists a matrix $U \in \mathbb{R}^{n \times n}$ with orthonormal columns and a symmetric positive semidefinite $P \in \mathbb{R}^{n \times n}$ so that $A = UP$.

Proof. Suppose $U_A^T A V_A = \Sigma_A$ is the thin SVD of A . It is easy to show that if $U = [U \ V]$ and $P = V_A \Sigma_A V_A^T$, then $A = UP$ and U and P have the required properties. D

We refer to U as the *orthogonal polar factor* and P as the *symmetric polar factor*. Note that $P = (A^T A)^{1/2}$ and if $\text{rank}(A) = n$, then $U = A(A^T A)^{-1/2}$. An important application of the polar decomposition is the orthogonal Procrustes problem (see §6.4.1).

Various iterative methods for computing the orthogonal polar factor have been proposed. A quadratically convergent Newton iteration for the square nonsingular case proceeds by repeatedly averaging the current iterate with the inverse of its transpose.

$$\begin{aligned} X_0 &= A && (\text{Assume } A \in \mathbb{R}^{n \times n} \text{ is nonsingular}) \\ \text{for } k &= 0, 1, \dots && \\ x_{k+1} &= (x_k + x_k^T)^{-1} / 2 \\ \text{end} \end{aligned} \tag{94.11}$$

To show that this iteration is well defined we assume that for some k the matrix X_k is nonsingular and that $X_k = U_k P_k$ is its polar decomposition. It follows that

$$X_{k+1} = \frac{1}{2} (X_k + X_k^{-T}) = \frac{1}{2} (U_k P_k + U_k P_k^{-1}) = U_k \left(\frac{P_k + P_k^{-1}}{2} \right). \tag{94.12}$$

Since the average of a positive definite matrix and its inverse is also positive definite it follows that X_{k+1} is nonsingular. This shows by induction that (94.11) is well-defined and that the P_k satisfy

$$P_{k+1} = (P_k + P_k^{-1}) / 2, \quad P_0 = P.$$

This is precisely the Newton sign iteration (9.4.1) with starting matrix $P_0 = P$. Since

$$\| X_k - U \|_2 = \| U(P_k - I) \|_2 = \| P_k - I \|_2$$

and $P_k = \text{sign}(P)$, by quadratic convergence we conclude that X_k matrices in (9.4.11) converge to U quadratically.

Extensions to the rectangular case and various ways to accelerate (9.4.11) are discussed in Higham (1986), Higham and Schreiber (1990), Gander (1990), and Kenney and Laub (1992). In this regard the matrix sign function is (once again) a handy tool for deriving algorithms. Note that if $A = U_A \Sigma_A V_A^T$ is the SVD of $A \in \mathbb{J}_{m,n}$ and

$$Q = \frac{1}{\sqrt{2}} \begin{bmatrix} U_A & 0 \\ 0 & V_A \end{bmatrix} \begin{bmatrix} I_n & I_n \\ I_n & -I_n \end{bmatrix}$$

then Q is orthogonal and

$$Q^T \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} Q = \begin{bmatrix} \Sigma_A & 0 \\ 0 & -\Sigma_A \end{bmatrix}.$$

It follows that

$$\text{sign} \left(\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \right) = Q \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix} Q^T = \begin{bmatrix} 0 & U \\ U^T & 0 \end{bmatrix}$$

where $U = U_A V_A^T$ is the orthogonal polar factor of A .

There is a well-developed perturbation theory for the polar decomposition. A sample result for square nonsingular matrices due to Li and Sun (2003) says that the orthogonal polar factors U and V for nonsingular $A, A \in \mathbb{J}_{m,n}$ satisfy the bound

$$\| U - \tilde{U} \|_F \leq \frac{4 \| A - \tilde{A} \|_F}{\sigma_{n-1}(A) + \sigma_n(A) + \sigma_{n-1}(\tilde{A}) + \sigma_n(\tilde{A})}.$$

hsgsb 78 ht f.Tellk' tTf e.b

Given $A \in \mathbb{J}_{m,n}$, a solution to the matrix equation $e^X = A$ is a logarithm of A . Note that if $X = \log(A)$, then $X + 2k\pi i$ is also a logarithm. To remove this ambiguity we define the *principal logarithm* as follows. If the real eigenvalues of $A \in \mathbb{J}_{m,n}$ are all positive then there is a unique real matrix X that satisfies $e^X = A$ with the property that its eigenvalues satisfy $\{X\} \subset \{z \in \mathbb{C} : -\pi < \operatorname{Im}(z) \leq \pi\}$.

Of course, the eigenvalue-based methods of §9.2 are applicable for the $\log(A)$ problem. We discuss an approximation method that is analogous to Algorithm 9.3.1, the scaling and squaring method for the matrix exponential.

As with the exponential, there are a number of different series expansions for the log function that are of computational interest. The simplest is the Maclaurin expansion:

$$\log(A) - M(A) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{(A - J)^k}{k}.$$

To apply this formula we must have $\rho(A - I) < 1$ where $\rho(\cdot)$ is the spectral radius.
The Gregory series expansion for $\log(x)$ yields a rational approximation:

$$\log(A) - G_q(A) = -2 \sum_{k=0}^q \frac{1}{2k+1} ((I - A)(I + A)^{-1})^{2k+1}.$$

For this to converge, the real parts of A's eigenvalues must be positive.

Diagonal Padé approximants are also of interest. For example, the (3,3) Padé approximant is given by

$$\log(A) - r_3(A) = D(A)^{-1} N(A)$$

where

$$D(A) = 6I + 9(A - I) + 36(A - I)^2 + 3(A - I)^3,$$

$$N(A) = 6I + 6(A - I)^2 + 11(A - I)^3.$$

For an approximation of this type to be effective, the matrix A must be sufficiently close to the identity matrix. Repeated square roots are one way to achieve this.

```

k=0
Ao=A
while ||A - I|| > tol
    k=k+1
    Ak=A!^(1/2)
end

```

The Derman-Beavers iteration (948) can be invoked to compute the matrix square roots. If we next compute $F = \log(A_k)$ by using (say) an appropriately chosen Padé approximant, then $\log(A) = 2^k \log(A_k) / 2^k F$. This solution framework is referred to as *inverse scaling and squaring*. There are many details associated with the proper implementation of this procedure and we refer the reader to Cheng, Higham, Kenney, and Laub (2001), Higham (2001), and Higham (FOM, Chap. 11).

Problems

- P9.4.1 f3.. f18..38c86.1..8..1..11 ..638..8f .1 ..f. .11. 1>38 ..1..1.
 $R =)/r - iAA = S10$ TOAE=Ad(AA!LrS) TrAErISB =S10ln'AbnEQxA =.rISQ = 1..A
- P9.4.2 .316 .3..> ..0R.1P .386.66 R = =T)6rJ'
- P9.4.3 .316 .3... .R = A(A^2)^{-1/2}
- P9.4.4 e8..>Pr2..c1. R.1P1.
- P9.4.5 re.38886..6 P . 8.8..16 R.1P 6f 68 ..6f f8 8.1.8..8 N.
- P9.4.6 .316 .3..>68.... .38c86.1..r2..8..11..8..1.. R. P.1 ..8.... .1... 88 ..c28
 $...b A, n/A) A_k^{-1} J Q-S=TrTA A!OEA=I_k$
- P9.4.71 .8 A Q!Sxb = 4A=rAr/AIS=EqrSE=S= (A - AT)/2, f = A + AT)/2
- P9.4.8 .316 .3.. 38.18.. f8.1..1..1.. 1> 16..6....b .. 26.r.81P.6or. A= U1P1) F
 $A = U2P2.FArIS S=E,AaSbS=TrISQ=PAOUU1 = P2P1^{-1}.0 U U2= P1P2^{-1}/A n/A=cA$
 $ATAQ==A=G$

P9.4.9 N..8. ..8.8. N. 8p..8...86N. P6888. .8.8 .8..P.8. $A = UP$ l. . MA2the 2 en 7l.Re
gp St 24n7 AMpS7.MpaU mMlx.xcl8X

P9.4.10 N..8. .8.8. N. 8p..8n88g 6 8.8 . . 01.10 .8P.P.8.8.b

P9.4.11 8. 18.P86 < n iAMTl&2&l. eAAI,SITcxcl8IA E E x^n .

P9.4.12 18PA St.8 +.St.+tt,AxMcSltcxjjeA=acxAxMcIT , sl. x2x xAMA,aATA ctxt
tt,AxMcSltcxniA=anXAcaaRa.R = X² 1 +/(L/R/RQXO = I .ae

$$X_{k+1} = (X_k - AX_k^{-1})/2$$

74tp X_k g 2rnSl7.An,h 2(tlt g StpM7t74t en7l.r X .p Jnl7 rnR

P9.4.13 . 8P6.P

$$X(t) = C1a=1 + L_2 \sqrt{A^{-1}} t c8,n T$$

tl,iAt xAcacxi0A SMISOK,(t) = - AX(t), $X(0) = C_1$, $\dot{X}(0) = C_2$ l===- A RaRttt,AxMcA
SltcxciAQ8cxAv

Notes and References for §9.4

P.8.P .6. P .8.P.8. . .8.8.8. . .8.P8..8.P .. P..6. foM88...8or

C P.. . (2005). 4| p 7mMp6n7l.Ata SL.p Handbook of Linear Algebra , nZlrSAa , AdFu
A2S,,8 .8e .. OOnA.lxlau unu§11-1-§11-13.

NnJteal7(n7 SmaAara N4t2rm7Mn7l.r am)5r5A7.Mpn5S .7a nJJ1.An7.M55A1NST

o9 fhtla (1987). 42M,Im5) 7d7.enmA omAAAn72rn7mM@m7(7(fn7emR2.)p cNp47.MpSLinear
Alg. Applic. 85 267-279.

.e29 -ppth npSe9,9 Inr. (1991). 4on7.M5nD7ln7m4t7iMSa,l 7(t 4n7lmr2.) p |pA 7mpMS SIAM
J. Matrix Anal. Appl. 12, 273-291.

.rg29-tp9thS ee,e Inr.S npS N4e NnJnSMJM(1992). 44n7l.R 2.)5 e,)Ml.7(ea ,e o.AAn7.x2N
7mMp6n7A J. Math. Contr l Info. 9, 331-344.

.e29 -ppth n5S ee,e Inr. (1992). 4lp 2An,p,)t27Mp m@t74MS ,e NM,nltAMeJMa.7mMpS 7it
4n7l.r 2.)p |5A7.M5 SIAM J. Matrix Anal. Applic. 13, 688-706.

o9 fhtla S 9 2pS 59 4t4ecnpp (1997). 4.4t 4n 7lmRm)p|5A7.Mp 4t7(Mri n5S 74t .MeJN7n7.M5
M.DpIsln572r.aJnAtaSIAM J. Matr x Anal. Applic. 18, 615-632.

ke fmpS,3.3 teet, (1998). 4gamp7it 4n 7emRm)5|pA7mM57MMeJr7t Dp .n572N.aJnAtaSL
SIAM J. Matr x Anal. Applic. 19, 2205-2225.

,e,e 2.)ine (1994). 4.it 4n7lmr2.)p ttAMeJMat7.M5S D7p7.Mp 7M4t NM,nltAMeJMa.7.MpL
Lin. Alg. Applic. 212/213, 3-20.

,e,e 2)ne Ste2e 4nA/thS ,l 4nA/thS npSce .maatrl (2004). 4.MeJN7.p) 7(t NM,nltAMeJMa.7.Mp
npS74t 4n7l.r 2m)5ttAMeJMat7.M54n7l.r ClMNJ SIAM J. Matr x Anal. Applic. 25, 1178-
1192.

5n .MN@JtA7a MT(t n7l.m2rn tMM7lM.te nlt S.aAraatS.pk

xet9 ttpen5 npS e9,8tnItla (1976). 4.(S 4n7l.r 2m)p|pA7.Mp npS.MeJN7n7.M5a5 2ha7teaSL
Appl. Math. Comput., 2, 63-94.

I f8/eA/n5S 292neen ,p) (1983). 4e 2A4Nl4t7(MS ,e 74t 22rnet oMM7l.n 4n7 r8ELin. Alg.
Applic. 52/53, 127-140.

,9,, 2.)ne (1986). 4,t2 7M5a4t7(MS ,l 7(S 4n7l.R 22Nn dMM7lMath. Comput. 46 537-550.

,e,e 2.)ne (1987). 4.MeJN7.p) otn, 22rnlt oMM7M.n otn, 4n 7l.rSLLin. Alg. Applic. 88/89
405-430.

,9,e 2.)ne (1997). 427n,t D7tln7mMpa7it 4n7l.R 22Nnlt oMM7INumer. Algorithms 15, 227-242.
7-7- Ir (1998). 4e NnS=eJlMr.en7mM4t74MS ,l 22rnet oMM7Mheet7e.A NMam7.ttrnp.7t
4n7l.Ata SIAM J. Matr x Anal. Applic. 19, 833-845.

,9,e 2m)(neS t42A/thS ,9 4nA/thS n5S c9 ..aatrl (2005). 4|pA7.Mpa NetateI.p) 4n7l.r ClMNJa
n5S D7tln7.Mpb 74t 4n7l.R 22Nn dMM7lSIAM J. Matr x Anal. Applic. 26, 849-877.

.9 re2CrMnpS ,e ,n 2m)4n(2006). 4e 2A4rolS,t27Mp 4t74MS ,e 7it 4n7emr 7ioMM7 n5S m7a
D5ItlatSIAM J. Matrix Anal. Applic. 28, 788-804.

fe 4tp. (2004). 4.4t 4n7er 22rnlt oMM7Men,t2 |5A7.Mpn1 NteaJtA7.Itkitit7.An, otaN17a
n5S .MeJr7n7.M5n, DaanSIAM J. Matrix Anal. Applic. 26, 362-376.

06men at r6to(Ae2) -2009.. 45tll.tS .Mc.roiAi63p E I.raoNMi&EA1Ail6RBLSIAM J. Matrix Anal. Applic. 31, 1279–1302.

- .Mc.roCAC6MFA0t.CaMEut JM0A\$t.Mc.Ma6CIM\$pS 6CplAal3ACIM\$pId.MltS .pS ..p 26).A8 -1986.. 4.Mc.roCl,) i.t 2M0A1l3c.3.li6Mp .6i. e.J.a6.c C.M,aBSIAM J. Sci. Statist. Comp. 7, 1160–1174.
- o.2. 2.ltl.tl ApS fc., NAl0tii -1988.. 4f0M.y (t.CMlaS..uMlh ApS .McJroCAi6MBLSIAM J. Numer. Anal. 25, 189–205.
- ... 2l).Ac ApSoci. I.lld.tl -1990.. 4xAaiNM1At.Mc.Mali6Mp ME a\$ilAlh 1Ail 6rBLSIAM J. Sci. Statist. Comput. 11, 648–655.
- ,cpc 2.).A c ApSN.NAJ&cl illMro1994.. 4e NAlla0tæ1)Ml6Cuc5l .McJroi6p) i.1 NM0At.Mc3 J3a6iMpbBar lIel Comput. 20, 1161–1173.
- e.e. tro lro0b999.. ep.i.croc DCtlAi6Mp i.t 1Ail.R ,M1Alt.M8 .Ma6iMpbETNA 8, 21–25.
- e. kAppA ApS k. 4ropi.tre-Aw -2002.. 4CtplA0.3tS NM1At.McJMa6CIM\$pId i.t JlMRl8Ai6Mp MED 1Ail6R xRJMptp@OBLSIAM J. Matrix Anal. Applic. 23, 840–862.
- f. ftw3ylt.l9 ApS-. kltiAre -2006.. 4e..d 3r.9Ai6MpMEqid..ta ApSAxAc61h M€ApStl 1di.MSa 5l NM1qt.Mc.M a.CM@BIT 46, 345–366.
- s1 fhtla ApS2. 6ro 2008.. 4e .t. 2.A16p) 5d ,t.i Mp1auitlAiMp5l iu1 NM1Alt.Mc.Mali.Mp ApS uiaF.y.AIS liA61tibLSIAM J. Matrix Anal. Applic. 30, 822–843.
- ... 2.).Ac B. 4du1B ApS.6aattrol -2010.. 4.ut AFMpl.A1CtplA0lts N31Alt.Mc.M nli6MBL SIAM J. Matrix Anal. Applic. 31, 2163–2180.

xMdApA,A0ha6aaiM.ti.td MlpMü.d .30AdSt.Mc.Ma6i6M,A, .t .Mc.roitS lp A.p.it proc.td ME aitJaBtdS

e. CtMd)t ApS-.. uylA8MI-1996.. 4u .ut NM0#lMcJMa6i6Mp xlplCu0l88JroiA.0tg BLSIAM J. Matrix Anal. Applic. 17, 348–354.

e. CtMl)t ApS-.. uylAcMI-1997.. 4eSStpSro9SDa1 NM1Alt.Mc.Mnli6Mp x6p.Gh .McJroiA.0t,BL SIAM J. Matrix Anal. Appl. 18, 264–264.

.utdt 6aA.Mpa.SulA0t 06itdAi6Mp dlFtS .6uu .M. Cut.MaAd .i.Mln.Ap)t ropStl.dlirod.Ai6MpS

o. 4ACu6Aa1993.. 42tlriol.Ai63p fMropS6l i.t NM0A1l39.Mal i6MBLSIAM J. Matrix Anal. Applic. 14, 588–597.

o.e.. ft.-1997.. 4otaA 6Talirod.AilMpMropS6l i tp 6iAdhN3aAdA.iMB@BIT 37, 67–75.

x ..Ai6p ..Ai da6FB ClAiMp-2000.. 4lp i.d .MpSli6Mp proc .cla eaam.6AidS.6Cu Cut NMbAl xACMl63Ai6MpA1Allr Blnumer. Lin. Alg. 7, 337–354.

.. ftl ApS.- Irop 2003.. 4.t. NtdCrol.Ai.MfMroFS5d ,p liAlh N30AdACM@BL SIAM J. Matr x Anal. Applic. 25, 362–372.

x6pA00kBaAlbaMp.dlplp) i.t cACllr 0M)Al6CuApS 6Ma.roiAi6MpAh .t 5ropS6FS

f=.. 2t0iMp-1968.. 4ftM)Al6i)c, MEAid..un Rn Pr c. AMS 19, 733–736.

ft. tl -1996.. 4.Mpa6StlAi.MpId pMc.roi6p) otA1 ftM)Alli.ca MEAill.ta BAc61iMp6cftM)Al6CucaB ApS Iyd.e2hc9tCll. ftM)BcucBLLin. Alg. Applic. 244, 35–54.

fi(t6t B 1M6plBapS e. JlA-1996.. 4.Mc.roiAC.3,A1.t. p1rota 5l otA0 ftM)Al6i.ca M#ACll ta BLSIAM J. Matrix Anal. Applic. 17, 570–593.

.. I) -tp,th ApSe. ,.ftAro.-1998.. 4e I.urode2 .ti e0)MlC.c 5l .Mc.roi6p) i.t IM)Alliuc ApC xR.Mplpi6A0M@4ACl.rBLSIAM J. Matr x Anal. Applic. 19, 640–663.

ft. t6t -1998.. 4odqa 2A9.0iMp6Apf3)Ad64c M# Iu8J1i.6. 1Aid6RBLLin. Alg. Applic. 281, 227–246.

ft.tlt. ApSe. NAJ6p!2000.. 4.MpSli6Mp6p) pSNAS-eJ.lMRlcAi6MpMEutftM)ql.C.c M#4ACllRBL SIAM J. Matrix Anal. Applic. 21, 913–930.

... 2l).Ac -2001.. 4xIA0roAi6p)NAS-e.JlMRc,q,ia M€udlqid6R ftM)Al.cBL SIAM J. Matr Anal. Applic. 22, 1126–1135.

2.2. .utp)B ,p. 2l).Ac B.I. -tpptbh ApSe.p. ftAro.-2001.. 4e.lMr6cAClp) i.t IM)Al.Cuc of a Ha =AOfoNJDyynno AasineN. Matrix Anal. Applic. 22, 1112–1125.

Chapter 10

Large Sparse Eigenvalue Problems

edP		
edPI	G	G
edPn		
edPo		
edPr		
edPs	M	

The Lanczos process computes a sequence of partial tridiagonalizations that are orthogonally related to a given symmetric matrix A . It is of particular interest if A is large and sparse because, instead of updating A along the way as in the Householder method of §8.2, it simply relies on matrix-vector products. Equally important, information about A 's extremal eigenvalues tends to emerge fairly early during the iteration, making the method very useful in situations where just a few of A 's largest or smallest eigenvalues are desired, together with the corresponding eigenvectors.

The derivation and exact arithmetic attributes of the method are presented in §10.1, including its extraordinary convergence properties. Central to the discussion is the connection to an underlying Krylov subspace that is defined by the starting vector. In §10.2 we point out connections between Gauss quadrature and the Lanczos process that can be used to estimate expressions of the form $u^T f(A) u$ where $f(A)$ is a function of a large, sparse symmetric positive definite matrix A . Unfortunately, a "math book" implementation of the Lanczos method is practically useless because of roundoff error. This makes it necessary to enlist the help of various "workarounds," which we describe in §10.3. A sparse SVD framework based on Golub-Kahan bidiagonalization is detailed in §10.4. We also introduce the idea of a randomized SVD. The last two sections deal with the more difficult unsymmetric problem. The Arnoldi iteration is a Krylov subspace iteration like Lanczos. To make it effective, it is necessary to extract valuable "restart information" from the Hessenberg matrix sequence that it produces. This is discussed in §10.5 together with a brief presentation of the unsymmetric Lanczos

framework. In the last section we derive the Jacobi-Davidson method, which combines Newton's idea with Rayleigh-Ritz refinement.

Reading Notes

Familiarity with Chapters 5, 7, and 8 is recommended. Within this chapter there are the following dependencies:

$$\begin{array}{cccc} \S 10.1 & \S 10.3 & \S 10.5 & \S 10.6 \\ \S 10.2^c & \S 10.4^c & & \end{array}$$

General references for this chapter include Parlett (SEP), Stewart (MAE), Watkins (MEP), Chatelin (EOM), Cullum and Willoughby (LALSE), Meurant (LCG), Saad (NMLE), Kressner (NMSE), and EIG_TEMPLATES.

10.1 The Symmetric Lanczos Process

Suppose $A \in \mathbb{R}^{n \times n}$ is large, sparse, and symmetric and assume that a few of its largest and/or smallest eigenvalues are desired. Eigenvalues at either end of the spectrum are referred to as *extremal eigenvalues*. This problem can be addressed by a method attributed to Lanczos (1950). The method generates a sequence of tridiagonal matrices $\{T_k\}$ with the property that the extremal eigenvalues of $T_k \in \mathbb{R}^{k \times k}$ are progressively better estimates of A's extremal eigenvalues. In this section, we derive the technique and investigate some of its exact arithmetic properties.

One way to motivate the Lanczos idea is to be reminded about the shortcomings of the power method that we discussed in §8.2.1. Recall that the power method can be used to find the dominant eigenvalue, 1 and an associated eigenvector x_1 . However, the rate of convergence is dictated by $\|1 - 2/\lambda_2\|_k^2$ where λ_2 is the second largest eigenvalue in absolute value. Unless there is a sufficient magnitude gap between these two eigenvalues, the power method is very slow. Moreover, it does not take advantage of "prior experience". After k steps with initial vector v^0 , it has visited the directions defined by the vectors $Av^0, \dots, A^k v^0$. However, instead of searching the span of these vectors for an optimal estimate of x_1 , it settles for $A^k v^0$. The method of orthogonal iteration with Ritz acceleration (§8.3.7) addresses some of these concerns, but it too has a certain disregard for prior iterates. What we need is a method that "learns from experience" and takes advantage of all previously computed matrix-vector products. The Lanczos method fits the bill.

hymhmhb. KPLf b1D:e3f§ eb

The derivation of the Lanczos process can proceed in several ways. So that its remarkable convergence properties do not come as a complete surprise, we motivate the method by considering the optimization of the Rayleigh quotient

$$r(x) = \frac{x^T A x}{x^T x}, \quad x \neq 0$$

Recall from Theorem 8.12 that the maximum and minimum values of $r(x)$ are $\lambda_1(A)$ and $\lambda_n(A)$, respectively. Suppose $\{q_i\} \subset \mathbb{R}^n$ is a sequence of orthonormal vectors and define the scalars M_k and m_k by

$$M_k = \lambda_1(Q_k^T A Q_k) = \max_{y \neq 0} \frac{y^T (Q_k^T A Q_k) y}{y^T y} = \max_{\|y\|_2=1} r(Q_k y) \leq \alpha(A),$$

$$m_k = \alpha_k(Q_k^T A Q_k) = \min_{y \neq 0} \frac{y^T (Q_k^T A Q_k) y}{y^T y} = \min_{\|y\|_2=1} r(Q_k y) \geq \alpha_k(A),$$

where $Q_k = [q_1 \ I \ \dots \ I_{d_k}]$. Since

$$\text{ran}(Q_1) \subset \text{ran}(Q_2) \subset \dots \subset \text{ran}(Q_n) = \mathbb{R}^n$$

it follows that

$$\begin{aligned} M_1 &\leq M_2 \leq \dots \leq M_n = \alpha(A), \\ m_1 &= m_2 = \dots = m_n = \alpha_n(A). \end{aligned}$$

Thus, the proposed optimization framework will ultimately converge. However, the challenge is to choose the q-vectors in such a way that M_k and m_k are high-quality estimates well before k equals n .

Searching for a good q_k prompts consideration of the gradient:

$$Vr(x) = \frac{2}{x^T x} (Ax - r(x)x). \quad (10.11)$$

Suppose $U_k \in \text{span}\{q_1, \dots, q_k\}$ satisfies $M_k = r(U_k)$. If $Vr(U_k) = 0$ then $(r(U_k), U_k)$ is an eigenpair of A . If not, then from the standpoint of making M_{k+1} a large as possible it makes sense to choose the next trial vector q_{k+1} so that

$$\nabla r(u_k) \in \text{span}\{q_1, \dots, q_{k+1}\}. \quad (10.12)$$

This is because $r(x)$ increases most rapidly in the direction of the gradient $Vr(x)$. The strategy will guarantee that M_{k+1} is greater than M_k hopefully by a significant amount. Likewise, if $V_k \in \text{span}\{q_1, \dots, q_k\}$ satisfies $r(V_k) = m_k$, then it makes sense to require

$$Vr(v_k) \in \text{span}\{q_1, \dots, q_{k+1}\} \quad (10.13)$$

since $r(x)$ decreases most rapidly in the direction of $-Vr(x)$.

Note that for any $x \in \mathbb{R}^n$ we have

$$Vr(x) \in \text{span}\{x, Ax\}.$$

Since the vectors U_k and V_k each belong to $\text{span}\{q_1, \dots, q_k\}$, it follows that the inclusions (10.12) and (10.13) are satisfied if

$$\text{span}\{q_1, \dots, q_k\} = \text{span}\{q_1, Aq_1, \dots, A^{k-1}q_1\}.$$

This suggests we choose q_{k+1} so that

$$\text{span}\{q_1, \dots, q_{k+1}\} = \text{span}\{q_1, Aq_1, \dots, A^{k-1}q_1, A^k q_1\}$$

and thus we are led to the problem of computing orthonormal bases for the *Krylov subspaces*

$$(A, q, k) = \text{span}\{q, Aq, \dots, A^{-1}q\}.$$

These are just the range spaces of the Krylov matrices

$$K(A, q_1, k) = [q_1 | Aq_1 | A^2q_1 | \dots | A^{k-1}q_1]$$

that we introduced in §8.3.2. Note that $K(A, q, k)$ is precisely the subspace that the power method "overlooks" since it merely searches in the direction of $A^{k-1}q$.

hyhhgib 7 TtTt"kttP TtfT Tkb

In order to generate an orthonormal basis for a Krylov subspace we exploit the connection between the tridiagonalization of A and the QR factorization of $K(A, q, n)$. Recall from §8.3.2 that if $Q^T A Q = T$ is tridiagonal and $Q^T Q = I_n$, then

$$K(A, q, n) = Q^T K(A, q, n) = Q[e_1 | Te_1 | T^2 e_1 | \dots | T^{n-1} e_1]$$

is the QR factorization of $K(A, q, n)$ where e_1 and q are respectively the first columns of I_n and Q . Thus the columns of Q can effectively be generated by tridiagonalizing A with an orthogonal matrix whose first column is q .

Householder tridiagonalization, discussed in §8.3.1, can be adapted for this purpose. However, this approach is impractical if A is large and sparse because Householder similarity updates almost always destroy sparsity. As a result, unacceptably large, dense matrices arise during the reduction. This suggests that we try to compute the elements of the tridiagonal matrix $T = Q^T A Q$ directly. Toward that end, designate the columns of Q by

$$Q = [q_1 | \dots | q_n]$$

and the components of T by

$$T = \begin{bmatrix} \alpha_1 & \beta_1 & & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ \ddots & \ddots & \ddots & & \\ \vdots & \ddots & \ddots & & \beta_{n-1} \\ 0 & \cdots & \beta_{n-1} & \alpha_n \end{bmatrix}.$$

Equating columns in $AQ = QT$, we conclude that

$$Aq_k = \beta_{k-1}q_{k-1} + \alpha_k q_k + \beta_k q_{k+1}, \quad (\beta_0 \neq 0),$$

for $k = 1:n-1$. The orthonormality of the q -vectors implies

$$\alpha_k = q^T A q.$$

(Another way to see this is that $T_{ij} = q^T A q_j$.) Moreover, if we define the vector r_k by

$$r_k = (A - \alpha_k I)q_k - \beta_{k-1}q_{k-1}$$

and if it is nonzero then

$$q_{k+1} = r_k / \beta_k$$

where

$$\beta_k = \pm \| r_k \|_2.$$

If $r_k = 0$ then the iteration breaks down but (as we shall see) not without the acquisition of valuable invariant subspace information.

By properly sequencing the above formulas and assuming that $q \in \mathbb{R}^n$ is a given unit vector, we obtain what may be regarded as "Version 0" of the Lanczos iteration.

1b) $=IUm)$ srj sxsattj $\{y=t\}$ lz - 5j) =q by U.=qn) Given a symmetric matrix $A \in J^{n \times n}$ and a unit 2nom vector $q_1 \in I^n$, the following algorithm computes a matrix $Q_k = [q_1 | \dots | q_k]$ with orthonormal columns and a tridiagonal matrix $T_k \in I^{K \times K}$ so that $AQ_k = Q_k T_k$. The diagonal and superdiagonal entries of T_k are a_1, \dots, a_k and f_1, \dots, f_k respectively. The integer k satisfies $1 \leq k \leq n$.

$k = Q / 0 = 1, \varphi = 0 \text{ ro } q_1$

while k= 0 **or** f_k= 0

$$d_k I = r_k/f_k$$

$k = k + 1$

$$ak = \frac{1}{k} A \frac{1}{k}$$

$$rk = (A - \alpha k) \bar{c} - \beta k \bar{1} \bar{c} \bar{1}$$

$$f_k = \prod_{j=1}^k P_j$$

end

There is no loss of generality in choosing f_k to be positive. The \mathbf{q}_k vectors are called Lanczos vectors. It is important to mention that there are better ways numerically to organize the computation of the Lanczos vectors than Algorithm 10.1.1. See §10.3.1.

hyHhmrb 8 eT ttf Tkb tttb Aekb ukDtteb

The Lanczos iteration halts before complete tridiagonalization if q_1 is contained in a proper invariant subspace. This is one of several mathematical properties of the method that we summarize in the following theorem.

Theorem 10.1.1. The Lanczos iteration (Algorithm 10.1.1) runs until $k = m$, where

$$m = \text{rank}(A, q1n).$$

Moreover, for $k = 1:m$ we have

$$AQ_k = Q_k T_k + r_k e_k^T \quad (10.14)$$

where $Q_k = [q_1 | \dots | q_k]$ has orthonormal columns such that $\text{span}(A, q_1, k) = E_k$,

and

$$T_k = \begin{bmatrix} \alpha_1 & \beta_1 & & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ \ddots & \ddots & \ddots & & \\ \vdots & \ddots & \ddots & & \beta_{k-1} \\ 0 & \cdots & \beta_{k-1} & \alpha_k \end{bmatrix}. \quad \{10.15\}$$

Pr b The proof is by induction on k . It clearly holds if $k=1$. Suppose for some $k>1$ that the iteration has produced $Q_k = [q_1 | q_2 | \dots | q_k]$ with orthonormal columns such that

$$\text{ran}(Q_k) = \mathcal{K}(A, q_1, k).$$

It is easy to see from Algorithm 10.1.1 that equation {10.14} holds and so

$$Q_k^T A Q_k = T_k + Q_k^T r_k e_k^T. \quad \{10.16\}$$

Suppose i and j are integers that satisfy $1 \leq i \leq j \leq k$. From the equation

$$q_j^T A q_i = q_j^T (\beta_{i-1} q_{i-1} + \alpha_i q_i + \beta_i q_{i+1}) = \beta_{i-1} q_j^T q_{i-1} + \alpha_i q_j^T q_i + \beta_i q_j^T q_{i+1}$$

and the induction assumption $Q_k^T A Q_k = I_k$ we see that

$$q_i^T A q_j = q_j^T A q_i = \begin{cases} 0, & \text{if } i > j - 1, \\ \beta_{j-1}, & \text{if } i = j - 1, \\ \alpha_j, & \text{if } i = j. \end{cases}$$

If follows that $Q_k^T A Q_k = T_k$ and from {10.16} we have $\text{rank}(Q_k) = 0$.

If $\text{rank}(Q_k) = 0$ then $q_{k+1} = r_k / \|r_k\|$ is orthogonal to q_1, \dots, q_k . It follows that $q_{k+1} \in \mathcal{K}(A, q_1, k)$ and

$$q_{k+1} \in \text{span}\{A q_k, q_k, q_{k-1}\} \subseteq \mathcal{K}(A, q_1, k+1).$$

Thus, $Q_{k+1}^T Q_{k+1} = I_{k+1}$ and

$$\text{ran}(Q_{k+1}) = \mathcal{K}(A, q_1, k+1).$$

On the other hand, if $\text{rank}(Q_k) = 0$ then $A Q_k \subseteq \text{span}\{Q_k\} \cap I_k$. This says that $\text{ran}(Q_k) = \mathcal{K}(A, q_1, k)$ is invariant for A and so $\text{rank}(Q_{k+1}) = \text{rank}(\mathcal{K}(A, q_1, n))$. \square

To encounter a zero f_k in the Lanczos iteration is a welcome event in that it signs the computation of an exact invariant subspace. However, valuable approximate invariant subspace information tends to emerge long before the occurrence of a small f_k . Apparently, more information can be extracted from the tridiagonal matrix T_k and the Krylov subspace spanned by the columns of Q_k .

$1 \leq i \leq k$, $\|Q_k\|_F = \sqrt{\lambda_1 + \dots + \lambda_k}$

Recall from §8.1.4 that if S is a subspace of \mathbb{R}^n , then with respect to S we say that (θ_i, y_i) is a Ritz pair if $r \in \mathbb{R}^{J \times n}$ if $w \in S$. If $S = K(A, \cdot, j, k)$, then the Lanczos process can be used to compute the associated Ritz values and vectors. Suppose

$$S \approx T_k = e_k \cdot \text{diag}(Q_1, \dots, Q_k) \quad (10.17)$$

is a Schur decomposition of the tridiagonal matrix T_k . If

$$Y_k = [y_1 | \dots | y_k] = Q_k S_k \in \mathbb{R}^{n \times k},$$

then for $i = 1:k$ it follows that $(Q_i Y_i)$ is a Ritz pair because

$$Q_k^T (AY_k - Y_k \Theta_k) = (Q_k^T A Q_k) S_k - Q_k^T (Q_k S_k) \Theta_k = T_k S_k - S_k \Theta_k = 0.$$

Two theorems in §8.1 concern Ritz approximation and are of interest to us in the Lanczos setting. Theorem 8.1.14 tells us that the problem of minimizing $\|A Q_k - Q_k B\|_2$ over all k -by- k matrices B is solved by setting B at $T_k \cup Q_k^T A Q_k$. Thus, the θ_i are the eigenvalues of a "best possible matrix" that happens to be tridiagonal. Theorem 8.1.15 can be used to provide a bound for $\|AY_i - Q_i Y_i\|_2$. However, we can actually do better. Using (10.16) we have

$$AY_i - Q_i Y_i = (A Q_k - Q_k T_k) S_{k+1} \cup r_k e_k^T S_{k+1}$$

from which it follows that

$$\|AY_i - \theta_i y_i\|_2 = |\beta_k| |s_{ki}|. \quad (10.18)$$

Note that since S_k is orthogonal, $\|s_{ki}\|_2 = 1$.

We can use (10.18) to obtain a computable error bound. If E is the rank-1 matrix

$$E = -s_{ki} \cdot r_k y_i^T,$$

then

$$(A + E) y_i = Q_i Y_i.$$

It follows from Corollary 8.1.6 that

$$\min_{\mu \in \lambda(A)} |\theta_i - \mu| \leq |\beta_k| |s_{ki}|$$

for $i = 1:k$.

Golub (1974) describes the construction of a more informative rank-1 perturbation E . Use Lanczos tridiagonalization to compute $A Q_k - Q_k T_k + r_k e_k^T$ and then set $E = r_k w w^T$, where w is ± 1 and $w \cdot a_k \neq 0$. It follows that

$$(A + E) Q_k = Q_k (T_k + \tau a^2 e_k e_k^T) + (1 + \tau a b) r_k e_k^T.$$

If $0 < 1 + \tau a b$ then

$$\bar{T}_k = T_k + \tau a^2 e_k e_k^T$$

is a tridiagonal matrix whose eigenvalues are also eigenvalues of $A + E$. Using Theorem 8.18, it can be shown that the interval $[A_1(I_k), A_{k-1}(I_k)]$ contains an eigenvalue of A if $n = 2k$. These bracketing intervals depend on the choice of $\tau\alpha^2$. Suppose we have an approximate eigenvalue λ_k of A . One possibility is to choose $\tau\alpha^2$ so that

$$\det(\tilde{T} - \lambda I_k) = (\alpha_k + \tau\alpha^2 - A)\mathbf{P}_{k-1}(A) - f_k^2 \mathbf{P}_k(A) = 0$$

where the polynomials $P_k(x) = \det(T_i - xI)$ can be evaluated at A using the three-term recurrence (8.42). (This assumes that $\mathbf{P}_k(A) = 0$.) The idea of characterizing an approximate eigenvalue λ_k as an exact eigenvalue of a nearby matrix $A + E$ is discussed in Lehmann (1963) and Householder (1988).

hyhhhib .ktf Sc" Sf \$b7) \$eKb

The preceding discussion indicates how eigenvalue estimates can be obtained via the Lanczos process, but it reveals nothing about the approximation quality of T_k 's eigenvalues, a function of k . Results of this variety have been developed by Kaniel, Paige, Saad, and others and the following theorem is a sample from this body of research.

Theorem 10.1.2 Let A be an n -by- n symmetric matrix with Schur decomposition

$$Z^T A Z = \text{diag}(A_1, \dots, A_n), \quad A = \begin{pmatrix} A_1 & & \\ & \ddots & \\ & & A_n \end{pmatrix}, \quad Z = [Z_1 \ I \ \dots \ I \ Z_n]. \quad (10.1.9)$$

Suppose k steps of the Lanczos iteration (Algorithm 10.1.1) are performed and that T_k is the tridiagonal matrix (10.1.5). If $\lambda = A_1(T_k)$, then

$$A_1 := (i : : A_1 - (A_1 - A_n) \left(\frac{\tan(\phi)}{c_k(1+2p)} \right)^2)$$

where $\cos(\phi_1) = |q_1^T z_1|$,

$$\rho_1 = \frac{\lambda_1 - \lambda_2}{A_2 - A_n}, \quad (10.1.10)$$

and $c_{k-1}(x)$ is the Chebyshev polynomial of degree $k-1$.

From Theorem 8.12 we have

$$(i = \max_{y \neq 0} \frac{y^T T_k y}{y^T y} = \max_{\substack{\text{in a} \\ \# = 0}} \frac{(Q_k y)^T A (Q_k y)}{(Q_k y)^T (Q_k y)} = \max_{0 \neq w \in \mathcal{K}(A, q_1, k)} \frac{w^T A w}{w^T w}.$$

Since A_1 is the maximum of $w^T A w / w^T w$ over all nonzero w , it follows that $(i = A_1)$. To obtain the lower bound for $(i$, note that

$$1 = \max_{p \in P_{k-1}} \frac{q(p(A)A)p(A)q}{q^T p(A)^2 q_1},$$

where P_{k-1} is the set of degree $(k-1)$ polynomials and $p(x)$ is the amplifying polynomial. Given the eigenvector expansion $q_1 = d_1 z_1 + \dots + d_n z_n$ where $d = q^T Z$, it follows that

$$\frac{q(p(A)A)p(A)q}{q^T p(A)^2 q_1} = \frac{\sum_{i=1}^n d_i p(A)^2 A_i}{\sum_{i=1}^n d_i p(A)^2} \geq \frac{\lambda_1 d_1^2 p(\lambda_1)^2 + \lambda_n \delta^2}{d_1^2 p(\lambda_1)^2 + \delta^2} = A_1 - \frac{(A_1 - A_n) \delta^2}{d^T p(A)^2}.$$

where

$$\delta^2 = \sum_{i=2}^n d_i^2 p(\lambda_i)^2.$$

If the polynomial p has the property that it is large at $x = \lambda_1$ compared to its value at $\lambda_2, \dots, \lambda_n$, then we get a better lower bound for the Ritz value (*i.e.* This is the act of finding an *amplifying polynomial* and a good choice is to set

$$p(x) = c_{k-1} \left(1 + 2 \frac{x - \lambda_n}{\lambda_2 - \lambda_n} \right)$$

where $c_{k-1}(z)$ is the $(k-1)$ st Chebyshev polynomial generated via the recursion

$$c_k(z) = 2zc_{k-1}(z) - c_{k-2}(z), \quad c_0 = 1, c_1 \neq z.$$

These polynomials are bounded by unity on $[-1, 1]$, but grow very rapidly outside this interval. By defining $p(x)$ this way, it follows that $|p(\lambda_i)| \leq 1$ for $i = 2:n$ and $p(\lambda_1) = c_{k-1}(1 + 2\rho_1)$ where ρ_1 is defined by (10.1.10). Thus,

$$\delta^2 \leq \sum_{i=2}^n d_i^2 = 1 - d?$$

and so

$$\lambda_1 - (\lambda_1 - \lambda_n) \frac{1 - d_1^2}{d_1^2} \frac{1}{(c_{k-1}(1 + 2\rho_1))^2}.$$

The desired lower bound is obtained by noting that $\tan(c_1)^2 \geq (1 - d_1^2)/d_1^2$. \square

An analogous result pertaining to T_k 's smallest eigenvalue is an easy corollary.

Corollary 10.1.3. *Using the same notation as in the theorem, if $\lambda_k(T_k)$, then*

$$\lambda_n \leq 2 \leq \lambda_n + (\lambda_1 - \lambda_n) \left(\frac{\tan(c_n)}{c_{k-1}(1 + 2\rho_n)} \right)^2$$

where

$$\rho_n = \frac{\lambda_{n-1} - \lambda_n}{\lambda_1 - \lambda_{n-1}}$$

and $\cos(c_n) = q_1^T z_n$.

Proof. Apply Theorem 10.1.2 with A replaced by $-A$. \square

The key idea in the proof of Theorem 10.1.2 is to take the amplifying polynomial $p(x)$ to be the translated Chebyshev polynomial, for then $p(A)q_1$ amplifies the component of q_1 in the direction of the eigenvector z_1 . A similar idea can be used to obtain bounds for an interior Ritz value θ_i . However, the results are not as satisfactory because the new amplifying polynomial involves the product of the Chebyshev polynomial c_{k-i} and the polynomial $(x - \lambda_1) \cdots (x - \lambda_{i-1})$. For details, see Kaniel (1966) and Paige (1971) and also Saad (1980), who improved the bounds. The main theorem is as follows.

Theorem 10.1.4. Using the same notation as Theorem 10.1.2, if $1 \leq i \leq k$ and $\theta_i = \lambda_i(T_k)$, then

$$\lambda_i \geq \theta_i \geq \lambda_i - (\lambda_1 - \lambda_n) \left(\frac{\kappa_i \tan(\phi_i)}{c_{k-i}(1 + 2\rho_i)} \right)^2$$

where

$$\rho_i = \frac{\lambda_i - \lambda_{i+1}}{\lambda_{i+1} - \lambda_n}, \quad \kappa_i = \prod_{j=1}^{i-1} \frac{\theta_j - \lambda_n}{\theta_j - \lambda_1}, \quad \cos(\phi_i) = |q_1^T z_i|.$$

Proof. See Saad (NLE, p. 201). /a

Because of the κ_i factor and the reduced degree of the amplifying Chebyshev polynomial, it is clear that the bounds deteriorate as i increases.

hyhnga b 7\\$b lk\\$b h\\$ Qktb f SeeDebQ\\$b sttf fleb h\\$ Qktb

It is instructive to compare θ_1 with the corresponding power method estimate of λ_1 . (See §8.2.1) For clarity, assume $\lambda_1 > \dots > \lambda_{n-1}$. In the Schur decomposition (10.1.7), after $k-1$ power method steps applied to q_1 , a vector is obtained in the direction of

$$\mathbf{v} = A^{k-1} q_1 = \sum_{i=1}^n d_i \lambda_i^{k-1} z_i$$

along with an eigenvalue estimate

$$\gamma_1 = \frac{v^T A v}{\sqrt{v^T v}}.$$

By setting $p(x) = x^{k-1}$ in the proof of Theorem 10.1.2, it is easy to show that

$$\lambda_1 \geq \gamma_1 \geq \lambda_1 - (\lambda_1 - \lambda_n) \tan(\phi_1)^2 \left(\frac{\lambda_2}{\lambda_1} \right)^{2(k-1)}. \quad (10.1.11)$$

Thus, we can compare the quality of the lower bounds for θ_1 and γ_1 by comparing

$$L_{k-1} \equiv \frac{1}{\left[c_{k-1} \left(2 \frac{\lambda_1}{\lambda_2} - 1 \right) \right]^2} \geq \frac{1}{[c_{k-1}(1 + 2\rho_1)]^2}$$

and

$$R_{k-1} = \left(\frac{\lambda_2}{\lambda_1} \right)^{2(k-1)}.$$

Figure 10.1.1 compares these quantities for various values of k and λ_2/λ_1 . The superiority of the Lanczos bound is self-evident. This is not a surprise since θ_1 is the minimum of $r(x) = x^T Ax/x^T x$ over all of $\mathcal{K}(A, q_1, k)$, while $\gamma_1 = r(v)$ for a particular v in $\mathcal{K}(A, q_1, k)$, namely, $v = A^{k-1} q_1$.

λ_i/λ_2	k= 5	k= 10	k= 15	k= 20	k= 25
1.50	11x04 5xD2	20x09 6xD4	39x06 1xD5	74x02 2xD7	14x07 5xD9
1.10	27x02 4xD1	55x05 1xD1	i.1x07 6xD2	21x09 2xD2	43x03 1xD2
1.01	56x01 9xD1	10x01 8xD1	15x02 7xD1	20x03 6xD1	25x04 6xD1

Figure 10.11 $L_k i / R_k 1$

Problems

- P10.1.1 ..ddg.8 A $R^{n \times n}$ $C_{\beta\beta\gamma_0^T} A^T U \gamma_0^T (A - C \begin{pmatrix} 0 & \gamma_0^T \\ \gamma_0^T & 0 \end{pmatrix}) A^T$ $\lambda_1 = \frac{\gamma_0^T A^T}{\gamma_0^T \gamma_0}$ $\lambda_2 = \frac{\gamma_0^T A^T}{\gamma_0^T \gamma_0}$
- P10.1.2 18. A $R^{n \times n}$ $C_{\beta\beta\gamma_0^T} A^T U \gamma_0^T (A - C_{11}^T A^T)$ R^n $(C_{11}^T A^T)^T = A$
- P10.1.3 J.86 l. r1.8.r4.b A E $R^{n \times n}$ $(\beta_{-0}^T \gamma_0^T) \gamma_0^T (\beta_{-0}^T \gamma_0^T) \gamma_0^T$ $(A - C_{11}^T A^T)^T = A$
- P10.1.4 ..86 J. ..8 r.8b , 188.8. 10.1.1 ma $\text{http://www.MPEB.ac/A00taipIAI.Api.ar.a.A.t}$
5l ARA XAK -
- P10.1.5 18. A $R^{n \times n}$ $C_{\beta\beta\gamma_0^T} A^T U \gamma_0^T (A - C_{11}^T A^T)^T \gamma_0^T \beta_{-0}^T$ $\lambda_1 = \frac{\gamma_0^T A^T}{\gamma_0^T \gamma_0}$ $\lambda_2 = \frac{\gamma_0^T A^T}{\gamma_0^T \gamma_0}$
- P10.1.6 ..ddg.8 A $R^{n \times n}$ $C_{\beta\beta\gamma_0^T} A^T U \gamma_0^T (A - C_{11}^T A^T)^T \gamma_0^T \beta_{-0}^T$ $\lambda_1 = \frac{\gamma_0^T A^T}{\gamma_0^T \gamma_0}$ $\lambda_2 = \frac{\gamma_0^T A^T}{\gamma_0^T \gamma_0}$
- P10.1.7 ..ddg.8 T $R^{n \times n}$ $C_{\beta\beta\gamma_0^T} A^T U \gamma_0^T (A - C_{11}^T A^T)^T \gamma_0^T \beta_{-0}^T$ $\lambda_1 = \frac{\gamma_0^T A^T}{\gamma_0^T \gamma_0}$ $\lambda_2 = \frac{\gamma_0^T A^T}{\gamma_0^T \gamma_0}$

Notes and References for §10.1

88.r.8..8..8.. 81.88.r4 1.4 8..8..l8 r... r. 3...8.. (k(c) dro2eh1dro (01.).(e 6n3nidc 1er61lo=0dr6z.nTeo).o 1nt

1. 0dr6z.n(1950). 4ep DittAi.MptiuMS 5l i8t 2M0ri.MPEB ut xm)tpIA0rtNIM.0tc M.ftpmtAl t,tltpimA0 ApSDpit)lA0l.tlAiMlaSL J. Res. Nat. Bur. Stand. 45, 255-282.

xMlStiA.0aq.Mri iut MpItl)tp.t MEBt o,3 IA0rtaS attk

2. -Ap.t0 (1966). 4xai.cAita 5l 2Mct .Mc.riAi.MpA0 t .upm.rta mft.ptAl e0)t.lASL Math. Comput. 20, 369-378.

AxAmra(A1971) qn 2ERAS 2 TMB Qk R= TMB 2 i1 R n LPrn,RWk -t Ank i(B)IAI 2 22

HBCat (1980). 4lp ,ut sACta MEMpTratl)tp, M.,t ftAp.3MaApS,,t f1M.y ftAp.1Malt,3SawAA
SIAM J. Numer. Anal. 17, 687-706.

.t .Mppi,6MpatC.ltp ftAp.1Ma,llS 6})MpaA0l!6MypMIC.3)MpaAM0hpMclA0ApS,t ,tMlh M.c
-tp,a Alt Cla.rlatS lpS

,c,c ftt.cApp (1963). 4l.Cl cA0t x6)tp.tl, tlpas.0ltaarp) tpwLNumer. Math. 5, 246-272.
e12-2Mrat.M1St(1968). 41Mctp,a ApS..Alq.C,lla,, sMMCD DpSnumer. Math. 11, 126-128.
Cc21CMr.(1974). 2Mct ,ata MEutftAp.0Me0)MIC.c 6p,rctl..A0 ptAl e0)u.lAwLlp Topics
in Numerical Analysis, ,1p-2-1l00tl tSc ue.AStcl. Nlttaaut. 1M lyc
.c.- NAl)twf1,1 ,Al0tCCApS 2ceFraApS 5MlaC(1995). 4e.lMRlcAct 2M0r6MpApS xl)tpIA0rt
fMrpSaMMC -lh0MpxAtawppNumer. Lin. Alg. Applic. 2, 115-133.

10.2 Lanczos, Quadrature, and Approximation

To deepen our understanding of the Lanczos process and to build an appreciation for its connections to other areas of applied mathematics, we consider an interesting approximation problem that has broad practical implications. Assume that $A \in \mathbb{R}^{n \times n}$ is a large, sparse, symmetric positive definite matrix whose eigenvalues reside in an interval $[a, b]$. Let $f(\cdot)$ be a given smooth function that is defined on $[a, b]$. Given $u \in \mathbb{R}^n$, our goal is to produce suitably tight lower and upper bounds b and B so that

$$b \leq u^T \cdot f(A) \cdot u \leq B. \quad (102.1)$$

In the approach we develop, the bounds are Gauss quadrature rule estimates of a certain integral and the evaluation of the rules requires the eigenvalues and eigenvectors of a Lanczos-produced tridiagonal matrix.

The $\text{ulf}(A)$ estimation problem has many applications throughout matrix computations. For example suppose x is an approximate solution to the symmetric positive definite system $Ax = b$ and that we have computed the residual $r = b - Ax$. Note that if $x^* = A^{-1}b$ and $f(\cdot) = 1/\cdot$, then

$$\|x - x^*\| = (x^* - x)^T (x - x^*) = (A^{-1}(b - Ax))^T (A^{-1}(b - Ax)) = r^T f(A)r.$$

Thus, if we have a $\text{ulf}(A)$ estimation framework, then we can obtain $Ax = b$ error bounds from residual bounds.

For an in-depth treatment of the material in this section, we refer the reader to the treatise by Golub and Meurant (2010). Our presentation is brief, informal, and stresses the linear algebra highlights.

hymighb gfe Dlf Tkbkbf esb .e kf fSb

Without an integral in sight, it is mystifying as to why (102.1) involves quadrature at all. The key is to regard $\text{ulf}(A)u$ as a Riemann-Stieltjes integral. In general, given a suitably nice integrand $f(x)$ and weight function $w(x)$, the Riemann-Stieltjes integral

$$I(f) = \int_b^a f(x)dw(x)$$

is a limit of sums of the form

$$S_{N^{ij}} \sum_{j=1}^N f(q)(w(x_j) - w(x_{j+1}))$$

where $a = x_N < \dots < x_1 = b$ and $x_{\mu+1} \leq c_\mu \leq x_\mu$. Note that if w is piecewise constant on $[a, b]$, then the only nonzero terms in S_N arise from subintervals that have a "wjump". For example suppose $a = \lambda_n < \lambda_2 < \dots < \lambda_1 = b$ and that

$$w(\lambda) = \begin{cases} W_{t+1} & \text{if } \lambda < a, \\ w_\mu & \text{if } \lambda_\mu \leq \lambda < \lambda_{\mu-1}, \quad \mu = 2:n, \\ W_t & \text{if } b \leq \lambda, \end{cases} \quad (10.2.2)$$

where $0 \leq w_{n+1} \leq \dots \leq w_1$. By considering the behavior of S_N as $N \rightarrow \infty$, we see that

$$\int_a^b f(\lambda) dw(\lambda) = \sum_{\mu=1}^n (w_\mu - w_{\mu+1}) \cdot f(\lambda_\mu). \quad (10.2.3)$$

We are now set to explain why $\mathbf{Uf(A)u}$ is "secretly" a Riemann-Stieltjes integral. Let

$$A = XAX^\top \quad A = \text{diag}(A_1, \dots, A_n), \quad (10.2.4)$$

be a Schur decomposition of A with $\lambda_n \leq \dots \leq \lambda_1$. It follows that

$$\mathbf{Uf(A)u} = (\mathbf{X}\mathbf{f}\mathbf{u})^\top \mathbf{T} \cdot f(\Lambda) \cdot (\mathbf{X}\mathbf{f}\mathbf{u}) = \sum_{\mu=1}^n \mathbf{X}\mathbf{f}\mathbf{u}^\top \mathbf{J} \cdot f(\lambda_\mu).$$

If we set

$$w_\mu = \mathbf{X}\mathbf{f}\mathbf{u}^\top + \dots + \mathbf{X}\mathbf{f}\mathbf{u}^\top, \quad \mu = 1:n+1, \quad (10.2.5)$$

in (10.2.2), then (10.2.3) becomes

$$\int_a^b f(\lambda) dw(\lambda) = \sum_{\mu=1}^n \mathbf{X}\mathbf{f}\mathbf{u}^\top \mathbf{J} \cdot f(\lambda_\mu) = \mathbf{Uf(A)u} \quad (10.2.6)$$

Our plan is to approximate this integral using Gauss quadrature

hyhihib ukSb otEe erK ESB5Ette tf EeSb9E dSebtttb ukDtteb

Given an accuracy-related parameter k , an interval $[a, b]$, and a weight function $w(\lambda)$, a Gauss-type quadrature rule for the integral

$$I(f) = \int_a^b f(\lambda) dw(\lambda)$$

involves a carefully constructed linear combination of k -evaluations across $[a, b]$. The evaluation points (called nodes) and the coefficients (called weights) that define the linear combination are determined to make the rule correct for polynomials up to a certain degree that is related to k . Here are four examples

1. Gauss. Compute weights w_1, \dots, w_k and nodes t_1, \dots, t_k so if

$$I(a!) = \sum_{i=1}^k w_i f(t_i) \quad (10.2.7)$$

then $I(J) = Ia(!)$ for all polynomials f that have degree $2k-1$ or less.

2. Gauss-Radau(a). Compute weights W_{a, w_1, \dots, w_k} and nodes t_1, \dots, t_k so if

$$Ia(f) = W_a f(a) + \sum_{i=1}^k w_i f(t_i) \quad (1028)$$

then $I(J) = Ia(f)$ for all polynomials f that have degree $2k$ or less.

3. Gauss-Radau(b). Compute weights W_{b, w_1, \dots, w_k} and nodes t_1, \dots, t_k so if

$$Ia(f) = W_b f(b) + \sum_{i=1}^k w_i f(t_i) \quad (1029)$$

then $I(J) = Ia(f)$ for all polynomials f that have degree $2k$ or less.

4. Gauss-Lobatto. Compute weights $W_a, W_b, W_{w_1, \dots, w_k}$ and nodes t_1, \dots, t_k so if

$$Ia(f) = W_a f(a) + W_b f(b) + \sum_{i=1}^k w_i f(t_i) \quad (10210)$$

then $I(f) = Ia(f)$ for all polynomials f that have degree $2k+1$ or less.

Each of these rules has a neatly specified error. It can be shown that

$$I_a^b f(\lambda) d\omega(\lambda) = \begin{cases} Ia(!) & + Ra(!), \\ Ia(f) & + R_{a,f}(J), \\ Ia(f) & + R_{a,f}(J), \\ Ia(f) & + Ra(f), \end{cases}$$

where

$$Ra(!) = \frac{f^{(2k)}(\eta)}{(2k)!} \int_a^b \left[\prod_{i=1}^k (\eta - t_i) \right]^2 d\omega(\lambda), \quad a < \eta < b$$

$$R_{a,f}(f) = \frac{f^{(2k+1)}(\eta)}{(2k+1)!} \int_a^b \left[\prod_{i=1}^k (\eta - t_i) \right]^2 d\omega(\lambda), \quad a < \eta < b$$

$$R_{GR(b)}(f) = \frac{f^{(2k+1)}(\eta)}{(2k+1)!} \int_a^b \left[\prod_{i=1}^k (\eta - t_i) \right]_2^2 d\omega(\lambda), \quad a < \eta < b$$

$$Ra(f) = \frac{f^{(2k+2)}(\eta)}{(2k+2)!} \int_a^b \left[(\eta - a)(\eta - b) \prod_{i=1}^k (\eta - t_i) \right]^2 d\omega(\lambda), \quad a < \eta < b$$

If the derivative in the remainder term does not change sign across $[a, b]$, then the rule can be used to produce a bound. For example, if $f(\lambda) = 1/\lambda^2$ and $0 < a < b$ then

$f^{(2k)}$ is positive, $f^{(2k+1)}$ is negative, and we have

$$I_G(f) = \int_D w(\lambda) dw(\lambda) = I_{GR(a)}(f).$$

With this strategy, we can produce lower and upper bounds by selecting and evaluating the right rule. For this to be practical, the behavior of f 's higher derivatives must be known and the required rules must be computable.

hymhrb 7\\$b 7 stst" kttfb .ktt\\$ ff sktb

It turns out that the evaluation of a given Gauss quadrature rule involves a tridiagonal matrix and its eigenvalues and eigenvectors. To develop a strategy that is based upon this connection, we need three facts about orthogonal polynomials and Gauss quadrature.

Fact 1. Given $[a, b]$ and $w(\lambda)$, there is a sequence of polynomials $p_0(\lambda), p_1(\lambda), \dots$ that satisfy

$$\int_a^b p_i(\lambda) \cdot p_j(\lambda) \cdot dw(\lambda) = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases}$$

with the property that the degree of $p_k(\cdot)$ is k for $k \geq 0$. The polynomials are unique up to a factor of ± 1 and they satisfy a 3-term recurrence

$$\gamma_k p_k(\lambda) = (\lambda - w_k) p_{k-1}(\lambda) - \gamma_{k-1} p_{k-2}(\lambda)$$

where $p_{-1}(\lambda) = 0$ and $p_0(\lambda) = 1$.

Fact 2. The zeros of $p_k(\lambda)$ are the eigenvalues of the tridiagonal matrix

$$T_k = \begin{bmatrix} \omega_1 & \gamma_1 & 0 & \cdots & 0 \\ \gamma_1 & \omega_2 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \omega_{k-1} & \gamma_{k-1} \\ 0 & \cdots & 0 & \gamma_{k-1} & \omega_k \end{bmatrix}.$$

Since the γ_i are nonzero, it follows from Theorem 8.4.1 that the eigenvalues are distinct.

Fact 3. If

$$S^T T_k S = \text{diag}(\theta_1, \dots, \theta_k) \quad (102.11)$$

is a Schur decomposition of T_k , then the nodes and weights for the Gauss rule (102.7) are given by $t_i = \theta_i$ and $w_i = s_{1i}^2$ for $i = 1:k$. In other words,

$$I_G(f) = \sum_{i=1}^k s_{1i}^2 \cdot f(\theta_i). \quad (102.12)$$

Thus, the only remaining issue is how to construct T_k so that it defines a Gauss rule for (102.6).

1.c.c)A 4 N++ATN}L= }-Nf cA,n)AE}1 Q+A

We show that if we apply the symmetric Lanczos process (Algorithm 10.1.1) with starting vector $q_1 = u/\|u\|_2$, then the tridiagonal matrices that the method generates are exactly what we need to compute $I_G(f)$.

We first link the Lanczos process to a sequence of orthogonal polynomials. Recall from §10.1.1 that the k th Lanczos vector Q_k is in the Krylov subspace $\mathcal{K}(A, q_1, k)$. It follows that $q_k = p_k(A)q_1$ for some degree k polynomial. From Algorithm 10.1.1 we know that

$$\beta_k q_{k+1} = (A - \alpha_k I)q_k - \beta_{k-1} q_{k-1}$$

where $\beta_0 q_0 = 0$ and so

$$\beta_k p_{k+1}(A)q_1 = (A - \alpha_k I)p_k(A)q_1 - \beta_{k-1} p_{k-1}(A)q_1.$$

From this we conclude that the polynomials satisfy a 3-term recurrence

$$\beta_k p_{k+1}(\lambda) = (\lambda - \alpha_k) p_k(\lambda) - \beta_{k-1}^2 p_{k-1}(\lambda). \quad (10.2.13)$$

These polynomials are orthogonal with respect to the $u^T f(A)u$ weight function defined in (10.2.5). To see this, note that

$$\begin{aligned} \int_a b p_i(\lambda) p_j(\lambda) dw(\lambda) &= \sum_{\mu=1}^n [X^T u]_\mu^2 \cdot p_i(\lambda_\mu) \cdot p_j(\lambda_\mu) \\ &\quad \cdot (X^T u)^T (p_i(\Lambda) \cdot p_j(\Lambda)) \cdot (X^T u) \\ &= u^T (X \cdot p_i(\Lambda) \cdot X^T) (X \cdot p_j(\Lambda) \cdot X^T) u \\ &= u^T (p_i(A) p_j(A)) u \\ &\quad \cdot (p_i(A) u)^T (p_j(A) u) = \|u\|_2^2 \delta_{ij}, \quad 0 \end{aligned}$$

Coupled with (10.2.13) and Facts 1–3 this result tells us that we can generate an approximation $J_A f(A)$ to $u^T f(A)u$ as follows

Step 1: With starting vector $q_1 = u/\|u\|_2$, use the Lanczos process to compute the partial tridiagonalization $AQ_k = Q_k T_k Q_k^T$. (See (10.1.4).)

Step 2: Compute the Schur decomposition $S_k T_k S_k^T = \text{diag}(\lambda_1, \dots, \lambda_k)$.

Step 3: Set $J_A f(A) = \sum_{j=1}^k f(\lambda_j) + \text{sf}_k(Q_k)$.

See Golub and Welsch (1969) for a more rigorous derivation of this procedure.

hypipib .kEDf st"b f esbot Deei9 tttDb 9DPSb

Recall from (10.2.1) that we are interested in upper and lower bounds. In light of our remarks at the end of §10.2.2, we need techniques for evaluating other Gauss quadrature rules. By way of illustration, we show how to compute $I_{G_R(a)}$ defined in (10.2.8). Guided by Gauss quadrature theory, we run the Lanczos process for k steps if we were setting out to compute $I_G(f)$. We then must determine a_{k+1} so that if

$$\tilde{T}_{k+1} = \left[\begin{array}{cc|c} & 0 & \cdots & 0 & 0 \\ 1 & \alpha_2 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \alpha_{k-1} & \beta_{k-1} \\ 0 & \cdots & & \beta_{k-1} & \alpha_k \\ \hline 0 & \cdots & \cdots & 0 & \beta_k \\ & & & & \tilde{\alpha}_{k+1} \end{array} \right]$$

then $a \in \mathbb{G}_{+1}$). By considering the top and bottom halves of the equation

$$\tilde{T}_{k+1} \begin{bmatrix} x \\ -1 \end{bmatrix} = a \begin{bmatrix} x \\ -1 \end{bmatrix}, \quad x \in \mathbb{R}^k,$$

it is easy to verify that $\tilde{\alpha}_{k+1} = a + \beta_{k+1}^2 e_k^T (T_k - aI_k)^{-1} e_k$ works.

hs oiha b ,hsb Ff ſe tf dʒet q ſakedb

All the necessary tools are now available to obtain sufficiently accurate upper and bounds in (10.2.1). At the bottom of the loop in Algorithm 10.1.1, we use the current tridiagonal (or an augmented version) to compute the nodes and weights for the lower bound rule. The rule is evaluated to obtain b . Likewise, we use the current tridiagonal (or an augmented version) to compute the nodes and weights for the upper bound rule. The rule is evaluated to obtain B . The while loop in Algorithm 10.1.1 can obviously be redesigned to terminate as soon as $B - b$ is sufficiently small.

Problems

$$\text{P10.2.1} \quad f_8 \ f_8 \ . \quad 8\cdot8 \ . \ R \quad N_8 \quad 8 \quad 89 \quad \text{Nro8} \mid p_k R_8 = 2xp_{k-1} - \dots = p_{k-2}(x) \\ (-) \quad x^k + \binom{k}{e} \cdot f_8 + \left(w^k x^e, -L_C x^{2e-1} \right)^{12} \cdot 3 = -x^k \mid a(a) \mid (1)$$

P10.2.2 fe⁸⁶srobR4. roN 1.8 R4^{10.2.5, 13.)3.} 3MIM8.ro Iap(b) GnO 1.

Notes and References for §10.2

f8 82d§8.88.8N 81 9fd.. N I 9I 8 9s R §R 1 . 8 §8.88or

Nt NsSl. N5 | (2019). *Matrices, Moments, and Quadrature with Applications*, **SIAM**

ot9dAdII..p i..9 TradtTraTraMF).s,3dh

*C, - CM0roX962). Kf3roFqd 6)uFITra0roMqd.S.Tra)M,Tfh98t3d.II xTraid.IIt9 M8.ro3tXhi.t
fto 1ti.MS B.Math. Comput. 16, 438-445.*

CMOrDraFS. t aII.(1969). 20B6MTC Trans. Trasf. elettron. S. Math. Comput. 23, 221-230.

C1-PCM0rd(1974). $fMro,S$ 1 *id.r* $\beta M8uF39$ *Rocky Mountain J. Math.* 4, 207-211.

St fMMqFSC = -CM0rd (1978). Kq0t ro8tu.II I40B u 0uIIMp9idrolisMqTraIIIMTraqd6MM-I. e3dTra0 tTraqTalg. Applic. 21 245-260.

,) -Trarioi9yAFSC=-EM0ro(1983). a (a (2 ,)1 ∈ (Lin. Alg. Applic. 52/53, 439-455.

- k menu cat Qt, Gr I (1991). 4xai.cAimp) iut IAI)ta1 2mp)r0A5A0.ta M.IAI)t 2JAlat 4Ail.ta
I.A 4MSl.tS 4Mctpia SINumer. Algs. 1, 353–374.
tcN. IArL.t (1996). 4epimeC Araam 6ASLA,rlt or0taSIMath. Comput. 65, 739–747.
k fAm ApS2CM0r(1997). 4fMrpSd iut gAt Miut upItlat ApSi ttitlc.pApi M2hcctill.
N3a.ilIt tt.plit 4Ailm.taSLAnnals Numer. Math. 4, 29–38.
4c ftp3. ApS Cc2c (1999). 4fMrpSd iut xpila M4A,lmR p.i.Mpa2miueJJ0..AiMpaiM
Nlt.MpSimMplp)BIT 39, 417–438.
tc .A0It ii.S Cc2c CMbr.S3 fraClA)S ApSIc ot..ut0 (2000). 4McJriAiMp M.CAraaM-lMplMS
6rASlAirt or0taSIMath. Comput. 69, 1035–1052.
tcN. IArL.t (2001). 4.McJriAiMpM.CAraae.hJt 6rASlftirlt xMlcr 0wSLJ. Comput. Appl. Math.
127, 201–217.

10.3 Practical Lanczos Procedures

Rounding errors greatly affect the behavior of the Lanczos iteration. The basic difficulty is caused by loss of orthogonality among the Lanczos vectors, a phenomenon that muddles the issue of termination and complicates the relationship between A's eigenvalues and those of the tridiagonal matrices T_k . This troublesome feature, coupled with the advent of Householder's perfectly stable method of tridiagonalization, explains why the Lanczos algorithm was disregarded by numerical analysts during the 1950s and 1960s. However, the pressure to solve large sparse eigenproblems coupled with the computational insights set forth by Paige (1971) changed all that. With many fewer than n iterations typically required to get good approximate extremal eigenvalues, the Lanczos method became attractive as a sparse matrix technique rather than a competitor of the Householder approach.

Successful implementation of the Lanczos iteration involves much more than a simple encoding of Algorithm 10.1.1. In this section we present some of the ideas that have been proposed to make the Lanczos procedure viable in practice.

hygrhb \$nDT\$nb uf kxt"\$b ttb -k edb

With careful overwriting in Algorithm 10.1.1 and exploitation of the formula

$$\alpha_k = q_k^T (Aq_k - \beta_{k-1}q_{k-1}),$$

the whole Lanczos process can be implemented with just a pair of n-vectors:

$$w = q_1, \mathbf{V} = Aw, \alpha_1 = w^T \mathbf{V}, \mathbf{V}_+ = \mathbf{V} - \alpha_1 w, \| \mathbf{V}_+ \|_2, k = 1$$

while $\beta_k \neq 0$

for $i = 1:n$

$$t = w_i, w_i = \mathbf{V}_+ / \|\mathbf{V}_+\|_2, v_i = -\beta_k t$$

end

$\mathbf{V} = \mathbf{V}_+ + Aw$

$$k = k + 1, \alpha_k = w^T \mathbf{V}, \mathbf{V} = \mathbf{V}_+ - \alpha_k w, \beta_k = \| \mathbf{V}_+ \|_2$$

end

(10.3.1)

At the end of the loop body, the array w houses q_k and \mathbf{V} houses the residual vector $r_k = Aq_k - \alpha_k q_k - \beta_{k-1}q_{k-1}$. See Paige (1972) for a discussion of various Lanczos implementations and their numerical properties. Note that A is not modified during

the entire process and that is what makes the procedure so useful for large sparse matrices.

If A has an average of v nonzeros per row, then approximately $(2v + 8n)$ flops are involved in a single Lanczos step. Upon termination the eigenvalues of T_k can be found using the symmetric tridiagonal QR algorithm or any of the special methods of §85 such as bisection. The Lanczos vectors are generated in the vector w . If eigenvectors are required, then the Lanczos vectors must be saved. Typically, they are stored in secondary memory units.

hygrib 9Dt tkeb .. k\\$ f \\$eb

The development of a practical, easy-to-use Lanczos tridiagonalization process requires an appreciation of the fundamental error analyses of Paige (1971, 1976, 1980). An examination of his results is the best way to motivate the several modified Lanczos procedures of this section.

After j steps of the iteration we obtain the matrix of computed Lanczos vectors $\hat{Q} = [\hat{q}_1 \hat{I} \cdots \hat{I} \hat{q}_k]$ and the associated tridiagonal matrix

$$\hat{T} = \begin{bmatrix} \hat{\alpha}_1 & \hat{\beta}_1 & & \cdots & 0 \\ \hat{\beta}_1 & \hat{\alpha}_2 & \ddots & & \vdots \\ \ddots & \ddots & \ddots & & \\ \vdots & \ddots & \ddots & & \hat{\beta}_{k-1} \\ 0 & \cdots & & \hat{\beta}_{k-1} & \hat{\alpha}_k \end{bmatrix}.$$

Paige (1971, 1976) shows that if \hat{r}_k is the computed analog of r_k , then

$$A\hat{Q}_k = \hat{Q}_k \hat{T}_k + \hat{r}_k e_k^T + E_k \quad (1032)$$

where

$$\|E_k\|_2 \approx \|u\| \|A\|_2. \quad (1033)$$

This shows that the equation $A\hat{Q}_k = Q_k T_k + r_k e_k^T$ is satisfied to working precision.

Unfortunately, the picture is much less rosy with respect to the orthogonality among the \hat{q}_i . (Normality is not an issue. The computed Lanczos vectors essentially have unit length.) If $\hat{\beta}_k = \|(\hat{r}_k)\|_2$ and we compute $\hat{q}_{k+1} = \text{fl}(\hat{r}_k / \hat{\beta}_k)$, then a simple analysis shows that

$$\hat{\beta}_k \hat{q}_{k+1} \approx \hat{r}_k + w_k$$

where

$$\|w_k\|_2 \approx \|u\| \|\hat{r}_k\|_2 \approx \|u\| \|A\|_2.$$

Thus we may conclude that

$$|\hat{q}_{k+1}^T \hat{q}_i| \approx \frac{|\hat{r}_k^T \hat{q}_i| + \|u\| \|A\|_2}{|\hat{\beta}_k|}$$

for $i = 1:k$. In other words, significant departures from orthogonality can be expected when $\hat{\beta}_k$ is small, even in the ideal situation where $\hat{r}_k^T \hat{Q}_k$ is zero. A small $\hat{\beta}_k$ implies

cancellation in the computation of \hat{r}_k . We stress that loss of orthogonality is due to one or several such cancellations and is not the result of the gradual accumulation of roundoff error.

Further details of the Paige analysis are given shortly. Suffice it to say now that loss of orthogonality always occurs in practice and with it, an apparent deterioration in the quality of \hat{T}_k 's eigenvalues. This can be quantified by combining (1032) with Theorem 8.1.16. In particular, if we set

$$F_1 = \hat{r}_k e_k^T + E_k, \quad X_1 = \hat{Q}_k, \quad S = \hat{T}_k,$$

in that theorem and assume that

$$\tau = \| \hat{Q}_k^T \hat{Q}_k - I_k \|_2$$

satisfies $\tau = 1$, then there exist eigenvalues $\mu_1, \dots, \mu_k \in \lambda(A)$ such that

$$|\mu_i - \lambda_i(T_k)| \leq \sqrt{2} (\| \hat{r}_k \|_2 + \| E_k \|_2 + \tau(2 + \tau) \| A \|_2)$$

for $i = 1:k$. An obvious way to control the τ factor is to orthogonalize each newly computed Lanczos vector against its predecessors. This leads directly to our first "practical" Lanczos procedure

hyhrmr b sttf flob jTf)b.kE dS Sb8s kf)k' ktd bftf bktb

Let $r_0, \dots, r_{k-1} \in \mathbb{R}^n$ be given and suppose that Householder matrices H_0, \dots, H_{k-1} have been computed such that $(H_0 \cdots H_{k-1})^T [r_0 \ I \ \cdots \ Ir_{k-1}]$ is upper triangular. Let $[q_1 \ I \ \cdots \ Ir_k]$ denote the first k columns of the Householder product $(H_0 \cdots H_{k-1})$. Now suppose that we are given a vector $r_k \in \mathbb{R}^n$ and wish to compute a unit vector q_{k+1} in the direction of

$$w = r_k - \sum_{i=1}^k q_i^T r_k) q_i \in \text{span}\{q_1, \dots, q_k\}^\perp.$$

If a Householder matrix H_k is determined so $(H_0 \cdots H_k) \mathbf{1} [r_0 \ I \ \cdots \ Ir_k]$ is upper triangular, then it follows that column $(k+1)$ of $H_0 \cdots H_k$ is the desired unit vector.

If we incorporate these Householder computations into the Lanczos process, then we can produce Lanczos vectors that are orthogonal to machine precision:

$$r_0 = q_1 \text{ (given unit vector)}$$

Determine Householder H_0 so $H_0 r_0 = e_1$.

for $k = 1:n - 1$

$$\alpha_k = q_k^T A q_k$$

$$r_k = (A - \alpha_k I) q_k - \beta_{k-1} q_{k-1}, \quad (\beta_0 q_0 \equiv 0) \tag{1034}$$

$$w = (H_{k-1} \cdots H_0) r_k$$

Determine Householder H_k so $H_k w = [w_1, \dots, w_k, \beta_k, 0, \dots, 0]^T$.

$$q_{k+1} = H_0 \cdots H_k e_{k+1}$$

end

This is an example of a complete reorthogonalization Lanczos scheme. The idea of using Householder matrices to enforce orthogonality appears in Golub, Underwood, and Wilkinson (1972). That the computed \hat{q}_i in (1034) are orthogonal to working precision follows from the roundoff properties of Householder matrices. Note that by virtue of the definition of q_{k+1} , it makes no difference if $\beta_k = 0$. For this reason, the algorithm may safely run until $k = n - 1$. (However, in practice one would terminate from a much smaller value of k .)

Of course, in any implementation of (1034), one stores the Householder vectors v_k and never explicitly forms the corresponding matrix product. Since we have $H_k(1:k, 1:k) = I$, there is no need to compute the first k components of the vector w in (1034) since we do not use them. (Ideally they are zero.)

Unfortunately, these economies make but a small dent in the computational overhead associated with complete reorthogonalization. The Householder calculations increase the work in the k th Lanczos step by $O(kn)$ flops. Moreover, to compute q_{k+1} , the Householder vectors associated with H_0, \dots, H_k must be accessed. For large n and k , this usually implies a prohibitive level of memory traffic.

Thus, there is a high price associated with complete reorthogonalization. Fortunately, there are more effective courses of action to take, but these require a greater understanding of just how orthogonality is lost.

ayhroslb usPS ff Tf \$9\$ kf dk' ktP bktf bktb

A remarkable, ironic consequence of the Paige (1971) error analysis is that loss of orthogonality goes hand in hand with convergence of a Ritz pair. To be precise, suppose the symmetric QR algorithm is applied to A and renders computed Ritz values $\hat{\theta}_1, \dots, \hat{\theta}_k$ and a nearly orthogonal matrix of eigenvectors $Eh = (Sp)$. If

$$\hat{Y}_k = [\hat{y}_1 | \cdots | \hat{y}_k] = f((\hat{Q}_k \hat{S}_k)),$$

then it can be shown that for $i = 1:k$ we have

$$|\hat{q}_{k+1}^T \hat{y}_i| \approx \frac{\|A\|_2}{|\hat{\beta}_k| |\hat{s}_{ki}|} \quad (1035)$$

and

$$\|A\hat{y}_i - \hat{\theta}_i \hat{y}_i\|_2 \approx |\hat{\beta}_k| |\hat{s}_{ki}|. \quad (1036)$$

That is, the most recently computed Lanczos vector \hat{q}_{k+1} tends to have a nontrivial and unwanted component in the direction of any converged Ritz vector. Consequently, instead of orthogonalizing \hat{q}_{k+1} against all of the previously computed Lanczos vectors we can achieve the same effect by orthogonalizing it against the much smaller set of converged Ritz vectors.

The practical aspects of enforcing orthogonality in this way are discussed in Parlett and Scott (1979). In their scheme, known as selective reorthogonalization, a computed Ritz pair $\{\hat{\theta}, \mathbf{f}\}$ is called "good" if it satisfies

$$\|A\hat{y} - \mathbf{B}\hat{f}\|_2 \leq \sqrt{\mathbf{u}} \|A\|_2.$$

As soon as \hat{q}_{k+1} is computed, it is orthogonalized against each good Ritz vector. This is much less costly than complete reorthogonalization, since, at least at first, there are many fewer good Ritz vectors than Lanczos vectors.

One way to implement selective reorthogonalization is to diagonalize' at each step and then examine the $\| \mathbf{Q} \mathbf{Q}_k \|_2$ in light of (1035) and (1036). A more efficient approach for large k is to estimate the loss of orthogonality measure $\| I_k - S^T S \|_2$ using the following result.

Lemma 10.3.1. Suppose $S_+ = [S_d]$ where $S \in \mathbb{R}^{n \times k}$ and $d \in \mathbb{R}^n$. If

$$\| I_k - S^T S \|_2 \leq \mu \quad |1 - d^T d| \leq \delta,$$

then

$$\| I_{k+1} - S_+^T S_+ \|_2 \leq \mu_+$$

where

$$\mu_+ = \frac{1}{2} \left(\mu + \delta + \sqrt{(\mu - \delta)^2 + 4 \| S^T d \|_2^2} \right).$$

Proof. See Kahan and Parlett (1974) or Parlett and Scott (1979). \square

Thus, if we have a bound for $\| I_k - \mathbf{Q} \mathbf{Q}_k \|_2$, then by applying the lemma with $S_+ = \mathbf{Q}_k$ and $d = \mathbf{r}_{k+1}$ we can generate a bound for $\| I_{k+1} - \mathbf{Q}_{k+1} \mathbf{Q}_{k+1}^T \|_2$. (In this case we assume that I_k has been orthogonalized against the set of currently good Ritz vectors.) It is possible to estimate the norm of \mathbf{Q}_{k+1} from a simple recurrence that spares one the need to access q_1, \dots, q_k . The overhead is minimal, and when the bounds signal loss of orthogonality, it is time to contemplate the enlargement of the set of good Ritz vectors. Then and only then is' diagonalized.

hypergib 7\\$b ok ef bAT\\$f tPDSb.kf f\\$b

Considerable effort has been spent in trying to develop a workable Lanczos procedure that does not involve any kind of orthogonality enforcement. Research in this direction focuses on the problem of "ghost" eigenvalues. These are multiple eigenvalues of' that correspond to simple eigenvalues of A . They arise because the iteration essentially restarts itself when orthogonality to a converged Ritz vector is lost. (By way of analogy, consider what would happen during orthogonal iteration (828) if we "forget" to orthogonalize.)

The problem of identifying ghost eigenvalues and coping with their presence is discussed by Cullum and Willoughby (1979) and Parlett and Reid (1981). It is a particularly pressing problem in those applications where all of A 's eigenvalues are desired, for then the above orthogonalization procedures are expensive to implement.

Difficulties with the Lanczos iteration can be expected even if A has a genuinely multiple eigenvalue. This follows because the' are unreduced, and unreduced tridiagonal matrices cannot have multiple eigenvalues. The next practical Lanczos procedure that we discuss attempts to circumvent this difficulty.

hypmrga b uf kfdb stf flkb eP' kTf).b

Just as the simple power method has a block analogue in simultaneous iteration, so does the Lanczos algorithm have a block version. Suppose $n = np$ and consider the

decomposition

$$Q^T A Q = \bar{T} = \begin{bmatrix} M_1 & B_1^T & \cdots & 0 \\ B_1 & M_2 & \ddots & \vdots \\ \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & B_{r-1}^T \\ 0 & \cdots & B_{r-1} & M_r \end{bmatrix} \quad (1037)$$

where

$$Q = [X_1 \ I \ \cdots \ I \ X_r], \quad X_i \in \mathbb{R}^{n \times p},$$

is orthogonal, each $M_i \in \mathbb{R}^{p \times p}$, and each $B_i \in \mathbb{R}^{p \times n}$ is upper triangular. Comparison of blocks in $AQ = Q\bar{T}$ shows that

$$AX_k = X_{k-1}B_{k-1}^T + X_kM_k + X_{k+1}B_k$$

for $k = 1:r$ assuming $X_0B_0^T = 0$ and $X_{r+1}B_r = 0$. From the orthogonality of Q we have

$$M_k = X_k^T AX_k$$

for $k = 1:r$. Moreover, if we define

$$R_k = AX_k - X_kM_k - X_{k-1}B_{k-1}^T \in \mathbb{R}^{n \times p},$$

then

$$X_{k+1}B_k = R_k$$

is a QR factorization of R_k . These observations suggest that the block tridiagonal matrix \bar{T} in (1037) can be generated as follows

$$X_1 \in \mathbb{R}^{n \times p} \text{ given with } X_1^T X_1 = I_p$$

$$M_1 = X_1^T AX_1$$

$$\text{for } k = 1:r - 1$$

{1038}

$$R_k = AX_k - X_kM_k - X_{k-1}B_{k-1}^T \quad (X_0B_0^T = 0)$$

$$X_{k+1}B_k = R_k \quad (\text{QR factorization of } R_k)$$

$$M_{k+1} = X_{k+1}^T AX_{k+1}$$

end

At the beginning of the k th pass through the loop we have

$$A[X_1 \ I \ \cdots \ I \ X_k] = [X_1 \ I \ \cdots \ I \ X_k] \bar{T}_k + R_k [0 \ I \ \cdots \ I \ 0] I_p, \quad \{1039\}$$

where

$$\bar{T}_k = \begin{bmatrix} M_1 & B_1^T & \cdots & 0 \\ B_1 & M_2 & \ddots & \vdots \\ \ddots & \ddots & \ddots & \ddots \\ \vdots & \ddots & \ddots & B_{k-1}^T \\ 0 & \cdots & B_{k-1} & M_k \end{bmatrix}.$$

Using an argument similar to the one used in the proof of Theorem 10.1.1, we can show that the X_k are mutually orthogonal provided none of the R_k is rank-deficient. However if $\text{rank}(R_k) < p$ for some k , then it is possible to choose the columns of X_k such that $X_k^T X_i = 0$ for $i = 1:k$. See Golub and Underwood (1977).

Because f_k has bandwidth p , it can be efficiently reduced to tridiagonal form using an algorithm of Schwartz (1968). Once tridiagonal form is achieved, the Ritz values can be obtained via the symmetric QR algorithm or any of the special methods of §8.4. In order to decide intelligently when to use block Lanczos, it is necessary to understand how the block dimension affects convergence of the Ritz values. The following generalization of Theorem 10.1.2 sheds light on this issue.

Theorem 10.3.2. Let A be an n -by- n symmetric matrix with Schur decomposition

$$Z^T A Z = \text{diag}(A_1, \dots, A_n), \quad \lambda_1 \geq \dots \geq \lambda_n, \quad Z = [z_1 \ I \ \dots \ I_{z_n}].$$

Let μ_1, \dots, μ_p be the p largest eigenvalues of the matrix f_k obtained after k steps of (10.3.8). Suppose $Z_1 = [z_1 \ I \ \dots \ I_{z_n}]$ and

$$0 < \cos(\phi_p) = \sigma_p(Z_1^T X_1),$$

the smallest singular value of $Z_1^T X_1$. Then for $i = 1:p$,

$$\lambda_i \geq \mu_i \geq \lambda_i - (\lambda_1 - \lambda_n) \left(\frac{\tan(\theta_p)}{\rho_i - (1 + \tau_p)} \right)^2$$

where

$$\rho_i = \frac{\lambda_i - \lambda_{p+1}}{\lambda_{p+1} - \lambda_n}$$

and $c_{k-1}(z)$ is the Chebyshev polynomial of degree $k-1$.

Proof. See Underwood (1975). Compare with Theorem 10.1.2. \square

Analogous inequalities can be obtained for f_k 's smallest eigenvalues by applying the theorem with A replaced by $-A$. Based on the theorem and scrutiny of (10.3.8), we conclude that

- the error bounds for the Ritz values improve with increased p ;
- the amount of work required to compute f_k 's eigenvalues is proportional to $k p^2$;
- the block dimension should be at least as large as the largest multiplicity of any sought-after eigenvalue.

Determination of the block dimension in the face of these tradeoffs is discussed in detail by Scott (1979). We mention that loss of orthogonality also plagues the block Lanczos algorithm. However, all of the orthogonality enforcement schemes described above can be extended to the block setting.

1, APA:A t3ITA E1 Q+A n3f.= G-DIA uG-FA.o-} = -GfA

The block Lanczos algorithm (1038) can be used in an iterative fashion to calculate selected eigenvalues of A. To fix idea, suppose we wish to calculate the p largest eigenvalues. If $X \in \mathbb{R}^{n \times p}$ is a given matrix having orthonormal columns, then it can be refined as follows.

Step 1. Generate $X_2, \dots, X_s \in \mathbb{R}^{n \times p}$ via the block Lanczos algorithm

Step 2. Form $T_s = [X_1 | \dots | X_s]^T A [X_1 | \dots | X_s]$, an sp-by-sp matrix that has bandwidth p

Step 3. Compute an orthogonal matrix $U = [U_1 | \dots | U_p]$ such that $U^T T_s U = \text{diag}(\dots)$ with $1 \leq \lambda_i < \lambda_{i+1}$

Step 4. Set $X_1^{(\text{new})} = [X_1 | \dots | X_s] [u_1 | \dots | u_p]$.

This is the block analog of the s-step Lanczos algorithm which has been extensively analyzed by Cullum and Donath (1974) and Underwood (1975). The same idea can be used to compute several of A's smallest eigenvalues or a mixture of both large and small eigenvalues. See Cullum (1978). The choice of the parameters s and p depends upon storage constraints as well as upon the block-size implications that we discussed above. The value of p can be diminished as the good Ritz vectors emerge. However, this demands that orthogonality to the converged vectors be enforced.

Problems

P10.3.1 8. **R t 8t4.5** (10.3.8) $\left(\begin{array}{cc} a & 0 \\ 0 & a \end{array} \right) \geq 0 \quad a > 0$

P10.3.2 **re1** $\text{rank}(R_k) < p$. F7.1171g. 13 n3[003. 3. $\text{ft3ran}(X_1 | \dots | X_k)$] \rightarrow " . | +n] A?

Notes and References for §10.3

mf8 8f. R8. f8 4 2. 9 R9 . . s1.s1 .8 8 r0b8.R.R4....d8.90 R4

3 3a R8 5P 18 d.9.9R8 PR.8 ..18. R.0 e1e8s.1 ..81d ..8 .9.R48. .89. R0.R0. R9.1 .8

.d8.ro 4rsrob d.d8.. R44§..8or

3a3 R.R05 3820.9R . .R g1r0 18 R8isof8. .rof8R.84d.8 2.87. Inst.

Math. Applic. 10, CkC.Cng
 $\dots - 2A)t (1976). \dots) \theta .0 a (\geq 6., a \neq 0) \lambda = n \lambda^2 3 \theta r V 4 u =$
 $u = J\text{-Inst. Math. Applic. } 18, 341-349.$
 $\geq \geq , - \frac{1}{4} = :6 \geq \dots \geq 0 , \frac{(2a)^2}{(a)} (, a (\geq (6.-) a \geq a, 0 \geq) \geq)$
 $(\text{Lin. Alg. Applic. } 4, 15y \text{ Ir d})$

OLM ReP1SSgt/E72AgA2yEeAAiMqA229.1. nA,MiaA. 1. + Agg

Evlval22.jJ.u ol+ 2l niqA yAn al,3grMP2yAllirMrhngE+gt6. th. Comput. 33,
 $n. n-o$

$f), .Alot,3 u23t, .8 MFApSft.1. 13l.,jul 7sg n.1 l9 xa38ft3.,) 3.t ftclta3 x,t7TraAOrt3u 3ut$
 $ftAp.3M.3)8 Math. Comput. 38, 153-166.$

$f,n .Alot33 A,S 1. n = p = iT MOPp.pP + ZJ ke01 00 =)=p=L0OC+ Oa z3=y On$
 $> a Ap3ZEyn a TSin. Alg. Applic. 68, so, nnle$

Bt)Eu (O2 ot tBF aCO/O ly, liB H(Q,OQ2h QAdlheQ l2abSe .u Q=B/en at
 UoE(OShQA(/2D oxotr CQn SIAM J. Matrix Anal. Appl. 13 NJINNIIM
 .MIAIMiaiae -MsMiBlf JJ7.9amAiaa- It iae Ala) Iri.A.MielAa3grlM1mCat.narA
 hMAaltMlyClausActa Numerica 15, -os , -n=

e .tAO.C. MElAC6.A0Ap.3Maelt0AC6S5lcA CMpcAh TraSrpS 6p{

p1=-r0rc ApSo1e1 .60 0Mr)u.h .n1 n.1 Lanczos Algorithms for Large Symmetric Eigenvalue Computations: Vol. I: Theory, 2ue4 Nr.06.ACmMpA0 10JutNAB
 p1f1MpS 41 ..S t- x006AmApSo1 N0t8cMpa.s((=-Proceedings of the Cornelius Lanczos International Centenary Conference, 2ue4 Nr. 06.AC6MpNSmQ0J.6ASNe

xMLASma.raamMtraMrtMrtMICuM)MpA0m3AC6Mp aatxtixaB

.1.ra NA6).ts.(o1. 1 4NIA.Cm.A0t MCut2hcctClm. ftAp.3MdlM.tas .6CtMICuM MpA0636M BSL
 10, sg7 s,1

C121 CM0or1BpStl.MMSS ApS p121.m0ymqMps(o.n.1 4.ut ftAp.3Me0)Ml6Cc 5l C t 2hrrixctCs
 16. Az = λBx NIMTralBdotJMIC2.e,re.2e onenolBttJAICctpC MEMcJrCtl 2.6tp.tS 2CAp5LS
 ,p mItlamCkSAp5LSS.e1

f1,1 NAl0tC4pS t12MCQs(o. 1 ftAp.3Me0)M6C.c .mC.2t0t.CmI4lCuM)MpA063AC6MpS.
 Comput. 33, I NT5DM

.MEMClaf N, M3aigtltlf nsTCCAM.laa- lt 3grlM1mGrlAlM.yrlraig1- i.11A.mL
 let u6in. Alg. Appl. 61, s1sBs7n1

.6yuMrC Aph ltMICuM)MpA063AC6GnMtaaAh t6CutlCMixMp6CMI 0MaM EMIC.M)MpA0636B2rmC
 ACCut AJJlJMlmAGpaCApMltot CMStI6at A.uoct C.AC.600 mStpC6np.Mpl)ts t6)tpIA0rta ApS
 EA0atr0Cm.10CmRattk

.1 -AuAp ApS f1,1 NAl0tCG(o.. =42M. xAl 2.Mr0S CM6C. Cut ftAp.3MdlM.taa,L 6p Sparse
 Matrix Computations, p101frp.. ApS t1p1oMat.ts a1.B.AStc6. NltaaSo. 7MlySs7s h-
 p1r0rc ApS1e. 600Mj.h .s(o. 1 4ftAp.3M4pS CMcJrCACmMp2Jt.67tS upCtIA0ME Cut
 2Jt.Clrrix M.ftAl)ts2JtAlat otA0 2hcctCl6. 4AClm.taBpSparse Matrix Proc., 12d1 ApSCra.ra
 2Ct.ALC.tSa1B2uel NrTr4C6MSPN.60AS0J.mABe1

f1,1 NAl0tC4pSp1-rot6S .s(gs.1 4gA.ypm) C.t NIM)tsME.t ftAp.3M0)Ml6Cuc5l ftAl)t 2hrrix5
 ctClm. x6)tpJlMTraS1IMA J. Num. Anal. 1, s7, -,,.

xMAltaCAIC6pMAct.3ly CM ar.taaEr,B6CraC tRJ0MfGtAJJlMR6cACtPAI6Ap6r.aJA.t mp535
 cAC6M6pAaCw Trattpt.2rmLtS TraBt6CdlAcnMpaGaATraMrCt dM SmpB adt0

t1 .A0ItCCmS ft1 ApS t1t0S 2Mltpatp .s((-. 1 4ep ucJ06.6y0htaCAICtS ftAp.3Ma4tC.MS u3
 ftAl)t 2hcctCl6. xm)tpIA0rtNIM.0ocaSLETNA 2, sB.1

-1 r ApS21 2mcMpn111. ra4.um.yectaCAICtAp.3M4tCu3S 5l ftAl)t 2hcctCl6. x6)tpIA0rt N3M
 0tca SLISIAM J. Matrix Anal. Appl. 22, .1nS...1

.ud Tra0MfAp.3Ma A0)Ml6C6a.raatS6p0

J. ep < $\langle \mathbf{p} \mathbf{x} \mathbf{LB} \mathbf{-}, \mathbf{=}, \mathbf{a} \mathbf{NO} \mathbf{D} \mathbf{Y} \mathbf{r} \mathbf{=}, \mathbf{m} \mathbf{=}, \mathbf{n} \mathbf{y} \mathbf{o} \mathbf{y} \mathbf{=}, \mathbf{Jv} \mathbf{I} \mathbf{x} \mathbf{e} \mathbf{=}, \mathbf{x} \mathbf{f} \mathbf{p} \mathbf{v} \mathbf{J} \mathbf{y} \mathbf{<} \mathbf{y} \mathbf{On} \mathbf{<} \mathbf{x} \mathbf{On} \mathbf{a} \mathbf{=}, \mathbf{y} \mathbf{Oov} \mathbf{-}, \mathbf{Jy} \mathbf{G} \mathbf{a} \mathbf{=}, \mathbf{a} \mathbf{=<} \mathbf{p} \mathbf{0} \mathbf{6} \mathbf{=}, \mathbf{=}, \mathbf{D} \mathbf{J} \mathbf{0} \mathbf{y} \mathbf{e} \mathbf{f} \mathbf{a} \mathbf{=}, \mathbf{y} \mathbf{0} \mathbf{0} \mathbf{a} \mathbf{+}, \mathbf{y} \mathbf{0} \mathbf{0} \mathbf{ex} \mathbf{x} \mathbf{Ov} \mathbf{H} \mathbf{a} \mathbf{v} \mathbf{=}, \mathbf{Jn} \mathbf{Oo} \mathbf{=}$
 Proceedings of the 1974 IEEE Conference on Decision and Control, N.Mtp6R&B.1, .1(rA

o1 ,pStl.MMS .s(o. 1 4ep uCtACmfpM.y ftAp.3M4tCuMSl Cut2M0rCMMpAl)- 2JAlat 2hrrixv
 l6. xm)tpIs0rt N3MTralBdotJMIC2.e,re.2e o,'-, StpJAICctpC MEMcJrCtl 2.mtp.tS 2CAp43S
 ,p61 tlamCkS Ap5LSS.e1

C121 CM0rTra1 ,pStl.MMS .s(oo.1 4.ut f0M.: ftAp.3M4tC.MS 5l .McJrC6p) x6)tpIA0rtaS3np
 Mathematical Software III, p10mt .cS1Se.AStc6. N3taaSh. 7MlySJJ17.- 7001

p1 .r0rrt4 og.1 2mcr0C AptMMrizJrCACnMpxt. MEute0)tlA6.A00h ftAl)taCApS2cA00
 xm)tpIA0rtM A ftAl)t 2JAlat 2hcctCl6. 1ACl6FSLBIT 18, n.,Bn.o,1

e1 or t.s(o.1 4DcJ0tctpCACmfpM.y ftAp.3Me0)MlmC.ca5l .McJrCACmMp6)tp b'
 rta MfpAl)t 2JAlat 2hcctCl6. 4ACl6.taSLMath. Comput. 33, .g1 . .go1

.ut .0M.y ftAp.3M0)MlmCtpSIACtaAshctClm. TraApS cACl6R t6MplA0rta.Ap TraMcJrC
 mpApH MEtItlA0.Aha1 lpt AJJlMA..mSta.lm.tS 6p0

21o 12...AIC3 .s(.g. . 4g6S6A)MpA0m3AC6M2pcrxtClmfApS 4ACl6R NLnumer. Math. 12, n7s n ra

XarTe orS)Qa(.Q(a)e((nu_ QnQna e(QTQ)(18.Qnn elcaelSe(. Iae (QnQhA QnacSu
UoE(QQ2eons z)i gsesoel ehele3neull of ieahisestl enf s1onf 3meleciahiloesstsel
et aelutreheMsn'5ia5 f heneasbrieicesnepiselenof 3hesetjntrosel cabneias)tsl f cerief tal
if chil linid spinsel aerid"ciotol stheVeed

- a-el ilcaeI reuelrtlosl iad r.l.l ifie7al (1976). 4.A0.r0AC6M₄,Mrix A0MSta M.l.,Apa ,a.p)
AIAp,3MatCuMSSAA Sp6se Matrix Computations, p b, frp.u ApSt,p1 oMat tSa) lasire7sl
messMed it nMplfty fh01
FAlpnf ssstalarl al Tchel(1980). 4.8t 2Jt.CIA1 gApa5lcAC.Mp ftAp,3MatC8MS5l Cut,rrixt...A0
2M0rC.Mp,ftAl)t 2JAlatCptlAa63tS2hrixctil6. x.)tpIA1rt NLM.OtrixaAMath. Comput. 35, 1251–
1268.
ocf-1Ml)qp (1991). 4.MrixJri6p)DlCtl.Mt6)tlIA0rta M.ftAl)t 1Ail..ta SAAAn. Alg. Applic. 154-156,
289–309.
o1C1 Cl.rizaSp1C1ftt.a BAps21tc 2.rixM(p1994). 4e 2869tS f1M.y ftAp,3Mæl)Ml.C8rix5l 2MbI.p)
2JAlat 2hrixrxtClCptlA1.3tS x6)tpJLM.OtrixaSSAM J. Matrix Anal. Applic. 15, 228–272.

10.4 Large Sparse SVD Frameworks

The connections between the SVD problem and the symmetric eigenvalue problem are discussed in §861. In light of that discussion, it is not surprising that there is a Lanczos process for computing selected singular values and vectors of a large sparse, rectangular matrix A . The basic idea is to generate a bidiagonal matrix B that is orthogonally equivalent to A . We show how to do this in §54 using Householder transformations. However, to avoid large dense submatrices along the way, the Lanczos approach generates the bidiagonal entries directly.

hymstlb dkPdF st tttb REESxbuTt Tt'k ttP ftQTktb

Suppose $A \in \mathbb{R}^{m \times n}$ with $m \geq n$ and recall from §548 that there exist orthogonal $E \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ so that

$$u^T A V = B = \begin{bmatrix} 0 & 1 & /1 & \cdots & \cdots & 0 \\ 0 & 0 & 2 & /2 & \cdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & & 0 & Q_1 & 1 & f_n & 1 \\ 0 & \cdots & \cdots & 0 & 0 & & Q_1 \end{bmatrix}. \quad (104.1)$$

Since A and B are orthogonally related, they have the same singular values.

Analogously to our derivation of the symmetric Lanczos procedure in §10.1.1, we proceed to outline a sparse matrix-friendly method for determining the diagonal and superdiagonal of B . The challenge is to bypass the generally full intermediate matrices associated with the Householder bidiagonalization process (Algorithm 54.2). We expect to extract good singular value/vector information long before the full bidiagonalization is complete.

The key is to develop useful recipes for the one's and 's of the matrix equations $AV = UB$ and $AT^T = VBT$. Given the column partitionings

$$U = [u_1 \ I''' \ I U_m], \quad V = [V_1 \ \dots \ V_n],$$

we have

$$Av_k = \alpha_k u_k + \beta_{k-1} u_{k-1}, \quad (1042)$$

$$ATu_k = \alpha_k v_k + \beta_k v_{k+1} \quad (1043)$$

for $k=1:n$ with the convention that $\beta_0 u_0 = 0$ and $\beta_n v_{n+1} = 0$. Define the vectors

$$r_k = Av_k - \beta_{k-1} u_{k-1}, \quad (1044)$$

$$pk = ATu_k - \alpha_k v_k. \quad (1045)$$

Using (1042), (1044), and the orthonormality of the u -vectors, we have

$$\alpha_k = \pm \| r_k \|_2,$$

$$u_k = r_k / \alpha_k, \quad (\alpha_k \neq 0).$$

Note that if $\alpha_k = 0$ then from (1041) it follows that $A(:,1:k)$ is rank deficient. Similarly we may conclude from (1043) and (1045) that

$$\beta_k = \pm \| pk \|_2,$$

$$v_{k+1} = pk / \beta_k, \quad (\beta_k \neq 0).$$

If $\beta_k = 0$ then it follows from the equations $AV = UB$ and $ATU = VBT$ that

$$AU(:,1:k) = V(:,1:k)B(1:k,1:k), \quad (1046)$$

$$ATV(:,1:k) = U(:,1:k)B(1:k,1:k)T, \quad (1047)$$

and thus

$$ATAV(:,1:k) = V(:,1:k)B(1:k,1:k)^T B(1:k,1:k).$$

It follows that $a(B(1:k,1:k)) \subseteq a(A)$.

Properly sequenced, the above equations mathematically define the Golub-Kahan process for bidiagonalizing a rectangular matrix.

Wif nl2d2itUmi-1 tayALrr,0012 yq nGyl0An0nylo2 Given a matrix $A \in \mathbb{R}^{m,n}$ with full column rank and a unit 2-norm vector $v_c \in \mathbb{R}^n$, the following algorithm computes the factorizations (1046) and (1047) for some k with $1 \leq k \leq n$. The first column of V is v_c .

$k = 0$, $p_0 = v_c$, $\beta_0 = 1$, $u_0 = 0$

while $\beta_k = 0$

$$v_{k+1} = p_k / \beta_k$$

$k = k + 1$

$$r_k = Av_k - \beta_{k-1} u_{k-1}$$

$$\alpha_k = \| r_k \|_2$$

$$u_k = r_k / \alpha_k$$

$$p_k = A^T u_k - \alpha_k v_k$$

$$\beta_k = \| p_k \|_2$$

end

This computation was first described by Golub and Kahan (1965). If $V_k = [v_1 | \dots | v_k]$, $U_k = [u_1 | \dots | u_k]$, and

$$B_k = \begin{bmatrix} \alpha_1 & \beta_1 & \cdots & \cdots & 0 \\ 0 & \alpha_2 & \beta_2 & \cdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & & 0 & k_1 & \beta_{k-1} \\ 0 & \cdots & 0 & 0 & \alpha_k \end{bmatrix}, \quad (1048)$$

then after the k th pass through the loop we have

$$AV = U_k B_k, \quad (1049)$$

$$A^T U_k = V_k B_k + P_k, \quad (10410)$$

assuming that $\alpha_i > 0$. It can be shown that

$$\text{span}\{v_1, \dots, v_k\} = K(A^T A, k), \quad (10411)$$

$$\text{span}\{u_1, \dots, u_k\} = K(A A^T, k). \quad (10412)$$

Thus, the symmetric Lanczos convergence theory presented in 10.15 can be applied. Good approximations to A 's largest singular values emerge early, while the small singular values are typically more problematic, especially if there is a cluster near the origin. For further insight, see Luk (1978), Golub, Luk, and Overton (1981), and Björck (NMS, §7.6).

$$1) c_A = G Q A N V V^T p G^T - G^T A$$

The Ritz idea can be applied to extract approximate singular values and vectors from the matrices U_k , V_k and B_k . We simply compute the SVD

$$F_k^T B_k G_k = \Gamma = \text{diag}(\gamma_1, \dots, \gamma_k) \quad (10413)$$

and from the matrices

$$Y_k = V_k \Gamma = [y_1 | \dots | y_k],$$

$$Z_k = U_k F_k = [z_1 | \dots | z_k].$$

It follows from (1049), (10410), and (10413) that

$$AY_k = z_k,$$

$$A^T z_k = Y_k + P_k F_k$$

and so for $i = 1:k$ we have

$$Ay_i = \gamma_i z_i, \quad (10414)$$

$$A^T z_i = \gamma_i y_i + [F_k]_{ki} \cdot p_k. \quad (10415)$$

It follows that $A^T A y_i = \gamma_i^2 z_i + \|F_k\|_2^2 p_k$ and thus $\{\Gamma, y_i\}$ is a Ritz pair for $A^T A$ with respect to $\text{ran}(V_k)$.

In mPA E A f mGf.:}2 (tnGf .}2A ...o i-GA

In §8.6 we showed that there is a connection between the SVD of a matrix $A \in J_{mn}$ and the Schur decomposition of the symmetric matrix

$$C = \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}. \quad (104.16)$$

In particular, if α is a singular value of A , then both α and $-\alpha$ are eigenvalues of C and the corresponding singular vectors "make up" the corresponding eigenvectors.

Likewise, a given bidiagonalization of A can be related to a tridiagonalization of C . Assume that $m = n$ and that

$$[U_1/U_f] AV = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}, \quad i \in J_{mn}$$

is a bidiagonalization of A with $U_1 \in J_{mn}$, $U_f \in J_{mn}$, and $V \in J_{mn}$. Note that

$$Q = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}$$

is orthogonal and

$$\tilde{T} = Q^T C Q = \begin{bmatrix} 0 & \tilde{B} \\ \tilde{B}^T & 0 \end{bmatrix}.$$

This matrix can be symmetrically permuted into tridiagonal form. For example, in the 4 by 3 case, if $P = h([5162734])$, then the reordering $\tilde{T} \rightarrow P\tilde{T}P^T$ has the form

$$\left[\begin{array}{cccc|cc} 0 & 0 & 0 & 0 & \alpha_1 & \beta_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & \alpha_2 & \beta_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & \alpha_3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right] \rightarrow \left[\begin{array}{cccc|cc} 0 & \alpha_1 & 0 & 0 & 0 & 0 & 0 \\ \alpha_1 & 0 & \beta_1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & \alpha_2 & 0 & 0 & 0 \\ 0 & 0 & \alpha_2 & 0 & \beta_2 & 0 & 0 \\ 0 & 0 & 0 & \beta_2 & 0 & \alpha_3 & 0 \\ 0 & 0 & 0 & 0 & \alpha_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right].$$

This points to an interesting connection between Golub-Kahan bidiagonalization (Algorithm 104.1) and Lanczos tridiagonalization (Algorithm 101.1). Suppose we apply Algorithm 104.1 to $A \in J_{mn}$ with starting vector v_c . Assume that the procedure runs for k steps and produces the bidiagonal matrix B_k displayed in (104.8). If we apply Algorithm 101.1 to the matrix C defined by (104.16) with a starting vector

$$q_1 = \begin{bmatrix} & C \end{bmatrix} E J_{mn} \quad (104.17)$$

then after $2k$ steps the resulting tridiagonal matrix T_{2k} has a zero diagonal and off-diagonal specified by $[\alpha_1, \beta_1, \alpha_2, \dots, \alpha_{k-1}, \beta_{k-1}, \alpha_k]$.

1) If $A \in \mathbb{C}^{n \times n}$, then $\|A\|_F = \sqrt{\lambda_{\max}(A^T A)}$

In §11.4.2 we show how the Golub-Kahan bidagonalization can be used to solve sparse linear systems and least squares problems. It turns out that in this context, lower bidiagonalization is more useful:

$$U \tilde{T} A V = \begin{bmatrix} a_{11} & 0 & \cdots & \cdots & 0 \\ 1 & a_{21} & 0 & \cdots & \vdots \\ \vdots & \beta_3 & \ddots & \ddots & \vdots \\ \vdots & & \ddots & & \vdots \\ \cdots & \cdots & & & \cdots \\ \cdots & \cdots & & & \cdots \end{bmatrix}.$$

1A) t4A rAI -oAA } LhQ dA Eu(} TArff 7qG } mA

The need to extract information from unimaginably large datasets has prompted the development of matrix methods that involve randomization. The idea is to develop matrix approximations that are very fast to compute because they rely on limited, random samplings of the given matrix. To give a snapshot of this increasingly important paradigm for large-scale matrix computations, we consider the problem of computing a rank- k approximation to a given matrix $A \in \mathbb{R}^{m \times n}$. For clarity we assume that $k = \text{rank}(A)$. Recall that if $A = \tilde{Z}\tilde{\Sigma}\tilde{Y}^T$ is the SVD of A , then

$$\tilde{A}_k = \tilde{Z}_1\tilde{\Sigma}_1\tilde{Y}_1^T = \tilde{Z}_1\tilde{\Sigma}_1^T A \quad (1042)$$

where $Z_1 = Z(:, 1:k)$, $f_1 = f(1:k, 1:k)$, and $Y = Y(:, 1:k)$, is the closest rank- k matrix to A , measured in either the 2-norm or Frobenius norm. We assume that A is so large that the Krylov methods just discussed are impractical.

Dineen, Kannan, and Mahoney (2006) propose a method that approximates the intractable A_k with a rank- k matrix of the form

$$A_k = CUR, \quad C \in \mathbb{R}^{m \times c}, U \in \mathbb{R}^{c \times r}, R \in \mathbb{R}^{r \times n}, k \leq c, k \leq r \quad (1042)$$

where the matrices C and R are comprised of randomly chosen values taken from A . The integers c and r are parameters of the method. Discussion of the CUR decomposition (1042) nicely illustrates the notion of random sampling in the matrix context and the idea of a probabilistic error bound.

The first step in the CUR framework is to determine C . Each column of this matrix is a scaled, randomly-selected column of A :

Determine column probabilities $Q_j = \|A(:, j)\|_2 / \|A\|_F^2$, $j = 1:n$.

for $t = 1:c$

Randomly pick $\text{col}(t) \in \{1, 2, \dots, n\}$ with Q_j the probability that $\text{col}(t) = _a$.

$C(:, t) = A(:, \text{col}(t)) / \sqrt{c Q_{\text{col}(t)}}$

end

It follows that $C = A(:, \text{col})D_C$ where $D_C \in \mathbb{R}^{c \times c}$ is a diagonal scaling matrix.

The matrix R is similarly constructed. Each row of this matrix is a scaled, randomly-selected row of A :

Determine row probabilities $X_{1:i} = \|A(i, :)\|_2 / \|A\|_F^2$, $i = 1:m$.

for $t = 1:r$

Randomly pick $\text{row}(t) \in \{1, 2, \dots, m\}$ with $X_{1:i}$ the probability that $\text{row}(t) = _a$.

$R(t, :) = A(\text{row}(t), :) / \sqrt{r X_{\text{row}(t)}}$

end

The matrix R has the form $R = D_R A(\text{row}, :)$ where $D_R \in \mathbb{R}^{r \times r}$ is a diagonal scaling matrix.

The next step is to choose a rank- k matrix U so that $A_k = CUR$ is close to the best rank- k approximation A_k . In the CUR framework, this requires the SVD

$$C = Z\Sigma Y^T = Z_1\Sigma_1 Y_1^T + Z_2\Sigma_2 Y_2$$

where $Z_1 = Z(:, 1:k)$, $E1 = E(1:k, 1:k)$, and $Y = Y(:, 1:k)$. The matrix U is then given by

$$U = \Phi \Psi^T, \quad \Phi = Y_1 \Sigma_1^{-2} Y_1^T, \quad \Psi = D_R C(row, :).$$

With these definitions, simple manipulations confirm that

$$C\Phi = Z_1 E^T Y, \quad (1042)$$

$$\Psi^T R_1^T (D_R (Z_1(row, :) \Sigma_1 Y_1^T + Z_2(row, :) E2Y))^T D_R A(row, :), \quad (1042)$$

and

$$CUR = (C\Phi)(\Psi R) = Z_1 (D_R Z_1(row, :)) (D_R A(row, :)). \quad (1042)$$

An analysis that critically depends on the selection probabilities $\{q\}$ and $\{p\}$ shows that $\text{ran}(Z_1) \approx \text{ran}(\tilde{Z})$ and $(D_R Z_1(row, :))^T (D_R A(row, :)) \approx Z_1^T A$. Upon comparison with (1042) we see that $CUR \approx Z_1 Z_1^T A \approx \tilde{Z}_1 \tilde{Z}_1^T A = k$. Moreover, given $r > Q$ and k , it is possible to choose the parameters r and c so that the inequality

$$\|A - CUR\|_F \leq \|A - \tilde{A}_k\|_F + \epsilon \|A\|_F$$

holds with probability $1 - \delta$. Lower bounds for r and c that depend inversely on r and δ are given by Drineas, Kannan, and Mahoney (2006).

Problems

P10.4.1 e8NR1. (10.4.6), (10.4.7), (10.4.9), ApS(10.4.10).

P10.4.2 38.8.d84.R4ro8. (10.3.1), StIt03. }p 6c.0tctpiA,Mp ME0)Ml6i.8 10.4.1 , Ai 6pIMaIta A c6p.croc ,ro8.td MEBu,Ml Mya.Ata=

P10.4.3 .f8robo.rR1 rank(A) - n , , M $\in \mathbb{C}^{n \times n}$, $f()$, $1 \leq r \leq n$, B P (10.4.18) .AppM,,AIt A 3tlMMp 6iaS6A)M,Aa)

P10.4.4 3.8.8 (10.4.19). ...Ai ...A, hMro qAhMrbl(:, 1:k) = V(:, 1:k) [f1] = 0 6p)Ml.i.8 10.4.2?

P10.4.5 1..8.81. ro8. (10.4.11)-(10.4.12), ,3. , ,Ai d)M.i.9 10.4.2 t AIt

$$\text{span}\{v_1, \dots, v_k\} = J(\Lambda V A V^T), \quad \text{span}\{u_1, \dots, u_k\} = J(\Lambda U U^T)$$

P10.4.6 .ld d8.8C $A, S \in \mathbb{C}^{n \times n}$ (10.4.16) A, S (10.4.17) loa.t.i.Itah= rqR2)M. uAi

$$\kappa(C, q_1, 2k) - a \cdot Ap \left\{ [C]_q [AG]_q [AV], \dots, [(AV)]_q' [A^T]^k [V]_q \right\}.$$

!)(P10.4.7) !k - !k [W§10.4.3 A.Mroi ,mro.S6A)MpADB- r. 2iAit ApS lMI1 A,AaM)Mrdta-0,a.ut, i.t N16)ore2AropStdq)Mp{063},6Mp root.P

P10.4.7 e8NR . ro8. (10.4.24).

Notes and References for §10.4

fig. .8N848td.8f8. R8.t84. s1N8.l.1orf. .R.R8..1 .dg.688P .4N5.6 §7.6). ..t dtotIA,t ME1 IAp. Ma8.taa imIt .6S.})MpA0.3},6MfBlt.,qp)ro0Al caIl.r . da, dtatpitS .p0

C121 C30}pS = -A.Ap (1965). m.A0.roGAp ,u 26p)ro0AAdA0rota4pS2atroSMre pIMI1A id6rB11 SIAM J. Numer. Anal. Ser. B, 2, 205–224.

.u .SM. roa6p M.0e0).Ap .6S6A)M,A0.3},6MM aMAt a.}la 06pAl aha,t9a }pS 0twi a.roAlta .dM.0tca aiAd,tS .6,u JAtdo

1616Røle ayi 2r hjt bhas8Eosla 12ksUJA(ot ZSSja r2UædUlib)ra(5 SIAM J. Numer. Anal. 11, N.T IJ.

aA tmigIA ilA pltit iSIIp pmPmApA iSp1v,pP P aapPaPpPpl pipePtaIttPpap +ASMAfAapA +ASMePirlaig taMAT

AvAhinrAae I v3vLilaeA.tN.dh „LVI. 3a 3grlMnp 1LSi.tAPaAi.9csipPlataeLSi.tA ,Ap LsliAt6 ACM Trans. Math. Softw. 8,5r TNc

I. S.iapPaPjSgiAapipPlaIttTAAT

.v.v .lgIS .vav,IB. iae I v,v,iA.pla 4N.dN .3 wglabiaa- lt nAp.le 1 AlCSI pPpmA Lna gi. BigsAiae Al.AtslaePaenarIgBAp.t l. i Iip.Pb6 ACM Trans. Math. Softw. 7, Nr N7.2

9VAI gglCv3vangIrmStae I v,iBA 4.d5h2 .3 ,iaa- lt 3grlPpmi. AliSspnaiLrsgim BiglAiaeBAaplif,i.rA nipPa SIAM J. Sci. Stat. Comput. 4 (Tr N.c

I vw)M.tNI h2.,ir),LaigALSi.tA nsgiglBiglAliSspipPlat International J. Supercomputing Appl. 6 N5,c

I vwAMiae I v,3IAMSiaN..5h .3 wglB,iaa- lt L-E I Ap.le + npSeiSpPilA pmlrlaign-pnla,6a Proceedings of the Cornelius Lanczos International Centenary Conference, oA0dm)8zuv IDE4 Nro.8n.AimM pmlrlaign-ASt0JumMew

k. pmAAS t. „r 3n17.- 4eDrix JhahodaiAlitS otrnpts f.S.)MpA0.3AimMap.3Mati8MS 5d .Mrixroi,p) A NAdi.A0mp)rooAdA0rt tt.McJM ami.Mp SIAM J. Matrix Anal. Applic. 25, I,7r I 7..,apAMAtppSigaiprlat pia- lt SPePirlaign- Rplle3T

Evh,Ait iae 9v3Lniilat 4N.dN .3 wpleirlaigP- ipPlarArIgi.rhpliHMM,iMrA LaigAntaApm- ipPllggItle hlsGAC6 SIAM J. Sci. Stat. Comput. 2 ,Tr ,dc

EVAigiAppnv .lgsS. iae,v lAnaAU.9tpniiprla pAasiAiPi,iaa- lt wnePilaigM n-pPla,6IT 39,7J57N.c C Đ ðp %@ AE8 v J 0Đ Đ8ÔcA3Đ ð)0Đ Å @5"R T•Đ |

.vEVLpa iae.v -i 4JJlc,l+ LliaB Iip.Pb8SSMlbPiipPlaPam,iaa- lt wPerlaign- ipnlahlaAtt+ npSSSgipPla SIAM J. Sci. Comput. 21, II.Tr IIc

I. tBApaf pmAr leAaliSltPpPlaA+ IMP'S.Ae laT

hvE.PaA.lv 5iaia. iae Rvav

10.5 Krylov Methods for Unsymmetric Problems

If A is not symmetric, then the orthogonal tridiagonalization $Q^T A Q = T$ does not exist in general. There are two ways to proceed. The Arnoldi approach involves the column-by-column generation of an orthogonal Q such that $Q^T A Q = H$ is the Hessenberg reduction of §7.4. The unsymmetric Lanczos approach computes the columns of matrices Q and P so that $P^T A Q = T$ is tridiagonal and $P^T P = I$. Methods based on these ideas that are suitable for large, sparse, unsymmetric eigenvalue problems are discussed in this section.

hyhnlb 78utesfb eetkPtsb .e kf Seeb

One way to extend the Lanczos process to unsymmetric matrices is due to Arnoldi (1951) and revolves around the Hessenberg reduction $Q^T A Q = H$. In particular, if $Q = [Q_1 \cdots Q_k]$ and we compare columns in $AQ = QH$, then

$$A_{ik} = \sum_{j=1}^{k-1} h_{kj} q_j \quad 1 \leq k \leq n-1$$

Isolating the last term in the summation gives

$$h_{k+1,k} = A_{k+1,k} - \sum_{j=1}^k h_{kj} q_j = r_k$$

where $h_{k+1,k} = A_{k+1,k}$. It follows that if $r_k = 0$ then QH is specified by

$$Q_{k+1} = r_k h_{k+1,k}$$

where $h_{k+1,k} = \|r_k\|_2$. These equations define the Arnoldi process and in strict analogy to the symmetric Lanczos process (Algorithm 10.1) we obtain the following

method (using MATLAB syntax). If $A \in \mathbb{R}^{n,n}$ and $q_1 \in \mathbb{R}^n$ a unit norm then the following algorithm computes a matrix $Q = [q_1 \cdots q_n] \in \mathbb{R}^{n,n}$ with orthonormal columns and an upper Hessenberg matrix $H = (h_{ij}) \in \mathbb{R}^{n,n}$ with the property that $AQ = QH$. The integer t satisfies $1 \leq t \leq n$.

$$k = 0 \quad r_0 = \|q_1\|_2$$

while ($h_{k+1,k} \neq 0$)

$$Q_{k+1} = r_k h_{k+1,k}$$

$$k = k + 1$$

$$r_k = A_{kk}$$

for $i = 1 \dots k$

$$h_{ki} = q_i r_k$$

$$r_k = r_k - h_{ki} q_i$$

end

$$h_{k+1,k} = \|r_k\|_2$$

end

$$t = k$$

The q_k are called *Arnoldi vectors* and they define an orthonormal basis for the Krylov subspace $\mathcal{K}(A, q_1, k)$:

$$\text{span}\{q_1, \dots, q_k\} = \text{span}\{q_1, Aq_1, \dots, A^{k-1}q_1\}. \quad (1051)$$

The situation after k steps is summarized by the equation

$$AQ_k = Q_k H_k + r_k e_k^T \quad (1052)$$

where $Q_k = [q_1 | \cdots | q_k]$, $e_k = I_k(:, k)$, and

$$H_k = \begin{bmatrix} h_{11} & h_{12} & \cdots & \cdots & h_{1k} \\ h_{21} & h_{22} & \cdots & \cdots & h_{2k} \\ \mathbf{0} & h_{32} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & & \vdots \\ \mathbf{0} & \cdots & \cdots & h_{k,k-1} & h_{kk} \end{bmatrix}.$$

Any decomposition of the form (1052) is a *k -step Arnoldi decomposition* if $Q_k \in \mathbb{R}^{n \times k}$ has orthonormal columns, $H_k \in \mathbb{R}^{k \times k}$ is upper Hessenberg, and $Q_k^T r_k = \mathbf{0}$.

If $y \in \mathbb{R}^k$ is a unit 2-norm eigenvector for H_k and $H_k y = \lambda y$, then from (1052)

$$(A - \lambda I)x = (e_k^T y)r_k$$

where $x = Q_k y$. Since $r_k \in \mathcal{K}(A, q_1, k)^\perp$, it follows that (λ, x) is a Ritz pair for A with respect to $\mathcal{K}(A, q_1, k)$. Note that if $v = (e_k^T y)r_k$, then

$$(A + E)x = \lambda x$$

where $E = -vx^T$ with $\|E\|_2 = \|y_k\| \|r_k\|_2$. Recall that in the unsymmetric case computing an eigenvalue of a nearby matrix does *not* mean that it is close to an exact eigenvalue.

Some numerical properties of the Arnoldi iteration are discussed by Wilkinson (AEP, p. 38). The history of practical Arnoldi-based eigensolvers begins with Saad (1986). Two features of the method distinguish it from the symmetric Lanczos process

- Arnoldi vectors q_1, \dots, q_k must all be refined in step k and the computation of q_{k+1} involves $O(kn)$ fops excluding the matrix-vector product Aq_k . Thus, there is a steep penalty associated with the generation of long Arnoldi sequences.
- Extremal eigenvalue information is not available forthcoming as in the symmetric case. There is no unsymmetric Kaniel-Paige-Saad convergence theory.

These realities suggest a framework in which we use the Arnoldi iteration idea with repeated, carefully chosen restarts and a controlled iteration maximum. We described such a framework in conjunction with the block Lanczos procedure in §10.3.7.

1A4AA If $\mathbf{B}_m \mathbf{A} = \mathbf{G}(\mathbf{A}, \alpha, \gamma)$ then $\mathbf{f} = \mathbf{G}(\mathbf{A}, \alpha, \gamma) \mathbf{x}$

Consider running Arnoldi for r_+ steps and then restarting the iteration with a new initial vector q_+ chosen from the span of the Arnoldi vectors q_1, \dots, q_m . Because of the Krylov correction (105.1), q_+ has the form

$$q_+ = p(\mathbf{A})q_1$$

for some polynomial of degree m . It is instructive to examine the action of $p(\mathbf{A})$ in terms of \mathbf{A} 's eigenvalues and eigenvectors. Assume for clarity that $\mathbf{A} \in \mathbb{R}^{n \times n}$ is diagonalizable and that $\mathbf{A}z_i = \lambda_i z_i$ for $i = 1:n$. If q_1 has the eigenvector expansion

$$q_1 = a_1 z_1 + \dots + a_n z_n,$$

then q_+ is a scalar multiple of

$$z = a_1 p(\lambda_1) z_1 + \dots + a_n p(\lambda_n) z_n.$$

Note that if $p(\lambda_\alpha) \gg p(\lambda_1)$, then relatively speaking q_+ is much richer in the direction of z_α than in the direction of z_1 . More generally, by carefully choosing $p(\lambda)$ we can design q_+ so that its component in certain eigenvector directions is emphasized while its component in other eigenvector directions is deemphasized. For example, if

$$p(\lambda) = c \cdot (\lambda - \mu_1)(\lambda - \mu_2) \cdots (\lambda - \mu_p) \quad (105.3)$$

where c is a constant, then q_+ is a unit vector in the direction of

$$z = c \cdot \sum_{k=1}^n a_k \left(\prod_{i=1}^p (\lambda_k - \mu_i) \right) z_k.$$

It follows that z_1 is deemphasized relative to z_α if λ_β is near to one of the "filter values" μ_1, \dots, μ_p and λ_α is not. Thus, the act of picking a good restart vector q_+ from $\mathcal{K}(\mathbf{A}, q_1, m)$ is the act of picking a filter polynomial that tunes out unwanted portions of the spectrum. Various heuristics for doing this have been developed based on computed Ritz vectors. See Saad (1980, 1984, 1992).

1A4APA 51 f3nmA α, γ }f - GfA

We describe an Arnoldi restarting procedure due to Sorensen (1992) that implicitly determines the filter polynomial (105.3) using the QR iteration with shifts. (See §7.52.) Suppose $\mathbf{H}_e \in \mathbb{R}^{n \times n}$ is upper Hessenberg, μ_1, \dots, μ_p are scalars, and the matrix \mathbf{H}_+ is obtained via the shifted QR iteration

$$\begin{aligned} \mathbf{H}(\mathbf{Q}) &= \mathbf{H}_e \\ \text{for } i &= 0:p \\ H^{(i-1)} - \mu_i I &= \mathbf{V} \quad (\text{Givens QR}) \\ H^{(i)} &= \mathbf{V} + \mu_i I \\ \text{end} \\ H_+ &= H^{(p)} \end{aligned} \quad (105.4)$$

Recall from §7.4.2 that each $H^{(i)}$ is upper Hessenberg. Moreover, if

$$V = V_1 \cdots V_p \quad (1055)$$

then

$$H_+ = V^T H_c V. \quad (1056)$$

The following result shows that the filter polynomial (1053) has a relationship to (1054).

Theorem 10.5.1. If $V = V_1 \cdots V_p$ and $R = R_p \cdots R_1$ are defined by (1054), then

$$VR = (H_c - \mu_1 I) \cdots (H_c - \mu_p I). \quad (1057)$$

Proof. We use induction, noting that the theorem is obviously true if $p = 1$. If $\tilde{V} = V_1 \cdots V_{p-1}$ and $\tilde{R} = R_{p-1} \cdots R_1$, then

$$\begin{aligned} VR &= V(WR)V\tilde{R} = V(H_c - \mu_1 I)\tilde{R} = V(V^T H_c \tilde{V} - \mu_p I)\tilde{R} \\ &= (H_c - \mu_p I)\tilde{V}\tilde{R} = (H_c - \mu_p I)(H_c - \mu_1 I) \cdots (H_c - \mu_{p-1} I), \end{aligned}$$

where we used the fact that $H^{(p-1)} = \tilde{V}^T H_c \tilde{V}$. \square

Note that the matrix R in (1057) is upper triangular and so it follows that

$$V(:, 1) = p(H_c)e_1$$

where $p(\cdot)$ is the filter polynomial (1053) with $c = 1/R(1, 1)$.

Now suppose that we have performed m steps of the Arnoldi iteration with starting vector q_1 . The Arnoldi factorization (1052) says that we have an upper Hessenberg matrix $H_c \in \mathbb{R}^{n \times m}$ and a matrix $Q_c \in \mathbb{R}^{n \times m}$ with orthonormal columns such that

$$AQ_c = Q_c H_c + r_c e_m^T. \quad (1058)$$

Note that $Q_c(:, 1) = q_1$ and $r_c \in \mathbb{R}^n$ has the property that $Q_c^T r_c = 0$. If we apply (1054) to H_c , then by using (1055) and (1056) the preceding Arnoldi factorization transforms to

$$AQ_+ = Q_+ H_+ + r_c e_m^T V \quad (1059)$$

where

$$Q_+ = Q_c V.$$

If q_+ is the first column of this matrix, then

$$q_+ = Q_+(:, 1) = Q_c V(:, 1) = c \cdot Q_c (H_c - \mu_1 I) \cdots (H_c - \mu_p I) e_1.$$

Equation (1058) implies that

$$(A - \mu I)Q_c e_1 = Q_c (H_c - \mu I) e_1$$

for any $\mu \in J$ and so

$$q_+ = c(A - \mu_1 I) \cdots (A - \mu_p I) Q_c e_1 = p(A) q_1.$$

This suggests the following framework for repeated restarting

Repeat:

With starting vector \mathbf{q}_j , perf $m-m$ steps of the Arnoldi iteration obtaining $\mathbf{Q} \in \mathbb{R}^{n \times m}$ and $\mathbf{H} \in \mathbb{R}^{m \times m}$.

Determine filter values μ_1, \dots, μ_p . (105.10)

Perf $m-p$ steps of the shifted QR iteration (105.4) obtaining the Hessenberg matrix \mathbf{H}_+ and the orthogonal matrix \mathbf{V} .

Replace \mathbf{q}_j with the first column of $\mathbf{Q}\mathbf{V}$.

However, we can do better than this. The orthogonal matrices $\mathbf{V}, \dots, \mathbf{V}_p$ that arise in (105.4) are each upper Hessenberg. (This is easily deduced from the structure of the Givens rotations in Algorithm 5.25.) Thus, \mathbf{V} has lower bandwidth p and so $V_{(m-l:m-p-1)} = 0$. It follows from (105.9) that if $j = m-p$ then

$$\mathbf{A}\mathbf{Q}_{+}(:, l_j) = \mathbf{Q}_{+}(:, l_j)\mathbf{H}_{+}(l_j, l_j) + \mathbf{V}_{lj}\mathbf{T}_{lj}$$

is a j -step Arnoldi decomposition. In other words, we are all set to perf m step $j+1$ of the Arnoldi iteration with starting vector \mathbf{q}_j . There is no need to launch the restart from step 1. This leads us to the following modification of (105.10):

With starting vector \mathbf{q}_j , perf $m-m$ steps of the Arnoldi iteration obtaining $\mathbf{Q} \in \mathbb{R}^{n \times m}$, $\mathbf{H} \in \mathbb{R}^{m \times m}$, and $\mathbf{T} \in \mathbb{R}^n$ so $\mathbf{A}\mathbf{Q} = \mathbf{Q}\mathbf{H} + \mathbf{T}_j$.

Repeat:

Determine filter values μ_1, \dots, μ_p .

Perf $m-p$ steps of the shifted QR iteration (105.4) applied to \mathbf{H} obtaining $\mathbf{H}_+ \in \mathbb{R}^{m \times m}$ and $\mathbf{V} = (\mathbf{v}_j) \in \mathbb{R}^{n \times m}$.

Replace \mathbf{Q}_j with the first j columns of $\mathbf{Q}\mathbf{V}$.

Replace \mathbf{H} with $\mathbf{H}_+(l_j, l_j)$.

Replace \mathbf{T}_j with $\mathbf{V}_{lj}\mathbf{T}_{lj}$.

Starting with $\mathbf{A}\mathbf{Q} = \mathbf{Q}\mathbf{H} + \mathbf{T}_j$, perf m steps $j+1, \dots, j+p = m$ of the Arnoldi iteration obtaining $\mathbf{A}\mathbf{Q}_m = \mathbf{Q}_m\mathbf{H}_m + \mathbf{T}_m$.

Set $\mathbf{Q} = \mathbf{Q}_m$, $\mathbf{H} = \mathbf{H}_m$ and $\mathbf{T} = \mathbf{T}_m$.

In light of our remarks in §105.2, the filter values μ_1, \dots, μ_p should be chosen in the vicinity of \mathbf{A} 's "unwanted" eigenvalues. In this regard it is possible to formulate useful heuristics that are based on the eigenvalues of the $m \times m$ Hessenberg matrix \mathbf{H}_+ . For example, suppose the goal is to find the three smallest eigenvalues of \mathbf{A} in absolute value. If $p = m-3$ and $\sigma(\mathbf{H}_+) = \{\lambda_1, \dots, \lambda_m\}$ with $|\lambda_1| \geq \dots \geq |\lambda_m|$, then it is reasonable to set $\mu_i = \lambda_i$ for $i = 1:p$.

The Arnoldi iteration with implicit restarts has many attractive attributes. For implementation details and further analysis, see Lehoucq and Sorensen (1996), Morgan (1996), and the ARPACK manual by Lehoucq, Sorensen, and Yang (1998).

1. matA = Tridiag(1, -2, 1); b = matA * f; tol = 1e-6; maxit = 100;

An alternative restart procedure due to Stewart (2001) relies upon a carefully ordered Schur decomposition of the Hessenberg matrix H_m that is produced after m steps of the Arnoldi iteration. Suppose we have computed

$$AQ_m = Q_m H + r/m$$

and that $m = j + p$, where j is the number of A 's eigenvalues that we wish to compute. Let

$$UH_m V = \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix}$$

be the Schur decomposition of A and assume that the eigenvalues have been ordered so that the eigenvalues of $T_{11} \in \mathbb{R}^{j \times j}$ are of interest and the eigenvalues of $T_{22} \in \mathbb{R}^{(m-j) \times (m-j)}$ are not. (For clarity we ignore the possibility of complex eigenvalues.) The Arnoldi decomposition above transforms to

$$AQ_+ = Q_+ T + r/m U$$

where $Q_+ = Q_m U$. It follows that

$$AQ_+(:, 1:j) = Q_+(:, 1:j)Tu + mT$$

where $T = U(m, 1:j)$. It is possible to determine an orthogonal $Z \in \mathbb{R}^{n \times j}$ so that $Z^T T_{11} Z$ is upper Hessenberg and $Z^T u = b$. (See P1052) It follows that

$$A(Q_+ Z) = (Q_+ Z)(Z^T T_{11} Z) + r_c(Z^T u)^T$$

is a j -step Arnoldi factorization. We then set Q_+ , H and r_j to be $Q_+ Z$, $Z^T T_{11} Z$ and AQ_+ respectively and perform Arnoldi steps $j+1$ through $j+p = m$. For more detailed discussion see Stewart (MAE, Chap. 5) and Watkins (FMC, Chap. 9).

1. matA = Tridiag(1, -2, 1); b = matA * f; tol = 1e-6; maxit = 100;

Another way to extend the symmetric Lanczos process is to reduce A to tridiagonal form using a general similarity transformation. Suppose $A \in \mathbb{R}^{n \times n}$ and that a nonsingular matrix Q exists such that

$$Q^{-1}AQ = T = \begin{bmatrix} \alpha_1 & \gamma_1 & & \cdots & & 0 \\ \beta_1 & \alpha_2 & \ddots & & & \vdots \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & & \\ 0 & \cdots & & f_{n-1} & & \mathbf{1}' \\ & & & & & \alpha_n \end{bmatrix}.$$

With the column partitionings

$$Q = [q_1 | \cdots | q_n],$$

$$Q^T T = \tilde{Q} = [\tilde{q}_1 | \cdots | \tilde{q}_n],$$

we find upon comparing columns in $AQ = QT$ and $A^T \tilde{Q} = \tilde{Q}T^T$ that

$$\begin{aligned} Aq_k &= \gamma_{k-1}q_{k-1} + \alpha_k q_k + \beta_k q_{k+1}, & \text{"op } Q \\ A^T \tilde{q}_k &= \beta_{k-1} \tilde{q}_{k-1} + \alpha_k \tilde{q}_k + \gamma_k \tilde{q}_{k+1}, & \text{fdo } Q \end{aligned}$$

for $k = 1, \dots, n-1$. These equations together with the biorthogonality condition

$$\tilde{Q}^T Q = I_n$$

imply

$$\alpha_k = \tilde{q}_k^T A q_k$$

and

$$\begin{aligned} \beta_k q_{k+1} &= r_k = (A - \alpha_k I)q_k - \gamma_{k-1}q_{k-1}, \\ \gamma_k \tilde{q}_{k+1} &= \tilde{r}_k = (A - \alpha_k I)^T \tilde{q}_k - \beta_{k-1} \tilde{q}_{k-1}. \end{aligned}$$

There is some flexibility in choosing the scale factors β_k and γ_k . Note that

$$1 = \tilde{q}_{k+1}^T q_{k+1} = (\tilde{r}_k / \gamma_k)^T (r_k / \beta_k).$$

It follows that once β_k is specified, then γ_k is given by

$$\gamma_k = \mathbf{r}^T \mathbf{r}_k / \beta_k.$$

With the "canonical" choice $\beta_k = \|r_k\|_2$ we obtain

q_1 , **ii** given unit 2-norm vectors with $\tilde{q}_1^T q_1 = 0$

$k = 0$, $q_0 = Q r_0 = q_1$, **io**= Q so $\tilde{q}_1 = q_1$

while ($r_k \neq 0$ and $\mathbf{rk}(Q)$ $\neq k$) **and** ($\mathbf{rk}(r_k) \neq 0$

$$\beta_k = \|r_k\|_2$$

$$\gamma_k = \tilde{r}_k^T r_k / \beta_k$$

$$q_{k+1} = r_k / \beta_k$$

$$\tilde{q}_{k+1} = \tilde{r}_k / \gamma_k$$

$$k = k + 1$$

$$\alpha_k = \tilde{q}_k^T A q_k$$

$$r_k = (A - \alpha_k I)q_k - \gamma_{k-1}q_{k-1}$$

$$\tilde{r}_k = (A - \alpha_k I)^T \tilde{q}_k - \beta_{k-1} \tilde{q}_{k-1}$$

end

(1051)

If

$$T_k = \begin{bmatrix} \alpha_1 & \gamma_1 & & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ \ddots & \ddots & \ddots & & \\ 0 & \cdots & & \beta_{k-1} & \gamma_{k-1} \end{bmatrix},$$

then the situation at the bottom of the loop is summarized by the equations

$$A[q_1 \dots q_k] = [q_1 \dots q_k] T_k + r_{k\text{ef}}, \quad (105.12)$$

$$AT[Q_1 \dots Q_k] = [Q_1 \dots Q_k] T + r_{k\text{ef}}. \quad (105.13)$$

If $T_k = Q$ then the iteration terminates and $\text{span}\{q_1, \dots, q_k\}$ is an invariant subspace of A . If $T_k = Q$ then the iteration also terminates and $\text{span}\{Q_1, \dots, Q_k\}$ is an invariant subspace of AT . However, if neither of these conditions is true and $r_k r_k = 0$ then the tridiagonalization process ends without any invariant subspace information. This is called serious breakdown. See Wilkinson (AEP, p. 39) for an early discussion of the matter.

hygi ha b 7\\$b skd se)S ttb ft\\$b

It is interesting to examine the serious breakdown issue in the block version of (105.1). For clarity assume that $A \in \mathbb{R}^{mn}$ with $n = rp$. Consider the factorization in which we want $\{Q_i I_r\}$

$$Q^T A Q = \begin{bmatrix} M_1 & & & & & & & O \\ B_1 & M_2 & & & & & & \vdots \\ \ddots & \ddots & \ddots & & & & & \\ \vdots & \ddots & \ddots & \ddots & & & & \\ O & \cdots & & & B_{r-1} & C_{r-1} & & M_r \end{bmatrix} \quad (105.14)$$

where all the blocks are $p \times p$. Let $Q = [Q_1 \dots Q_r]$ and $Q = [Q_1 \dots Q_r]$ be compatible partitionings of Q and Q . Comparing block columns in the equations $AQ = QT$ and $AT = QT$, we obtain

$$\begin{aligned} Q_{k+1} B_k &= A Q_k - Q_k M_k - Q_k C_{-1} = R_k \\ Q_{k+1} C_k &= A^T Q_k - Q_k M_k^T - Q_k B_k = S_k \end{aligned}$$

Note that

$$M_k = Q^T A Q_k$$

If $S_k^T R_k = C$ and $Q_{k+1} B_k E$ if E is nonsingular and we compute $B_k C_k \in$ so that

$$C_k^T B_k = S_k^T R_k,$$

then

$$Q_{k+1} = R_k B_k^{-1}, \quad (105.15)$$

$$Q_{k+1} = S_k C_k^{-1} \quad (105.16)$$

satisfy $Q_{k+1} Q_{k+1}^T = I_p$. Serious breakdown in this setting is associated with having a singular $S_k^T R_k$.

One way of solving the serious breakdown problem in (105.1) is to go after a factorization of the form (105.14) in which the block sizes are dynamically determined. Roughly speaking, in this approach matrices Q_{k+1} and Q_{k+1}^T are built up column

by column with special recursions that culminate in the production of a nonsingular $Q_{+}[Q_{k+1}]$. The computations are arranged so that the biorthogonality conditions $Q^T Q_{k+1} = O$ and $Q^T Q_{k+1} = O$ hold for $i = 1:k$.

A method of this form belongs to the family of look-ahead Lanczos methods. The length of a look-ahead step is the width of the Q_{k+i} and Q_{k+i} that it produces. If that width is one, a conventional block Lanczos step may be taken. Length-2 look-ahead steps are discussed in Parlett, Taylor, and Liu (1985). The notion of incurable breakdown is also presented by these authors. Freund, Gutknecht, and Nachtigal (1993) cover the general case along with a host of implementation details. Floating point considerations require the handling of "near" serious breakdown. In practice, each M_k that is 2-by-2 or larger corresponds to an instance of near serious breakdown.

Problems

P10.5.1 784...c..l86 1|88.8. .051dc.|8..!.. 8.|8.8..1.. 8b|8 1.4 8.84.8N1.1..
.c.l. .01.a.6.8 .. d.8..8-8.. |88.8. .!.. .88..!8 ...8 .!1.. r. |8 1.8.1 .84.8..
1.1.8.1. .0Pa.1

P10.5.3 N1.8. 8b.d.8 81.1. .84.8.rN6l14l18.8.14. 1.4. 81.8..c8. .0Pa.d
 ..8.d. .86. 6c.8. .8.81.c.1.d.48 1.r.18§ro.8

$$A = \begin{bmatrix} I & 2 \\ I & 2 \\ L & 5 \end{bmatrix}.$$

P10.5.4 ..d d8.8H ER^{n × n} R^j))..+n)| . . R^{j..]M R^{j,.. } |HR^{j+R^j, ..+|R^{j+n}}}}

P10.5.5 .I86 .I. .I. t7 ...8.c.l. rN8c.8....8. .88. .8. d.8.8.8..c.c..8... ...4..8 1..I8
.....8.14 4l.

Notes and References for §10.5

fe8-8.lc. .. c.d.8.8...c8. 1..l.6... cN1P58 8.18 .8. 48.d.8!8.c.8.8..8.
8b....8.c4 or...8.8!8.5 ..86... N1P5.. f. dc.. RP35.8.8.8 ..cM8d.8..8.18
... 04... 8 .k .848. 8.0. l 1. 18.86c.. 8.1 6. c4.8

8ro3.8.8.8. 000 2c.. 8.c4..N8.l8. .1... P1.....8 3.8..8.. 67Acta Numerica ii-
 ..(M,g-ra

.8t M16)6pA1pM1S!6StAlai A..tAltS 6pk

x), elpM1S64., R-4.8t NL6f6.1t MTlp6c6 3o, uiolAi6Mfpi8t 2M1ri6MfEBt 4Ail6R x6)tpIAlrt NLM.1tc SLQuarterly of Applied Mathematics 9, soMn.

2AASati i8t aiA)t 5l i8d StIolM.ctpi MTA.i6.A1 6c.1tcopiAilMpaSattS

7n 2AASr sg1R=45Al6AilMpMElpM1S1,4t8MS 5l .Mc.rilp) xl)tp tlc tpi, M B Ad)t, pahcc til6. 4Ail6ta .B II in. Alg. Applic. 34, n(n(.

7-*2AASrs*(g-R) 4.8t.ha8tI e.tltlAi6Mp t .8p6²rta ,l 2M1I6p)Mpahtctil6. x6)tpIA1rt NIM.3
1tcx SIMath. Comput. 12 o ag1

7. *2AASr4g(R14-lh1MI2ra.At* *4ti8MSaMp2r.tl.Mc.riolaSL* *SIAM J. Sci. Stat. Comput., i S*
sn11Msn1rn

otEt ltp.tg_5l_6c.16.li_lta1Ali6p) 6pi8t_elpnM1S6MpitRi_6n.1rStS

- q6IBGraea ly,1i, blkS)ls QcA))osra raPhuribA lSh(laoQGsec Out1 keyArt { SIAM J. Matrix Anal. Applic. 13, 7, oM7g .
- o- f1ftt.Mr.. A,S t,.. 2Mltpatp.s(.. c 4t-(1 iMpqt Op.ru [l c, 8J,m..i0t,iAlitS lltlAiMp(Tra SIAM J. Matrix Anal. Applic. 17, og(Bgnsc
- o1f. 4M)Ap .s(.. c 4l, otaiAli.p) iut elpM0S.4tiuMS 5l ftAl)t ,Mpa88-il.. x.3tpIA1r1 NIM3 0t8a(ThMath Comput. 65, sns7M871P
- ltl.tl)tp ApSe-2Jtp.t .s((o. c 4ucJ0m.mighaiAlitS elpM,S..uu Nrol..AiMp5l i. u 2umE uplIti gAp5l8A i.Mp(ThMath. Comput. 218, ..oM.g(P
- o.f. ft0Mr..Bt1 .., 2Mlcpat,(ApS.1 7Ap) .s((g. ARPACk Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods, 2ue4 N8.0.AimMpN(.0ASl0J.mA(Ne.
- e. 2iAiMJMrlMa(2AAApS-1 r .. ((g.c 4thpA8m. qu.y owniAli.p) MElu- t}J.S,M,ApS it u8J1. .i8taiAli-S el,M,Sm4tiuMSa SIAM J. Sci. Comput. 19, nndMn-,P
- o. f1 ItuMr.. .n11s.c 4u8J1.mi0hotaiAlitS elpM0Sm 4tiuMS 2r.aAAS .i-lA i.M (TrSIAM J. Matrix Anal. Applic. 23, , s ,nc
- q0t -lh,M1e2.0rl AJJ13A0 iMelpM0Sthacli,3 a JlMJMatSS
- C1.1 2it.Ali .n11s.. 4e -lhaMI 2.0rk0 Ml.i08 5l ftAl)t x.)t,LM.a --n(TraSIAM J. Matrix Anal. Applic., 23, .1sM,-
- qut lAiMpA0lpM0S.JlM.daa.pIM1It, iut aumEPsIm,ItlimStAupi.., MA8t.Mly elpM,S..a AJJ0mts CMo8cCl.r (A - μI)⁻¹(,ttS
- e. or t.s(g.-c 4oAi.M,A8-lh0MIe0)Ml.ca 5l x.)opIA08t .M8JriA.i.Mp=Thin. Alg. Applic. 58, 7(s ..-1,n
- e- or0t .s(((-.. 4oAi.M,A1-lh0MIe0)Ml.iuc a 5l ,3p,hccilm. xm)o9Icart 2lM.au8a uix+Ailmr NAmal Thm. Alg. Applic. 197, ng7 .n,
- en srut .s(((-.c 4q.t oAimMpA0 -#)MlC.c 5l ,3p,h88tilm. x.)tpIA1rt 2lM.1u8a u .M0J1tr 2umE5l sta0 4Ail..1a (ThBIT 34, s.,Bso)
- 4Ailmr Ep.imM,JlM,t8a iuAi .pIM,It 0Al)- nJAst 8Aulm.ta.Ap tASSlttaauSr,,p) lh1Mftq,3Ma mStq attS
- 71 2AqS .s((n.c 4e,c,h,a MElM8u -lh8MI 28.aJA.o e.JlMr.8AimMpriMi0t 1Ail.r xrJMpptimA0(Tra SIAM J. Numer. Anal. 29, n1(Mngc
- 41 2M.ulry ApS.1 18.m.u .s((o. c 4l -lh,MI 2r.nJA.t eJlMr.m8Ai.3paMhut 4Ail.r xrJMpptimA1 lJtAI Ml(ThSIAM J. Numer. Anal. 34, s(sB.(n.,
- 51 tbray. p(eracLtp.Arc ApS ft1-p.3.ptlcpAp .s((g.. 4,amp),M,3liuM)MpA0ftA,3Ma5t.iMla mp iut .M8J8iAiMp MEAilmr|p., MpTn SIAM J. Sci. Comput. 19, 7 ,)
- ,1 tt0 frMpMft1ft3Jt3(AfaNoirroAMll,.c 4.Mc 8AiMpMEnt xrJMpptpi.A1M#tAl)t 2JAlat 2yt. 28till.. 4Ailm.uuBTThAM J. Sci. Comp. 27, nogB#7-
4. xt18App ApS l.Cx,ai .n11..P 4e otaiAli-S -lh1MI 28.aJA.t 4ti0MS 5l iot xIA,8Ai.M,ME 4Ailmr|p.i.M,a (ThSIAM J. Numer. Anal. 44, n-gs n,1-)
- J. > a0O= oA=AO3- n3n=N2ppDXn = O3= Gra3=ywf= Z f 3Xa s=vO@ nXa = = 0
- ff= @#HThAM J. Sci. Comput. 27, s-7gSs-0)
- liutl elpM0S.lt,AitS JAJoin m,18StS
- qn 2r.y8t .so = 4qut elpM0S#ti0MS 5l ,MlTreA#Ail..ta (SIAM J. Matrix Anal. Applic. 15, -o(B-g=
- ..1 qMuApSft-1qltCutp .s((.. P4.A1.r0ACmMpENatrSMaJu.il h iut elpM0S.utlAi.M,(ThSIAM J. Sci. Comput. 17, sBs,)
- q1C. lm.i ApS ft1,1gt.iu.p .n11..) 4IAI)uB2.clt .M8J i}i.3p3ENatrSMnJt.ilA,a,) eoNe.- ApSx) SIAM J. Sci. Comput. 23, ,(sB.1.,
5. 2tlpApS u3(-xM8Ap(),S1 qM8w .n11 o.) 4NAIA00#lM0S. x.)u,a31I tla .mi0 xpp#4tS 2.A0Aa .ih IA COM.A0M8c8pm.AimMpAllAp)t 80,i (ThParallel Computing 33, ,ns, , -1 .
- qut rpah88til.. ftAp.3MdlM.taa ApS lt8AitS aMA0yASmStwAlt p..0h Jltat,itS m,S
- f1, NA0tii(t1 qAh0MI(ApS.z. S3N2D pDyS= = .- ya @anDyZ= 3K + = x eXX Ov= 0 Ha = E#H00 Comput. 44, s1,M.n-)
- o1.1 2t8pS (41 Criypt.wi(ApS.1 , .ui)A0 .so(7.. 4e, u8J0u8tpiAi.M, MEut ftMMMyetAS ftAp.3Me,)Mlmi.8 5l ,MpI-tlc .i.Ap 4Ailm.ta(SpAM J. Sci. Stat. Comput. 14, .7o.s,gc

Zeeo[®]SiO₂

T2Zdt (o(o)r))(O (=EP. • = T• = k(• (P E)=T))(kTP(EA xPNQxr T)(Q(• (OP KT• QN• D-
 nE •))nTE=.(4 ((= I((kTP(T)OE. • r (ONSIAM J. Numer. Anal. 19, -g,M,1,P
 t(ft(fM0thS 0L4hSC(2(CM1r. ApS1(2- CroiyFt.ui (o(o) x• e=I+QuESP=(P. • =EDsFDE(xEKN• (• E=) (•)I(• (T=) ((• HTkNkTksOT(N Numer. Algorithms 1, nsM-p
 C-e(Ct6ai (f0))))IDPks E (=).(B , =kTnk qTAT=(• Ei• .(N SIAM J. Matrix Anal.
 Applic. 12, 7,n.M 07)
 - fdt3lpayls u0AgApS2(IASM y(O((o)r))n• TDSeqO=(D (E=ED xOPD=(D (Es (=EP. • =
)(4))(• .TK(=NNumer. Algorithms 1, n.sMng-p
 I(-19 ApS e(..udMpM.M0M(dOf(r)))= (• ((=EP. • =)(T)P ESN(EP(0.(OEKD • E
 (E))Q((IKO(NParallel Comput. 17, o 7MoBg
 f-, NAltis (o(o))IDP KT• (k qSDs=(• E=)I• .(=EDTEs(=) IO=s=kT• EONAM J. Matrix
 Anal. Applic. 13, -o , (n
 1(CroiyFt.ui (o((o)T) ((- OOD#NO H(kNQ=I((kTP (=EP. • E• PO=EDQ=kOD- (• .
 .TKN+ fEkkx SIAM J. Matrix Anal. Applic. 13, (- .1p
 1rg Croiypo.0)((()r) ((- QOA#(O I• (kN((=I((QTP=eP. • = (E• PEEQ=kOD- (• .
 .TK(=I-Ekkx SIAM J. Matrix Anal. Applic. 15, O OZ- O
 z.) a (o(((r)(. • E)=)I• s• (kNQ=EP. •))(• ESN(- . x(E=I((kEsP (s(En=)E1(• r).N
 Math. Comput. 62, n1(-nn.
 -- 2r.y0t (o((()))(• l=(, • DTFP=kT• (k(4(E=I((kTP (=EP. •))(• .TKN (• Math. Comput.
 64, s,o,oBs,ggp
 k(p6q(O((o)r)#NOpEnOE(QPO 0.(4E)=T. (D=(P. • EON• Dr. (=EO(E=I((kTP xs(((E• r I
)4=(FSIAM J. Matrix Anal. Applic. 16, -7 ,n,p
 1---.0rS o-x(FSD0l.SAFSC-2-CM0)((()r))=E(xeO IDP KT• I• .(E= =EDPls)(0s I
 P=kT• E=kTnk=Pk• s• E=E)AM Review 37, ,sn M,71p
 .M9.r16p) o6tpIA0rot,Meopah99tid6. idlS6)Mpa@A,d6.ta 6nS6a.roaatSpS
 t(e(f6Flgt(Ct86)pAFls qq(Slaatrod ((()T)#N(QESPN))r .,kN• D. kNQ• E=I((kEsP
 qTAT=(• Ei)(En=-(O r .+I--SIAM J. Matrix Anal. Applic. 27, s,7rsoc

10.6 Jacobi-Davidson and Related Methods

We close the chapter with a brief discussion of the Jacobi-Davidson method, a solution framework that involves a mix of several important ideas. The starting point is a reformulation of the eigenvalue problem as a nonlinear systems problem, a maneuver that enables us to apply Newton-like methods. This leads in a natural way to a method of Jacobi that can be used to compute eigenvalue-eigenvector pairs of symmetric matrices that have a strong diagonal dominance. Eigenproblems of this variety arise in quantum chemistry and it is in that venue where Davidson (1975) developed a very successful generalization of the Jacobi procedure. It builds a (non-Krylov) nested sequence of subspaces and incorporates Ritz approximation. By restricting the Davidson corrections to the orthogonal complement of the current subspace, we arrive at the Jacobi-Davidson method developed by Sleijpen and van der Vorst (1996). Their technique does not require symmetry or diagonal dominance. Thus, in terms of abstraction, exposition in this section starts from the general, descends to the specific, and then climbs back out to the general. All along the way we are driven by practical, algorithmic concerns. Our presentation draws upon the insightful treatments of the Jacobi-Davidson method in Sorensen (2002) and Stewart (MAE, pp. 404–420).

We mention that full appreciation of the Jacobi-Davidson method and its versatility requires an understanding of the next chapter. This is because a critical step in the method requires the approximate solution of a large sparse linear system and preconditioned iterative solvers are typically brought into play. See §115

1t 5A1A TEEA iff f pG} oA b u.A s§1 ofTA

Consider then by-neigenvalueproblem $Ax = \lambda x$ and how we might improve an approximate eigenpair $\{x_c, \lambda_c\}$. Note that if

$$A(x_c + \delta x_c) = (\lambda_c + \delta \lambda_c)(x_c + \delta x_c),$$

then

$$(A - \lambda_c I) \delta x_c - \delta \lambda_c x_c = -r_c + \delta \lambda_c \cdot \delta x_c, \quad (10.6.1)$$

where

$$r_c = Ax_c - \lambda_c x_c$$

is the current residual. By ignoring the second-order term $\delta \lambda_c \cdot \delta x_c$ we arrive at the following specification for the corrections δx_c and $\delta \lambda_c$:

$$(A - \lambda_c I) \delta x_c - \delta \lambda_c x_c = -r_c. \quad (10.6.2)$$

This is an underdetermined system of nonlinear equations that has a very uninteresting solution obtained by setting $\delta x_c = -x_c$ and $\delta \lambda_c = 0$. To keep away from this situation we add a constraint so that if

$$\begin{bmatrix} x_+ \\ \lambda_+ \end{bmatrix} = \begin{bmatrix} x_c \\ \lambda_c \end{bmatrix} + \begin{bmatrix} \delta x_c \\ \delta \lambda_c \end{bmatrix}, \quad (10.6.3)$$

then the new eigenvector approximation x_+ is nonzero. One way to do this is to require

$$w^T x_+ = 1,$$

where $w \in \mathbb{R}^n$ is an appropriately chosen nonzero vector. Possibilities include $w = x_c$, which forces x_+ to have unit 2-norm, and $w = 1$, which forces its first component to be one. Regardless, if x_c is also normalized with respect to w , then

$$w^T \delta x_c = w^T (x_+ - x_c) = 0. \quad (10.6.4)$$

By assembling (10.6.2) and (10.6.4) into a single matrix-vector equation we obtain

$$\begin{bmatrix} A - \lambda_c I & -x_c \\ w^T & 0 \end{bmatrix} \begin{bmatrix} \delta x_c \\ \delta \lambda_c \end{bmatrix} = - \begin{bmatrix} r_c \\ 0 \end{bmatrix}. \quad (10.6.5)$$

This is precisely the Jacobian system that arises if Newton's method is used to find a zero of the function

$$F \left(\begin{bmatrix} x \\ \lambda \end{bmatrix} \right) = \begin{bmatrix} Ax - \lambda x \\ w^T x - 1 \end{bmatrix}.$$

Its solution is easy to specify:

$$\delta \lambda_c = \frac{w^T (A - \lambda_c I)^{-1} r_c}{w^T (A - \lambda_c I)^{-1} x_c}, \quad (10.6.6)$$

$$\delta x_c = -(A - \lambda_c I)^{-1} (r_c - \delta \lambda_c x_c). \quad (10.6.7)$$

Unfortunately, the required linear equation solving is problematic if A is large and sparse and this prompts us to consider the approximate Newton framework.

The idea behind approximate Newton methods is to replace the Jacobian system with a nearby, look-alike system that is easier to solve. One way to do this in our problem is to approximate A with a matrix M with the proviso that systems of the form $(M - \tilde{A})z = r$ are "easy" to solve. If $N = M - A$, then (1065) transforms to

$$\begin{bmatrix} M - \lambda_c I & -x_c \\ w^T & 0 \end{bmatrix} \begin{bmatrix} \delta x_c \\ \delta \lambda_c \end{bmatrix} = - \begin{bmatrix} r_c - N \cdot \delta x_c \\ 0 \end{bmatrix}.$$

Continuing with the approximate Newton mentality, let us throw away the inconvenient $N \cdot \delta x_c$ term that is part of the right-hand side. This leaves us with the system

$$\begin{bmatrix} M - \lambda_c I & -x_c \\ w^T & 0 \end{bmatrix} \begin{bmatrix} \delta x_c \\ \delta \lambda_c \end{bmatrix} = - \begin{bmatrix} r_c \\ 0 \end{bmatrix}, \quad (1068)$$

and the following computefriendly recipes for the corrections

$$8e^{-\frac{w^T(M - \tilde{A})^{-1}re}{w^T(M - \tilde{A})^{-1}xe}} \quad (1069)$$

$$8xe^{-\frac{1}{-(M - \tilde{A})^{-1}(re - 8Axe)}}. \quad (10610)$$

Of course, by cutting corners in Newton's method we risk losing quadratic convergence. Thus, the design of an approximate Newton strategy must balance the efficiency of the approximate Jacobian solution procedure with a possibly degraded rate of convergence. For an excellent discussion of this tension in the context of the eigenvalue problem see Stewart (MAE, pp. 364-9).

Now suppose

$$A = \begin{bmatrix} \alpha & & \\ & \ddots & \\ c & & \end{bmatrix}. \quad \text{Tr} \in \mathbb{R}, c \in \mathbb{R}^{n-1}, \text{Tr} \in \mathbb{R}^{(n-1) \times (n-1)} \quad (10611)$$

is symmetric and strongly diagonally dominant. Assume that α is the largest element on the diagonal in absolute value. Our ambition is to compute x (close to Tr) and λ_E so that

$$\begin{bmatrix} \alpha & c^T \\ c & A_1 \end{bmatrix} \begin{bmatrix} 1 \\ z \end{bmatrix} = \lambda \begin{bmatrix} 1 \\ z \end{bmatrix}. \quad (10612)$$

Because of the dominance assumption, there is no danger in assuming that the sought-after eigenvector is nicely normalized by setting its first component to 1. Partition $8e$, X_E , and x_+ as follows

$$8e = \begin{bmatrix} 8e_+ \\ 8e_- \end{bmatrix}. \quad X_E = \begin{bmatrix} 1 \\ G \end{bmatrix}. \quad X_+ = \begin{bmatrix} 1 \\ q \end{bmatrix}.$$

By substituting (10611) and $w = e_1$ into the Jacobian system (1065), we get

$$\left[\begin{array}{c|cc} \alpha - \lambda_c & c^T & -1 \\ c & A_1 - \lambda_c I & -z_c \\ \hline 1 & 0 & 0 \end{array} \right] \left[\begin{array}{c} \delta \mu_c \\ \delta z_c \\ \delta \lambda_c \end{array} \right] = - \left[\begin{array}{c} \alpha + c^T z_c - \lambda_c \\ (A_1 - \lambda_c I) z_c + c \\ 0 \end{array} \right],$$

i.e.,

$$\left[\begin{array}{c} z \\ (A_1 - \lambda_c I) z_c + c \\ \alpha + c^T z_c - \lambda_c \end{array} \right]. \quad (10613)$$

It is easy to verify that this is the Jacobian system that arises if Newton's method is used to compute a zero of

$$f\left(\begin{bmatrix} z \end{bmatrix}\right) = \begin{bmatrix} \alpha & c^T \\ c & A_1 - \lambda_c I \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \lambda \begin{bmatrix} 1 \end{bmatrix}.$$

If $A_1 = M_1 - N_1$, then (10613) can be rearranged as follows

$$(M_1 - \lambda_c I) z_+ = -c + N_1 z_c + \{\delta \lambda_c \cdot z_c + N_1 \cdot \delta z_c\},$$

$$\lambda_+ = \alpha + c^T z_+.$$

The Jacobi orthogonal component correction (JOCC) method is defined by ignoring the terms enclosed by the curly brackets and taking α to be the diagonal part of A_1 :

$$\lambda_1 = \alpha, z_1 = \mathbf{Dn} \mathbf{1}, \rho_1 = \|c\|_2, k = 1$$

while $\rho_k > \text{td}$

$$(M_1 - \lambda_k I) z_{k+1} = -c + N_1 z_k$$

$$\lambda_{k+1} = \alpha + c^T z_{k+1} \quad (10614)$$

$$k = k + 1$$

$$\rho_k = \|A_1 z_k - \lambda_k z_k + c\|_2$$

end

The name of the method stems from the fact that the corrections to the approximate eigenvectors

$$x_k = \begin{bmatrix} 1 \\ z_k \end{bmatrix},$$

are all orthogonal to e_1 . Indeed, it is clear from (10614) that each residual

$$r_k = (A - \lambda_k I)x_k$$

has a zero first component:

$$r_k = \begin{bmatrix} \alpha & c^T \\ c & A_1 \end{bmatrix} \begin{bmatrix} 1 \\ z_k \end{bmatrix} - \lambda_k \begin{bmatrix} 1 \\ z_k \end{bmatrix} = \begin{bmatrix} (A_1 - \lambda_k I) z_k + c \end{bmatrix}. \quad (10615)$$

Hence, the termination criterion in (106.14) is based on the size of the residual.

Jacobi intended this method to be used in conjunction with his diagonalization procedure for the symmetric eigenvalue problem. As discussed in §8.5 after a sufficient number of sweeps the matrix A is very close to being diagonal. At that point, the JOCC iteration (106.14) can be invoked after a possible $PAPT^T$ update to maximize the $(1,1)$ entry.

hyha hrb 7hsb etf Ttktb hs f)ktb

As with the JOCC iteration, Davidson's method is applicable to the symmetric diagonally dominant eigenvalue problem (106.12). However, it involves a more sophisticated placement of the residual vectors. To motivate the main idea, let V_k be the diagonal part of A and use (106.15) to rewrite the JOCC iteration as follows:

$$x_1 = e_1, \lambda_1 = Ax_1, r_1 = Ax_1 - \lambda_1 x_1, V_1 = [e_1], k = 1$$

while $\|r_k\| > \text{tol}$

Solve the residual correction equation

$$(M - \lambda_k I)\delta v_k = -r_k.$$

Compute an improved eigenpair $\{\lambda_{k+1}, x_{k+1}\}$ so $r_{k+1} \in \text{ran}(V_k)$:

$$\delta x_k = \delta v_k, x_{k+1} = x_k + \delta x_k, \lambda_{k+1} = \lambda_k + c^T \delta x_k$$

$$k = k + 1$$

$$r_k = Ax_k - \lambda_k x_k$$

end

Davidson's method uses Ritz approximation to ensure that r_k is orthogonal to e_1 and $\delta v_1, \dots, \delta v_{k-1}$. To accomplish this, the boxed fragment is replaced with the following:

Expand the current subspace $\text{ran}(V_k)$:

$$s_{k+1} = (I - v_k v_k^T) \delta v_k$$

$$v_{k+1} = s_{k+1} / \|s_{k+1}\|_2 \quad v_{k+1} = [v_k | v_{k+1}]$$

Compute an improved eigenpair $\{\lambda_{k+1}, x_{k+1}\}$ so $r_{k+1} \in \text{ran}(V_{k+1})^\perp$:

$$(V_{k+1}^T A V_{k+1}) t_{k+1} = \theta_{k+1} t_{k+1} \quad (\text{a suitably chosen Ritz pair})$$

$$\lambda_{k+1} = \theta_{k+1}, x_{k+1} = v_{k+1} t_{k+1}$$

(106.16)

There are a number of important issues associated with this method. To begin with, V is an n -by- k matrix with orthonormal columns. The transition from V_k to V_{k+1} can be effectively carried out by a modified Gram-Schmidt process. Of course, if k gets too big, then it may be necessary to restart the process using V_k as the initial vector.

Because $r_k = Ax_k - \lambda_k x_k = A(V_k t_k) - \theta_k (V_k t_k)$, it follows that

$$V_k^T r_k = (V_k^T A V_k) t_k - \theta_k t_k = 0$$

i.e., r_k is orthogonal to the range of V_k , as required.

We mention that the Davidson algorithm can be generalized by allowing M to be a more involved approximation to A than just its diagonal part. See Crouzeix, Philippe, and Sadkane (1994) for details.

hyga hsb 7esb Atfkf 1 etf 1te ktb 3t . §kdb

Instead of forcing the correction δx_c to be orthogonal to e_1 as in the Davidson setting, the *Jacobi-Davidson method* insists that δx_c be orthogonal to the current eigenvector approximation x_c . The idea is to expand the current search space in a profitable, unexplored direction.

To see what is involved computationally and to connect with Newton's method, we consider the following modification of (1.65):

$$\begin{bmatrix} A - \lambda_c I & -x_c \\ x_c^T & 0 \end{bmatrix} \begin{bmatrix} \delta x_c \\ \delta \lambda_c \end{bmatrix} = - \begin{bmatrix} r_c \\ 0 \end{bmatrix}. \quad (106.17)$$

Note that this is the Jacobian system associated with the function

$$F\left(\begin{bmatrix} x \\ \lambda \end{bmatrix}\right) = \begin{bmatrix} Ax - \lambda x \\ (x^T x - 1)/2 \end{bmatrix}$$

given that $x_c^T x_c = 1$. If x_c is so normalized and $\lambda_c = x_c^T A x_c$, then from (106.17) we have

$$\begin{aligned} (I - x_c x_c^T)(A - \lambda_c I)(I - x_c x_c^T)\delta x_c &= -(I - x_c x_c^T)(r_c - \delta \lambda_c x_c) \\ &= -(I - x_c x_c^T)r_c \\ &= -(I - x_c x_c^T)(Ax_c - \lambda_c x_c) \\ &= -(I - x_c x_c^T)Ax_c \\ &= -(Ax_c - \lambda_c x_c) = -r_c. \end{aligned}$$

Thus, the correction δx_c is obtained by solving the projected system

$$(I - x_c x_c^T)(A - \lambda_c I)(I - x_c x_c^T)\delta x_c = -r_c \quad (106.18)$$

subject to the constraint that $x_c^T \delta x_c = 0$.

In Jacobi-Davidson, approximate projected systems are used to expand the current subspace. Compared to the Davidson algorithm, everything remains the same in (106.16) except that instead of solving $(M - \lambda_c I)\delta v_k = -r_k$ to determine δv_k , we solve

$$(I - x_k x_k^T)(M - \lambda_k I)(I - x_k x_k^T)\delta v_k = -r_k, \quad (106.19)$$

subject to the constraint that $x_k^T \delta v_k = 0$. The resulting framework permits greater flexibility. The initial unit vector x_1 can be arbitrary and various Chapter 11 iterative solvers can be applied to (106.19). See Sleijpen and van der Vorst (1996) and Sorensen (2002) for details.

The Jacobi-Davidson framework can be used to solve both symmetric and non-symmetric eigenvalue problems and is important for the way it channels sparse $Ax = b$.

technology to the sparse $Ax = b$ problem. It can be regarded as an approximate Newton iteration that is "steered" to the eigenpair of interest by Ritz calculations. Because an ever-expanding orthonormal basis is maintained, restarting has a key role to play in the Arnoldi setting (§105).

1A5A4A TFoA Tf{id qmA nf fm E1A

We briefly discuss the trace-min algorithm that can be used to compute the k smallest eigenvalues and associated eigenvectors for the n -by- n symmetric-definite problem $\mathbf{A}\mathbf{x} = \lambda \mathbf{B}\mathbf{x}$. It has similarities to the Jacobi-Davidson procedure. The starting point is to realize that if $\mathbf{V}_0 \in \mathbb{R}^{n \times k}$ solves

$$\min_{V \in \mathbb{R}^{n \times k}} \text{tr}(V^T A V),$$

then the required eigenvalues/eigenvectors are exposed by $V^T A V = \text{diag}(\mu_1, \dots, \mu_k)$ and $A_{(:,j)} = \mu_j B V^T_{(:,j)}$, for $j = 1:k$. The method produces a sequence of V matrices, each of which satisfies $V^T B V = \cdot$. The transition from V to $V+$ requires the solution of a projected system

$$(I - Q_c Q_c^T) A (I - Q_c Q_c^T) Z_c = A V_c$$

where $Z \in \mathbb{R}^{n \times k}$ and $QR = BV$ is the thin QR factorization. This system, analogous to the central Jacobi-Davidson update system (106–19), can be solved using a suitably preconditioned conjugate gradient iteration. For details, see Sameh and Wisniewski (1989) and Sameh and Tong (2000).

Problems

P10.6.1 P8668... .8. .8..8 R085c1..c.. J.. ..d8. P8..8..8..

P10.6.2 1...8 1

$$\begin{bmatrix} T_D \\ \vdots \end{bmatrix}$$

1 xT20211 x(b1))uMA1A(-Ab Y-J+-

Notes and References for §10.6

fe8-88d8N8..84.c.8.8. 18 .8.18.. 8> .18418F6 .848..8.. n.86.. R1PRP0 cP00

8§3an8.8.8. R0000 2c.0..4.. R8 I8.. 151.8 P..8....8 3.8.08.. R7 *Acta Numerica* 11,
.N.d M

Eiinela CAp.le)iSAMhaagI eAT

9d. EiiReIaNT₂x.a.A.pAMipA₁g1gip1lai A. l. pAnLAp 91rAaiiglAe ALMMA, SlaeM
 na9nr)aiAapll niMrA₁g1gltCCApM₁RMnA. Comput. Phys. i Sgo(-
 o-f-1M)Ap ApStPI-I.Mii .s(g., , 4CtplA0l3AimMp9A16S9MpA₁8MS 5l .Mc.rimy xm)tpIA0
 rt9 M.IJA9t Ihccilm. 1Aidm,t9S, SIAM J. Sci. Stat. Comput. 7, gso gn,3
 pP09tp SNPbMl)tp9tpSAPs,1 I6c Mp9 s((1, 3 ,w9mp)8t lpt refloWtp Ilcmmp 00Mp.)rlAimMp
 .x.u , upitlA9i6MpB, Chem. Phys. Letters 169, 75 TI4
 l.wxnlMriaPnI 2v ..AaAMigR- ipnLiiReIaQ, nApM₁ALCSI plaoNrAaiiglA, niMrA
 2la,tCCApM₁ipMlalG. Comput. Phys. 101, IdT-INr

uL ZdtOre 2y,34r **SfOTUaSt1at** qpl+tra ueAtt(UDa(uled+F Uyue lheatSS) PuS Se({5Numer. Math. 64, .(,MnX.=

1. *dMr3tmBfc Nu.0m..tBapS1. IAS Ts((e-, (Kqutu IIS,Mpltu MS* SIAM J. Sci. Comput. 15, .nMo.-

e- *IidAuM.Mr0MtB ITraApS .. . l,9utdTX „c Ko3TrNtDMpSMp.p) 5d ftAd), I.AdatB Ihc8didl9 xl)tpIA0rt NdMTra08d. Comput. Appl. Math. 64, s(o-n ,-*

.ut *Md.),pA,A9MTraItAI.S,MSA A.uAda .pS*

CcftcCI0,.Ftp TrapS-e- λp St5MdaiTs((. , - Ke ,A9M.UtAI.2a3p uitdA,lMp iu35d ftlptAd xl)tpIA0rot2dM.0.8aB SIAM J. Matrix Anal. Applic. 1 -Bn--n,Tr

MaL0.9Ai.MpaApStRitpa.Mpa3 Mi.td.dMTra0tcaBS

C.ft.C. I0t.FtpBe.C.ft. fMMBtco. xMyyt8ABApS 2.e. p Std 5MdniT((.,) K,A9MTra0tcaBSnMp hJt 1,CuMSa5d CtpdA0m3t8, tp.dM.0u8a TrapS M0hpMc.A0tp.dM.0t8a BBIT 36, O.d2pM' U

1U a 1U<10B o+ yu= 00A =oAa =OH030AB2Zn R - r3Jd.a =oBox!G.Zn=

= •oA r= S3 ,H.v•30A=1= a =a0a10A0Aa < 3ryY0< .A0x0TNA &, ((

t.o . MyytcaAB.ftnCI0tBro.tpAp rroneApStd Mdai((g,- A9MTra0tcaBSnMpS6k

e0) Md.,u8a5d ,ut o,Sro9i.MpMEA,d.R Ntp9l0aB SIAM J. Sci. Computut. 20, (- ,n

NredTratpApS1,xn 2M9ua,tp.A9u tml1-, Tr4e 1) AIIM..ItAI.S,MS 5d IM0I.p).M8.atR Ih8s

ctCd.9 xm)tpIA0rtNdM.0t8aB SIAM J. Sci. Comput. 25, s. BX oC-

.ut *idA9t1c.p 8tiuMS m&t,A0,S lpS*

e. *IA8t. ApS,- .ap .tay. Tsogn, 1 4e gA9t 1lp.-.9 ,3pea)Md.,u8 5d i.c CtptdA0l9,2 xl),p3 dMTra0tcaBS* SIAM J. Numer. Anal. 19, sn-7M.n,(

e- *IA8tu ApS z. n= -ly 222R y10 Hn= 3i 3R y< A3x+tX= JY e2X0xA0g= 0Aa < 300 - 3y0=OxON Comput. Appl. Math. 123, . ;no,Tr*

Chapter 11

Large Sparse Linear System Problems

eePe
eeP
efR
eoSo
eeP
fePs

This chapter is about solving linear systems and least squares problems when the matrix in question is so large and sparse that we have to rethink our powerful dense factorization strategies. The basic challenge is to live without the standard 2-dimensional array representation where there is a 1:1 correspondence between matrix entries and storage cells.

There is sometimes sufficient structure to actually compute an LU, Cholesky, or QR factorization by using a sparse matrix data structure and by carefully reordering equations and unknowns to control the fill-in of nonzero entries during the factorization process. Methods of this variety are called *direct methods* and they are the subject of §11.1. Our treatment is brief, touching only some of the high points of this well-developed area. A deeper presentation requires much more graph theory and implementation-based insight than we can provide in these few pages.

The rest of the chapter is concerned with the *iterative method* framework. These methods produce a sequence of vectors that typically converge to the solution at a reasonable rate. The matrix A "shows up" only in the context of matrix/vector multiplication. We introduce the strategy in §11.2 through discussion of the "classical" methods of Jacobi, Gauss-Seidel, successive over-relaxation, and Chebyshev. The discrete Poisson problem from §4.8.3 is used to reinforce the major idea.

Krylov subspace methods are treated in the next two sections. In §11.3 we derive the method of conjugate gradients that is suitable for symmetric positive definite linear systems. The derivation involves the Lanczos process, the method of steepest descent, and the idea of optimizing over a nested sequence of subspaces. Related methods for

symmetric indefinite systems, general systems, and least squares problems are covered in §11.4.

It is generally the case that Krylov subspace methods are successful only if there is an effective *preconditioner*. For a given $Ax = b$ problem this essentially requires the design of a matrix M that has two properties. It must capture key features of A and it must be relatively easy to solve systems of the form $Mz = r$. There are several major families of preconditioners and these are surveyed in §11.5 and §11.6, the latter being dedicated to the mesh-coarsening/multigrid framework.

Reading Path

An understanding of the basics about LU, Cholesky, and QR factorizations is essential. Eigenvalue theory and notions of matrices have a prominent role to play in the analysis of iterative $Ax = b$ solvers. The Krylov methods make use of the Lanczos and Arnoldi iterations that we developed in Chapter 10.

Within this chapter, there are the following dependencies:

§11.2 . . . §11.3 . . . §11.4 . . . §11.5
 §11.6ⁱⁱ

§11.1 is independent of the others. The books by Axelsson (ISM), Greenbaum (IMSL), Saad (ISPLA), and van der Vorst (IMK) provide excellent background. The software "templates" volume LIN_TEMPLATES (1998) is very useful for its concise presentation of all the major iterative strategies and for the guidance it provides in choosing a suitable method.

11.1 Direct Methods

In this section we examine the direct method framework where the goal is to implement solution procedures that revolve around careful implementation of the Cholesky, QR, and LU factorizations. Central themes, all of which are detailed more fully by Davis (2006), include the importance of ordering to control fill-in, connections to graph theory, and how to reason about performance in the sparse matrix setting.

It should be noted that the band matrix methods discussed in §4.3 and §4.5 are examples of sparse direct methods.

hhghhhb 9§ 3§e 8tf tf sktb

Data structures play an important role in sparse matrix computations. Typically, a real vector is used to house the nonzero entries of the matrix and one or two integer vectors are used to specify their "location". The *compressed-column* representation serves as a good illustration. Using a dot-on-grid notation to display sparsity patterns, suppose

$$A = \begin{array}{|c|c|c|c|} \hline & \bullet & & \bullet \\ \hline & & \bullet & \\ \hline & & & \bullet \\ \hline & & \bullet & \\ \hline & \bullet & & \bullet \\ \hline & & \bullet & \\ \hline & & & \bullet \\ \hline \end{array} .$$

The compressed-column representation stores the nonzero entries column by column in a real vector. If A is the matrix, then we denote this vector by $A.val$, eg,

$$A.val = \boxed{a_{11} \ a_{41} \ a_{52} \ a_{23} \ a_{33} \ a_{63} \ a_{14} \ a_{44} \ a_{25} \ a_{55} \ a_{65}}.$$

An integer vector $A.c$ is used to indicate where each column "begins" in $A.val$:

$$A.c = \boxed{1 \ 3 \ 4 \ 7 \ 9 \ 12}.$$

Thus, if $k = A.c(j) : A.c(j+1) - 1$, then $v = A.val(k)$ is the vector of nonzero components of $A(:,j)$. By convention, the last component of $A.c$ houses $mz(A) + 1$ where

$$mz(A) = \text{the number of nonzeros in } A.$$

The row indices for the nonzero components in $A(:,1), \dots, A(:,n)$ are encoded in an integer vector $A.r$, eg,

$$A.r = \boxed{1 \ 4 \ 5 \ 2 \ 3 \ 6 \ 1 \ 4 \ 2 \ 5 \ 6}.$$

In general, if $k = A.c(j) : A.c(j+1) - 1$, then $Aval(k) = A(A.r(k), j)$.

Note that the amount of storage required for $A.r$ is comparable to the amount of storage required for the floating point vector $A.val$. Index vectors represent one of the overheads that distinguish sparse from conventional dense matrix computations.

hhhhlib 6E § tf Tktebtbb ePfkf tf Tkteb

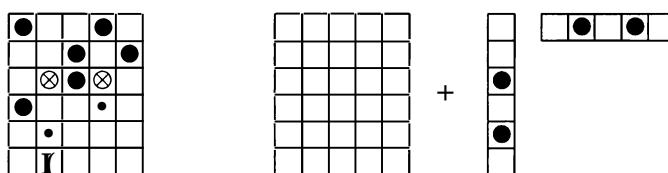
Consider the gaxy operation $y = y + Ax$ with A in compressed-column format. If $A \in \mathbb{R}^{m \times n}$ and the dense vectors $y \in \mathbb{R}^m$ and $x \in \mathbb{R}^n$ are conventionally stored, then

```
for j = 1:n
    k = A.c(j) : A.c(j+1) - 1
    y(A.r(k)) = y(A.r(k)) + A.val(k) * x(j)
end
```

(11.1)

overwrites y with $y + Ax$. It is easy to show that $2mz(A)$ fops are required. Regarding memory access, x is referenced sequentially, y is referenced randomly, and A is referenced through $A.r$ and $A.c$.

A second example highlights the issue of memory allocation. Consider the outer-product update $A = A + uv^T$ where $A \in \mathbb{R}^{m \times n}$, $u \in \mathbb{R}^m$, and $v \in \mathbb{R}^n$ are each stored in compressed-column format. In general, the updated A will have more nonzeros than the original A , eg,



for A in order to house the result. Moreover, the expansion of the vectors A_{val} and A_{ar} to accommodate the new nonzero entries is a nontrivial overhead. On the other hand, if we can predict the sparsity structure of $A + UV^T$ in advance and allocate space accordingly, then the update can be carried out more efficiently. This amounts to storing zeros in locations that are destined to become nonzero, e.g.,

$$\begin{aligned} A_{\text{val}} &= \boxed{\text{all } a_1 \quad 0 \quad \text{all } a_2 \quad \text{all } a_3 \quad \text{all } a_4 \quad 0 \quad a_{44} \quad 0 \quad \text{all } a_5 \quad \text{all } a_6}, \\ A_{\text{ar}} &= \boxed{1 \quad 3 \quad 5 \quad 8 \quad 12 \quad 15}, \\ A_{\text{r}} &= \boxed{1 \quad 4 \quad | \quad 3 \quad 5 \quad | \quad 2 \quad 3 \quad 6 \quad | \quad 1 \quad 3 \quad 4 \quad 5 \quad | \quad 2 \quad 5 \quad 6}. \end{aligned}$$

With this assumption, the outer product update can proceed as follows:

```

for  $\beta = 1:mz(v)$ 
     $j = vr(\beta)$ 
     $\alpha = 1$ 
    for  $f = Adj(j):Adj + j - 1$ 
        if  $O = mz(u) < Ar(f) = ur(f)$ 
             $A_{\text{val}}(f) = A_{\text{val}}(f) + u_{\text{val}}(O) \cdot v_{\text{val}}(f)$ 
             $O = O + 1$ 
        end
    end
end

```

(11.12)

Note that $A_{\text{val}}(f)$ houses a_1 and is updated only if UV is nonzero. The index O is used to reference the nonzero entries of U and is incremented after every access.

The overall success of a sparse matrix procedure typically depends strongly upon how efficiently it predicts and manages the fill-in phenomenon.

hhglib FetsTt"ob 3Tdtltohttb f)\$b.) ld\$edKb3ff leT ftf Tktb

The first step in the outer-product Cholesky process involves computation of the factorization

$$A = \begin{bmatrix} \alpha & v^T \\ v & B \end{bmatrix} = \begin{bmatrix} \sqrt{\alpha} & 0 \\ v/\sqrt{\alpha} & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & A^{(1)} \end{bmatrix} \begin{bmatrix} \sqrt{\alpha} & v^T/\sqrt{\alpha} \\ 0 & I \end{bmatrix} \quad (11.13)$$

where

$$A^{(1)} = B - \frac{vv^T}{\alpha}. \quad (11.14)$$

Recall from §4.2 that this reduction is repeated on the matrix $A^{(1)}$.

Now suppose A is a sparse matrix. From the standpoint of both arithmetic and memory requirements, we have a vested interest in the sparsity of $A^{(1)}$. Since B is sparse, everything hinges on the sparsity of the vector v . Here are two examples that dramatize what is at stake.

Example 1:

$$\begin{array}{|c|c|c|c|c|} \hline & \bullet & \bullet & \bullet & \bullet \\ \hline \bullet & & \bullet & & \\ \hline \bullet & & \bullet & & \\ \hline \bullet & & & \bullet & \\ \hline \bullet & & & \bullet & \\ \hline \bullet & & & & \bullet \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|} \hline & \bullet & & & \\ \hline \bullet & & \bullet & & \\ \hline \bullet & & \bullet & \bullet & \\ \hline \bullet & & \bullet & \bullet & \\ \hline \bullet & & \bullet & \bullet & \\ \hline \bullet & & \bullet & \bullet & \\ \hline \end{array} \begin{array}{|c|c|c|c|c|} \hline & \bullet & \bullet & \bullet & \bullet \\ \hline \bullet & & \bullet & \bullet & \bullet \\ \hline \bullet & & \bullet & \bullet & \bullet \\ \hline \bullet & & \bullet & \bullet & \bullet \\ \hline \bullet & & \bullet & \bullet & \bullet \\ \hline \bullet & & \bullet & \bullet & \bullet \\ \hline \end{array}$$

Example 2:

$$\begin{array}{|c|c|c|c|c|} \hline & & & & \bullet \\ \hline & & \bullet & & \\ \hline & \bullet & & \bullet & \\ \hline & & \bullet & & \\ \hline & & & \bullet & \\ \hline & & & \bullet & \\ \hline \bullet & \bullet & \bullet & \bullet & \bullet \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|} \hline & & & & \bullet \\ \hline & & \bullet & & \\ \hline & \bullet & & \bullet & \\ \hline & & \bullet & & \\ \hline & & & \bullet & \\ \hline & & & \bullet & \\ \hline \bullet & \bullet & \bullet & \bullet & \bullet \\ \hline \end{array} \begin{array}{|c|c|c|c|c|} \hline & & & & \bullet \\ \hline & & & \bullet & \\ \hline \bullet & \bullet & \bullet & \bullet & \bullet \\ \hline \end{array}$$

In Example 1, the vector \mathbf{r} associated with the first step is dense and that results in a full $A_{\text{L}\backslash\text{U}}$. All sparsity is lost and the remaining steps essentially carry out a dense Cholesky factorization. Example 2 tells a happier story. The first vector is sparse and the update matrix $A_{\text{L}\backslash\text{U}}$ has the same "arrow" structure as A . Note that Example 2 can be obtained from Example 1 by a reordering of the from PAPT where $P = I_n - \text{diag}(A)$. This motivates the *Sparse Cholesky challenge*:

The Sparse Cholesky Challenge

Given a symmetric positive definite matrix $A \in \mathbb{R}^{n \times n}$, efficiently determine a permutation p of $1:n$ so that if $P = I_n(:, p)$, then the Cholesky factor in $A_{(pp)} = pAPT = c^T c$ is close to being optimally sparse.

Choosing P to actually minimize $\text{nnz}(G)$ is a formidable combinatorial problem and is therefore not a viable option. Fortunately, there are several practical procedures based on heuristics that can be used to determine a good reordering permutation P . These include (1) the Cuthill-McKee ordering (2) the minimum degree ordering and (3) the nested dissection ordering. However, before we discuss these strategies, we need to present a few concepts from graph theory.

shhshshb o tEeeb ttib uEtxeTRkb

Here is a sparse symmetric matrix A and its adjacency graph \mathcal{G}_A :

$$A = \begin{array}{|c|c|c|c|c|} \hline & \bullet & \bullet & \bullet & \bullet \\ \hline \bullet & & \bullet & \bullet & \bullet \\ \hline \bullet & \bullet & & \bullet & \bullet \\ \hline \bullet & \bullet & \bullet & & \bullet \\ \hline \bullet & \bullet & \bullet & \bullet & & \bullet \\ \hline \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ \hline \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\ \hline \end{array} \quad \begin{array}{c} (11.15) \\ \begin{array}{c} \text{Graph } \mathcal{G}_A: \\ \text{Nodes: } 1, 2, 3, 4, 5, 6, 7, 8, 9 \\ \text{Edges: } (1,2), (1,3), (1,4), (2,3), (2,5), (3,4), (3,5), (3,6), (4,5), (4,6), (4,7), (5,7), (5,8), (6,7), (6,8), (7,8) \end{array} \end{array}$$

In an adjacency graph for a symmetric matrix, there is a node for each row number

entry a_{ii} is nonzero. In general, a graph $G(V, E)$ is a set of labeled nodes V together with a set of edges E , e.g.,

$$V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\},$$

$$E = \{(1, 4), (1, 6), (1, 7), (2, 5), (2, 8), (3, 4), (3, 5), (4, 6), (4, 7), (4, 9), (5, 8), (7, 8)\}.$$

Adjacency graphs for symmetric matrices are *undirected*. This means there is no difference between edge (i, j) and edge (j, i) . If P is a permutation matrix, then, except for vertex labeling, the adjacency graphs for A and $PAPT^T$ "look the same."

Node i and node j are *neighbors* if there is an edge between them. The *adjacency set* for a node is the set of its neighbors and the cardinality of that set is the *degree* of the node. For the above example we have

Node	1	2	3	4	5	6	7	8	9
Degree	3	2	2	5	3	2	3	3	1

Graph theory is a very powerful language that facilitates reasoning about sparse matrix factorizations. Of particular importance is the use of graphs to predict structure, something that is critical to the design of efficient implementations. For a much deeper appreciation of these issues than what we cover below, see George and Liu (1981), Duff, Erisman, and Reid (1986), and Davis (2006).

hhmhm 7\$b .Df)TRshf assb 6 n\$Tt"b

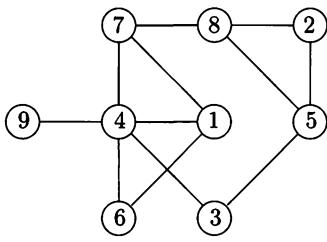
Because bandedness is such a tractable form of sparsity, it is natural to approach the Sparse Cholesky challenge by making $A = PAPT^T$ a "banded" as possible" subject to cost constraints. However, this is too restrictive a Example 2 in §11.1.3 shows Profile minimization is a better way to induce good sparsity in G . The *profile* of a symmetric $A \in \mathbb{R}^{n \times n}$ is defined by

$$\text{profile}(A) = n + \sum_{i=1}^n (i - f_i(A))$$

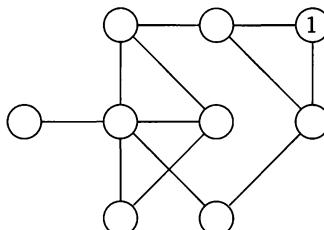
where the *profile indices* $f_1(A), \dots, f_n(A)$ are given by

$$f_i(A) = \min\{j : 1 \leq j - i \leq 0\}. \quad (11.16)$$

For the 9by-9 example in (11.15), $\text{profile}(A) = 37$. We use that matrix to illustrate a heuristic method for approximate profile minimization. The first step is to choose a "starting node" and to relabel it as node 1. For reasons that are given later, we choose node 2 and set $S_0 = \{2\}$:



Original G_A



Labeled: S_0

We then proceed to label the remaining nodes as follows

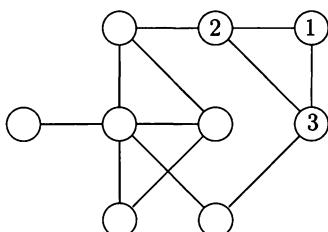
Label the neighbors of S0 Those neighbors make up S1

Label the unlabeled neighbors of nodes in S1. Those neighbors make up S2.

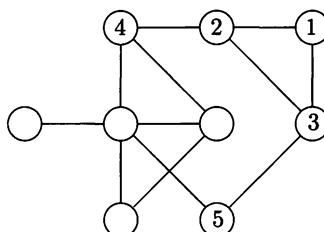
Label the unlabeled neighbors of nodes in S_2 . Those neighbors make up S_3 .

etc.

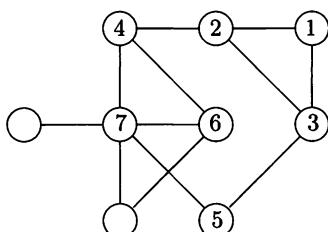
If we follow this plan for the example, then $S_1 = \{8, 5\}$, $S_2 = \{7, 3\}$, $S_3 = \{1, 4\}$, and $S_4 = \{6, 9\}$. These are the *level sets* of node 2 and here is how they are determined one after the other:



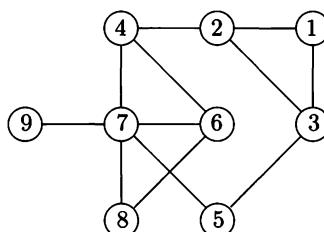
Labeled: S_O 81



Labeled: S_0 , S_1 , S_2



Labeled: S_0, S_1, S_2, S_3



Labeled: S_0, S_1, S_2, S_3, S_4

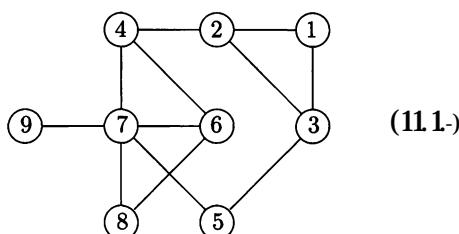
By "concatenating" the level sets we obtain the *Cuthill-McKee reordering*:

P.

2	.	5	I	3	1	4	6	9
---	---	---	---	---	---	---	---	---

S_0 S_1 S_2 S_3 S_4

Observe the band structure that is induced by this ordering



Note that $\text{profile}(A(\cdot, \cdot)) = 25$. Moreover, $A(\cdot, \cdot)$ is a 5by5 block tridiagonal matrix.

with square diagonal blocks that have dimension equal to the cardinality of the level sets S_0, \dots, S_4 . This suggests why a good choice for S_0 is a node that has "far away" neighbors. Such a node will have a relatively large number of level sets and that means the resulting block tridiagonal matrix $A(p,p)$ will have more diagonal blocks. Heuristically, these blocks will be smaller and that implies a tighter profile. See George and Liu (1981, Chap. 4) for a discussion of this topic and why the *reverse Cuthill-McKee ordering* (p(n-1)) typically results in less fill-in during the Cholesky process.

hhphga b 7\\$b hTtTqEqb e\\$ "S\\$b F. t\\$Tt'b

Another effective reordering scheme that is easy to motivate starts with the update recipe (11.14) and the observation that the vector v at each step should be as sparse as possible. This version of Cholesky with pivoting for $A = GGT$ realizes this ambition:

Step 1. $P \in \mathbb{R}^{n \times n}$

$k = 1:n-2$

Step 2. Choose a permutation $P_k \in \mathbb{R}^{(n-k+1) \times (n-k+1)}$ so that if

$$P_k A(k:n, k:n) P_k^T = \begin{bmatrix} \cdot & \cdot \\ \vdots & \mathbf{t} \end{bmatrix}$$

then v is as sparse as possible

Step 3. $P = \text{diag}(J_{k-1}, P_k) \cdot P$

11.18

Step 4. Reorder $A(k:n, k:n)$ and each previously computed G-column

$$A(k:n, k:n) \leftarrow P_k A(k:n, k:n) P_k^T$$

$$A(k:n, 1:k-1) \leftarrow P_k A(k:n, 1:k-1)$$

Step 5. Compute $G(k:n, k)$: $A(k:n, k) \leftarrow A(k:n, k) / \sqrt{A(k, k)}$

Step 6. Compute $A(k+1:n, k+1:n)$

$$A(k+1:n, k+1:n) = A(k+1:n, k+1:n) - A(k+1:n, k) A(k+1:n, k)^T$$

end

The ordering that results from this process is the *minimum degree ordering*. The terminology makes sense because the pivot row in step 5 is associated with a node in the adjacency graph $g(A(k:n, k:n))$ whose degree is minimal. Note that this is a greedy heuristic approach to the Sparse Cholesky challenge.

A serious overhead associated with the implementation of (11.18) concerns the outer-product update in Step 5. The memory allocation discussion in §11.12 suggests that we could make a more efficient procedure if we knew in advance the sparsity structure of the minimum degree Cholesky factor. We could replace Step 0 with

Step 0. Determine the minimum degree permutation P and represent $A(P, P)$ with "placeholder" zeros in those locations that fill in

This would make Steps 1–3 unnecessary and obviate memory requests in Step 5. Moreover, it can happen that a collection of problems need to be solved each with the same sparsity structure. In this case, a single Step 0 works for the entire collection thereby amortizing the overhead. It turns out that very efficient O procedures have been developed. The basic idea revolves around the intelligent exploitation of two facts that completely characterize the sparsity of the Cholesky factor in $A = GGT$.

Fact 1: If $j \neq i$ and g_j is nonzero, then $\sum_k g_{kj} b_k$ is nonzero assuming no numerical cancellation.

Fact 2: If g_{ik} and g_{jk} are nonzero and $k \neq j \neq i$, then $\sum_l g_{il} b_l$ is nonzero assuming no numerical cancellation. See Parter (1961).

The caveats about no numerical cancellation are required because it is possible for an entry in G to be "luckily zero". For example, Fact 1 allows for the formula

$$g_i = (j - \sum_k g_{kj}) / g_j$$

with the assumption that the summation does not equal j .

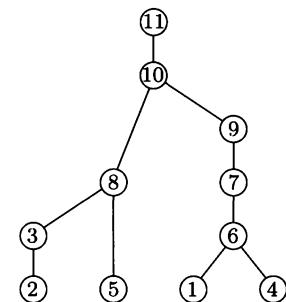
The systematic use of Facts 1 and 2 to determine G 's sparsity structure is complicated and involves the construction of an elimination tree (e-tree). Here is an example taken from the detailed presentation by Davis (2006 Chap. 4):

•											
•	•										
•	•										
•		•									
•		•									
•			•								
•			•								
•			•								
•			•								
•			•								
•			•								
•			•								

The matrix A

•											
•	•										
•	•										
•		•									
•		•									
•			0								
•			0								
•			0	0							
•			0	0							
•			0	0	0						
•			0	0	0						
•			0	0	0						

A's Cholesky factor



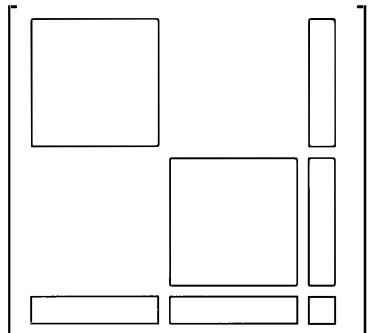
A's elimination tree

The "0" entries are nonzero because of Fact 2. For example, b_8 is nonzero because a_{88} and a_{18} are nonzero. The e-tree captures critical location information. In general, the parent of node i identifies the row of the first subdiagonal nonzero in column i . By encoding this kind of information, the e-tree can be used to answer various path-in-graph questions that relate to fill-in. In addition, the leaf nodes correspond to those columns that can be eliminated independently in a parallel implementation.

shpsptlb k\\$ ef\\$tbe\\$f\\$tbtbFet\\$eTt'eb

Suppose we have a method to determine a permutation P so that PAJ has the following block structure:

$$P_0 AP_0^T = \begin{bmatrix} A_1 & 0 & C_1 \\ 0 & A_2 & C_2 \\ \text{er} & \text{c} & \end{bmatrix} =$$



Through the schematic we are stating "A1 and A2 are square and roughly the same size and C1 and C2 are relatively thin". Let us refer to this maneuver as a "successful dissection". Suppose $P_1 \setminus A_1 P$ and $P_2 \setminus A_2 P$ are also successful dissections. If $P = \text{diag}(P_1, P_2, I)$, P_Q then

$$PAPT = \begin{bmatrix} D & D \\ c & :c \\ & D \\ & D \\ & D \\ & D \\ & cQ \end{bmatrix}$$

The process can obviously be repeated on each of the four big diagonal blocks. Note that the Cholesky factor inherits the recursive block structure.

In the end, the ordering produced is an example of a *nested dissection ordering*. These orderings are fill-reducing and work very well on grid-related elliptic partial differential equation problems; see George and Liu (1981, Chap. 8). In graph terms, the act of finding a successful permutation for a given dissection is equivalent to the problem of finding a good *vertex cut* of (A) . Davis (2006 pp. 128–130) describes several ways in which this can be done. The payoff is considerable. With standard discretizations, many 2-dimensional problems can be solved with $O(n^{3/2})$ work and $O(n \log n)$ fill-in. For 3-dimensional problems, the typical costs are $O(n^2)$ work and $O(n^{4/3})$ fill-in.

hhmh**b** u3e e**s**b 59b ttgb f)**s**bu3te e**s**b Stef b u Dte**s**b .e kf P**s**b

Suppose we want to minimize $\|Ax - b\|_2$ where $A \in \mathbb{R}^{m \times n}$ has full column rank and is sparse. If we are willing and able to form ATA , then we can apply sparse Cholesky technology to the normal equations $ATAx = Atb$. In particular, we would compute a permutation P so that $P(ATA)P^T$ has a sufficiently sparse Cholesky factor. However, a downside of forming the matrix ATA is that it is dense even though A is sparse.

A is sparse. (Consider the case when A has a dense row)

If we prefer to take the QR approach, then it still makes sense to reorder the columns of A , for if $APT = QR$ is the thin QR factorization of APT , then

$$P(A^T A)P^T = R^T R,$$

i.e., R^T is the Cholesky factor of $P(ATA)P^T$. However, this poses serious issues that revolve around fill-in and the Q matrix. Suppose Q is determined via Householder QR. Even though P is chosen so that the final matrix R is reasonably sparse, the intermediate Householder updates $A = H_k A$ tend to have high levels of fill-in. A corollary of this is that Q is almost always dense. This can be a showstopper especially if $n \gg n$ and motivates the *Sparse QR challenge*:

The Sparse QR Challenge

Given a sparse matrix $A \in \mathbb{R}^{n \times n}$, efficiently determine a permutation p of $1:n$ so that if $P = I_n(:,p)$, then the R -factor in the thin QR factorization $A(:,p) = APT = QR$ is close to being optimally sparse. Use orthogonal transformations to determine R from $A(:,p)$.

Before we show how to address the challenge we establish its relevance to the sparse least squares problem. If $APT = QR$ is the thin QR factorization of $A(:,p)$, then the normal equation system $ATb = ATAx_{LS}$ transforms to

$$P(A^T b) = (P(A^T A)P^T)P x_{LS} = R^T R P x_{LS}.$$

Solving the normal equations with a QR-produced Cholesky factor constitutes the *seminormal equations* approach to least squares. Observe that it is not necessary to compute Q . If followed by a single step of iterative improvement, then it is possible to show that the computed x_{LS} is just as good as the least squares solution obtained via the QR factorization. Here is the overall solution framework:

Step 1. Determine P so that the Cholesky factor of $P(ATA)P^T$ is sparse.

Step 2. Carefully compute the matrix R in the thin QR factorization $APT = QR$.

Step 3. Solve $RTy_0 = P(ATb)$, $Rz_0 = y_0$, $x_0 = P^{-1}z_0$

Step 4. Improve $r = b - Ax_0$, $RTy_1 = P(ATr)$, $Rz_1 = y_1$, $e = P^{-1}z_1$, $x_{LS} = x_0 + e$.

To appreciate Steps 3 and 4, think of x_0 as being contaminated by unacceptable levels of error due to the pitfalls of normal equations. Noting that $ATAx_0 = ATb - ATr$ and $ATAe = ATr$, we have

$$A^T A(x_0 + e) = A^T b - A^T r + A^T r = A^T b.$$

For a detailed analysis of the seminormal equation approach see Björck (1987).

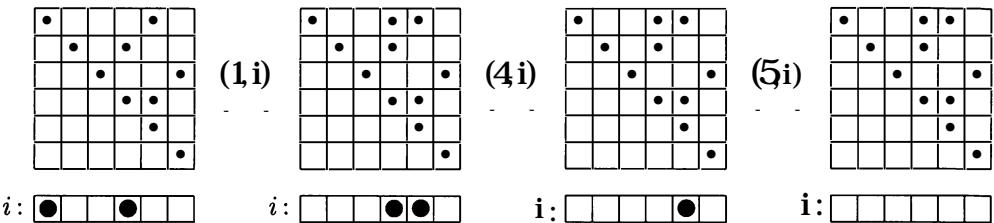
Let us return to the Sparse QR challenge and the efficient computation of R using orthogonal transformations. Recall from §5.2.5 that with the Givens rotation approach there is considerable flexibility with respect to the zeroing order. A strategy for introducing zeros into $A \in \mathbb{R}^{m \times n}$ one row at a time can be organized as follows:

```

for i = 2m
    for j = 1:min{i - 1, n}
        if aij ≠ 0
            Compute a Givens rotation G such that G [ ajj ] = [ × ]
            aij   0
            Update [ ajj ... ajn ] = G [ ajj ... ajn ]      (11.19)
            aij ... ain   1:n
        end
    end
end

```

The index i names the row that is being "rotated into" the current R matrix. Here is an example that shows how the j -loop oversees that process if $i > n$.



Notice that the rotations can induce fill-in both in R and in the row that is currently being zeroed. Various row reordering strategies have been proposed to minimize fill-in "along the way" to the final matrix R . See George and Heath (1980) and Björck (NMLS, p. 244). For example before (11.19) is executed, the rows can be arranged so that the first nonzero in each row is never to the left of the first nonzero in the previous row. Rows where the first nonzero element occurs in the same column can be sorted according to the location of the last nonzero element.

hhhhhb uEt. eS eI

The first step in a pivoted LU procedure applied to $A \in \mathbb{R}^{n \times n}$ computes the factorization

$$PAQ^T = \begin{bmatrix} \alpha & w^T \\ v & B \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ v/\alpha & I_{n-1} \end{bmatrix} \begin{bmatrix} \alpha & w^T \\ 0 & A^{(1)} \end{bmatrix} \quad (11.10)$$

where P and Q are permutation matrices and

$$A^{(1)} = B - \frac{1}{\alpha} vw^T. \quad (11.11)$$

In §3.4 we discussed various choices for P and Q . Stability was the primary issue and everything revolved around making the pivot element α sufficiently large. If A is sparse, then in addition to stability we have to be concerned about the sparsity of $A^{(1)}$. Balancing the tension between stability and sparsity defines the *Sparse LU challenge*:

The Sparse	Challenge
Given a matrix $A \in J^{n \times n}$, efficiently determine permutations p and q of $1:n$ so that if $P = I_n(:, p)$ and $Q = I_n(:, q)$, then the factorization $A(p, q) = PAQ\Gamma = LU$ is reasonably stable and the triangular factors L and U are close to being optimally sparse.	

To meet the challenge we must interpolate between a pair of extreme strategies:

- Maximize stability by choosing P and Q so that $\|A\|_F = \max_i \|A_{i,:}\|_1$.
- Maximize sparsity by choosing P and Q so that $\text{nnz}(A(\cdot))$ is minimized.

Markowitz pivoting provides a framework for doing this. Given a threshold parameter τ that satisfies $0 \leq \tau \leq 1$, choose P and Q in each step of the form (11.1.10) so that $\text{nnz}(A(\cdot))$ is minimized subject to the constraint that $\|A_{i,:}\|_1 \geq \tau \|A_{i,:}\|_F$ for $i = 1:n - 1$. Small values of τ jeopardize stability but create more opportunities to control fill-in. A typical compromise value is $\tau = 1/10$.

Sometimes there is an advantage to choosing the pivot from the diagonal, i.e., setting $P = Q$. This is the case when the matrix A is *structurally symmetric*. A matrix A is structurally symmetric if a_{ij} and a_{ji} are either both zero or both nonzero. Symmetric matrices whose rows and/or columns are scaled have this property. It is easy to show from (11.1.10) and (11.1.11) that if A is structurally symmetric and $P = Q$ then $A(\cdot)$ is structurally symmetric. The Markowitz strategy can be generalized to express a preference for diagonal pivoting if it is "safe". If a diagonal element is sufficiently large compared to other entries in its column, then P is chosen so that $(PAQ\Gamma)_{11}$ is that element and structural symmetry is preserved. Otherwise, a sufficiently large off-diagonal element is brought to the $(1, 1)$ position using a $PAQ\Gamma$ update.

Problems

P11.1.1 **NB** 4.8 N. If . rog. 8. 4dd8N. r 4.1. ... 82 = b 12Av nBn = n(EAx)
 $nAD(TnEA = Ax)D(laEF3n =$

P11.1.2 **b** 8.4.8 **r** 4. 4 **a** **r** **8b** **8dl** **1**

$$M = \begin{bmatrix} A_1 & C_1 \\ 0 & C_2 \\ 0 & C_3 \end{bmatrix},$$

- o. **iov0= ae=**):< < aN_{CP} Direct Methods for Sparse Matrices, 2.lmp)tle5tlbA)S ,t2 7Mly1
 2. N.aaApC3yts(g.-3 Sparse Matrix Technology, eACtcm. NltaaSt2 7Mly.
 u12trS e. xlmacApSp.- ot.C .s(g.. - Direct Methods for Sparse Matrices, lR5lC ,pmTratlamCh
 NltaaSftMpSPm.
- e cMlt lt_tpC CltACctpCC(ACCAI)tCalAC.C.MptlaS .lMTrpaCnC mpCMAp)t M..c. 0tctpCACmMp
 maartaSpCuw Ap tR.t0btPAppMCACt6.0.MA.uh mCt500M2mp)S
- .e. tATram1..3 Direct Methods for Sparse Linear Systems, 2ue4 Nr.bm.E.MpdSum0AS(mASvera
 .ut .pCtL0A .tC2tp)IA(C(tMlApCa.Alat cACl.R .Mc.rCACmMp a Quma Mpahc.M0m.
 ACmIm3AC.M(pCtCm.C700a p.tbh atC5lC. mpk
- p.ra21 ft.r .s(1. -4ut oM0tM.x0.c.pACmMp .p 2qlat xACMlmGnMpSIAM J. Matrix Anal.
 Applic. 11, N5/NTI4
 9vMhISAM.N. .hMRea.PaMIa.IPA)iMtAi.MnAIc)I.i .Platus SIAM J. Matrix Anal.
 Applic. 15, 7IT.4
 svAQPtAatiiao9vav.vPIf J.H. .3grlM2CB)Aat. 9gnCpalahAAAtMS)iMtA attP2
 CA.MhiMnaAtu6M J. Matrix Anal. Applic. 29uN75 NDI4
- lAgIn2Ag(AaA)AMta)MlQVAJIA.PhaUIJAr
- avav .irAMJ.4.InanCn- PnMlQbAi stCCAMhi.MPb16M J. Sci. Comput. 23,
 so(Msgs.3
 p.-1otmCApCp.e1 2.MCCh11..3 4otCr.mp)Cut.MCALfApC2mSCM.A2.Alat ,pahcctClm. 4ACl,RS3
 SIAM J. Matrix Anal. Applic. g1Mgns3
- x/mtpC.c.btcpCACmMp. C(t c.p mcrc Ct)ltt .CtAAlt C.a.raatC.pS
- N10. ectaCMh.S.e. tATramApCura2r1 .s((..3 4ep e..IMR.cACt 4mpmcrc tt)ltt llCtImp) eb)
 l.CucSLSIAM J. Matrix Anal. Applic. 17, dd7J.4
 av3Nei2Ptovlv.PgAMsUv,iMnClMaei 9v.2r f J.J. . .3 AlgICa3))MlbPCi.APaPCC
 EArMAMAMBmMmCu6M Trans. Math. Softw. 30, 7,7M7o.3
- xMlAp MTratlTra.t2JAMtbtwCarAltaS aut fe.ly ,4ft2S (A. 1 ... ApCA0aMS
- p.e. CtMl)t ApS4... 2tAC(.s(1. 3 42M0rCmMpAlat ftmptAftAa22rAlta NIM.0tca ,amp) CmTratpa
 oMCd.MpaSlin. Alg. Applic. 34, (- g73
 I. fe.ly ApS u12.tr1 .s(g1. 3 4e tmlt.C 4tCuMC5l Cut2M0rCmMpAlat ftmptAfttwC 22rnva
 NIM.0tcaSLLin. Alg. Applic. 34, -7M03
 e. CtMl)t ApSx.) .s(g 7.3 4lp oM2 ApS .M0rcltCtImp)a 5l 2.Alat fttwC2.rAlta NIM.btcaS3
 SIAM J. Numer. Anal. 20, 7n.-7-3
 4... 2tAC(.s(g.-3 4,rcil.40 4tCuMC5l ftAl)t 2.Alat ftmptAfttwC 2rAlta NIM.btcaSgSIAM J.
 Sci. Stat. Comput. 5, Tr.N5.
 n0w)CMabT. .s.iSngP.SaiUttPlt .2AI A.mle. sACnalMCgj.Plat 1M,A. sd.M-
 hMISgAcLin. Alg. Applic. 88/89, 5 ,d4
- a2AJAtndi i tSiMtA)MlaAelMIA ntigt tiSgAP&Ptal ttA9ar
- 9avEACCAlA9PtAati9lv.PgAMP-svP,iae9vaM,9If N.j 4.3 sI)AMalegSM-m
 1.sSiMtAiM.PHP12haruSIAM J. Matrix Anal. Applic. 20, on o,-
 ft.Clm)MnS.ra ttcc t0SApC 6.jftmn1lo.3 3 4NAlA002bc.M0.. xACMlm3AC5lMpAlat ft, 2,Cu
 2CAC.NmTrp)YCSIAM J. Sci. Comput. 3, sing(Ms7s-3
 ft.Clm)MlmpForC.0.tlCSApS41 .MapAlC.n11g.3 42hc.M0.. ApSxRAC2Clr.Crlt NltS..C.Mp5l .Alat
 CAraraAp0.cnprAC.Mp5mC(NAlCmMp)SgSIAM J. Matrix Anal. Applic. 30, s, n s,-3
- Frontal methods Alt A2Ah M.Ml)Ap3.p) MrCtleJlMCrCSACtaaMCuACutlt ar0C.p.c). 0tctpCACmMp
 .a.lm.(mfpApCtAClRM.tlAC.Mp5cAptrTratC(AGndmC..A0 ME(t aCApCJM.pCt5lMp)S s e
- C T 1B .22.03aw F= 2*x .x*x * x0=a t= = x x=-xTx1x= xap= =x • x x
 SIAM Review 34, gnMs1(3
 t.p. NmtltApSp1C1ft2.a .s((o. 3 42.Alat 4r0CmMpCAdpy otTratAbmfp) .CML3A6MgSIAM J.
 Matrix Anal. Applic. 18uNrNdi4
 av3Nei2ntiaJ .NEM f N. 4 .3 AlCSPaAeaPlaigtIIgndig IA.mJ IMr attPCA.Hn-
 s)imtAi.MnaAtu6M Trans. Math. Softw. 25uNIJ4

OnsAen)ansds reter)h)Aosate)alSt(,erSSah sr S)e s)ddlen In (ez
 OnPsAea ot laAhdck qy, 2-{ blidsh s-en)e VdHion lm ra oCote kors .5 ACM
Trans. Math. Softw. 16 5J5r 5Lx
 T4EM1ise wa iM N4.s pyAwgl@nianIaiNMClf stCCApMna IipMnAAt6ew
 52, -,M-013

xAl8h.A.tla Ml JAlA00t0 *eAlAaRMc.roiAimMtiaiAi Alt rn@ 2mi. pitltaimp)* mStAa mp.8roStk
 41.1 2tAiuux3.)w AlS f3.3 NthiMp((s i -4NAlA00t8)Mlmiuca5l 2.Alat ftmptA2haitcaWASIAM
 Review 33, IJ-, 7J4
 94IMP aSMpe l4 sayMAN, SAM, ..nryathiMiagAitAmlgAtOaplMPDnia.QQ M. J. Sci.
 Stat. Comput. 13, NN XNIM

IMi tSiMtA CipMhnlf alaenpnAtpn6nlsAMMgntntiae MAaiSAMISgACAT
 14M.MhCAiae 94.4 A.nt yN.dMAlsenpn18ICSApCipn1MsSiMtApMnaAt6M J.
Sci. Stat. Comput. 2, 7g-M7g3
 43 elmM8p8B ttccct8w ApS tr@23Ps(g(i 3 42M0Im@)Alat ftmptA2haitca 2miu2.Alat fA.2AlS
 dllMbwLSIAM J. Matrix Anal. Applic. 10, N.7.N.J4
 A4.4 wntamN.Jk4,aaMACAspigenpnBtpnCipn1Ms)iMtApMnaAt6Q J. Matrix Anal.
 Applic. 11, 5N1 5I14
 I4a4 wAMM34N.apli. ise .4a4 spA.iMjjJkM3urlMnpn,CAIC)IpnsSiMtApMnaAt6Q
 liaO 3SSMlbCipn1Ms)iMtApMnaAt6Q *Trans. Math. Softw.* 31, II-I7.4

11.2 The Classical Iterations

An iterative method for the $Ax = b$ problem generates a sequence of approximate solutions $\{x^{(k)}\}$ that converges to $x = A^{-1}b$. Typically, the matrix A is involved only in the context of matrix-vector multiplication and that is what makes this framework attractive when A is large and sparse. The critical attributes of an iterative method include the rate of convergence, the amount of computation per step, the volume of required storage, and the pattern of memory access. In this section, we present a collection of classical iterative methods, discuss their practical implementation, and prove a few representative theorems that illuminate their behavior.

hhhimhb 7\$b Atf kf Thittb ot DeeiuSTtsfb f \$tf Tkteb

The simplest iterative method for the $Ax = b$ problem is the *Jacobi iteration*. The 3by-3 instance of the method can be motivated by rewriting the equations as follows

$$\begin{aligned}x_1 &= (b_1 - a_{12}x_2 - a_{13}x_3)/a_{11}, \\x_2 &= (b_2 - a_{21}x_1 - a_{23}x_3)/a_{22}, \\x_3 &= (b_3 - a_{31}x_1 - a_{32}x_2)/a_{33}.\end{aligned}$$

Suppose $x^{(k-1)}$ is a "current" approximation to $x = A^{-1}b$. A natural way to generate a new approximation $x^{(k)}$ is to compute

$$\begin{aligned}x_1^{(k)} &= (b_1 - a_{12}x_2^{(k-1)} - a_{13}x_3^{(k-1)})/a_{11}, \\x_2^{(k)} &= (b_2 - a_{21}x_1^{(k-1)} - a_{23}x_3^{(k-1)})/a_{22}, \\x_3^{(k)} &= (b_3 - a_{31}x_1^{(k-1)} - a_{32}x_2^{(k-1)})/a_{33}.\end{aligned}\tag{11.2.1}$$

Clearly, A must have nonzeros along its diagonal for the method to be defined. For general n we have

for $i = 1:n$

$$x_i^{(k)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k-1)} - \sum_{j=i+1}^n a_{ij} x_j^{(k-1)} \right) / a_{ii} \quad (11.22)$$

end

Note that the most recent solution estimate is not fully exploited in the updating of a particular component. For example, $x_1^{(k-1)}$ is used in the calculation of $x_2^{(k)}$ even though $x_1^{(k)}$ is available. If we revise the process so that the most current estimates of the solution components are always used, then we obtain the *Gauss-Seidel iteration*:

for $i = 1:n$

$$x_i^{(k)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^n a_{ij} x_j^{(k-1)} \right) / a_{ii} \quad (11.23)$$

end

As with Jacobi, $a_{ii}, a_{i1}, \dots, a_{in}$ must be nonzero for the iteration to be defined.

For both of these methods, the transition from $x^{(k-1)}$ to $x^{(k)}$ can be succinctly described in terms of the strictly lower triangular, diagonal, and strictly upper triangular parts of the matrix A . Denote these three matrices by L_A , D_A , and U_A respectively, e.g.,

$$L_A = \begin{bmatrix} 0 & 0 & 0 \\ a_{21} & 0 & 0 \\ a_{31} & a_{32} & 0 \end{bmatrix}, \quad D_A = \begin{bmatrix} a_{11} & 0 & 0 \\ 0 & a_{22} & 0 \\ 0 & 0 & a_{33} \end{bmatrix}, \quad U_A = \begin{bmatrix} 0 & a_{12} & a_{13} \\ 0 & 0 & a_{23} \\ 0 & 0 & 0 \end{bmatrix}.$$

It is easy to show that the Jacobi step (11.22) has the form

$$a_1 x_1^{(k)} = x_1^{(k-1)} + b \quad (11.24)$$

where $a_1 = D_A$ and $b = -(L_A + U_A)$. On the other hand, the Gauss-Seidel step (11.23) is defined by

$$K_1 x_1^{(k)} = x_1^{(k-1)} + b \quad (11.25)$$

with $K_1 = I - D_A$ and $i K_1 = I - U_A$.

.., 1,1 .. r

The Jacobi and Gauss-Seidel methods have obvious block analogs. For example, if A is a 3by-3 block matrix with square, nonsingular diagonal blocks, then the system

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

can be rewritten as follows:

$$\begin{aligned} .1x_1 &= b_1 - A_{12} x_2 - A_{13} x_3 \\ A_{22} x_2 &= b_2 - A_{21} x_1 - A_{23} x_3 \\ .3x_3 &= b_3 - A_{31} x_1 - A_{32} x_2 \end{aligned}$$

From this we obtain the *block Jacobi iteration*

$$\begin{aligned} A_{11}x_1^{(k)} &= b_1 - A_{12}x_2^{(k-1)} - A_{13}x_3^{(k-1)}, \\ A_{22}x_2^{(k)} &= -A_{21}x_1^{(k-1)} - A_{23}x_3^{(k-1)}, \\ A_{33}x_3^{(k)} &= b_3 - A_{31}x_1^{(k-1)} - A_{32}x_2^{(k-1)}, \end{aligned}$$

and the *block Gauss-Seidel iteration*

$$\begin{aligned} A_{11}x_1^{(k)} &= b_1 - A_{12}x_2^{(k-1)} - A_{13}x_3^{(k-1)}, \\ A_{22}x_2^{(k)} &= b_2 - A_{21}x_1^{(k)} - A_{23}x_3^{(k-1)}, \\ A_{33}x_3^{(k)} &= b_3 - A_{31}x_1^{(k)} - A_{32}x_2^{(k)}. \end{aligned}$$

In contrast to the point versions of these iterations, a genuine linear system must be solved for $x_i^{(k)}$. These can be solved directly using LU or Cholesky factorizations or approximately solved via some iterative method. Of course, for this framework to make sense, the diagonal blocks must be nonsingular.

`hessianb uEfTf Tt"bttb kf Se"StSb`

Many iterative methods for the $Ax = b$ problem can be written in the form

$$Mx^{(k)} = Nx^{(k-1)} + b \quad \{11.26\}$$

where $A = M - N$ is a *splitting* and $x^{(0)}$ is a starting vector. For the iteration to be practical, it must be easy to solve linear systems that involve M . This is certainly the case for the Jacobi method where M is diagonal and the Gauss-Seidel method where M is lower triangular.

It turns out that the rate of convergence associated with {11.26} depends on the eigenvalues of the *iteration matrix*

$$G = M^{-1}N.$$

By subtracting the equation $Mx = Nx + b$ from {11.26} we obtain

$$M(x^{(k)} - x) = N(x^{(k-1)} - x).$$

Thus, there is a simple connection between the error at a given step and the error at the previous step. Indeed, if

$$e^{(k)} = x^{(k)} - x,$$

then

$$e^{(k)} = M^{-1}Ne^{(k-1)} = Ge^{(k-1)} = G^k e^{(0)}. \quad \{11.27\}$$

Everything hinges on the behavior of G^k as $k \rightarrow \infty$. If $\|G\| < 1$ for some choice of norm, then convergence is assured because

$$\|e^{(k)}\| = \|G^k e^{(0)}\| \leq \|G^k\| \|e^{(0)}\| \leq \|G\|^k \|e^{(0)}\|.$$

However, it is the largest eigenvalue of G that determines the asymptotic behavior of G^k . For example, if

$$G = \begin{bmatrix} \lambda & \alpha \\ 0 & \lambda \end{bmatrix},$$

then

$$G^k = \begin{bmatrix} \lambda^k & \alpha\lambda^{k-1} \\ 0 & \lambda^k \end{bmatrix}. \quad (11.28)$$

We conclude that for this problem $G^k \rightarrow 0$ if and only if the eigenvalue λ satisfies $|\lambda| \leq 1$. Recall from (7.1.1) the definition of spectral radius

$$\rho(G) = \max\{\lambda : \lambda \in \sigma(G)\}.$$

The following theorem links the size of $p(M^{-1}N)$ to the convergence of (11.26).

Theorem 11.2.1. Suppose $A = M - N$ is a splitting of a nonsingular matrix $A \in \mathbb{R}^{n \times n}$. Assuming that M is nonsingular, the iteration (11.26) converges to $\mathbf{x} = A^{-1}\mathbf{b}$ for all starting n -vectors \mathbf{x} if and only if $p(G) \leq 1$ where $G = M^{-1}N$.

Proof In light of (11.27), it suffices to show that $G^k \rightarrow 0$ if and only if $p(G) \leq 1$. If $G\mathbf{x} = \mathbf{0}$, then $G^k\mathbf{x} = \mathbf{0}$. Thus, if $G^k \neq 0$ then we must have $\|G\|_F \geq 1$, i.e., the spectral radius of G must be less than 1.

Now assume $p(G) \leq 1$ and let $G = QTQ$ be its Schur decomposition. If $D = \text{diag}(t_{11}, \dots, t_{nn})$ and $E = A - D$, then it follows from (7.3.15) that

$$\|G^k\|_2 \leq (1 + \mu)^{n-1} \left(\rho(G) + \frac{\|E\|_F}{1 + \mu} \right)^k$$

where μ is any nonnegative real number. It is clear that we can choose this parameter so that the upper bound converges to zero. For example, if G is normal, then $E = 0$ and we can set $\mu = 0$. Otherwise, if

$$\mu = \frac{2\|E\|_F}{1 - p(G)},$$

then it is easy to verify that

$$\|G^k\|_2 \leq \left(1 + \frac{2\|E\|_F}{1 - p(G)} \right) \frac{\|E\|_F}{2} \left(1 + \frac{p(G)}{2} \right)^k \quad (11.29)$$

and this guarantees convergence because $1 + p(G)/2 < 1$.

The 2by2 example (11.28) and the inequality (11.29) serve as a reminder that the spectral radius does not tell us everything about the powers of a normal matrix. Indeed, if G is nonnormal, then it is possible for G^k (and the error $\|\mathbf{x} - \mathbf{x}_k\|$) to grow considerably before decay sets in. The ℓ -pseudospectral radius introduced in §7.9.6 provides greater insight into this situation.

To summarize what we have learned so far, two attributes are critical if a method of the form (11.26) is to be of interest:

- The underlying splitting $A = -N$ must have the property that linear systems of the form $M^{-1}N = d$ are relatively easy to solve
- A way must be found to guarantee that $\rho(M^{-1}N) < 1$

To give a flavor for the kind of analysis that attends the second requirement, we state and prove a pair of convergence results that apply to the Jacobi and Gauss-Seidel iterations.

hhlihsb p1t" kttfb pksttf \$ttsb Atf kf 1bf \$tfskb

One way to establish that the spectral radius of the iteration matrix G is less than one is to show that $\|G\| \leq 1$ for some choice of norm. This inequality ensures that all of G 's eigenvalues are inside the unit circle. As an example of this type of analysis, consider the situation where the Jacobi iteration is applied to a strictly diagonally dominant linear system. Recall from §4.1.1 that $A \in \mathbb{R}^{n \times n}$ has this property if

$$\sum_{\substack{j=1 \\ j \neq i}}^n |a_{ij}| < 1 \quad i = 1:n.$$

Theorem 11.2.2. If $A \in \mathbb{R}^{n \times n}$ is strictly diagonally dominant, then the Jacobi iteration (11.2.4) converges to $x = A^{-1}b$.

Proof. Since $G_J = -D^{-1}(L + U)$ it follows that

$$\|G_J\|_\infty = \|D_A^{-1}(L_A + U_A)\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^n \left| \frac{a_{ji}}{a_{ii}} \right| = c_1$$

The theorem follows because no eigenvalue of A can be bigger than $\|A\|_\infty$. D

Usually, the "more dominant" the diagonal the more rapid the convergence, but there are counterexamples. See P11.2.3.

.. , 1, : § [1 , 1

A more complicated spectral radius argument is needed to show that Gauss-Seidel converges for matrices that are symmetric positive definite.

Theorem 11.2.3. If $A \in \mathbb{R}^{n \times n}$ is symmetric and positive definite, then the Gauss-Seidel iteration (11.2.5) converges for any $x^{(0)}$.

Proof. We must verify that the eigenvalues of $G_{GS} = -(D + L)^{-1}U$ are inside the unit circle. This matrix has the same eigenvalues as the matrix



where $L = D^{1/2} A D^{-1/2}$. If

$$-(I + L)^{-1} L V = \lambda v \quad v^H v = 1$$

then $-V H V = A(I + V L V)$. If $V L V = a + b$, then

$$|\lambda|^2 = \frac{a+b}{1+a+b} = \frac{a^2+b^2}{1+2a+a^2+b^2}$$

However, since $D^{1/2} A D^{-1/2} = I + L + L^T$ is positive definite, it is not hard to show that $0 < 1 + V L V + V L^T V = 1 + 2a$ and hence that $|\lambda| < 1$.

We mention that to bound $\rho(M_{V_{GS}})$ away from 1 requires additional information about A . The required analysis can be quite involved.

ss hima b e1efDeekb kb tb hktSdb .e kf d\$b

It is instructive to consider application of the Jacobi and Gauss-Seidel methods to the symmetric positive definite linear system

$$(I_{n_1} \otimes T_{n_2} + T_{n_1} \otimes I_{n_2}) u = b \quad (11.210)$$

where

$$T_m = \begin{bmatrix} 2 & -1 & \cdots & 0 \\ -1 & 2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & -1 \\ 0 & \cdots & -1 & 2 \end{bmatrix} \in \mathbb{R}^{m \times m}. \quad (11.211)$$

Systems with this structure arise from discretization of the Poisson equation on a rectangular grid, see §4.8.3. Recall that it is convenient to think of the solution vector as doubly subscripted. Associated with grid point (i,j) is the unknown $U(i,j)$. When the system is solved, the value of $U(i,j)$ is the average of the values associated with its north, east, south, and west "grid neighbors." Boundary values are known and fixed and this permits us to reformulate (11.210) as a 2-dimensional array averaging problem.

Given $U(0:n_1+1, 0:n_2+1)$ with fixed values in its top and bottom rows and fixed values in its leftmost and rightmost columns, determine $U(l:n_1, l:n_2)$ such that

$$U(l:j) = \frac{U(l:j-1) + U(l:j+1) + U(l-1:j) + U(l+1:j)}{4}$$

for $i = 1:n_1$ and $j = 1:n_2$

It is much easier to reason about Jacobi and Gauss-Seidel from this point of view. For example, the update

```

V=U
for i = 1:n1
    for j = 1:n2
        U(i,j) = (V(i-1,j)+V(i,j+1)+V(i+1,j)+V(i,j-1))/4
    end
end

```

corresponds to one step of Jacobi while

```

for i = 1:n1
    for j = 1:n2
        U(i,j) = (U(i-1,j)+U(i,j+1)+U(i+1,j)+U(i,j-1))/4
    end
end

```

is the corresponding update associated with Gauss-Seidel. The organization of both methods reflects the ultimate exploitation of matrix structure. *The matrix A is nowhere in sight!* We simply take advantage of the Kronecker structure at the block level and the 1-2-1 structure of the underlying tridiagonal matrices.

The array-update point of view for the model problem that we are considering makes it easy to appreciate why the Jacobi process is typically easier to vectorize and/or parallelize than Gauss-Seidel. The Jacobi update of $U(1:n1, 1:n2)$ is a matrix averaging

$$\frac{U(1:n1, 1:n2-1) + U(2:n1, 1:n2) + U(1:n1, 2:n2+1) + U(2:n1-1, 1:n2)}{4}.$$

The use the most-recent-estimate attribute of the Gauss-Seidel method makes it harder to describe the update at such a high level.

Now let us analyze the spectral radius $(M^{-1}N)$. Closed-form expressions for T_m eigenvalues permit us to determine this important quantity. Note that

$$T_m = 2I - E_m$$

where

$$E_m = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ 1 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 \\ 0 & \cdots & 1 & 0 \end{bmatrix}.$$

Since

$$A = I_{n1} \otimes T_2 + T_n \otimes I_{n2} = 4I_{n1n2} - (\frac{1}{n1} E_n) - (\frac{1}{n2} E_m), \quad (11.2.12)$$

the Jacobi splitting $A = M - N$ is given by

$$M = 4I_{n1n2}$$

$$N = (\frac{1}{n1} E_n) + (\frac{1}{n2} E_m).$$

Using results from our fast eigenvalue system discussion in §4.8.6 it can be shown that

$$\mathbf{S} = E_m S_m = D_m = \text{diag} \{\mu_1^{(m)}, \dots, \mu_m^{(m)}\} \quad (11.2.13)$$

where S_m is the sine transform matrix $[S_m]_{kj} = \sin(kj\pi/(m+1))$ and

$$\mu_k^{(m)} = 2 \cos \left(\frac{\pi k}{m+1} \right), \quad k = 1:m \quad (11.2.14)$$

It follows that

$$(S_{n_1} \otimes S_{n_2})^{-1} (M_j^{-1} N_j) (S_{n_1} \otimes S_{n_2}) = (I_{n_1} \otimes D_{n_2} + D_{n_1} \otimes I_{n_2}) / 4.$$

By using the Kronecker structure of this diagonal matrix and (11.2.14), it is easy to verify that

$$\rho(M_j^{-1} N_j) = \frac{2 \cos(7/(n_1+1)) + 2 \cos(7/(n_2+1))}{4}. \quad (11.2.15)$$

Note that this quantity approaches unity as n_1 and n_2 increase.

As a final exercise concerning the model problem, we use its special structure to develop an interesting alternative iteration. From (11.2.12) we can write $\mathbf{A} = M_x - N_x$ where

$$M_x = 4I_{n_1 n_2} - (I_{n_1} \otimes E_{n_2}), \quad N_x = (E_{n_1} \otimes I_{n_2}).$$

Likewise, $\mathbf{A} = M_y - N_y$ where

$$M_y = 4I_{n_1 n_2} - (E_{n_1} \otimes I_{n_2}), \quad N_y = (I_{n_1} \otimes E_{n_2}).$$

These two splittings can be paired to produce the following transition from $u^{(k-1)}$ to $u^{(k)}$:

$$\begin{aligned} M_x v^{(k)} &= N_x u^{(k-1)} + \mathbf{b} \\ M_y u^{(k)} &= N_y v^{(k)} + \mathbf{b} \end{aligned} \quad (11.2.16)$$

Each step has a natural interpretation based on the underlying partial differential equation, see §4.8.4. The first step corresponds to treating the north and south values at each grid point as fixed, while the second step corresponds to treating the east and west values at each grid point as fixed. The resulting iteration is an example of an *alternating direction iteration*. See Varga (1962, Chap. 7). Since

$$u^{(k)} - x = (M_y^{-1} N_y)(v^{(k)} - x) = (M_y^{-1} N_y)(M_x^{-1} N_x)(u^{(k-1)} - x)$$

it follows that $e^{(k)} = G^k e^{(0)}$ where

$$\begin{aligned} G &= (M_y^{-1} N_y)(M_x^{-1} N_x) \\ &= (4I_{n_1 n_2} - E_{n_1} \otimes I_{n_2})^{-1} (I_{n_1} \otimes E_{n_2}) (4I_{n_1 n_2} - I_{n_1} \otimes E_{n_2})^{-1} (E_{n_1} \otimes I_{n_2}). \end{aligned}$$

Using (11.2.13) and (11.2.14) it is easy to show that

$$\begin{aligned} (S_{n_1} \otimes S_{n_2})^{-1} G (S_{n_1} \otimes S_{n_2}) &= \\ (4I_{n_1 n_2} - D_{n_1} \otimes I_{n_2})^{-1} (I_{n_1} \otimes D_{n_2}) (4I_{n_1 n_2} - I_{n_1} \otimes D_{n_2})^{-1} (D_{n_1} \otimes I_{n_2}) \end{aligned}$$

is diagonal and that

$$\rho(G) = \frac{\cos(7/(n_1+1)) \cos(7/(n_2+1))}{(2 - \cos(7/(n_1+1))(2 - \cos(7/(n_2+1)))} < 1 \quad (11.2.17)$$

11t .cA IS.A }IA L1 bfinA IS.A

The Gauss-Seidel iteration is very attractive because of its simplicity. Unfortunately, if the spectral radius of $M_{GS}^{-1}N_{GS}$ is close to unity, then it may be prohibitively slow. To address this concern, we consider the parameterized splitting $A = M_\omega - N_\omega$ where

$$M_\omega = \frac{1}{\omega}D_A + L_A \quad N_\omega = \left(\frac{1}{\omega} - 1\right)D_A + U_A. \quad (11.2.18)$$

This defines the method of successive over-relaxation (SOR):

$$\left(\frac{1}{\omega}D_A + L_A\right)x^{(k)} = \left(\left(\frac{1}{\omega} - 1\right)D_A + U_A\right)x^{(k-1)} + b. \quad (11.2.19)$$

At the component level we have

for $i = 1:n$

$$x_i^{(k)} = w \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^n a_{ij}x_j^{(k-1)} \right) / a_{ii} + (1 - w)x_i^{(k-1)}$$

end

Note that if $w = 1$, then this is just the Gauss-Seidel method. The idea is to choose w so that $\rho(M_\omega, N_\omega)$ is minimized. A detailed theory on how to do this is developed by Young (1971). For an excellent synopsis of that theory, see Greenbaum (IMSL, p. 149).

Observe that x is updated top to bottom in the SOR step. We can just as easily update from bottom to top:

for $i = n:-1:1$

$$x_i^{(k)} = w \left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k-1)} - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \right) / a_{ii} + (1 - w) \cdot x_i^{(k-1)}$$

end

This defines the backward SOR iteration

$$\left(\frac{1}{\omega}D_A + U_A\right)x^{(k)} = \left(\left(\frac{1}{\omega} - 1\right)D_A + L_A\right)x^{(k-1)} + b. \quad (11.2.21)$$

Note that this update can be obtained from (11.2.19) simply by interchanging the roles of L and U .

If A is symmetric ($U_A = L_A^T$), then the symmetric SOR (SSOR) method is obtained by combining the forward and backward implementations of the update as follows

$$\left(\frac{1}{\omega}D_A + L_A\right)y^{(k)} = \left(\left(\frac{1}{\omega} - 1\right)D_A - L_A^T\right)x^{(k-1)} + b, \quad (11.2.22)$$

$$\left(\frac{1}{\omega}D_A + L_A^T\right)x^{(k)} = \left(\left(\frac{1}{\omega} - 1\right)D_A - L_A\right)y^{(k)} + b. \quad (11.2.23)$$

It can be shown that if

$$M_{SSOR} = \frac{\omega}{2-w} \left(D + L A \right) D^{-1} \left(D + U \right) \quad (11.224)$$

then the transition from $x^{(k-1)}$ to $x^{(k)}$ is given by

$$x^{(k)} = x^{(k-1)} + \frac{1}{\omega} (b - Ax^{(k-1)}). \quad (11.225)$$

Note that M_{SSOR} is defined if $0 < w < 2$ and that it is symmetric. It is also positive definite if A has positive diagonal entries. Here is a result that shows SSOR converges if A is symmetric and positive definite.

Theorem 11.2.4. Suppose the SSOR method (11.2.22) and (11.2.23) is applied to a symmetric positive definite $Ax = b$ problem and that $0 < w < 2$. If

$$M_\omega = \frac{1}{\omega} D_A + L_A, \quad N_\omega = \left(\frac{1}{\omega} - 1 \right) D_A - L_A^T, \quad G = M_\omega^{-T} N_\omega^T M_\omega^{-1} N_\omega,$$

then G has real eigenvalues $p(G) < 1$, and

$$(x^{(k)} - x) = G^k (x^{(0)} - x). \quad (11.226)$$

Proof. From (11.2.22) and (11.2.23) it follows that

$$\begin{aligned} y^{(k)} - x &= M^{-1} N(x^{(k-1)} - x), \\ x^{(k)} - x &= M^{-T} N^T(y^{(k-1)} - x). \end{aligned}$$

From which it is easy to verify (11.2.26). Since D is a diagonal matrix with positive diagonal entries, there is a diagonal matrix D_1 so $D = D_1$. If $L_1 = D_1^{-1} L D_1^{-1}$ and $G_1 = D_1 G D_1^{-1}$, then with a little manipulation we have

$$G_1 = (I + \omega L_1^T)^{-1} (I + \omega L_1)^{-1} ((1 - \omega)I - \omega L_1) ((1 - \omega)I - \omega L_1^T).$$

We show that if $E \cdot (G_1) > 0$: $\therefore < 1$. If $G_1 v = . v$, then

$$((1 - \omega)I - \omega L_1)((I - \omega)I - \omega L_1^T)v = \omega(I + \omega L_1)(I + \omega L_1^T)v.$$

This is a generalized singular value problem, see §8.7.4. It follows that ω is real and nonnegative. Assuming that $V E R$ has unit norm, it is easy to show that

$$\lambda = \frac{\|(1 - \omega)v - \omega L_1^T v\|_2}{\|v + \omega L_1^T v\|_2} = 1 - \frac{\omega(2 - \omega)}{\|v + \omega L_1^T v\|_2^2}. \quad (11.227)$$

To complete the proof, note that $1 + 2\omega L_1^T v = (D_1^{-1})^T A (D_1^{-1} v)$ and that this quantity is positive. By hypothesis, $\omega(2 - \omega) > 0$ and so we have < 1 .

The original analysis of the symmetric SOR method is in Young (1970).

11 mmqA T cA .D cA+ cA LoGo Po7}G: Acp [ELA

Another way to accelerate the convergence of certain iterative methods makes use of Chebyshev polynomials. Suppose the iteration $Mx^{(j+1)} = Nx^{(j)} + \mathbf{b}$ has been used to generate $x^{(1)}, \dots, x^{(k)}$ and that we wish to determine coefficients $\nu_j(k)$, $j = 0:k$ such that

$$y^{(k)} = \prod_{j=0}^k \nu_j(k) x^{(j)} \quad (11.228)$$

represents an improvement over $x^{(k)}$. If $x^{(0)} = \dots = x^{(k)} = x$, then it is reasonable to insist that $y^{(k)} = x$. If the polynomial

$$p_k(z) = \prod_{j=0}^k \nu_j(k) z^j$$

satisfies $p_k(1) = 1$, then this criterion is satisfied and

$$y^{(k)} - x = \prod_{j=0}^k \nu_j(k) (x^{(j)} - x) = \prod_{j=0}^k \nu_j(k) (M^{-1}N)^j e^{(0)} = p_k(G) e^{(0)}$$

where $G = M^{-1}N$. By taking norms in this equation we obtain

$$\|y^{(k)} - x\|_2 \leq \|p_k(G)\|_2 \|e^{(0)}\|_2. \quad (11.229)$$

This suggests that we can produce an improved approximate solution if we can find a polynomial $p_k(\cdot)$ that (a) has degree k , (b) satisfies $p_k(1) = 1$, and (c) does a good job of minimizing the upper bound.

To implement this idea, we assume for simplicity that G is symmetric. (There are ways to proceed if this is not the case; see Manteufel (1977). Let

$$S^T G S = \text{diag}(\lambda_1, \dots, \lambda_n) = \Lambda$$

be a Schur decomposition of G and assume that

$$-1 < \alpha \leq \lambda_n \leq \dots \leq \lambda_1 \leq \beta < 1 \quad (11.230)$$

where α and β are known estimates. It follows that

$$\|p_k(G)\|_2 = \|p_k(\Lambda)\|_2 = \max_{\lambda_i \in \lambda(A)} |p_k(\lambda_i)| \leq \max_{\alpha \leq \lambda \leq \beta} |p_k(\lambda)|.$$

The degree- k Chebyshev polynomial $c_k(\cdot)$ can be used to design a good choice for $p_k(\cdot)$. We want a polynomial whose value on $[\alpha, \beta]$ is small subject to the constraint that $p_k(1) = 1$. Recall from the discussion in §10.1.5 that the Chebyshev polynomials are bounded by unity on $[-1, +1]$, but that their value is very large outside this range. As a consequence, if

$$\mu = -1 + 2 \frac{1-\alpha}{\beta-\alpha} = 1 + 2 \frac{1-\beta}{\beta-\alpha},$$

then the polynomial

$$P_k(z) = G_k \left(-1 + 2 \frac{z - \alpha}{\beta - \alpha} \right) / G_k(\mu)$$

satisfies $p_k(l) = 1$ and is bounded by $1/lk(\mu)$ on $[a, l]$. From the definition of $P_k(z)$ and inequality (11.229) we see

$$\| y^{(k)} - x \|_2 \leq \frac{\| x - x^{(0)} \|_2}{|c_k(\mu)|}.$$

The larger the value of μ the greater the acceleration of convergence.

In order for the whole process to be effective, we need a more efficient method for calculating $y^{(k)}$ than (11.22). The retrieval of the vectors $x_0^{(k)}, \dots, x_{\ell}^{(k)}$ becomes an unacceptable overhead as k increases. Fortunately, it is possible to derive a three-term recurrence among the $y^{(k)}$ by exploiting the three-term recurrence that exists among the Chebyshev polynomials. Assume (for simplicity) that $a = -1$ in (11.23) and that we are given $x_0^{(k)}, \dots, x_{\ell-1}^{(k)}$. Here is how the process plays out when it is used to accelerate the iteration $Mx^{(k+1)} = Nx^{(k)} + b$:

$$c = \frac{1}{2}Q = \frac{1}{2}/$$

$$y^Q = x^Q, My^Q = Ny^Q + b, r^Q \text{ ta } b - Ay^Q, k=1$$

while $\text{trd} \geq \text{td}$

$$C_{k+1} = \{2/\rangle\}C_k - C_{k-1}$$

$$W_{\pm 1} \approx 1 \pm C_k i/C_{k+1}$$

$$Mz^{(k)} \leftarrow r^{(k)}$$

$$v^{(k+1)} = v^{(k-1)} + w_k t(v^{(k)} + z^{(k)}) - v^{(k-1)}$$

$$k = k + 1$$

$$r^{(k)} = b - Av^{(k)}$$

end

Note that $y^0 = x^0$ and $y^1 = x^1$, but that thereafter the x^k are not involved. For the acceleration to be effective we need good lower and upper bounds in $\{1123\}$ and that is sometimes difficult to accomplish. The method is extensively analyzed in Golub and Varga (1961) and Varga (1962 Chap. 5).

Problems

P11.2.1 .868 k.48.. R.8.. R₈48.8.... N. 0 ..10.8...4 ..8.R2..88 ..8...8..n

$$\mathbf{P11.2.2} \quad \text{...} \quad \mathcal{A} = M - N \left(\begin{pmatrix} I^m, D_1 \end{pmatrix}_1 \sigma^{-1} \cdot D_{\sigma} \sigma_{1,1}, \quad)D_{-1} \rho(M^{-1}N) \right) \in \mathbb{O} \Theta \mathbb{B} \quad \text{...}$$

P11.2.3 8 .. 7a.8e..... 8 .8 ..

$$A_1 = \begin{bmatrix} 1 & n \\ -1/2 & 1 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 1 & -3/4 \\ 1/s_n & 1 \end{bmatrix}.$$

$$P11.2.4 \quad \text{8.8} \quad A = T_{n_1} \circ I_{n_2} \circ R_{J_{n_2}} + I_{n_1} \circ R_{T_{n_2}} \circ R_{I_{n_2}} + I_{n_1} \circ R_{I_{n_2}} \circ R_{T_{n_2}}, \quad \begin{matrix} 1 \\ 3 \\ n \end{matrix}, \quad \begin{matrix} 1 \\ 3 \\ n \end{matrix}$$

ocSet =�AenSe $Au = b \cdot rCA \cdot LC2r = EA \cdot E2 \cdot E = DA \cdot (2Ax - AE2r) \cdot BE \cdot x$
P11.2.5 1 P .8r2.8. rr7lro8 r.20r.. 8.rb $A = I_{n_1} \otimes T_{n_2} \cdot T_{n_1} \otimes I_{n_2} \cdot D \cdot (D \cdot t) \cdot I$
 $\cdot U(i,j) \cdot A \cdot CA2AE2A(FU(i-1,j), U(i,j+1), U(i+1,j), 2 \cdot U(i,j-1), i(Hb - 2A$
 $2, d(I) = IA) = FA(AEA -) \cdot LC \cdot CU(i,j) - 2 \cdot a \cdot E \cdot 2AE2X(o \cdot nA \cdot C) \cdot DA \cdot C(E, 2 \cdot IA \cdot E, A$
 $rCAEA = h \cdot BE \cdot x = -) \cdot EE) \cdot ArAE \cdot Exaln = 1 \cdot F2 \cdot (- = barC(x = a = m) = (2AAu = b \cdot rCA)$
 $LC2 - rCA = AE2 \cdot a = (FCAl \cdot (2Ax - AE2r) \cdot BE \cdot x)$

P11.2.6 38.r 82.8.r.8. ...28. $(I_{n_1} \otimes T_{n_2} + T_{n_1} \otimes I_{n_2})x = b$ 12 - 3CA 1AE21E2x = (FrCA
- 1AE21 0 12E x.E CA, 1r 2L - 1AE21 0 12E x.E CA, 1r 2L , 1re 2Af e, 1f e)

P11.2.7 3.8.8 (11213) 2 (11214).

P11.2.8 3.8.8 (11215).

P11.2.9 3.8.8 (11217).

P11.2.10 3.8.8 (11224) 2. (11225).

P11.2.11 38..r 8. 281-2T2E x

$$A = \begin{bmatrix} 1 & \rho \\ -\rho & 1 \end{bmatrix}.$$

$$\rho(M_{\omega}^{-1}N_{\omega}) < O \text{ } \mathbb{L} \bullet \text{ } a > a < @V.x . N x (M_{\omega}^{-1}N_{\omega})? \text{ } 11 + D' \text{ } \mathcal{D}^{\sigma} \text{ } \overline{D} \text{ } \overline{G} \text{ } , .1 \text{ } \mathcal{D}_1 \text{ } y u$$

$$A = \begin{bmatrix} & 1 & 1 \end{bmatrix}$$

$\Rightarrow \mathbf{k} \quad :: \quad \mathbf{R}^{n \times n}, \text{Gy}^{\sigma'} u \quad (1.)_1 \quad -s: \quad ";$

P11.2.12 f8 6.2 28r..8.2r..8 2.8.8.l2r8. 81 = LCAFAA AT. i(E 2 h(xA1 E(AbP
I)= xAFnA;) nA- lAFnA)A 21E(x b2r 0 n

$$u'' + \sigma u' = 0 \quad 0 < x < 1$$

$$\text{LCAEA}(0) = 102x \quad u(1) = 10Ax1 \quad mC = 1A2x1 \quad (\text{CAx- LEA})AA - a2(0)$$

$$-u_{i-1} + 2u_i - u_{i+1} \quad R(u_{i+1} - u_{i-1}) = Q \quad i = \text{OE}$$

$$\begin{aligned} \text{LCAE} &= ah/2, \quad u_0 = 10, \quad u_{n+1} = 10e^\sigma, \quad i_1 = \frac{\sigma}{S_{1,1}}, \quad R = \frac{(1-e^{-\sigma})}{2}, \quad D^\sigma = 1, \quad \text{IC} = 1 \\ &= \text{AE21 E21 AE} = (\mathbf{F} M^{-1} N)^{1,1} \quad M = (\mathbf{A} + \mathbf{AT})/2, \quad N = (\mathbf{AT} - \mathbf{A})/2^2 \end{aligned}$$

P11.2.13 38..r 8. 8.8..2r8.

$$y^{(k+1)} = \omega(By^{(k)} + d - y^{(k-1)}) \quad y^{(k-1)}$$

$\text{Bx} + \text{d} = \text{Ax} + \text{e}^{(k)}$ $\text{Ax} = \text{Bx} - \text{d}$ $\text{e}^{(k)} = \text{Bx} - \text{Ax} - \text{d}$

$$\text{P11.2.14 J..8. } \underline{\underline{686}}\ldots \underline{\underline{.8}} \underline{\underline{.8}} \underline{\underline{-.8}} \underline{\underline{2}} \underline{\underline{-.8}} \underline{\underline{5}} \underline{\underline{2}} \underline{\underline{.8}} \underline{\underline{1}} \underline{\underline{2}} \underline{\underline{r}} \underline{\underline{l}} \underline{\underline{.8}} \ldots \underline{\underline{.8}} \underline{\underline{.0}} \underline{\underline{8}} \ldots \underline{\underline{r}} \underline{\underline{.4}} \underline{\underline{x}} \ldots \underline{\underline{b}} \underline{\underline{1}} \underline{\underline{b}} \underline{\underline{L}} \underline{\underline{C}} \underline{\underline{A}} \underline{\underline{E}} \underline{\underline{=}} \underline{\underline{R}}^{m \times n},$$

$$Mx_{i+1} = Nx_i + \mathbf{A}^T \mathbf{b}$$

$$\|v\|_W = (\sqrt{W})^{-1}, \quad v^m = W^{1/2} v^m, \quad (v^m)^{\sigma} = W^{1/2} v^m, \quad (v^m)^{\sigma \sigma} = W^{1/2} v^m.$$

$\|x_{LS} - x_{i+1}\|_W \leq \|x_{LS} - x_i\|_W \cdot k' + (\|CL(L) - b\|_A B_A) \|CA\|_A \|x_i\|_A$

$$M = \begin{bmatrix} A \\ F \end{bmatrix}$$

$$x = \text{HT}(x) \oplus d \oplus w \oplus v$$

$$\text{P11.2.15} \quad .2..8.8 \quad T \quad \mathbf{R}^{n \times n} \quad y(.,y) \quad y\mathcal{D}_i, "o\mathcal{G}_i \quad)_1 +, +1,- \quad)\mathcal{D}_i^t, i+1 t_{i+1}, i \quad \text{O.E } i = 1 \dots 1$$

ZAr' sAsAhe v(ot2has bns D E $\mathbb{R}^{n \times n}$, n , (S $\text{DTD}^{-1} \mathbf{J} \otimes \mathbf{exT} \mathbf{O} = \mathbf{In}_{\mathbb{R}^n} = \mathbf{o} \mathbf{J} \mathbf{O} =$
 $t < - \mathbf{J} \mathbf{A} \mathbf{J} = \mathbf{0} \mathbf{e} \mathbf{Z} = \mathbf{0} \mathbf{.} \mathbf{O} = \mathbf{.} \mathbf{.} \mathbf{.} \mathbf{u}_1, \mathbf{.} \mathbf{.} \mathbf{.} \mathbf{u}_n:$

$$-u_{i-1} - 2u_i + u_{i+1} - \frac{\sigma h}{2}(u_{i+1} - u_i) = a \quad : \quad i = 1:n.$$

yooo wO a. $u_{n+1} = \beta$, σ_1 Q iae h₁ O r aeAMniilaelpillatia pm1t pM1elttPA CSA
 $tCC1pM1-stAr,iTdTl.iA 1MCs,iAM$ pAlAlaiplApmpa 1S1p1Mp1ipM1b4

Notes and References for

fig.8..1es..8..8.. 8≥8 ...1es 1..1 .8 18.688N.88..! Rn16d5 d@8.. .8≥8
 r..3rob1.8.l.8.

71na.... 4d80a Matrix Iterative Analysis, Nltpi69tre2AA1S p)At.M3Sl6s A.S.p-
 $t-11$ 1Mrop), (o. .=Iterative Solution of Large Linear Systems, e9ASd-69 NltaaSt. 13 lyp
 $ft-e3A)tcAp$ A,S $t-11Mrop$, (gs. =Applied Iterative Methods, e9ASTc69 NltaaSt. 1Mly3
 $-2A9:roa9$, ((- .=Iterative Solution of Large Sparse Systems of Equations, 1.15 p)ule5t0A)S t.
 1Ml:1

ea o ctpi6MptSS1Mrop), (os. uw i.t cMai 9Mc.ltutpa6It iltAi-tpi M. iullo ct iuMS3.ut
 $M.e7i M.llo iudMlh6aiM$)ro6t roatl6p9uMMa6p)o lt0ARAi5MAlActitl w. €XX =XXX X u X
 $= = = XX = axF=xuXX=xuX =+= = XXXXXX+xe+$

,O C Q, Q == XX€=+=·a = QnWva=2XX X X X U=XX F X XX=iQ= Xr= -x=xx=xQ= x x
 O E+ += n XX = xamXX C * Nm= Mm=12Xn, -nnn=
 $t111$ 1Mrop), (o1. = 4MpItl)tp9d NLM.dli5taM But Ihc-til69 ApS,phccitl69 lltle 0ARAi6Mt
 1tiuMSaSLMath. Comput. 24, o(7Mglo.

$t-11$ 1Mrop), (on = 4Ctp,LA053Ai6MptEM.tlih e ApS.Mpa5aitpillSt l6p)SLSIAM J. Numer. Anal.
 9...r ,7 2k

143M2.1gi 1eAtNT,A., a i Al+ApM. 18)Apl. L,l maqpmAmAlNfAlat1tpAaMeAM 1M
 htpplAEA,alpApM11AtuBer. Math. 12, (, .1=

e3 orout, s(o., 4llo 1diuMSa 5l iut x6)tpI,0rot NLM.Atc .6iu ftAl)t I.Alt 1Ail69ta SLMath.
 Comput. ngt, o s11

ft3eSAcA Ap23 ,MISAp, (g... 4uall0 .M0Mf06pS SLAM J. Sci. Stat. Comput. 7, -(1M,1.c
 11 6tl-Apt ApSo-I-5Al)A , ((1 . 4uauit l. ifcAA_w a OX x ox €XX = , x XXXX-LinX
 Alg. Applic. 182, n,o noo,

21 ftra.((. 4iA6l1Ail69ta ApS.uu5l Ctpul A63Ai5Mpt1 e..0 69Ai6Mpa uitMi6It1tiuMSa uS
 Ctp1A063Ai5Mpt1 Iro99taa6It1ltdAARAi6MptiuMS SLSIAM J. Numer. Anal. 37, . so=

ep ApA0ha6dE uit.hatI at-6re6itlAi6I9tiu3S A.oAa 6pS

C-2-CM1roApSopI15Al)A , (.s=4.uu.hautI It96reiwAI6I1 1tiu Ma,9ro99cla6It1lco0ARAiMmp
 uitlAi6It1oi.MSaS ApSIt9MptellStl 069uAlSaMpitlAi6I0tiuMSaS2Alia uApSuSNumer. Math.
 3, .-o-s, .B.,o Ms.gc

.u i.Mly 6a.lid-6atS Mpiut arc.i6MpiuAi iut ropStlA 6p1Ai53pcAil5R uAnltAA6t)tpIA1rota3
 2M. iM.lM9dos.u1p iu6gapMSiu1 9wt 686a9rooatSpS

.pe1 1ApitrostA , (oo. = 4.ut .9ut.h9uuI uitlAi63, 5l ,Mpa-h-ctil69 ft6d2l Ihait-aSL Numer.
 Math. 28, 11M7noc

13 5olcAppApS.n ,5tiuA pctl, (gl.. 4lp iut .Mpa1ro9i6Mpt-5c6itlAi6It 1diu MBB SLSIAM
 J. Numer. Anal. 20, s.,1 .sl.

.n ,mtiuAc-tl ApSo3Ip5Al)A , (g7. = 4.ut epA0ha6dM.yeait uitlAi6It1tiuMSa 5l ft5ptAllhaitca
 ELMcIrocc A.605ihutMlhSLNumer. Math. 41, o oMn1.

C323CM1roApS13 1ltliMp,s(gg.1 4.ut .Mpt1)tp9t MuptRA9i uit.hautI ApS om9uAlSaMtitlAi6
 1tiuMSa 5l IM1I6p)ft5ptAllhaitca SLSIUM. Math. 53, ,os ,1

t- A0Iti6S C32-C30ro.SApS f33 ot59ut0((- = 4ep eSAi6lautI uitlAi6It 1tiuMS 5l
 Mpa-ctil69 ft6pFAlhai,-a Ba' = = HODJlx n=Numer. Math. 67, n,-1=

x3 C6A96SE123CM AroAS p1f3-t00t1 , (g. ra4uppl ApS lroialtitlAi6Mptl iut.ut.hautI eA)
 l6iucSLSIAM J. Numer. Anal. 35, 711M7s=(

li.tl ctiuMSa 5l ropahccitl69 .IM.0tca Alt S6a9rooatSpS

1. **l1Uo aQ E 4gsao dU o :KEuo** (1992). *4eeIIo0-d}i5Mf ot aaRci63p x-i1MSn ,d Mpl2cd6i5}F ft67td 2h,Ccc SIAM J. Matrix Anal. Applic. 13, 979–991.*
2. $x_0 = \text{ap}ApS$ 2=C ar(1990). 4 Ccdci6I-C1MSa dII06e100b-SroIItSM7I2t0EeSFro36ft6F1a 2h,icca u§Math. Comput. 54, 671–700.
2. $x_0 = \text{a-p A S} = \mathcal{Z}$ ar(1990). 4 itd ,6I ,1MSd .hII05ec0ab-,roIIoS,M7I2taEe,Fro36ft6F1a 2h,icca u§Math. Comput. 56, 215–242.
- o=fdaF OphS = 2 (1992). oMNd3FroIIi6MMS ,d ft1d)-,MFahc cd5If6p)d 2hni-ca § SIAM J. Sci. Statist. Comput. 13, 168–193.
- C-dai6Itti13Sa,d IIM .0aR ccid6Ithni1ca AdtStCa6 69S*
- o. $dR < 0$. To a = (100). 5IDro1, u,odA 6Tratti03S,[d 290I6p].M . R2hcc-id6II ft}ptcd2ha3cn §u Numer. Lin. Alg. 7, 197–218.
- = $x=M.04ApS_k = .1I$ 6(2005). , CcdC6I1-i1MS dMaI673Mc.01RI_k-hccCd5II 2hai-cn d5a}pox-IICd6AMId 3Sd0)§ SIAM J. Matrix Anal. Applic. 26, 1150–1178.
- uicd}C6Icti.MS, [d ,69)ro0}dhaCo AdtS5 eronr6pS*
- e=tLR (1990). 4 0t3pIt)cd9II ME 6ptaaCc63p}dhuC,dAC5IdMIIc,ac,[d 2M0I6p26p3ro0Aphs aidroIIirod25ai 9nME 6F-axroAC53p, SIAM Review 32, 611–635.
- e2=.)M (2001). 4 3it3, NdM 6oaM,B.a6ii5p)n ME6pbro 2hccid5 N3n6 67b 5Sc.p6ic xA,d6IIta Numer. Math. 88, 603–606.
- NAddn C1AdcIIM7IIcdp6,1 } A00t0tc 8 1i6MpII0roSt0*
- tcp c xIc9, (1984). 42Adaaac2l ui }i5C1MS,§ Parallel Comput. 1, 3–18.*
- ,= N},-0ApS 2=p3dS1p (1984). dA NM6pM2,5at 2roeecc l8ideot ARAC63pC 3S MpAxro .29I-nMd Parallel Comput. 1, 207–222.
- o=p= -cMpa (1986). "A X< ' -1 oA= Yk4Pellr.Qy..1r0Qx-e+ x.o xa rC0= & o 3E< y= aey xSIAM J. Alg. Disc. Meth. 7, 337–347.
- ,= -1 IC1AFS Pa cl(1989). 2dM8tII 6MC131,d 2M0Tra6p3h p-id6f6pAd2h,Cccn Mplroa5d3IIcaa3dn§Parallel Computing 9, 291–312.
- ,= ec3S6M }ps x=xA (1995). e,AdA00ta ro ,12 6t0MS df03Iy g6S6AMpAaf6pcld 2haitca §u SIAM J. Sci. Comput. 16, 1451–1461.
- c 1}I- n-1p iAi it II37S5i53p(A) 1/ia,C)l, cap 1 /s(73ae1AppApqE3..i))T1AplAx=b l+AiA, p8A-E1lp.lld 8tpC 6q 76/i SAiiR Ea.2, pi+1 8leA,,C6lp(, / 1Ar
- 11.3 lghe.x EC'** (1985). 4ot ai53pfci.-F MpS6i6M,-cdn }p2 ,ut MpIt)tpII1 3EC1o paII3X10C1M, , ocE .3a6i5Ic t-.p6 oxa id}Ic Numer. Math. 46, 31–42.
- x=ed6M5 u=7ro c7 t=oro60(1992). 2i3. 6p6 cd6A d icd1i5I2M0I,da§ SIAM J. Matrix Anal. Applic. 13, 138–144.
- x69}0h i1t t tII8E 3ro 15F) MM7,c c 13lM i }aei5 \$ 3d 6p
- 2=M1p6}yM.ay5 (H78. ® h) (x 12.-k- < nH ln= - - Kj k11-+1= Ai=.nNumer. Math. 30, 301–314.
- N=-p 6)li (1993). xdd2p10hn6n M_ AiMpAdh Ccd1C0MPeaaMII}cic§dM 0e§2.Bt=i c,6aR^n tt.adC t7i 3E , --A 6176§ cd,5ih 3E apII1ta tp)§:9,=

11.3 The Conjugate Gradient Method

A difficulty associated with the SOR, Chebyshev semi-iterative, and related methods is that they depend upon parameters that are sometimes hard to choose properly. For example, the Chebyshev acceleration scheme requires good estimates of the largest and smallest eigenvalues of the underlying iteration matrix $M^{-1}N$. This can be a very challenging problem unless this matrix is sufficiently structured. In this section and the next we present various Krylov subspace methods that avoid this difficulty.

We start with the well-known conjugate gradient (CG) method due to Hestenes and Stiefel (1952) and which is applicable to symmetric positive definite systems

There are several ways to motivate and derive the technique. Our approach involves the method of steepest descent, Krylov subspaces, the Lanczos process, and tridiagonal systems solving. After developing the Lanczos implementation of the CG process, we proceed to establish its equivalence with the Hestenes-Stiefel formulation.

A brief comment about notation is in order. Most of the methods in the previous section are developed at the (i, j) level and this necessitated the use of superscripts to designate vector iterates. From now on, the derivations in this chapter can proceed at the vector level. Subscripts will be used to designate vector iterates, so instead of $\{x^{(k)}\}$ we now have $\{x_k\}$.

11mPm1A nA SV [11mQ]GA a7..20A

Suppose $A \in \mathbb{R}^{n \times n}$ is symmetric positive definite, $b \in \mathbb{R}^n$, and that we want to compute the solution x to

$$Ax = b \quad (1131)$$

Note that this problem is equivalent to solving the optimization problem

$$\min_{x \in \mathbb{R}^n} \psi(x) \quad (1132)$$

where

$$\psi(x) = \frac{1}{2}x^T Ax - x^T b \quad (1133)$$

This is because ψ is convex and its gradient is given by

$$\nabla \psi(x) = Ax - b$$

Thus, if x_c is an approximate minimizer of ψ , then x_c can be regarded as an approximate solution to $Ax = b$. To make this precise, we define the A-norm by

$$\|v\|_A = \sqrt{v^T Av}. \quad (1134)$$

Since

$$\psi(x_c) = \frac{1}{2}x_c^T Ax_c - x_c^T b = \frac{1}{2}(x_c - x)^T A(x_c - x) = \frac{1}{2}b^T A^{-1}b$$

and $\psi(x) = -b^T A^{-1}b/2$ it follows that

$$\phi(x_c) = \frac{1}{2}\|x_c - x_*\|_A^2 + \phi(x_*). \quad (1135)$$

Thus, an iteration that produces a sequence of ever-better approximate minimizers for ψ is an iteration that produces ever-better approximate solutions to $Ax = b$ as measured in the A-norm.

hhmrpb C\\$b h\\$ Qknb kb uQSS3 \\$eQbps e\\$tQb

Let us consider the minimization of ψ using the method of steepest descent with exact line searches. In this method the current approximate minimizer x_c is improved by

searching in the direction of the negative gradient, i.e., the direction of most rapid decrease. In particular, the improved approximate minimizer x_+ is given by

$$x_+ = x_c - \mu_c g_c,$$

where $g_c = Ax_c - b$ is the current gradient and μ_c solves

$$\min_{\mu \in \mathbb{R}} \langle (x_c - \mu g_c), g_c \rangle. \quad (11.36)$$

This is an exact line search framework. It is easy to show that

$$\mu_c = \frac{g_c^T g_c}{g_c^T A g_c}$$

and

$$\phi(x_c) = \phi(x_*) - \frac{1}{2} \frac{(g_c^T g_c)^2}{g_c^T A g_c}. \quad (11.37)$$

Thus the objective function is decreased if $\mu_c > 0$. To establish global convergence of the method, define

$$K_C = \frac{g_c^T A g_c}{g_c^T g_c} \cdot \frac{g_c^T A^{-1} g_c}{g_c^T g_c}$$

and observe that $g_c^T A^{-1} g_c = 2(x_c) + b^T A^{-1} b$ and

$$\phi(x_+) = \phi(x_c) - \frac{1}{2} \frac{1}{\kappa_c} g_c^T A^{-1} g_c = \phi(x_c) - \frac{1}{\kappa_c} \left(\phi(x_c) + \frac{1}{2} b^T A^{-1} b \right). \quad (11.38)$$

If $\lambda_{\max}(A)$ and $\lambda_{\min}(A)$ are the largest and smallest eigenvalues of A , then we have

$$K_C = \frac{g_c^T A g_c}{g_c^T g_c} \cdot \frac{g_c^T A^{-1} g_c}{g_c^T g_c} \leq \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)} = K_2(A).$$

If we subtract $\langle (x^*) = - (b^T A^{-1} b)/2$ from both sides of (11.38) and use (11.35), then we obtain

$$\|x_+ - x_*\|_A^2 \leq \left(1 - \frac{1}{K_2(A)}\right) \|x_c - x_*\|_A^2. \quad (11.39)$$

It follows by induction that the method of steepest descent with exact line search is globally convergent.

~~more details here~~ ie. via Cholesky factorization. Given a symmetric positive definite $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$, $Ax = b$ and a termination tolerance ϵ the following algorithm produces $x \in \mathbb{R}^n$ so that $\|Ax - b\|_2 \leq \epsilon$

$x = x_0, g = Ax_0 - b$

while $\|g\|_2 > \epsilon$

$\mu = (g^T g) / (g^T A g), x_+ = x_0 - \mu g, g = Ax_+ - b$

end

Unfortunately, a convergence rate characterized by $(1 - 1/K_2(A))^k$ is typically not good enough unless A is extremely well-conditioned.

11 nPAPAnA IN A+fiA L= } dfIA

We can improve upon the steepest descent idea by expanding the dimension of the search space each step. To pursue this idea we introduce the notion of an affine space. Formally, if $v \in E$ and $S \subseteq \mathbb{R}^n$ is a subspace, then

$$v + S = \{x | x = v + s, s \in S\}.$$

is an affine space. Note that in Algorithm 11.3.1, the step-k optimization is over the affine space $\mathbb{R}^k + \text{span}\{-(x_k)\}$.

Given $Ax_0 = b$ our plan is to produce a nested sequence of subspaces

$$S_1 \subset S_2 \subset S_3 \subset \dots$$

that satisfy $\dim(S_k) = k$ and to solve the problem

$$\min_{x \in x_0 + S_k} \langle x \rangle \quad (11.3.10)$$

each step along the way. If x_k is the step-k minimizer, then because of the nesting we have $\langle x_1 \rangle \leq \langle x_2 \rangle \leq \dots \leq \langle x_k \rangle = \langle x \rangle$. Since $S_n = \mathbb{R}^n$ we ultimately obtain $x = A^{-1}b$. Even though this is a finite-step solution framework, it may not be attractive if n is extremely large. The challenge is to find a subspace sequence that promotes rapid decrease in the value of $\langle x \rangle$ if r then we may be able to terminate. ¶

$$= \begin{bmatrix} & & & \cdots & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & & & & \ddots & \\ & & & & & \ddots \\ \vdots & & & \ddots & & \ddots \\ & & & & \ddots & \end{bmatrix},$$

and a vector $r_k \in \text{span}(Q_k)$ so that

$$A k = Q_k k + r_k. \quad (11.3.13)$$

Note that the tridiagonal matrix

$$Q_k^T A Q_k = T_k$$

is positive definite. The solution to the optimization problem (11.3.10) via Lanczos is particularly simple if we set $q_0 = r_0$, where $r_0 = b - A x_0 - g_0$ and $g_0 = \|r_0\|_2 \mathbf{1}$. Since the columns of Q_k span $S_k = K(A, Q_k)$, it follows that the act of minimizing $\langle (x_0 + Q_k y) \rangle$ over $x_0 \in S_k$ is equivalent to minimizing $\langle (x_0 + Q_k y) \rangle$ over all vectors $y \in \mathbb{R}^k$. Since

$$\begin{aligned} \langle (x_0 + Q_k y) \rangle &= \frac{1}{2} \langle (x_0 + Q_k y)^T A (x_0 + Q_k y) \rangle - \langle (x_0 + Q_k y) \rangle^T b \\ &= y^T (Q_k^T A Q_k) y - y^T (Q_k^T b) + \langle x_0 \rangle \end{aligned}$$

and for $Q_k(1) = r$, it follows that the minimizer satisfies

$$k k = Q_k r_0 = f \in \mathbb{R}$$

and so $X_k = x_0 + Q_k k$. Building on Algorithm 10.11 this leads to a preliminary version of the conjugate gradient (CG) method

$$k = Q_k o = b - Ax_0, f_0 = \|r_0\|_2 \mathbf{1}, o = 0$$

while $k = 0$

$$q_{k+1} = r /$$

$$k = k + 1$$

$$k = q A k$$

$$T_k k = f_0$$

$$x_k = Q_k$$

$$r_k = (A - k)q - \beta_{k-1} q_{k-1}$$

$$!k = \|r\|_2 \mathbf{1}$$

end

$$X^* = X_k$$

{11.3.14}

As it stands, this formulation is not suitable for large problems because X_k is computed as an explicit n -by- k matrix-vector product and this requires access to all previously computed Lanczos vectors. However, before we develop a slick recursion for X_k that circumvents this problem, we establish some important properties that are associated with the iteration.

Theorem 11.3.1. If k_* is the dimension of the smallest invariant subspace that contains r_0 then the conjugate gradient iteration (11.3.14) terminates with $X_k = x_*$.

Proof. From Theorem 10.11 we know that the Lanczos iteration terminates after generating Q_k if $(A q, k)$ is an invariant subspace. If $q_1 = r_0 / \|r_0\|_2 \mathbf{1}$ then q_k

must be generated for otherwise row would be contained in an invariant subspace with dimension less than k . Since we can write α as a linear combination of k eigenvectors it follows that the Krylov matrix $[r_0 | Ar_0 | A^2r_0 | \cdots | A^{k-1}r_0]$ has rank k . This implies $\beta_{k+1} = 0$ in (11.3.14) and so the iteration terminates with $x_* = x_{k+1}$.

An important ramification is that early termination can be expected if the matrix A is a lowrank perturbation of the identity matrix.

Corollary 11.3.2. Assume that $U \in \mathbb{R}^{n \times r}$, $D \in \mathbb{R}^{r \times r}$ is symmetric, and $r \leq n$. If $A = I_n + UD\bar{U}^\top$ is positive definite and the conjugate gradient iteration (11.3.14) is applied to the problem $Ax = b$ then at most $r + 1$ iterations are required to compute x .

Proof. If $v \in \mathbb{R}^n$ is in the nullspace of U^\top , then $Av = v$ and $v = 1$ is an eigenvalue of A with multiplicity at least $n - r$. It follows that A cannot have more than $r + 1$ distinct eigenvalues. Thus, r_0 is contained in an invariant subspace with dimension $r + 1 - 1$.

Recall that our derivation of (11.3.14) begins with a plan to improve upon the method of steepest descent. Instead of determining x_k from a 1-dimensional search in the direction of the $\nabla\phi(x_{k-1})$, the CG method determines x_k by searching over a Krylov subspace that includes $\nabla\phi(x_{k-1})$. It follows that a CG step is at least as good as a steepest descent step, as the following theorem shows.

Theorem 11.3.3. If x_* is the solution to the symmetric positive definite system $Ax = b$ and x_k and x_{k+1} are produced by the CG method (11.3.14), then

$$\|x_{k+1} - x_*\|_A \leq \left(1 - \frac{1}{\kappa_2(A)}\right)^{1/2} \|x_k - x_*\|_A.$$

Proof. Setting $x_c = x_k$ in (11.3.9) gives

$$\|x_+ - x_*\|_A \leq \left(1 - \frac{1}{\kappa_2(A)}\right)^{1/2} \|x_k - x_*\|_A,$$

where x_+ is the steepest descent successor to x_c . By using inequality (11.3.11) we have $\|x_{k+1} - x_*\|_A \leq \|x_+ - x_*\|_A$.

Just how these mathematical results color practical matters is detailed in §11. For now, we continue with our exact arithmetic derivation of the method.

hhmmib a) \$ h\\$ Qeknbkb.lk dD"NO\$mxNnSQezb usfkttb -\$x e1kb

Returning to the initial version of the CG method in (11.3.14), we work out the details associated with the tridiagonal solve $T_k y_k = \beta_0 e_1$ and the matrix-vector product $x_k = Q_k y_k$. For the overall implementation to be attractive for large sparse A , we need

a way to compute \mathbf{X}_k without having to access Lanczos vectors $\mathbf{q}_1, \dots, \mathbf{q}_k$. Since the tridiagonal matrix $\mathbf{T}_k = \mathbf{Q}_k^T \mathbf{A} \mathbf{Q}_k$ is positive definite, it has an LDL^T factorization. By comparing coefficients in $\mathbf{T}_k = \mathbf{L}_k \mathbf{D}_k \mathbf{L}_k^T \mathbf{f}$ where

$$\mathbf{L}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \ell_1 & 1 & 0 & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \ell_{k-1} & 1 \end{bmatrix}, \quad \mathbf{D}_k = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & d_k \end{bmatrix},$$

we find

```

d = Rh
for i = 2:k
    e_i-1 = f_i - d_i/d_i
    d_i = G_i - f_i - e_i-1
end

```

(11.3.15)

Given this factorization we see that if $\mathbf{V}_k \mathbf{E}_1^k$ solves

$$\mathbf{L}_k \mathbf{D}_k \mathbf{L}_k^T \mathbf{v}_k = \mathbf{f} \quad (11.3.16)$$

then $\mathbf{L}_k^T \mathbf{f} = \mathbf{V}_k$. If $\mathbf{c}_k \mathbf{E}_1^k \mathbf{x}_k$ satisfies

$$\mathbf{Q}_k^T \mathbf{f} = \mathbf{Q}_k \quad (11.3.17)$$

then

$$x_k = x_0 + Q_k y_k = x_0 + C_k L_k^T y_k = x_0 + C_k v_k. \quad (11.3.18)$$

This is an impractical recipe because the matrix \mathbf{Q}_k is full and involves all the Lanczos vectors. However, there are simple connections between \mathbf{Q}_k and \mathbf{Q}_k^T and between \mathbf{V}_k and \mathbf{V}_k^T that can be used to transform (11.3.18) into a very handy update recipe for x_k . Consider the lower bidiagonal system (11.3.16), eg,

$$\left[\begin{array}{ccc|c} d_1 & 0 & 0 & 0 \\ d_1 \ell_1 & d_2 & 0 & 0 \\ 0 & d_2 \ell_2 & d_3 & 0 \\ \hline 0 & 0 & d_3 \ell_3 & d_4 \end{array} \right] \left[\begin{array}{c} \nu_1 \\ \nu_2 \\ \nu_3 \\ \hline \nu_4 \end{array} \right] = \left[\begin{array}{c} \beta_0 \\ 0 \\ 0 \\ 0 \end{array} \right].$$

We conclude that

$$v_k = \left[\begin{array}{c} \nu_1 \\ \vdots \\ \frac{\nu_{k-1}}{\nu_k} \end{array} \right] = \left[\begin{array}{c} v_{k-1} \\ \hline \nu_k \end{array} \right] \quad (11.3.19)$$

where

$$v_k = \begin{cases} \beta_0/d_1 & \text{if } k=1 \\ -d_{k-1} \ell_{k-1} \nu_{k-1}/d_k & \text{if } k > 1 \end{cases}. \quad (11.3.20)$$

Next, we consider a column partitioning of equation (11.3.17), eg,

$$\begin{bmatrix} c_1 & | & c_2 & | & c_3 & | & c_4 \end{bmatrix} \begin{bmatrix} 1 & \ell_1 & 0 & 0 \\ 0 & 1 & \ell_2 & 0 \\ 0 & 0 & 1 & \ell_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} q_1 & | & q_2 & | & q_3 & | & q_4 \end{bmatrix}.$$

From this we conclude that

$$C_k = [C_{k-1} \mid c_k] \quad (11.3.21)$$

where

$$c_k = \begin{cases} q_1 & \text{if } k=1 \\ q_k - \ell_{k-1}c_{k-1} & \text{if } k>1 \end{cases}. \quad (11.3.22)$$

It follows from (11.3.19) and (11.3.21) that

$$x_k = x_0 + C_{1:k} = x_0 + C_{k-1}v_{k-1} + \nu_k c_k = \mathbf{x}_{k-1} + \nu_k c_k.$$

This is precisely the kind of recursive formula for x_k that we need to make the recipe (11.3.18) attractive for large sparse problems. Combining this expression with (11.3.20) and (11.3.22), we obtain the following implementation of (11.3.14).

~~messhAui-11.3.2 el 83.CIp7mp1p.1m37..pfop31O8.p(m1.183op If A E^{2-t} is symmetric positive definite, bEI N, and Ax₀ = b then this algorithm computes x_{*} EI so that Ax_{*} = b~~

$$\mathbf{k} = \mathbf{Q} r_0 = \mathbf{b}, \quad Ax_0, \beta_0 = \|r_0\|_2, \quad q_0 = \mathbf{Q} c_0 = \mathbf{0}$$

while $\beta_k \neq 0$

$$q_{k+1} = r_k / \beta_k$$

$$\mathbf{k} = \mathbf{k} + 1$$

$$\alpha_k = q^T A q_k$$

if $\mathbf{k} = 1$

$$d_1 = \alpha_1, \nu_1 = \beta_0 / d_1$$

$$c_k = q_1$$

else

$$\ell_{k-1} = \beta_{k-1} / d_{k-1}, \quad d_k = \alpha_k - \ell_{k-1} \ell_{k-1}^T, \quad \mathbf{V}\mathbf{k} = -\beta_{k-1} \nu_{k-1} / d_k$$

$$c_k = q_k - \ell_{k-1} c_{k-1}$$

end

$$x_k = x_{k-1} + \nu_k c_k$$

$$r_k = A q_k - \alpha_k q_k - \beta_{k-1} q_{k-1}$$

$$\beta_k = \| \mathbf{V}\mathbf{k} \|_2$$

end

$$x_* = x_k$$

Each iteration involves a single matrix-vector product and about $13n$ fops. It can be implemented with just a handful of length n storage arrays, we discuss in §11.3.8

11 cPc5A T oA 47}Lwol -+An7 oA.1 INf} oA

We make some observations about the gradients and search directions that arise during the CG iteration. First, we show that the gradients

$$g_k = Ax_k - b = \nabla\phi(x_k)$$

are mutually orthogonal, a fact that explains the name of the algorithm.

Theorem 11.3.4. If x_1, \dots, x_k are generated by Algorithm 11.8.2, then $g_i^T g_j = 0$ for all i and j that satisfy $1 \leq i < j \leq k$. Moreover, $g_k = v_k r_k$ where v_k and r_k are defined by the algorithm.

Proof. The partial tridiagonalization (11.3.13) permits us to write

$$g_k = Ax_k - b = A(x_0 + Q_k y_k) - b = -r_0 + (Q_k u_k + r_k e_k)y_k$$

Since $Q_k^T y_k = f$ and $Q_k u_k = r_0$, it follows that

$$g_k = (e_k^T y_k)r_k$$

Since each r_i is a multiple of $q+1$, it follows that the g_i are mutually orthogonal. To show that $g_k = v_k r_k$, we must verify that $e_k^T y_k = v_k$. From the equation

$$T_k y_k = (L_k D_k)(L_k^T y_k) = \beta_0 e_1$$

we know that $L f y_k = v_k$ where $(L_k D_k) v_k = \beta_0 e_1$. To complete the proof, recall from (11.3.19) that v_k is the bottom component of v_k and exploit the fact that $L f$ is unit upper bidiagonal. \square

The search directions c_1, \dots, c_k satisfy a different kind of orthogonality property.

Theorem 11.3.5. If c_1, \dots, c_k are generated by Algorithm 11.3.2, then

$$c_i^T A c_j = \begin{cases} 0 & \text{if } i \neq j, \\ d_j & \text{if } i = j, \end{cases}$$

for all i and j that satisfy $1 \leq i < j \leq k$.

Proof. Since $Q_k = C_k L$ and $d = Q_k^T A Q_k b$ we have

$$d = L_k (C_k^T A C_k) L_k^T$$

But $T_k = L_k D_k L f$ and so from the uniqueness of the LDLT factorization, we have

$$D_k = C_k^T A C_k$$

The column partitioning $C_k = [c_1 | \dots | c_k]$ implies that $C_k^T A C_k = [D_k]_{ii}$.

The theorem tells us that the search directions c_1, \dots, c_k are A -conjugate.

11APAA Ea + o + ob (L - mEAs 7p) B - mA

The preceding results permit us to rewrite Algorithm 11.32 in a way that avoids explicit reference to the Lanczos vectors and the entries in the ongoing LDLT factorization. In addition, we will be able to formulate the termination criterion in terms of the linear system residual $b - Ax_k$ instead of the more obscure "Lanczos residual vector" $(A - \alpha_k I)q_k - \beta_{k-1}q_{k-1}$. The key idea is to think of c_k as a search direction and ρ_k as a step length and to recognize that these quantities can be scaled. Consider the search direction update recipe

$$c_k = q_k - \ell_{k-1}c_{k-1}$$

from Algorithm 11.32. Since q_k is a multiple of g_{k-1} we see that

$$\boxed{(\text{search direction } k) = g_{k-1} + \text{scalar} \times (\text{search direction } k-1)}$$

If we write this as

$$p_k = g_{k-1} + \tau_{k-1}p_{k-1}, \quad (11.32)$$

then it follows from

$$Ap_k = Ag_{k-1} + \tau_{k-1}Ap_{k-1}$$

and Theorem 11.35 that

$$\tau_{k-1} = -\frac{p_{k-1}^T A g_{k-1}}{p_{k-1}^T A p_{k-1}} \quad (11.32)$$

and

$$p_k^T A g_{k-1} = p_k^T A p_k. \quad (11.32)$$

Since p_k is a multiple of c_k , the update formula $x_k = x_{k-1} + \rho_k c_k$ in Algorithm 11.32 has the form

$$x_k = x_{k-1} - \mu_k p_k$$

for some scalar μ_k . By applying A to both sides of this equation and subtracting we get

$$g_k = g_{k-1} - \mu_k A p_k.$$

Using Theorem 11.34 and equation (11.32) we see that

$$\mu_k = \frac{g_{k-1}^T g_{k-1}}{g_{k-1}^T A p_k} = \frac{g_{k-1}^T g_{k-1}}{p_k^T A p_k}.$$

From the equations $g_{k-1} = g_{k-2} - \mu_{k-1} A p_{k-1}$ and $g_{k-1}^T g_{k-2} = 0$ it follows that

$$g_{k-1}^T g_{k-1} = -\mu_{k-1} g_{k-1}^T A p_{k-1},$$

$$g_{k-2}^T g_{k-2} = \mu_{k-1} g_{k-2}^T A p_{k-1} = \mu_{k-1} p_{k-1}^T A p_{k-1}.$$

Substituting these equations into (11.32) gives

$$\tau_{k-1} = \frac{g_{k-1}^T g_{k-1}}{g_{k-2}^T g_{k-2}}.$$

By exploiting these recipes for p_k , x_k , g_k , μ_k , and τ_{k-1} , and redefining r_k to be the residual $b - Ax_k = -g_k$, we can rewrite Algorithm 11.32a as follows.

Algorithm 11.32b If $A \in \mathbb{R}^{n \times n}$ is symmetric positive definite, $b \in \mathbb{R}^n$, and $Ax_0 = b$, then this algorithm computes $x_* \in \mathbb{R}^n$ so that $Ax_* = b$.

```

 $k = 0$   $r_0 = b - Ax_0$ 
while  $\|r_k\|_2 > 0$ 
   $k = k + 1$ 
  if  $k = 1$ 
     $p_k = r_0$ 
  else
     $\tau_{k-1} = (r_{k-1}^T r_{k-1}) / (r_{k-2}^T r_{k-2})$ 
     $p_k = r_{k-1} + \tau_{k-1} p_{k-1}$ 
  end
   $\mu_k = (r_{k-1}^T r_{k-1}) / (p_k^T A p_k)$ 
   $x_k = x_{k-1} + \mu_k p_k$ 
   $r_k = r_{k-1} - \mu_k A p_k$ 
end
 $x_* = x_k$ 
```

This procedure is essentially the same as delineated in Hestenes and Stiefel (1952).

11 min{A m soA a7}i[G]BA 9} nB+A

Rounding errors lead to a loss of orthogonality among the residuals and finite termination is not guaranteed in floating point. For an extensive analysis of this fact, see Mourtant (1964). Thus, it makes sense to have a termination criterion based on (say) the size of $\|r_k\|$. With that in mind and being careful about required vector workspaces, we obtain the following more practical version of Algorithm 11.33.

```

 $k = 0$   $x = x_0$ ,  $r = b - Ax$ ,  $\rho_c = r^T r$ ,  $\epsilon = \text{tol} \cdot \|b\|_2$ 
while  $\|r\| > \epsilon$ 
   $k = k + 1$ 
  if  $k = 1$ 
     $p = r$ 
  else
     $\tau = \rho_c / \rho_{k-1}$ ,  $p = r + \tau p$ 
  end
   $w = Ap$ 
   $\mu = \rho_c / p^T w$ ,  $x = x + \mu p$ ,  $r = r - \mu w$ ,  $\rho_{k-1} = \rho_c$ ,  $\rho_c = r^T r$ 
end (11.32b)
```

Thus, a CG step requires one matrix-vector product, three solves, and two inner products. Four length n arrays are required. Note that if \mathbf{X} is the final iterate and \mathbf{x} is the exact solution, then

$$\| \mathbf{x}_c - \mathbf{x}_* \| = \| A^{-1}(b - Ax_c) \|_2 \leq \text{tol} \cdot \| A^{-1} \|_2 \| b \|_2 \leq \text{tol} \cdot \kappa_2(A) \| \mathbf{x}_* \|.$$

Thus, a stopping criterion ensures a relative error that is bounded by the product of tol and the condition number.

In practice, it is desirable to terminate the iteration long before approaches \mathbf{X} of then and Bau (NLA, p. 299) show that

$$\| \mathbf{x}_* - \mathbf{x}_t \| \leq \frac{\sqrt{\kappa_2(A)} - 1}{\sqrt{\kappa_2(A)} + 1} \| \mathbf{x}_* \| \quad (11.32)$$

Of course, it does not take much of a condition number for the upper bound to be hopelessly close to 1, so by itself, this result does not provide hope for an early exit. However, as we will see in 11.5 there is a way to induce speedy convergence by applying the method to an equivalent "preconditioned" system that is designed in such a way that (11.32) and/or Corollary 11.32 predict good things.

hhgrgib .kl yUtf \$b o tt T\$tf ebeEf T\$tf kbATA 1/7) AAT

There are two obvious ways to convert an unsymmetric $\mathbf{A} = \mathbf{b}$ problem into an equivalent symmetric positive definite problem

$$Ax = b \equiv \begin{cases} A^T A x = A^T b, \\ A A^T y = b, \quad x = A^T y. \end{cases}$$

Each of these conversions creates an opportunity to apply the method of conjugate gradients.

If we apply CG to the $\mathbf{A}^T \mathbf{A} = \mathbf{A} \mathbf{b}$ problem then at the k th step, a vector x_k is produced that minimizes

$$\phi_{A^T A}(x) = \frac{1}{2} x^T (A^T A)x - x^T (A^T b) = \frac{1}{2} \| Ax - b \|_2^2 - \frac{1}{2} b^T b$$

over the affine space

$$S_k = x_0 + \mathcal{K}(A^T A, A^T r_0, k) \quad (11.32)$$

where $r_0 = \mathbf{b} - \mathbf{A} \mathbf{x}_0$. The resulting algorithm is the conjugate gradient normed equation residual (CGNR) method.

If we apply the CG method to the "y-problem" $\mathbf{A} \mathbf{A}^T \mathbf{y} = \mathbf{b}$ then at the k th step a vector y_k is produced that minimizes

$$\phi_{\mathbf{A} \mathbf{A}^T}(y) = \frac{1}{2} y^T (\mathbf{A} \mathbf{A}^T)y - y^T b = \frac{1}{2} \| A^T y - A^{-1} b \|_2^2 - \frac{1}{2} b^T (A A^T)^{-1} b$$

over the affine space $\mathcal{O} + \mathcal{K}(\mathbf{A} \mathbf{A}^T, \mathbf{r}_0, k)$ where $\mathbf{r}_0 = \mathbf{b} - \mathbf{A} \mathbf{x}_0$. Setting $\mathbf{X} = \mathbf{A} \mathbf{T} \mathbf{y}_k$ this says that $\mathbf{x} = \mathbf{X}$ minimizes $\| \mathbf{x} - \mathbf{b} \|$ over the affine space defined in (11.32).

CG	CGNR	CGNE
$\mathbf{re} = \mathbf{b} - \mathbf{Ax}_0$	$r_c = \mathbf{b} - \mathbf{Ax}_0, z_c = \mathbf{A}^T r_c$	$\mathbf{re} = \mathbf{b} - \mathbf{Ax}$
$\mathbf{Pc} = \mathbf{re}$	$\mathbf{Pc} = \mathbf{Zc}$	$\mathbf{Pc} = \mathbf{ATre}$
$\mu_c = \frac{r_c^T r_c}{\mathbf{Pc}^T \mathbf{Ap}_c}$	$\mu_c = \frac{\mathbf{Zc}^T \mathbf{c}}{(\mathbf{Ap}_c)^T (\mathbf{Ap}_c)}$	$\mu_c = \frac{r_c^T r_c}{\mathbf{Pc}^T \mathbf{Pc}}$
$\mathbf{X}_+ = \mathbf{X}_c + \mu \mathbf{Pc}$	$\mathbf{X}_+ = \mathbf{X}_c + \mu \mathbf{pc}$	$\mathbf{X}_+ = \mathbf{X}_c + \mu \mathbf{pc}$
$r_+ = r_c - \mu \mathbf{Ap}_c$	$r_+ = r_c - \mu \mathbf{Ap}_c, z_+ = \mathbf{A}^T r_+$	$r_+ = r_c - \mu \mathbf{Ap}_c$
$\mathbf{T} = \frac{r_+^T r_+}{r_+^T \mathbf{r}_c}$	$\mathbf{T} = \frac{z_+^T z_+}{\mathbf{Zc}^T z_c}$	$\mathbf{T} = \frac{r_+^T r_+}{r_c^T r_c}$
$\mathbf{P}_+ = \mathbf{r}_+ + \mathbf{T} \mathbf{Pc}$	$\mathbf{P}_+ = \mathbf{Z}_+ + \mathbf{T} \mathbf{Pc}$	$\mathbf{P}_+ = \mathbf{A}^T \mathbf{r}_+ + \mathbf{T} \mathbf{Pc}$

Figure 11.3.1 The initializations and update formulas for the conjugate gradient (CG) method, the conjugate gradient normal equation residual (CGNR) method, and the conjugate gradient normal equation error (CGNE) method. The subscript "c" designates "current" while the subscript "+" designates "next".

The resulting method is called the conjugate gradient normal equation error (CGNE) method. It is also known as Cring's method.

Simple modifications of the CG update formulae in Algorithm 11.3.3 are required to implement CGNR and CGNE. We tabulate the initializations and updates of the three methods in Figure 11.3.1. Notice that CGNR and CGNE require procedures for \mathbf{A} -times vector and \mathbf{A}^T -times vector. See Saad (IMSL, pp. 251–254) and Greenbaum (IMSL, Chap. 7) for details and perspective on the squaring of the condition number that is associated with these methods. The CGNR method can be applied if \mathbf{A} is rectangular. Thus, it provides a normal equation framework for solving sparse, full rank, least squares problems. See Björck (SLE, pp. 288–293) for discussion and analysis. The CGNE method can also be applied to rectangular problems, but the underlying system must be consistent.

Problems

- P11.3.1 8 .4. .. 98.0 IR.. B.d\$2.49. 4619f. ..s.R fR.9fR04R86
 P11.3.2 1 9. = w r r a 'x ma2=u=1V-Wu == 3W E4r x= ar=a = rax a u a = a au 1au au a=r+u ux4V=uat= ' = • a a r = a
 P11.3.3 .Rfold .9.r..l\$. .9f. N 7 6 N o9fgl .r6feR.l.. 95dt
 P11.3.4 f8rob R9f.9obfR§ssd 8R9R61..g. R9 .d9 R4f.. .98

$$\| \mathbf{k}_k \| > \text{tol} (\| \mathbf{A} \| \mathbf{k}_k \| + \| \mathbf{b} \|),$$

$$, \mathbf{R} \| \quad \mathbf{M} \quad \mathbf{H}\mathbf{R} .\mathbf{J}\mathbf{R} \quad \| \mathbf{x} \quad \mathbf{r}, \mathbf{Ax} \neq \mathbf{b} .\mathbf{M}.\mathbf{aoc} \quad \text{lo0A3.loiMtol.}$$

Notes and References for §11.3

- P 4d..8l69o 9..f..9..R.§ R4fR..49R8R.4SI N. ...l. R RR.I.49 1 N 6 n . n3 f.8.R.R4.. .o.4or. 9f.486 l... R..9 .f8 R.or

KB: BteQae(at 1, CQ2 .uy, nly6heQArt(18 1aPhQe 0t)eaQ In GtJh T2emG(QN(1
J. Res. Nat. Bur. Stand. 49, J.r ,57Y

3mAneAl. MAriMenar aladri.A . rMie1MlPIA.ml&Aria- n.mnAlgl+ naSiSAM

9M5Anet.N.TV , a.mA A.mlel. Aladri.AMQe1AaMmAlOI.1la. niMrsSiMstAPt
l. nlaAiMl i.nlatr in Large Sparse Sets of Linear Equations, p.-1 ot6S .t Sc.B.AStcm. Nltaa-
.t. 7MlyBn7s n,-.

2M1t umacMlmAS rp6a6pNaJt.C6Ita Alt MiltS mp0

C.2. CM,r. ApSt1N1lAItAh hs(g.s. 42Mct 26aCMI.M.Cut.Mper)ACt ClASmtpApSIAp.9Ma4tCus
MSaASIAM Review 31, J NJIV

Ivlu.AtAaAtNJ N.Aladriatiae MienAatrlaQ History of Scientific Computing, eSSmap
t a0thBtASmp)4e1

21eau.h- 11e. 4ApCt,t,- ApSNx12Ah0Mls((n.s. 4e A MpMBl MpFr0r)ACtClAS6tpC4tCuMSa-Ap
SIAM J. Numer. Anal. 27, N,I r Ndv

,iAMmAAiMPlat il.mlMtnii) iai0tbAemAP A.hae

.MaA+ iMN.TV.amAAiAlAmrAhaAnAA.mlel. Aladri.AMielAat. ftlO.iAeM.MAPA
hlnat.lamASAAiNAr Nomer. Math. 24, dr.5v

3MAaaItT.N.TV f a2IAadAmA9lRaiiOSS3aMl6 mAlaiAMrAiaAl. mAla)Iri.A
M: 1AAlA.mleQ Q Inst. Math. Applic. 20, 7NBIV

o. y' O< o> 2eD0= < p > O = OevO.e + n p a xJ0=aAO> O oseF se Matrix
Techniques: Copenhagen, s(o.w 51e1 f Alytl .t S1.-2Jlmp)tle5tl0AB) ftl, mpc

4co 12taCtpa hs(gls. Conjugate Direction Methods in Optimization, 2Jlmp)tle5tl,A)- ft l,mp1
p1.0,r1 ApSo1.6, 0M.u.h hs(gls. 4rut IAp.9MaNtpMtpASep DpCtJltCACrjMtp Mpmpers
)ACt ClASltpClJC6169AGnMpLin. Alg. Applic. 29, 75l,Jv

3, iia eAMg ltae.M3nia eAMlMttNd72.3mAli.A l. AlaiAMrAhaAladri.AMielAaTr
Numer. Math. 48, „5X.7Jv

3x9M1P ia TxIMSBiiiae.xAMOP iaN.TV.3 2IA la Aladri.AMielAaAlaiAMrAaaAr
Numer. Math. 76 IJ.XI5Jv

3M9M1P ia e sMparAgSAIMjN .3 2IA la Aladri.AMielAaAlaiAMrAaaAlMff f 33
Numer. Math. 85, 77X.7d5v

sv9arAOStM 3M98inPiatiJJN .3 2IA la Ala)Iri.AMiJRAAlaiAMrAaaAlMff f r T
Numer. Math. 85, 7d .7Y

IMi -li.nar,SlRaeltal ttRla A.. tAAAlMiaf nA... + Agg

.val baB1+ tBnDlV.llIael1 9MMB1Ott1l. i 2A+ AgtAladri.AM:RAa30rlM1mPtr
Lin. Alg. Applic. 29, J. .1.Y

3MMAAaSII Pae -. s.MiBltNj N .hMAena.nA wAmliRlA. AhMAa1tRlaabtiae
Aladri.AMienAaAlPSI,iRatQ Q AM J. Matrix Anal. Applic. 13 NN TV5

- s.MiBliae hMhamIJJN ,a 9MMMP i.nla.lamAladri.AMienAdA.mleiae amt
n.alMBtla.nan.AMaaantAlPSI,i.lat,6ETNA 13, 7r dJY

M IAI Mie z. Ov= a e= 24 D9• Sa = nG = 0 = Pp yla' vOJ0< y = = J0-R. o=On
X= OnOyJ-Ov vJ0, Ata Numerica 15, ,TN,J,

3mAJPngt A.,MAQi.ACA.mleit iAMg iMiAe.mA lgg+ latP iof IStAl. mOnAMi.hM2

.Max s.A+ iMN.TV .Ala)Iri.AE 1MAa.iAaMlet 1Mslgi larsttAPt l. nRaAAlM i.nlatr T
Numer. Math. 21, Id, I.TY

EMBnAiMNdN.3mAwOlaAla)Iri.AMielAa30rlMmPae lAOi.AelA.mlel I QAlg.
Applic. 29, I.5r5IIv

9MEAaaBmae 5u tMaMMAaAMiOnMadr i EAMAaaf6 Lin. Alg. Applic. 88/89
NdiII.J.Y

3MwhatA.AMtaiAMv.sliAMN. N,a i Aladri.AMieRAa,3 ISAmle1MslOi larAlPOg2M
stCP.AMna nlaAiM lSttAPt Q Q 287, NJ: NI 5Y

avwiM.iae 3V IiaAI 1AOIJJZ JI g 1SOAAAlMtnAladri.AMienAa3grlMmP hiMi:
Op a nO= vOJ0N9A MOJ. Matrix Anal. Applic. 21, T7dr T.7V

AvniIJ.N. A.21 f tia 9MMM1ana8grlRmBSIAM J. Sci. Comput. 22, I N,X NNI v

3v3EISMI g QNw.la.lgnar.mAA.mlel. wbaBALachri.AMienAa.teTQ12, IN7Iev

ER tden cat tLArch X22En.bSh)AM 7hP 10 qZ1496 o laShue Outves usArt
 3sA OS caeet qfes 65ACM Trans. Math. Softw. 32, s.7.-76.
 7=IAASr11.Rc 4xm0itltSMp8ro)Aitota5SroA0reihe0)Ml6CuT25iu eJ0 5.Ai6Mp8LSIAM J. Matrix
 Anal. Applic. 28, g-, go1c
 .ut roatM.iuo TretiuMSMM0ItliA6p tl)tpIA0rot JlM.0tTrem&tiAm0tSp
 e- orout ApS.. .6.tl) r(s onRl 4.ut 1tiuMS MMp8ro)AitClASltbia,atS 5pupItlat uitlAimMp8LT
 12, -7, ,--
 e- xSt0TreAAppSI.-.ITre6iu.((.R - 4lp .Mp8ro)AitClASmtptireft3ytuMSa 5l x6)tpref6yNlM.0tTr&L
 BIT 36, -(-- ,lg p
 .ut Sta6)p M.atpal.0t aiMJ5p).llitdIA uw TreAphro.i0tiltaSittS
 I3x, eau.hS 13,3 2M0aiS) 1ApitroItOS ApS N-xIAh0Mtn1sR-4.ut oM0tM.Cut upptl NlMSro.inp
 IiMJJmpjmitlmA5l .Mpero)Ait ClASmtptiuitlAilMpa BIT 41, n. -,nB
 1. ellM06 n1Rt 4e IiMJJ6p5itlmMp5l iut .Mpero)Ait ClAS5tpi e0)Ml5iuTt6p A x6pmix0tTretpti
 1tiuMS 2ATret2MlySNumer. Math. 97, iON61

11.4 Other Krylov Methods

The conjugate gradient method can be regarded as a clever pairing of the symmetric Lanczos process and the LDL^T factorization. The "cleverness" is associated with the recursions that support an economical transition from x_k to x_{k+1} . In this section we move beyond symmetric positive definite systems and present instances of the same paradigm for more general problems:

$$\left(\begin{array}{c} \text{Krylov} \\ \text{process} \end{array} \right) + \left(\begin{array}{c} \text{Matrix} \\ \text{factorization} \end{array} \right) + \left(\begin{array}{c} \text{Clever} \\ \text{recursions} \end{array} \right) = \left(\begin{array}{c} \text{Sparse} \\ \text{matrix} \\ \text{method} \end{array} \right).$$

Methods for the symmetric indefinite problem (MINRES, SYMLQ), the least squares problem (LSQR, LSMR), and the square $Ax = b$ problem (GMRES, QMR, BiCG, CGS, BiCGStab) are briefly discussed. The Lanczos, Arnoldi, and unsymmetric Lanczos iterations are in the mix. Our goal is to communicate the main idea behind these methods. For deeper insight, practical intuition, and analysis, see Saad (ISPLA), Greenbaum (IMSL), van der Vorst (IMK), Fiedler, Golub, and Nachtigal (1992), and LIN_TEMPLATES.

LL44L

Assume that $A \in \mathbb{R}^{n \times n}$ is symmetric indefinite, i.e., $\lambda_{\min}(A) < 0$ or $\lambda_{\max}(A) < 0$. A consequence of this is that we cannot recast the $Ax = b$ problem as a minimization problem associated with $\phi(x) = x^T Ax / 2 - x^T b$. Indeed, this function has no lower bound. If $Ax = \lambda_{\min} x$, then $\phi(\alpha x) = \alpha^2 \lambda_{\min} - \alpha x^T b$ approaches $-\infty$ as α gets big.

This suggests that we switch to a more workable objective function. Instead of adopting the CG strategy of minimizing over the affine space $x_0 + \mathcal{K}(A, r_0, k)$, we propose to solve

$$\min_{x \in x_0 + \mathcal{K}(A, r_0, k)} \|b - Ax\|_2. \quad (114)$$

at each step. As in CG, we use the Lanczos process to generate the Krylov subspaces, setting $q_1 = r_0$ if $r_0 = b - Ax_0$ and $f_0 = \|r_0\|_2$. After k steps we have

$$AQ_k = Q_k T_k + \beta_k q_{k+1} e_k^T.$$

That is,

$$AQ_k = Q_{k+1}H_k, \quad (1142)$$

where $H_k \in \mathbb{R}^{k+1 \times k}$ is the Hessenberg matrix

$$H_k = \left[\begin{array}{cccc|c} \alpha_1 & \beta_2 & \cdots & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & 0 \\ \vdots & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \beta_{k-1} \\ 0 & \cdots & \cdots & \beta_{k-1} & \alpha_k \\ \hline 0 & \cdots & \cdots & 0 & \beta_k \end{array} \right]. \quad (1143)$$

Writing $\mathbf{x} = \mathbf{x}_0 + Q_k y$ and recalling that $\text{ran}(Q_k) = \mathcal{K}(A, G|k)$, we see that the optimization (1141) involves minimizing

$$\| A(x_0 + Q_k y) - b \|_2 = \| Q_{k+1}H_k y - (b - Ax_0) \|_2 = \| H_k y - \beta_0 e_1 \|_2$$

over all $y \in \mathbb{R}^k$. To solve this problem we take a hint from §5.26 and use the Givens QR factorization procedure. Suppose G_1, \dots, G_k are Givens rotations such that

$$G_k^T \cdots G_1^T H_k = \begin{bmatrix} R_k \\ 0 \end{bmatrix}, \quad R_k \in \mathbb{R}^{k \times k},$$

is upper triangular. If

$$G_k^T \cdots G_1^T (\rho e_1) = \begin{bmatrix} p_k \\ \vdots \end{bmatrix} \quad p_k \in \mathbb{R}^k,$$

and $y_k \in \mathbb{R}^k$ solves $R_k y_k = p_k$, then $x_k = \mathbf{x}_0 + Q_k y_k$ solves (1141) and the norm of the residual is given by $\| b - Ax_k \|_2 = |\rho_k|$. The transition

$$\{H_{k-1}, R_{k-1}, p_{k-1}, \rho_{k-1}\} \rightarrow \{H_k, R_k, p_k, \rho_k\}$$

can be realized with $O(1)$ fops after the k th Lanczos step is performed. The Givens rotation G_k can be determined from $[R_{k-1}]_{k-1,k-1}$. Note that after step $k-1$ we already have the first $k-2$ rows of R_k and the first $k-2$ components of p_k . The matrix R_k has upper bandwidth 2 and so the triangular system that determines y_k can be solved with $O(k)$ fops. Thus, in computing $x_k = \mathbf{x}_0 + Q_k y_k$ each step is not essential. On the other hand, it is possible to work out an $O(n)$ transition from x_{k-1} to x_k through recursions that involve Q_k and the QR factorization of H_k . (This corresponds to the LDLT-plus- Q_k recursions associated with CG developed in §11.35.) Either way, there is no need to access all the Lanczos vectors each step. Properly implemented, we have the MINRES method of Paige and Saunders (1975).

An alternative approach developed by the same authors works with the LQ factorization of the tridiagonal matrix T_k . We mimic the §11.34 in the CG derivation leading to (11314). However, the solution of the tridiagonal system

$$T_k y_k = \beta_0 e_1 \quad (1144)$$

is problematic because \mathbf{T}_k is no longer positive definite. This means that the \mathbf{LDL}^T factorization, together with the associated recursions, is no longer safe to use.

A way around this difficulty is to work with the transpose of the matrix equation $\mathbf{AQ}_k \mathbf{x} = \mathbf{Q}_k \mathbf{y}$. Suppose $\mathbf{X}_{k-1} \in \mathbb{R}^{n \times k}$ and $\mathbf{Q}_k \in \mathbb{R}^{n \times k}$ where \mathbf{Y}_k is the minimum norm solution to the $(k-1)$ -by- k underdetermined system

$$\mathbf{H}_{k-1}^T \mathbf{y}_k = \beta_0 \mathbf{e}_1. \quad (11.45)$$

It follows from $\mathbf{A} \mathbf{Q}_k \mathbf{y}_k = \mathbf{Q}_k \mathbf{y}$, $\mathbf{A} \mathbf{Q}_k \mathbf{X}_{k-1} = \mathbf{Q}_k \mathbf{Y}_k$ and $\mathbf{Q}_k^T \mathbf{A}^T \mathbf{H}_{k-1} \mathbf{Q}_k$ that

$$\mathbf{Q}_k \mathbf{y}_k = \mathbf{f}_{k-1} - \mathbf{H}_{k-1} \mathbf{Y}_k = \mathbf{0}.$$

Thus the residual $\mathbf{r}_{k-1} = \mathbf{b} - \mathbf{Ax}_k$ is orthogonal to $\mathbf{q}_1, \dots, \mathbf{q}_k$. Note that the underdetermined system (11.45) has full rowrank and that \mathbf{Y}_k can be determined via a Givens rotation lower triangularization, e.g.,

$$\begin{bmatrix} \alpha_1 & \beta_1 & 0 & 0 & 0 \\ \beta_1 & \alpha_2 & \beta_2 & 0 & 0 \\ 0 & \beta_2 & \alpha_3 & \beta_3 & 0 \\ 0 & 0 & \beta_3 & \alpha_4 & \beta_4 \end{bmatrix} G_1 G_2 G_3 G_4 = \begin{bmatrix} \times & 0 & 0 & 0 \\ \times & \times & 0 & 0 \\ \times & \times & \times & 0 \\ 0 & \times & \times & \times \end{bmatrix} \mathbf{I} = [L_{4|0}].$$

This is an LQ factorization and in general we have

$$\mathbf{H}_{k-1}^T G_1 \cdots G_{k-1} = [L_{k-1} \mid 0]$$

where L_{k-1} is lower triangular. (This is just the transpose of the Givens QR factorization of \mathbf{H}_{k-1} .) If $\mathbf{V}_{k-1} \mathbf{E}_j \mathbf{J}_k \mathbf{I}_l$ solves the necessarily nonsingular system $\mathbf{L}_{k-1} \mathbf{M}_{k-1} \mathbf{f}_{k-1} = \mathbf{0}$ then

$$\mathbf{y}_k = G_1 \cdots G_{k-1} \begin{bmatrix} w_{k-1} \\ 0 \end{bmatrix}.$$

The special structure of L_{k-1} (it has lower bandwidth equal to 2) and the Givens rotation sequence make it possible to realize the transition from \mathbf{X}_{k-1} to \mathbf{X}_k at with $O(n)$ work in a way that does not require access to all the Lanczos vectors. Collectively, these ideas define the SYMMQLQ method of Paige and Saunders (1975).

hhshcb Iu59b ttib Iu h9b eeb ISfe f b . teSb .. kPSeb

We show how the sparse least squares problem $\min \| \mathbf{Ax} - \mathbf{b} \|_2$ can be solved using the Paige-Saunders lower bidiagonalization process described in §10.4.4. Indeed, if we apply Algorithm 10.4.2 with $\mathbf{U}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$ where $\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}_0$ and $\mathbf{U}_0 = \mathbf{I} \|\mathbf{r}_0\|_2$, then after k steps we have a partial factorization of the form

$$\mathbf{AV}_k = \mathbf{UB}_k + \mathbf{P}_k \mathbf{f}$$

where $\mathbf{V}_k = [\mathbf{V}_1 \cdots \mathbf{V}_k] \in \mathbb{R}^{1 \times k}$ has orthonormal columns, $\mathbf{U}_k = [\mathbf{u}_1 \ \mathbf{u}_2 \ \cdots \ \mathbf{u}_k] \in \mathbb{R}^{1 \times k}$ has orthonormal columns, and $\mathbf{B}_k \in \mathbb{R}^{k \times k}$ is lower bidiagonal. If $\mathbf{P}_k \mathbf{E}_k \mathbf{R}_k^T$ is nonzero, then we can write

$$\mathbf{AV}_k = U_{k+1} \tilde{\mathbf{B}}_k$$

where $EhE \in \mathbb{R}^{k+1 \times k}$ is given by

$$\tilde{B}_k = \begin{bmatrix} \alpha_1 & 0 & \cdots & \cdots & 0 \\ \beta_1 & e_1 & \ddots & & 0 \\ \vdots & \ddots & & & \vdots \\ \vdots & & \ddots & & 0 \\ 0 & \cdots & \cdots & \beta_{k-1} & k \\ \hline 0 & \cdots & \cdots & 0 & \beta_k \end{bmatrix}. \quad (1146)$$

It can be shown that $\text{span}\{v_1, \dots, v_k\} = K(A^T A, A^T r_0, k)$. In the LSQR method of Paige and Saunders (1982), the k th approximate minimizer x_k solves the problem

$$\min_{x \in x_0 + K(A^T A, A^T r_0, k)} \|Ax - b\|_2. \quad (1147)$$

Thus, $x_k = x_0 + V_k y_k$ where $y_k \in \mathbb{R}^k$ is the minimizer of

$$\|A(x_0 + V_k y) - b\|_2 = \|U_{k+1} \tilde{B}_k y - (b - Ax_0)\|_2 = \|\tilde{B}_k y - \beta_0 e_1\|_2.$$

Givens QR can be used to solve this problem just as it is used in the MINRES context above. Suppose

$$G_k^T \cdots G_1^T \tilde{B}_k = \begin{bmatrix} R_k \\ 0 \end{bmatrix}, \quad G_k^T \cdots G_1^T (\beta_0 e_1) = \begin{bmatrix} p_k \\ \rho_k \end{bmatrix},$$

where G_1, \dots, G_k are Givens rotations, $R_k \in \mathbb{R}^{k \times k}$ is upper triangular, $p_k \in \mathbb{R}^k$, and $\rho_k \in \mathbb{I}$. Then, y_k solves $R_k y = p_k$ and

$$x_k = x_0 + V_k y_k = x_0 + W_k p_k$$

where $W_k = V_k R_k^{-1}$. It is possible to compute x_k from x_{k-1} via a simple recursion that involves the last column of W_k . Overall, we obtain the LSQR method of Paige and Saunders (1982). It requires only a few vectors of storage to implement.

The LSMR method provides an alternative to the LSQR method and is mathematically equivalent to MINRES applied to the normal equations $A^T A x = A^T b$. Like LSQR, the technique can be used to solve least squares problems, regularized least squares problems, underdetermined systems, and square unsymmetric systems. The norms of the vectors $r_k = b - Ax_k$ and $A^T r_k$ decrease monotonically, which allows for tractable early termination. See Fong and Saunders (2011) for more details.

hs hsmmb mh9A ubexb m§§ tfbAx = b

The Paige-Saunders MINRES method (§11.4.1) is a Lanczos-based technique that can be used to solve symmetric $Ax = b$ problems. The k th iterate x_k minimizes $\|Ax - b\|_2$ over $x_0 + K(A, b, k)$. We now present an Arnoldi-based iteration that does the same thing and is applicable to general linear systems. The method is referred to as the generalized minimum residual (GMRES) method and is due to Saad and Schultz (1986).

After k steps of the Arnoldi iteration (Algorithm 10.5.1) it is easy to confirm using (10.5.2) that

$$AQ_k = Q_{k+1}\tilde{H}_k \quad (11.48)$$

where the columns of

$$Q_{k+1} = [Q_k \ Iq_{k+1}]$$

are the orthonormal Arnoldi vectors and the upper Hessenberg matrix \tilde{H}_k is given by

$$\tilde{H}_k = \begin{bmatrix} h_{11} & h_{12} & \cdots & \cdots & h_{1k} \\ h_{21} & h_{22} & \cdots & \cdots & h_{2k} \\ 0 & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & h_{k,k-1} & h_{kk} \\ 0 & \cdots & \cdots & 0 & h_{k+1,k} \end{bmatrix} \in \mathbb{R}^{k+1 \times k}.$$

Moreover, if $q_1 = \text{r}_0/\|\text{r}_0\|$ where $\text{r}_0 = b - Ax_0$ and $\text{f}_0 = \|\text{r}_0\|_2$, then

$$\text{span}\{q_1, \dots, q_k\} = K(Ax_0).$$

In step k , the GMRES method requires minimization of $\|Ax - b\|_2$ over the affine space $x_0 + K(Ax_0)$. As with MINRES, we must find a vector $y \in \mathbb{R}^k$ so that

$$\|A(x_0 + Q_k y) - b\|_2 = \|Q_{k+1}\tilde{H}_k y - (b - Ax_0)\|_2 = \|\tilde{H}_k y - \beta_0 e_1\|_2$$

is minimized. If y_k is the solution to this $(k+1)$ -by- k least squares problem, then the k -th GMRES iterate is given by $x_k = x_0 + Q_k y_k$. Note that if Givens rotations G_1, \dots, G_k have been determined so that

$$G_k^T \cdots G_1^T \tilde{H}_k = \begin{bmatrix} R_k \\ 0 \end{bmatrix}, \quad R_k \in \mathbb{R}^{k \times k}, \quad (11.49)$$

is upper triangular and we set

$$G_k^T \cdots G_1^T (\beta_0 e_1) = \begin{bmatrix} p_k \\ \rho_k \end{bmatrix}, \quad (11.4.10)$$

where $p_k \in \mathbb{R}^k$ and $\rho_k \in \mathbb{R}$, then $R_k y_k = p_k$ and

$$|\rho_k| = \|Ax_k - b\|_2.$$

The transition

$$\{R_{k-1}, p_{k-1}, \rho_{k-1}\} \rightarrow \{R_k, p_k, \rho_k\}$$

is a particularly simple update that involves the generation of a single rotation G_k and exploitation of the identities $R_{k-1} = R_k(1:k-1, 1:k-1)$ and $p_k(1:k-1) = p_{k-1}$.

As a procedure for large sparse problems, the GMRES method inherits the usual Arnoldi concern: the computation of $H(1:k+1, k)$ requires $O(kn)$ flops and access to all previously computed Arnoldi vectors. For this reason it is necessary to build a restart strategy around the following multistep GMRES building block:

nskE III Pte Eonkh rs=EqPo OPnBso If $A \in \mathbb{R}^{n,n}$ is nonsingular, $b \in \mathbb{R}^n$, $Ax_0 = b$ and m is a positive iteration limit, then this algorithm computes $x \in \mathbb{R}^n$ where either \tilde{x} solves $Ax = b$ or minimizes $\|Ax - b\|_2$ over the affine space $x_0 + J(A)x_0$ (where $rx = b - Ax_0$)

$$k = 0, rx = b - Ax_0, \|rx\|_2 = \|rx\|_2$$

while ($\|rx\|_2 > 0$ and $k < m$)

$$Q_{k+1} = rk/rk$$

$$k = k + 1$$

$$T_k = Aq_k$$

$$\text{for } i = 1:k$$

$$hik = q_i^T r_k \quad (11.4.11)$$

$$T_k = T_k - hik Q$$

end

$$fk = \|T_k\|_2$$

$$hik = fk$$

Apply G_1, \dots, G_{k-1} to $H(1:k, k)$ and determine G_k, R_k, P_k and ρ_k

end

Solve $R_k x_k = P_k$ and set $x = x_0 + Q_k x_k$

If \tilde{x} is not good enough, then the process can be repeated with the new x_0 set to \tilde{x} . There are many important implementation details associated with this framework, see Saad (IMSLA, pp 164–184) and van der Vorst (IMK, pp 65–84).

hhsgsb GE Q1qf st "b Blqf Q1S.b.k df tkqsteb .kstQb kb - Sb

Before we present the next group of methods, it is instructive to connect the Krylov framework with polynomial approximation. Suppose the columns of $Q_k \in \mathbb{R}^{n,k}$ span $J(A, q, k)$. It follows that if $y \in \mathbb{R}^k$, then $Qy = \langle(A)q\rangle f$ for some polynomial \langle that has degree $k-1$ or less. This is because

$$Q_k = [Q_1 Aq_1 | \dots | A^{k-1} q_k] B$$

for some nonsingular $B \in \mathbb{R}^{k,k}$ and so if $a = By$, then

$$Qy = [q_1 | Aq_1 | \dots | A^{k-1} q_k] a = (a_1 + a_2 A + \dots + a_k A^{k-1}) q_k$$

Thus, the GMRES (and MINRES) optimization can be rephrased as a polynomial optimization problem. If J_k denotes the set of all degree k polynomials, then we have

$$\min_{x \in Q_k(A, q, k)} \|b - Ax\|_2 = \min_{r \in J_{k-1}} \|b - A(x_0 + \langle(A)\rangle r)\|_2$$

$$= \min_{r \in J_{k-1}} \|I - A \langle(A)\rangle r\|_2 b$$

$$\min_{\psi \in \mathbb{P}_k, \psi(0)=1} \|I - A \langle(A)\rangle \psi\|_2$$

This point of view figures heavily in the analysis of various Krylov subspace methods and can also be used to suggest alternative strategies.

$$\text{hhpsib uT.o dbouob uT. oe f t:obtttb 5h9b eb os t\$x tf bAx = b}$$

Just as the Arnoldi iteration underwrites GMRES, the unsymmetric Lanczos process (10.5.11) underwrites the next cohort of methods that we present. Suppose we complete k steps of (10.5.11) with $\tilde{\mathbf{q}}_k = \mathbf{r}_0/\|\mathbf{f}_0\|_2$, $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\mathbf{f}_0 = \mathbf{I}\mathbf{r}_0\|_2$, and $\mathbf{r}_J = \mathbf{0}$. This means we have the partial factorizations

$$\mathbf{A}\mathbf{Q}_k = \mathbf{Q}_k\mathbf{L}_k + \mathbf{R}_k, \quad \mathbf{Q}_k\mathbf{L}_k^T = \mathbf{Q}, \quad (11.4.12)$$

$$\mathbf{A}^T\mathbf{Q}_k = \mathbf{Q}_k\mathbf{U}_k + \mathbf{R}_k, \quad \mathbf{Q}_k\mathbf{U}_k^T = \mathbf{Q}, \quad (11.4.13)$$

where

$$\mathbf{Q}_k = [\tilde{\mathbf{q}}_1 | \cdots | \tilde{\mathbf{q}}_k], \quad \text{ran}(\mathbf{Q}_k) = \mathcal{K}(\mathbf{A}, \mathbf{r}_0, k),$$

$$\mathbf{Q}_k^T = [\tilde{\mathbf{q}}_1^T | \cdots | \tilde{\mathbf{q}}_k^T], \quad \text{ran}(\mathbf{Q}_k^T) = \mathcal{K}(\mathbf{A}^T, \mathbf{r}_0, k).$$

In addition, $\mathbf{Q}_k^T\mathbf{Q}_k = \mathbf{I}_k$ and $\mathbf{Q}_k^T\mathbf{A}\mathbf{Q}_k = \mathbf{T}_k$ is tridiagonal. Vectors $\tilde{\mathbf{q}}_k$ and \mathbf{t}_k and scalars f_k and T_k satisfy

$$\beta_k q_{k+1} = r_k, \quad \tau_k \tilde{q}_{k+1} = \tilde{r}_k$$

and can be generated with access to just the last two columns of \mathbf{Q}_k and \mathbf{Q}_k^T .

In step k of the bi-conjugate gradient (BiCG) method, an iterate $\mathbf{x}_k = \mathbf{x}_0 + \mathbf{Q}_k\mathbf{y}_k$ is produced where $\mathbf{y}_k \in \mathbb{K}^k$ solves the k -by- k tridiagonal system

$$T_k y_k = \tilde{Q}_k^T r_0.$$

It follows that

$$\tilde{Q}_k^T(b - Ax_k) = \tilde{Q}_k^T(b - A(x_0 + Q_k y_k)) = \tilde{Q}_k^T r_0 - T_k y_k = 0.$$

Thus, the residual associated with \mathbf{x}_k is orthogonal to the range of \mathbf{Q}_k .

Assume that \mathbf{T}_k has an LU factorization $\mathbf{T}_k = \mathbf{L}_k\mathbf{U}_k$ and note that \mathbf{L}_k is unit lower bidiagonal and \mathbf{U}_k is upper bidiagonal. It follows that

$$x_k = x_0 + Q_k T_k^{-1} \tilde{Q}_k^T r_0 = (Q_k U_k^{-1})(L_k^{-1}(\tilde{Q}_k^T r_0)).$$

Analogously to how we derived the CG algorithm, it is possible to develop simple connections between the matrix $(\mathbf{Q}_k\mathbf{U}_k^T)$ and its predecessor and between the vector $(\mathbf{L}_k^{-1}(\tilde{Q}_k^T r_0))$ and its predecessor. The end result is a procedure that can generate \mathbf{x}_k through simple recursions, which we report in Figure 11.4.1. We mention that the BiCG method is subject to serious breakdown because of its dependence on the unsymmetric Lanczos process. However, with the look-ahead idea discussed in §10.5.6 it is possible to overcome some of these difficulties. Notice that BiCG collapses to CG if \mathbf{A} is symmetric positive definite and $\mathbf{f}_0 = \mathbf{r}_0$. Also observe the similarity between the r and \tilde{r} updates and the p and \tilde{p} updates.

A negative aspect of the BiCG method is that it requires procedures for both \mathbf{A} -times vector and \mathbf{A}^T -times vector. (In some applications the latter is a challenge.)

BiCG	CGS	BiCGstab
$r_0 = b - Ax_0$	$r_0 = b - Ax_0$	$r_0 = b - Ax_0$
$r_{0,c} = 0$	$r_{0,c} = 0$	$r_{0,c} = 0$
$X_t = X_0$	$X_t = X_0$	$X_t = X_0$
$P_c = r_e = r_0$	$P_c = r_e = r_0$	$P_c = r_e = r_0$
$P_c = T_c = f_0$	$U_c = r_e$	
$\mu = \frac{\tilde{r}_c^T r_c}{\tilde{p}_c^T A p_c}$	$\mu = \frac{\tilde{r}_0^T r_c}{\tilde{r}_0^T A p_c}$	$\mu = \frac{r^T r_e}{r^T A P_c}$
$X_{t+} = X_t + \mu p_c$	$q_c = u_c - \mu A p_c$	$S_c = r_e - \mu A p_e$
$r_{++} = r_e - \mu A p_e$	$X_{t+} = X_t + \mu(u_c + Q)$	$w = \frac{s A s_e}{(A s_e)(A s_e)}$
$\tilde{r}_{++} = \tilde{r}_c - \mu A^T \tilde{p}_c$	$r_{++} = r_c - \mu A(u_c + q_c)$	$X_{t+} = X_t + \mu P_c + \omega s_c$
$\tau = \frac{\tilde{r}_{++}^T r_{++}}{\tilde{r}_c^T r_c}$	$\tau = \frac{\tilde{r}_0^T r_{++}}{\tilde{r}_0^T r_c}$	$r_{++} = S_c - w A s_c$
$P_{++} = r_{++} r_p c$	$U_{++} = r_{++} T Q$	$T = \frac{(\tilde{r}_0^T r_{++}) \mu}{(r^T r_e) w}$
$P_{++} = T_{++} T P_c$	$P_{++} = U_{++} + r(Q + T P_c)$	$P_{++} = r_{++} T(P_c - w A P_c)$

Figure 11.4.1 The initializations and update formulas for the biconjugate gradient (BiCG) method, the conjugate gradient squared (CGS) method, and the biconjugate gradient stabilized (BiCGstab) method. The subscript "c" designates "current" while the subscript "+" designates "next".

The conjugate gradient squared (CGS) method circumvents this problem and has some interesting convergence properties as well. The derivation of the method uses the polynomial point of view that we outlined in the previous section. It is easy to conclude from Figure 11.4.1 that after k steps of the procedure we have degree k polynomials ' k ' and ' k ' so that

$$\begin{aligned} r_k &= ' k(A) r_0 & p_k &= ' k(A) r_0 \\ \tilde{r}_k &= ' k(Ar) r_0 & \tilde{p}_k &= ' k(Ar) r_0 \end{aligned} \quad (11.4.14)$$

and ' k ' $Q = ' k(Q = 1$. This enables us to characterize expressions like $r[r_k]$ and $P_k A P_k$ in a way that involves only A -times vector:

$$\begin{aligned} r[r_k] &= (' k(Ar) r_0) (' k(A) r_0) = r^T (' k(A) r_0), \\ P_k A P_k &= (ck(Ar) r_0) A (ck(A) r_0) = r^T (Ac%k(A) r_0). \end{aligned}$$

It is possible to develop simple recursions among the polynomials $\{t^k\}$ and $\{ck\}$ that facilitate the transitions

rk 1- tL 1(A)ro - t (A)ro - rk,
Pk 1- cL 1(A)ro - c (A)ro - Pk

This leads to the conjugate gradient squared (CGS) method of Saad (1989). It produces iterates \mathbf{x}_k whose residuals r_k satisfy $r_k = \mathbf{t}(\mathbf{k})^T \mathbf{r}_0$. Note from Figure 11.4.1 that the updates rely on only matrix-vector products that involve only \mathbf{A} . Because of the squaring of the BiCG residual polynomial $\mathbf{t}(\mathbf{k})$, the method typically outperforms BiCG when it works, i.e., $\|\mathbf{t}(\mathbf{k})\|_2 \leq \|\mathbf{t}\|_2$. By the same token, it typically underperforms when BiCG struggles.

A third member in this family of $\mathbf{Ax} = \mathbf{b}$ solvers is the BiCGstab method of der Vast (1992). It addresses the sometimes erratic behavior of BiCG by producing iterates \mathbf{x}_k whose residuals satisfy

$$r_k = (1 - \omega_k A) \cdots (1 - \omega_1 A) \psi_k(A) r_0$$

where t_k is the BiCG residual polynomial defined in (11.4.14). The parameter w_k is chosen in step k to minimize $\|r_k\|_2$ given w_1, \dots, w_{k-1} and the vector $t_k(A)r_0$. The computations associated with this transpose-free method are given in Figure 11.4.1.

Yet another iteration that is built upon the unsymmetric Lanczos process is the quasi-minimum residual (QMR) method of Freund and Nachtigal (1991). As in BiCG, the k th iterate has the form $\mathbf{x}_k = \mathbf{x}_0 + Q_k \mathbf{y}_k$ where Q_k is specified by (11.4.12). This equation can be rewritten as $A \mathbf{Q}_k = \mathbf{Q}_k^{-1} \mathbf{r}$ where $E^{(k+1)} \mathbf{I} \mathbf{x}_k$ is tridiagonal. It follows that if $r_k = r_0 / f_k$ where $r_0 = b - A \mathbf{x}_0$ and $f_k = \mathbf{I} \mathbf{r}_0 / \mathbf{I} \mathbf{b}$ then

b- A(xo+ Q_ky) = ro- AQ_ky = ro- Q_k+1Iky = Q_{k+1}i(f_{oel}- T_{ky}).

In QMR, y is chosen to minimize $\|f - Ax\|_2$. Note that GMRES minimizes the same quantity because Q_{k+1} has orthonormal columns in Arnoldi.

Problems

P11.4.1 1...28 .. 88.81 8... c.8. d.8..4. 8..bd. c.8.8.c.a1...28 . .A ER^{n × n}
 $\begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix} = I_n$

$$P11.4.2 \dots d8.8 A \in \mathbb{R}^{n \times n}, v \in \mathbb{R}^n, (\cdot, \beta^{nn_0} n \in \mathbb{R}^{3n} \in \mathbb{R}^{3n}, \omega + (e - b) \in \mathbb{R}^{3n}) (I - \omega A)v$$

P11.4.3 N8 .. .8.c. .2 , 482d.8. aT.mAMA (+ G)+ ("+" .ΣE 1

Notes and References for §11.4

fe8..8.8... ...82.§ 68 ..8..8c.8. ... 6.8.1.81..816..128. 8. .8.c8.84...8 ..8.8 ..8.8
4.8.. 4. ..68..5 ..fe8.8 68..4822 .. LINT LAT S. 1=JO=naxKO>Oa axOXyp
a =>O O = = kFH Oa 38n3 => n00e x x = nQ = an uk kΘ = ,

m O+1x2x2W2T +2rxT pT = • 1x103q7,0 xx = = x2xxxe xx 4=Axta
Numer ca 1, Tr NJ

amAf 219s. s1II,V5 iae,sVI qj.A.lM0t eiApIhiRrA isAseAMRaRipAaAlf pmAtP
R.)lMpI,pqMAiet5NtgliApqe MAt,iMam8

1B1B2dt kOB ZsSter(pyin(, kZS=2a ra Z, mle 1te, 2e Zt ed ra Uken fSo=2(5
 SIAM J. Numer. Anal. 12, iNTT..
 AuAhrAise IM3MiI aeAMtMlt4-LV163s 3grM12.(ML)iMtAsAiMcl i51l1tise L)iMtA
 ,A,2 Lcd iM3t6CM' n. Math Sof w. 8 i5r (IM
 Ix3MLiss3AMME MCls ise 9M, l1) A. ddM- + lAla) Iri2A..Miel) 12 t)HhA2.let ,M
 r stC.A2M 1aAiMlt2H.t6SIAM J. Numer. Anal. 25, IT ..JM
 AuAhrAwm2IM2A' sM3Ms eANBIMtA.. t. -3)MIT P.121gs2IEt1e F1rIsigsA
 vlsset ,M5MtgEJ,tjaAt6Numer. Lin. Alg. Applic. 3 ((. NM
 IM3M1seIMN.TtMalg)s2lsrhMI) Aa21ls12,LVI.6 BIT 37, .73(Ji.
 .x3NEgN.dt. - If 21 9Lise 29l 3MDA22AM LIII,V 1aF1rAs)i 1MC) I 2i21ls6
 SIAM J. Sci. Comput. 19, NTT(Tdi.
 LuMAa,l+ A... t4 -Llg21rAsHMig1,At2LcsmAM{ gAgt 12y LVIAM J. Matrix
 Anal. Applic. 21, (77(Ti
 IM6lgANse .Max L2A+ iN12JiM12AMi212Asg1M1- i2Pda I, 219L96SIAM J. Matrix Anal.
 Appl. 21, 7(5 C7id
 M ,A1a1AgVM AAjdm-3 AsAMig1,JW1 3grM17g Numer. Lin. Alg. Applic. 15
 7,571J.
 PM,ak,ir. AuAhrAiae Eu12At,:glcI 1aJJtML2))P.rAMPnAM 122AMinlliaJ 21la
 lf ,IsAiMait2Lcd iM 1hMI,gA6 SIAM J. Matrix Anal. Applic. 31, d5NdIM
 Lu,AM1. AM1a1r3ise I u3MiI seAM JNIN-If 219L.V,6 3 5MtgE2 t)ahA2.Ee,M
 ,aeAQs 12M,ar1giMCCCH2M12AC6SIAM J. Sci. Comput. 33, NJ h27.
 EM,AM.lariae IM3M1seAMtJ(NM,LIr6 31,2AMin3AEM1 1ML)iMtA nLd iMAT
 hMI,gA6SIAM J. Sci. Comput. 33 I.JII .TN.
 aA1M1r1s1g 9L)iAMittA2IM21s6
 IMcieise IMsA1g2A,d7M,lrFL6 3 AsrMig1- lls1 Ig,rPesig3g1M12yM1Llg21a
 r att.)2MPalsAiMlt 2A.t6SIAM J. Sci. Stat. Comput. 7, d1 7.. d
 iae 2AM1AM32Aigf,ggl+,I) isigt1t6
 LMai.) ,AggMAMtAaA; 5MggAise AMEMAM. i ti - 119L1e 2AIPs1g higt
 slC lig1BIT 36 7737T.4
 3MMA3asC. B M2iBiae -. L2MiB4i.7tM 3a2la1saMAN A112AMrn1AA2A1thltt1gA
 ,M,1IFL. 11SIAM J. Matr x Anal. Applic. 17, ii.l ii
 5M,Ak A,Tt. -, nl9L itMfeAig1 9L SIAM J. Matrix Anal. Applic. 18 5J i i
 nMBM1lglu2iBise -. L2MiB4i.dM5Mtg2A-IAsaEfnoT P,Ar2, ise AEa2MrAsaA
 ff .11 9L6BIT 38 7571
 IMLiQeAJJJ -IM2AMig1tlf I 1a1C1g1AtPe1g2A2Pit'6Numer. Lin. Alg. 7, iT .5M
 f MAK,MA1JJJ -9T)MAtt1stwile M23.11 FLit lefig6BIT 40, II.2.
 EMig221M+ Pt,le M PaAgJI. -,n nAIHrIgiM1M8AM21H H dFL IH2le6
 Numer. Math. 91, iJ.I 71.i
 .x,1AAa,I ull- gl- a1Bae z. 1 z= «E= 221SP x b a j-0030p k< eHJ= J2jr.J0pa <
 AOz qo>SIAM J. Sci. Comput. 23, ir1Efr Nr w
 D,xaAa7/mwcLe .N11Awce3ul pAAr)185 RrvpalpuesreApxLzAAT 44, T. .dM
 AMANFAIMl- gl- s1BT sY2MiB4EJiM1le1, Ae.Mig La7gPe2,L.t, Ar 2clIMAt.
 ise wiab+ iM1eig12tf nL,1IFL. 11SIAM J. Matrix Anal. Applic. 28, N./Ny/w
 lp 3AaomlA3palAeAAT 18AednAa0BraAejAmr 1Gpanr 47)ee3TNA : x a + +
 = c n.4000 ix-x• W x:Q u@ 12w,x ut10•ex= • a o= = Va 1G Q=a =
 = - Sx• xW = x u r c
 m x Q x = x x x x u Q x =xx+x=Qx = x x x+ = Θ x x• u
 m c=c = ⇒ 00sQ x = 2= maeu x u Q=4x • u xneQ x • 1SIAM JuMatr= J
 Anal. Applic. 16 (N,1(NTN x
 3x lgAMiae r MtaAMAN.dt. -lAt2iM2A4 9L,ML.P2)e,1sAiMlttn1gt SIAM J. Sci.
 Comput. 19, μ: 17M
 BM1faa1s1N. t. -3 2A+BiM li 12 Int2iM2H3 FL6Numer. Lin. Alg. 6 i(TTM
 IMwM 11M1H. -1g)g1P2Ar2iM2Ae 9Liae 3M1lge1 nA2M2Eltt.gA2M1att2ACT
 If 9di2lat 11AM J. Matrix Anal. Applic. 21, N(1(N5.M

- tT uimoo ca Cr 4tAro <22nL b-e q 10 Clq TOT 1Z .comes=r /rs= C/s=rah sAe
 x(sas e(sas qm)LLG Numer. Lin. Alg. 7 ,0 .dx
- .c-I pB JJJ TuAaH MigmSpAl2ArAaaAlael pl.MEl tpEMpAe 9LkG Numer. Lin.
 Alg. 7, μNTENv
- lcwcnlMrEa IJUf unr 9L + mE AlipAlAtplMpm.Bk6 SIAM J. Sci. Comput. 24, IJr 5Tu
 nc 9CSMAATFayAalnll :Ae pyA MpsMnl 9LkWAM Review 45 I.jrI.,v
- .c-I pB JJJ Tu AEa2AMAaAlamMi lAtplMpAe 9L nApyl.3rCApA+I py rA.tSa)/k
 Numer. Lin. Alg. 12, mTmr 5JC
- iMI)M' aLI aEg I :r+s/mma 9hanc)HACAapi MAl2AAd ae
- .c.c aigBAMdrT.fC)gACAApi II pRA ur9Ln, yEe :I al; / HylgeMiat,MCipl 146
 SIAM J. Sci. Stat. Comput. 9, NlrN,5i
- 3c MHHCnC Sis ll.gl .n IaBk4LpMiB jjTT -2 ,MI aEg yi 2mHpy)nlem,AeMiCi
 Lay,I enp 9L,)QAAapi I 63 BIT 37, TJ, TNO
- h2cwMl+ia .c.c aigBHMNPY .nl 9L,a ,2,iMgtII aBMrtpAC5 SIAM J. Matrix Anal.
 Applic. 18, mT mu
- 5cwMMea .. 9Mm gNdTu a pmAM,..iRaA1f BiMI Rei)pl 2AMAlaempInlaAQ
 LpMip,rI Nt56 Lin. Alg. 5, μJ NINu
- Ic Li E. 5, a - NdTuEV.nl9Le i EIAAlVlitl II.iqlA:I eiqgrEMnp3Hta
 f alC)gApMpylrlangpl 1.k6umer. Lin. Alg. 3, 5I.r 5,5
- nc LL:EaBi Efc aEprE,k, 5,5)i. I iifae .c.c aigBAMujjTu -3 2A+3eE)pl 2A 9L
 3qlMl pM3aymA2hBy3aa,Miat56umer. Lin. Alg. 5, IPr I.T.
- , I AtAlJJ TuAEC),p' Al.2MrHaMlaet ,M.nl 9L SAM J. Matrix Anal. Applic.
 21, ddI J2O
- Bc itt5,c IIej LcEppEaEa.c, jarII ,IJJT v3grlMmpMce 13Af nl 9L ElpI aA:
 ,M1lig iae AIC)gABMhpAplaaf ByhAM,MCiakC)IpAM6 CM' ns. Math Sof w.
 31, II dI 5dv
- 3c.c wEB9M5 t/I)3a lc niapA,Ag JJ.Tu8 aAaJ c.A. 3aaAgM pIAln 2MrAaaA
 lf lAt iM,ue 9LkSIAM J. Matrix Anal. Applic. 26 ,Ir jd
- ,c rAI ayAgVc IAT TuuwMHiBel+ a,qil 9L,MLhr,gIM AtppSIAM J. Matrix Anal.
 Applic. 26 NJJN μ
- ay,, I E' gBaBMrif pYHnl 9LC)pmlktAAe
- LmClaalidc.iggl)l,q t - jjTx .Ea2ArAaa,hMI),MpIf)WgB.nl 9L iae nipM2-
 hqzRL, 2Eqk6n. Alg. Applic. 247, Tr NOv
- 3.c wiBHMhcEaaal ink 9cl, Att) - IJTU ..a ,)ME2I ,hdiMLlg2AMMM,M iaAe
 wqBBi.2i.plr r9L o SIAM J. Sci. Comput. 27, NJr N,I,u
- ncll { S.. n, Li.BiaH IJ,Tr.9tiapi.e aA- ianiBeE+ : I pmAglaBnl 9LnAplek6
 Lin. Alg. Applic. 419, I,r Ideu
- ,MFI .Eg, MA.a/r laI+ Ippyp1 mAAlL VnlkE. wI ApECS,pyEtmagI.He
 Aciaalt - gITu.Llqp2l.lr itpA: EII AEdMipI II tnI alI AfpANit56. Res. Nat.
 Bur. Stand 55.5
- R. R < O! InhOH De= n 1 1ya Q.QIG= J=aOin)Yh .t eZ r x oLN ceedings of the
 Dundee Biennial Conference on Numerical Analysis, 1974. c3, aiptla oTik)MfAM, 1Mgirk
 2H+1MB.
- hcllaA2Aqe jjT -A.Le3. p,6a- l/,lt) Llg2AM2latt.CApMII aAiMtp,CtkSIAM
 J. Sci. Stat. Comput. 10, 5, I.
- l, A.e i.e 2, 2iaypl rEqjuTuVnre3 Vs' mH .CiatI esigApyleM2l...AMCII pi
 ,I IHSMtpHt5Wumer. Math. 60 2N x 55j
- .4342a eAMMtnNjITwI .Al3we3. E. LCllpyqAla2MrI BiMmlfawI ,AMpyA
 LlgIpl la2/tCCHpMnaM LthpSIAM J. Sci. Stat. Comput. 13 ,5Nr ,u
- L,StAc,Aa)i)AMpmipAmpl hyA:ApypE: I .agI,e
- .cnc.cLgI oEa EcrdBBA i N5T iwmA.tps,gM,maAi9csipl lRa2I ar./tCC)pMma
 nipMI am(Mc)qAb)AapMk6TNA 1, r.5i
- lc ; "a - jjT G VE)l/A , V,ir mh, al O)tl ,ig 3ghMpyM2l...A MCI. pl ,SM
 Lt/pC6SIAM J. Sci. Comput. 14 ,TJ ,dl.

:E resat at 41u4bQ+hSXyEnLb5u w1 .OoR@le ra5u OShU22(5 ACM ' ns
 Math. Sof. w. 22, 7r TT;
 nx,AxAsariae a; xAmiaN .T; .n.,B Tw1A.sa3we3 w1A.sa3w BiMRiwAe la nI U.RA
 ,4aa lts.i..lar Bla.lt .6SIAM J. Sci. Comput. 21, (175(I.Jx
 n.,lgCA.9xn1qA..iae ARISSiSL.,IJNTVnl .w. Aeh.leAa.1laAa.a1csAtM.A slgs.Rla
 If 2lar ACR.R1ATCt.1. nsgRSgRrmr iae leA6SIAM J. Sci. Comput. 29, T W TdJx
 3YgIAaalsaxR. Snglae.x sielB ,IJ.BT Y whBA MtnlfwRAsa3w., RaAiMtAPt
 R.msqISqRrm.,iae leA6ETNA 16 NJ N.;
 .:x.x LaSAaxLlaAaQiae nYwx 2ilm xax3Gqw c x i Z = c = Gx = E x x Vx = x =
 m a =, E x x x AppNumer. Math. 60 NJ N.;
 n; .xs.B aAamJNYTEI 9b,iRaA6ETNA 96, (17rN,d;

11.5 Preconditioning

In general, a Krylov method for $\mathbf{Ax} = \mathbf{b}$ converges more rapidly if $\mathbf{A} \in \mathbb{R}^{n \times n}$ "looks like the identity" and preconditioning can be thought of as a way to bring this about. A matrix can look like the identity in several ways. For example, if \mathbf{A} is symmetric positive definite such that $\mathbf{A} = \mathbf{I} + \mathbf{AA}$, and $\text{rank}(\mathbf{AA}) = k^* < n$, then Theorem 11.31 plus intuition says that the CG method should produce a good approximate solution after about k^* steps. In this section we identify several major preconditioning strategies and briefly discuss some of their key attributes. Our goal is to impart a sense of what it takes to design or invoke a good preconditioner - an absolutely essential skill to have in many problem settings. For a more in-depth treatment, see Sa (IMSLS), Greenbaum (IMSL), van der Vorst (IMK) and LIN_TEMPLATES.

NLA / L

Suppose $\mathbf{M} = \mathbf{M}_1\mathbf{M}_2$ is nonsingular and consider the linear system $\tilde{\mathbf{A}}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}$ where

$$\tilde{\mathbf{A}} = \mathbf{M}_1^{-1}\mathbf{A}\mathbf{M}_2^{-1}, \quad \tilde{\mathbf{b}} = \mathbf{M}_1^{-1}\mathbf{b}$$

Note that if \mathbf{M} looks like \mathbf{A} , then $\tilde{\mathbf{A}}$ looks like \mathbf{I} . The proposal is to solve the "tilde problem" with a suitably chosen Krylov procedure and then determine \mathbf{x} by solving $\mathbf{M}_2\mathbf{x} = \tilde{\mathbf{x}}$. The matrix \mathbf{M} is called a preconditioner and it must have two attributes for this solution framework to be of interest:

Criterion 1. \mathbf{M} must capture the essence of \mathbf{A} , for if $\mathbf{M} = \mathbf{A}$, then we have $\mathbf{M}_1^{-1}\mathbf{A}\mathbf{M}_2^{-1} = \tilde{\mathbf{A}}$. (In settings where \mathbf{M} is specified through its inverse, it is more appropriate to say that \mathbf{M}^{-1} captures the essence of \mathbf{A}^{-1} .)

Criterion 2. It must be easy to solve linear systems that involve the matrices \mathbf{M}_1 and \mathbf{M}_2 because the Krylov process involves the operation $(\mathbf{M}_1^{-1}\mathbf{A}\mathbf{M}_2^{-1})$ -times vector.

Having a good preconditioner means fewer iterations. However, the cost of an iteration is an issue, as is the overhead associated with the construction of \mathbf{M}_1 and \mathbf{M}_2 . Thus, the enthusiasm for a preconditioner depends upon the strength of the inequality

$$\begin{pmatrix} \text{Set up} \\ \mathbf{M} \\ \text{cost} \end{pmatrix} + \begin{pmatrix} \text{Single} \\ \mathbf{A}-\text{iteration} \\ \text{cost} \end{pmatrix} \cdot \begin{pmatrix} \text{Number} \\ \text{of } \tilde{\mathbf{A}} \\ \text{iterations} \end{pmatrix} < \begin{pmatrix} \text{Single} \\ \mathbf{A}-\text{iteration} \\ \text{cost} \end{pmatrix} \cdot \begin{pmatrix} \text{Number} \\ \text{of } \mathbf{A} \\ \text{iterations} \end{pmatrix}.$$

There are several ways in which a preconditioner N can capture the essence of A . The difference $A - N$ could be small in norm or low in rank. More generally, if

$$A = [\text{friendly/important part}] + [\text{troublesome/lesser part}],$$

then the important part is an obvious candidate for a preconditioner subject to the constraint imposed by Criterion 2. For example, if A is symmetric positive definite, then its diagonal qualifies as an important part that is computationally friendly.

hhghib 78b .x \$ kt Tf Tk\$thob ttnb oh9A ubh§ f ekneb

Before we step through the various ways that a linear system can be preconditioned, we show how the CG and GMRES iterations transform in the presence of a preconditioner. For details related to other preconditioned Krylov methods, see LIN_TEMPLATES.

Suppose that $E \in \mathbb{R}^{n \times n}$ is a symmetric positive definite matrix that we choose to regard as a preconditioner for the symmetric positive definite linear systems $Ax = b$. Recall that there is a unique symmetric positive definite matrix C such that $M = C^{-1}E$. See §4.24. If

$$\tilde{A} = C^{-1}AC^{-1}, \quad \tilde{b} = C^{-1}b,$$

then we can solve $Ax = b$ by applying CG to the symmetric positive definite system $\tilde{A}x = \tilde{b}$ and then solving $Cx = x$. For this to be a practical strategy, we must be able to execute CG efficiently when it is applied to this "tilde" problem. Referring to Figure 11.31, here are the CG update formulas in this case:

$$\begin{aligned} \mu &= (r^T r) I / \|Ap\|^2, \\ X_+ &= X - wfc \\ r_+ &= Tc + \mu Ap \\ \tau &= (\tilde{r}_+^T \tilde{r}_+) / (\tilde{r}_c^T \tilde{r}_c), \\ f_+ &= re + rfc \end{aligned} \tag{11.5.1}$$

Typically A is dense and so we must clearly implement these five steps if a suitable level of efficiency is to be reached. Note that if $X = c^{-1}Ixc$ and $re = b - Axe$, then

$$re = b - Axe = c^{-1}(b - Axe) = c^{-1}Irc$$

By substituting this formula together with $r_+ = c^{-1}Irc$ and the definition of A into (11.5.1) we obtain

$$\begin{aligned} \mu &= (r^T M^T Ir) / (C^T If)^T TA(C^T If), \\ Cx_+ &= Cxc - \mu fe \\ c^{-1}Irc_+ &= c^{-1}Irc + \mu c^{-1}IAC^{-1}If \\ r &= (r^T M^T Ir) / (r^T M^T Ir), \\ f_+ &= c^{-1}Irc + TfC \end{aligned}$$

If we define $P_C = C^{-1}IfC$ and set $Z_C = NJ^T Irc$, then this transforms to

Solve $M_Z c = r e$

$$\mu = (r^T Zc) / (H^T A p_e),$$

$$X_t = X_e - \mu p_c$$

$$r_+ = r_c + \mu A p_e,$$

$$i = (r^T Z_t) I (r^T Zc),$$

$$P_t = Z_t + T p_c$$

and we arrive at the method of preconditioned conjugate gradients (PCG). Note that although the square root matrix $C = M^{1/2}$ figured heavily in the derivation of PCG, in the end its action is felt only through the preconditioner $M = C^2$.

In PILU6.ssj, nsat18Pc5 ULPm65PmAnjU) glj5LdUti If $A \in \mathbb{R}^{n \times n}$ and $E \in \mathbb{R}^{n \times n}$ are symmetric positive definite, $b \in \mathbb{R}^n$, and $Ax_0 = b$ then this algorithm computes $x^* \in \mathbb{R}^n$ so that $Ax^* = b$

$$k=0 \quad r_0 = b - Ax_0 \quad \boxed{\text{Solve } M_Z c = r_0}$$

while $\|r_k\|_2 > 0$

$$k=k+1$$

$$\text{if } k=1$$

$$P_k = Zc$$

else

$$T = (r^T P_k) / (r^T Z_k)$$

$$P_k = Z_k + T P_{k-1}$$

end

$$\mu = (r^T P_k) / (P_k^T A P_k)$$

$$X_k = X_{k-1} - \mu P_k$$

$$r_k = r_{k-1} - \mu A P_k$$

$$\boxed{\text{Solve } M_Z c = r_k}$$

end

$$x_* = x_k$$

To highlight the difference between PCG and CG (Algorithm 11.32) we have boxed the preconditioner system $M_Z c = r$. It follows that the volume of work associated with a PCG iteration is essentially the volume of work associated with an ordinary CG iteration plus the cost of solving the preconditioner system. It can be shown that the residuals and search directions satisfy

$$r_j^T M^{-1} r_i = 0, \quad p_j^T (C^{-1} A C^{-1}) p_i = 0, \quad (11.52)$$

for all $i = j$.

We now turn our attention to the preconditioned GMRES method. The idea is to apply the method to the system $(M^{-1} A)x = (M^{-1} b)$. Modifying Algorithm 11.42 in this way yields the following procedure.

CLUEk2o 11.5.2 .= 1i. 159 (6(15i9, f166id. foc .suo. If $A \in \mathbb{R}^{n,n}$ and $M \in \mathbb{R}^{n,n}$ are nonsingular, $b \in \mathbb{R}^n$, $Ax = b$ and ϵ_m is a positive iteration limit, then this algorithm computes $\tilde{x} \in \mathbb{R}^n$ where either i solves $Ax = b$ or minimizes $\|M^{-1}(Ax - b)\|_2^2$ over the affine space $x_0 + K(I - A^{-1}r_{0,r})$ where $r_0 = b - Ax_0$

$$k = 0 \quad r_0 = b - Ax_0, \quad \boxed{\text{Solve } Mz_0 = r_0}, \quad f_0 = \|r_0\|_2^2$$

while ($f_k > 0$ and $k < m$)

$$q_{k+1} = Z_k/f_k$$

$$k = k + 1$$

$$\boxed{\text{Solve } Mz_k = Aq_k}$$

for $i = l:k$

$$h_{ik} = q_i Z_k$$

$$Z_k = Z_k - h_{ik}q_i$$

end

$$f_k = \|Z_k\|_2^2 \quad h_{kk} = 1$$

Apply G_1, \dots, G_{l-1} to $H(l:k, k)$ and determine G_k, R_k, P_k and P_k

end

Solve $R_k y_k = P_k$ and set $\tilde{x} = x_0 + Q_k y_k$

Note that $P_k = M^{-1}(b - Ax_k)$ in this formulation

sshihmb 4f kf Thittb uu69b .e Sf kt t sf TktSeeb

We now begin a tour of the major preconditioning strategies. Since some strategies help motivate others, the order of presentation is pedagogical. It does not indicate relative importance, nor does it reflect how the research on preconditioning evolved.

Suppose $A \in \mathbb{R}^{n,n}$ is diagonally dominant or positive definite. For such a matrix, the diagonal tells much of the story and so it makes a certain amount of sense to consider perhaps the simplest preconditioner of all:

$$\pi = \text{diag}(a_{11}, \dots, a_{nn}).$$

Diagonal preconditioners are called Jacobi preconditioners. Recall from §11.22 that Jacobi's method is based on the splitting $\bar{A} = M - N$ where M is the diagonal of A . Indeed, for any iteration of the form $Ax_+ = Nx_+ + b$ we can regard N as a preconditioner. The requirement that

$$p(N^{-1}N) = p(M^{-1}(M - A)) = p(I - J_r^{-1}\bar{A}) < 1$$

is a way of saying that N^{-1} must "look like" A^{-1} . In this context, the SSOR preconditioner

$$M = (D - vL)D^{-1}(D - vLf)$$

is attractive for certain symmetric positive definite systems. Note that in this case M is also symmetric positive definite and so it can be used with PCG.

If $A = (A_{ij})$ is a p -by- p block matrix that is (block) diagonally dominant or positive definite, then the block Jacobi preconditioner $M = \text{diag}(A_{11}, \dots, A_{pp})$ is sometimes a natural choice.

$$11m4A) A \cdot I = 1G + \omega I_0 \quad \{7Aa7d . LG - G.b = +A$$

Sometimes A is near a data sparse matrix for which there is a fast solution procedure. Circulant preconditioners for symmetric Toeplitz systems are a nice example. For $a \in \mathbb{R}^n$, define the Toeplitz matrix $T(a) \in \mathbb{R}^{n \times n}$ and the circulant matrix $C(a) \in \mathbb{R}^{n \times n}$ by

$$T(a) = \begin{bmatrix} & & & a_3 \\ a_2 & a_1 & a_0 & a_2 \\ & a_3 & a_2 & a_1 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix}, \quad C(a) = \begin{bmatrix} & & & a_3 \\ a_2 & a_3 & a_0 & a_2 \\ & a_1 & a_2 & a_3 \\ a_3 & a_0 & a_1 & a_2 \end{bmatrix}, \quad (n=4).$$

Suppose we determine (ω) so that $\|T(a) - \omega I\|_F$ is minimized. A case can be made that $M = C(d)$ captures the essence of $T(a)$ and thus has potential as a preconditioner for the Toeplitz system $T(a)x = b$. Recall from §4.8.2 that circulant linear systems can be solved in $n \log n$ time using the fast Fourier transform. This style of Toeplitz system preconditioning was proposed by Chan (1989).

Because of their importance, there is a large body of work concerned with preconditioners for Toeplitz systems. An idea due to Chan and Strang (1989) is to set $M = C(d)$ where

$$a = \begin{bmatrix} a(0) \\ a(m-1) \\ \vdots \\ a(1) \end{bmatrix}$$

assuming that $n = 2m$ and $A = T(a)$ is positive definite. Intuition tells us that A 's central diagonals carry most of the information and so it makes sense that they define the preconditioner $C(d)$.

$$11hw4r - adElir Baa Fw\$osir nri Elir tF i8 wx zszwriFlr$$

Instead of determining M so $\|A - M\|_F$ is small, we can address Criterion 1 above by choosing M^{-1} so that $\|AM^{-1} - I\|_F$ is small. This is the idea behind sparse approximate inverse preconditioners. To be precise about the nature of the approximation, we define the $\text{sp}(\cdot)$ operator. For any $T \in \mathbb{R}^{n \times n}$ define $\text{sp}(T) \in \mathbb{R}^{n \times n}$ by

$$[\text{sp}(\cdot)]_{ij} = \begin{cases} 1 & \text{if } t_{ij} = 0 \\ 0 & \text{otherwise} \end{cases}.$$

Suppose $Z \in \mathbb{R}^{n \times n}$ is a given n -by- n matrix of zeros and ones with a manageable sparsity pattern and that we solve the constrained least squares problem

$$\min_{\text{sp}(T) = Z} \|AT - I\|_p$$

The constraint says that T is to have the same zero/nonzero structure as Z . Thus, the preconditioner M is specified through its inverse $M^{-1} = T$. A fringe benefit of this type of preconditioner design is that the $MZ = I$ system is solved via matrix-vector multiplication $Z \leftarrow T$. This is what makes this preconditioning approach attractive from the parallel computing point of view. Moreover, the actual columns of T can be computed in parallel because they are independent of each other.

It is important to appreciate that $T(:, k)$ is a constrained minimizer of $\|Ax - ek\|_2$. Let cds be the subvector of $1:n$ that identifies the nonzero components of $T(:, k)$. (These indices are determined by $Z(:, k)$.) Let $rows$ be a subset of $1:n$ that identifies the nonzero rows in $A(:, cds)$. If A solves the (generally very small) LS problem

$$\min \|A(rows, cds)r - ek(rows)\|_2$$

then $T(:, k)$ is zero except $T(rows, k) = r$. We mention that the sparsity pattern Z can be determined dynamically. For example, after completing the above column k calculation, it is possible to expand cd cheaply to include more nonzeros in $T(:, k)$. See Grote and Huckle (1997). Updating QR factorizations is part of their method.

hhhiha b .kPf t kTtPb .x \$f ktkT f Tkt\$xb

Suppose $A = Af_1 \cdot N_1$ is a splitting and that $p(G) < 1$ where $G = M_1^{-1}N_1$. Since $A = M_1(J - G)$, it follows that

$$A^{-1} = (I - G)^{-1}M_1^{-1} = \left(\sum_{k=0}^{\infty} G^k \right) M_1^{-1}.$$

This suggests another way to generate a preconditioner whose inverse resembles the inverse of A . We simply truncate the infinite series:

$$M^{-1} = \left(\sum_{k=0}^m G^k \right) M_1^{-1}.$$

It follows that

$$z = (I + G + G^2 + \dots + G^m) M_1^{-1}r$$

solves $Jz = r$. Moreover, there is a very simple way to compute this vector:

```

Zc=0
for k=1:m
    Mz+=Nizc+r
    Zc=Zc
end
Z=Zc

```

To see why this works, we note that $Z = Gz_c + d$ where $Af_1d = r$, and apply induction:

$$z_+ = Gz_c + d = G(I + G + \dots + G^{k-1})d + d = (I + G + \dots + G^k)d.$$

Thus, the $Mz = r$ calculation requires m steps of the iteration $f_1 z_+ = Niz_c + r$.

In the polynomial preconditioner paradigm, the given system $Ax = b$ is replaced by $M^{-1}Ax = M^{-1}b$ where the preconditioner M is defined by

$$M^{-1} = p(M_1^{-1}A)M_1^{-1}. \quad (11.53)$$

Here, p is a polynomial and M_1 is itself a preconditioner; e.g., the diagonal of A . In the above example, p was determined by the parameter m and the chosen $\|A\|_1$.

We mention that there are more sophisticated ways to design a good polynomial preconditioner. With $M_1 = I - p(A)A$, the goal is for $p(A)$ to look like A^{-1} , i.e., we want $I - p(A)A \approx q(A)$. Note that $I - p(A)A = q(A)$ where $q(z) = 1 - zp(z)$, so the challenge is to find $q \in \mathbb{P}_m$ with the property that $q(0) = 1$ and $q'(0)$ is small. There are several ways to address this optimization problem in practice, see Ashby, Manteufel, and Otto (1992) and Saad (1985).

hhiiTb .. mde't stb

The polynomial preconditioner discussion points to an important connection between the classical iterations and the preconditioned conjugate gradient algorithm. Many iterative methods have^a their basic step

$$x_k = x_{k-2} + \omega_k(\gamma_{k-1}z_{k-1} + x_{k-1} - x_{k-2}) \quad (11.54)$$

where $\|z_k\|_1 = \|r_k\|_1 = \|b - Ax_k\|_1$. For example, if we set $\omega_k = 1$ and $\gamma_k = 1$, then

$$x_k = M^{-1}(b - Ax_{k-1}) + x_{k-1},$$

i.e., $\|x_k\|_1 = \|x_{k-1}\|_1 + \|b - Ax_{k-1}\|_1$ where $A = M - N$. Following Concus, Golub, and O'Leary (1976), it is also possible to organize the preconditioned CG method with a central step of the form (11.54):

X 1= Q k= 0 o= A Axo
while rk= 0 ? a

$$\left. \frac{z_{k-1}^T M z_{k-1}}{z_{k-2}^T M z_{k-2}} \frac{1}{\omega_{k-1}} \right)^{-1}$$

1h4AqA V1 f 2oA.D3o +TA af d1 LGG:f +A

Assume that $A \in \mathbb{R}^{n \times n}$ is symmetric positive definite and that we are driven to consider the PCG method because A's Cholesky factor G has many more nonzero entries than the lower triangular portion of A. A natural idea for a preconditioner is to set $M = HHT$ where H is a sufficiently sparse lower triangular matrix so that if

$$R = HHT - A \quad (1156)$$

then

$$a_{ij} \neq Q \quad r_{ij} = 0 \quad (1157)$$

This means that $[HHT]_{ji} = a_{ij}$ for all nonzero a_{ij} . In this sense, $R = HHT$ captures the essence of A. To articulate what we mean by a "sufficiently sparse" H matrix, we specify a set P of subdiagonal index pairs and insist that

$$(i, j) \in P \Rightarrow h_{ij} = 0. \quad (1158)$$

Given P, any matrix H that satisfies (1156)-(1158) is an incomplete Cholesky factor of A.

It turns out that it is not always possible to compute H given P. To see what the issues are consider the outer-product implementation of the Cholesky factorization. Recall from §4.2 that it involves repeated application of the factorization

$$\begin{bmatrix} \alpha & v^T \\ v & B \end{bmatrix} = \begin{bmatrix} \sqrt{\alpha} & 0 \\ w & I_{n-1} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & A_1 \end{bmatrix} \begin{bmatrix} \sqrt{\alpha} & w^T \\ 0 & I_{n-1} \end{bmatrix} \quad (1159)$$

where $w = v/.$ and $A_1 = B - vv^T$. Indeed, if G_1 is the Cholesky factor of A_1 , then

$$G = \begin{bmatrix} \cdot & 0 \\ w & G_1 \end{bmatrix}$$

is the Cholesky factor of A. Now suppose $Z \in \mathbb{R}^{n \times n}$ is a matrix of zeros and ones with $z_{ij} = z_{ji} = 0$ if and only if $(j) \in P$. To ensure the existence of an incomplete Cholesky factor with respect to P, we need to guarantee that the following recursive function works

```

function H = incChol(A,Z,n)
    if n = 1
        H = V
    else
        alpha = A(1,1), v = A(2:n,1), B = A(2:n,2:n)
        w = (v/.) .* Z(2:n,1)
        A1 = (B - vv^T) .* Z(2:n,2:n), H1 = incChol(A ,Z(2:n,2:n),n- 1)
        H = [ sqrt(alpha) 0
               w H1 ]
    end

```

If Z is the matrix of all 1's, then this is just a recursive form of Cholesky factorization (Set $r = 1$ in Algorithm 424). As it stands, it is

$$\tilde{b}_{ij} = b_{ij} - \frac{\tilde{v}_i \tilde{v}_j}{\alpha}.$$

$$\frac{1}{\sqrt{\alpha}}$$

$$0 < x^T A x = 1 - \frac{v^T B^{-1} v}{\alpha}.$$

Since $B^{-1} \geq 0$ and $v \leq 0$, we have $\tilde{v}^T B^{-1} \tilde{v} \leq v^T B^{-1} v$ and so

$$\gamma \equiv 1 - \frac{\tilde{v}^T B^{-1} \tilde{v}}{\alpha} \geq 1 - \frac{v^T B^{-1} v}{\alpha} > 0.$$

Using the Sherman-Morrison formula

$$\tilde{B}^{-1} = \left(B - \frac{\tilde{v}\tilde{v}^T}{\alpha} \right)^{-1} = B^{-1} + \frac{1}{\gamma} B^{-1} \frac{\tilde{v}\tilde{v}^T}{\alpha} B^{-1}$$

we see that \tilde{B} is positive definite. D

A theorem of this variety can be found in the landmark paper by Meijerink and van der Vorst (1977).

So far we have just discussed incomplete Cholesky by position. The sparsity pattern for the incomplete factor is determined in advance through the set P and does not depend on the values in A . An alternative approach makes use of a drop tolerance $\tau > 0$ which is used to determine whether or not a "potential" b_{ij} is set to zero. As an example of this strategy, suppose we compute the matrix A_1 in incomplete Cholesky as follows

$$[A_1]_{ij} = \begin{cases} 0 & \text{if } |b_{ij} - w_i w_j| < \tau \sqrt{b_{ii} b_{jj}}, \\ b_{ij} - w_i w_j & \text{if } |b_{ij} - w_i w_j| \geq \tau \sqrt{b_{ii} b_{jj}}. \end{cases}$$

The idea is to drop unimportant entries in the update if they are small in a relative sense. Care has to be exercised in the selection of τ , so as not to induce an unacceptable level of fill-in. (Larger values of τ reduce fill-in) The drop tolerance approach is an example of incomplete Cholesky by value.

Lin and More (1999) describe a strategy that combines the best features of incomplete Cholesky by position and incomplete Cholesky by value. Recall in gaxy Cholesky (§4.25) that the triangular factor G is computed column by column. The idea is to adapt that procedure so that $H(j:n:j)$ has at most $N_j + p$ nonzeros, where N_j is the number of nonzeros in $A(j:n:j)$ and p is a nonnegative integer:

$f(r:j-1:n)$

$$v(j:n) = A(j:n:j) - H(j:n:j-1)H(j:j-1)^\top$$

$$H(j:j) = y$$

$N_j = \text{number of nonzeros in } A(j:n:j)$

Set to zero each component of $v(j:n)$ that is not one of the $N_j + p$

largest entries in $M(j:n:j)$

$$H(j+1:n:j) = v(j+1:n)/H(j:j)$$

end

It follows that the number of nonzeros in H is bounded by $p + N_1 + \dots + N_n$. Thus the value of p can be set in accordance with available memory. Note that $M(j:n:j)$ is defined by the "most important" entries in $v(j:n)$. The gaxy computation of this vector is a sparse gaxy, and it is critical that this structure be exploited.

$$A = \begin{bmatrix} A_1 & E_1^T & 0 \\ E_1 & A_2 & E_2^T \\ 0 & E_2 & A_3 \end{bmatrix} = \begin{bmatrix} G_1 & 0 & 0 \\ F_1 & G_2 & 0 \\ 0 & F_2 & G_3 \end{bmatrix} \begin{bmatrix} G_1^T \\ 0 \\ 0 \end{bmatrix}$$

$$- E_k \Lambda_k E_k^T$$

$$\begin{bmatrix} \tilde{G}_1 & 0 & 0 \\ \tilde{F}_1 & \tilde{G}_2 & 0 \\ 0 & \tilde{F}_2 & \tilde{G}_3 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix},$$

Each \mathbf{W}_k requires a G_k -systems solution and a \mathbf{e}_k -systems solution.

There remains the issue of choosing $\Lambda = \Lambda_1, \dots, \Lambda_{p-1}$. The central problem is how to determine a symmetric tridiagonal Λ so that if $T \in \mathbb{R}^{n \times n}$ is symmetric positive definite and tridiagonal itself, then $\Lambda \approx T^{-1}$. Possibilities include:

- Let $\Lambda = \text{diag}(1/t_1, \dots, 1/t_m)$,
- Let Λ be the tridiagonal part of T^{-1} , an $O(m)$ computation. See P11.55
- Let $\Lambda = \mathbf{L} \mathbf{U}$ where \mathbf{L} is the lower bidiagonal portion of K^{-1} where $T = \mathbf{KKT}$ is the Cholesky factorization. This is an $O(m)$ computation. See P11.56

For a discussion of these approximations and what they imply about the associated preconditioners, see Concus, Golub, and Meurant (1985).

hhlihhsb uttfSb 2Ttf buKef Sb 2 Sf kt Tf TkSb

A nonsingular 2by2block system of the form

$$K = \begin{bmatrix} A & B_1^T \\ B_2 & -C \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix},$$

where $A \in \mathbb{R}^{n \times n}$ is positive semidefinite and $C \in \mathbb{R}^{m \times m}$ is symmetric and positive semidefinite. An example of a saddle point problem Equilibrium systems (§4.4.6) are a special case.

Problems with saddle point structure arise in many applications and there is a host of solution frameworks. Various special cases create multiple possibilities for a preconditioner. For example, if A is nonsingular and $C = Q$ then

$$\begin{bmatrix} : & \mathbf{r} & \mathbf{1} \\ & \mathbf{B}_2^T & : & -\mathbf{1} \end{bmatrix} = \begin{bmatrix} : & & \\ & \mathbf{B}_2^T & : & -\mathbf{1} \end{bmatrix} \begin{bmatrix} : & & \\ & \mathbf{B}_2^T & : & -\mathbf{1} \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{B}_1 \\ \vdots & \end{bmatrix}. \quad S = -B_2^T A^{-1} B_1.$$

Possible preconditioners include

$$\mathbf{P}_{\text{fl}} = \begin{bmatrix} & & \\ & \mathbf{i} & \\ & & \end{bmatrix} \text{ or } \begin{bmatrix} & & \\ & \mathbf{i} & \\ & & \end{bmatrix} \text{ or } \begin{bmatrix} \tilde{A} & \\ B_2^T & \end{bmatrix} \begin{bmatrix} & \mathbf{A} & \mathbf{B}_1 \\ & \vdots & \end{bmatrix}$$

where $\mathbf{A} = A$ and $\mathbf{S} = S$.

If A and C are positive definite, $H_1 = (A + AT)/2$, $H_2 = (A - AT)/2$, and $B = B_1 - B_2$ then

$$\begin{bmatrix} A & B \\ -B^T & C \end{bmatrix} = \begin{bmatrix} H_1 & 0 \\ 0 & C \end{bmatrix} + \begin{bmatrix} H_2 & B \\ -B^T & 0 \end{bmatrix} \equiv K_1 + K_2$$

is a symmetric positive definite/skewsymmetric splitting. Preconditioners based on

$$\mathbf{M} = (\mathbf{d} + \mathbf{K}_2)^{-1}(\mathbf{d} - \mathbf{K}_1)(\mathbf{d} + \mathbf{K}_1)^{-1}(\mathbf{d} - \mathbf{K}_2)$$

where $a > 0$ have been shown to be effective. See the saddle point problems survey by Benzi, Golub, and Liesen (2005) for more details. Note that the above strategies are specialized ILU strategies.

11t 4t11A 9pmA 9i .pV. +GlmA a7d .K n|m67 +A

Domain decomposition is a framework that can be used to design a preconditioner for an $Ax = b$ problem that arises from a discretized boundary value problem (BVP). Here are the main idea behind the strategy:

Step 1. Express the given "complicated" BVP domain as a union of smaller, "simpler" subdomains $\Omega_1, \dots, \Omega_n$.

Step 2. Consider what the discretized BVP "looks like" on each subdomain. Presumably, these subproblems are easier to solve because they are smaller and have a computationally friendly geometry.

Step 3. Build the preconditioner M out of the subdomain matrix problems, paying attention to the ordering of the unknowns and how the subdomain solutions relate to one another and the overall solution.

We illustrate this strategy by considering the Poisson problem $u = f$ on an L-shaped domain Ω with Dirichlet boundary conditions. (For discretization strategies and solution procedures that are applicable to rectangular domains, see §4.84.)

Refer to Figure 11.5.1 where we have subdivided Ω into three non-overlapping rectangular subdomains Ω_1, Ω_2 , and Ω_3 . As a result of this subdivision, there are five

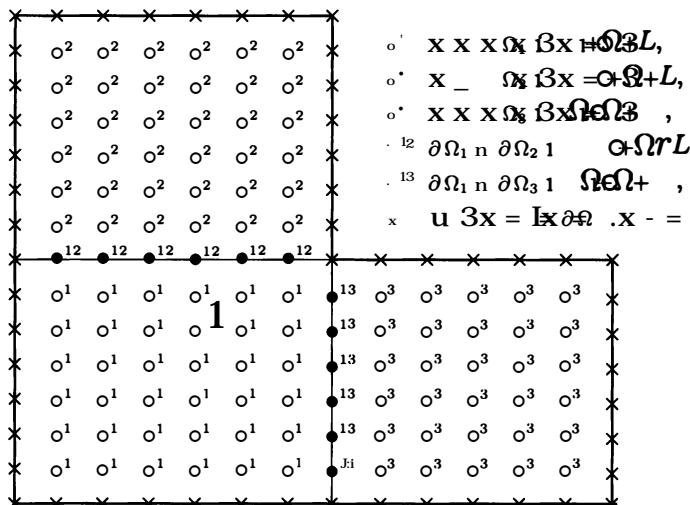


Figure 11.5.1 The Nonoverlapping subdomain framework

"types" of gridpoints (and unknowns). With proper ordering this leads to a block linear system of the form

$$Au = \begin{bmatrix} A_1 & 0 & 0 & B & C \\ 0 & A_2 & 0 & D & 0 \\ 0 & 0 & A_3 & 0 & E \\ F & H & 0 & Q_4 & 0 \\ G & O & K & 0 & Q_5 \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{bmatrix} = \begin{bmatrix} f_{\Omega_1} \\ f_{\Omega_2} \\ f_{\Omega_3} \\ J_1 \\ J_2 \end{bmatrix} = f \quad (11.5.1)$$

where A_1, A_2 and A_3 have the discrete Laplacian structure encountered in §4.8. Our notation is intuitive: $u_{\bullet,12}$ is the vector of unknowns associated with the \bullet grid points. Note that A can be factored as

$$A = \begin{bmatrix} I & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ FA_1^{-1} & HA_2^{-1} & 0 & I & 0 \\ GA_1^{-1} & 0 & KA_3^{-1} & 0 & I \end{bmatrix} \begin{bmatrix} A_1 & 0 & 0 & B & c \\ 0 & A_2 & 0 & D & 0 \\ 0 & 0 & A_3 & 0 & E \\ 0 & 0 & 0 & \mathbf{84} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{88} \end{bmatrix} = LU,$$

where $\mathbf{84}$ and $\mathbf{88}$ are the Schur complements

$$\mathbf{84} = Q_1 - FA_1^{-1}B, \quad \mathbf{88} = Q_3 - GA_1^{-1}C - KA_3^{-1}E.$$

If it were not for these typically expensive, dense blocks, the system $Au = f$ could be solved very efficiently via this LU factorization. Fortunately, there are many ways to manage problematic Schur complements. See Saad (IMSL, pp. 456–465). With appropriate approximations

$$\tilde{S}_4 \approx S_4, \quad \tilde{S}_5 \approx S_5,$$

we are led to a block ILU preconditioner of the form $M^- = LU$ where

$$M^- = \begin{bmatrix} A_1 & 0 & 0 & B & c \\ 0 & A_2 & 0 & D & 0 \\ 0 & 0 & A_3 & 0 & E \\ 0 & 0 & 0 & \mathbf{84} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{88} \end{bmatrix}.$$

With sufficient structure, fast Poisson solvers can be used during the L -solves while the efficiency of the U solver would depend upon the nature of the Schur complement approximations.

Although the example is simple, it highlights one of the essential ideas behind nonoverlapping domain decomposition preconditioners like M . Bordered block diagonal systems must be solved where (a) each diagonal block is associated with a subdomain and (b) the border is relatively "thin" because in the partitioning of the overall domain, the number of domain-coupling unknowns is typically an order of magnitude less than the total number of unknowns. A consequence of (b) is that $A - M$ has lowrank and this translates into rapid convergence in a Krylov setting. There are also significant opportunities for parallel computation because of the nearly decoupled subdomain computations. See Bjorstad, Gipp, and Smith (1996).

A similar strategy involves overlapping subdomains and we continue with the same example to illustrate the main idea. Figure 11.5.2 displays a partitioning of the same L-shaped domain into three overlapping subdomains. With proper ordering we obtain

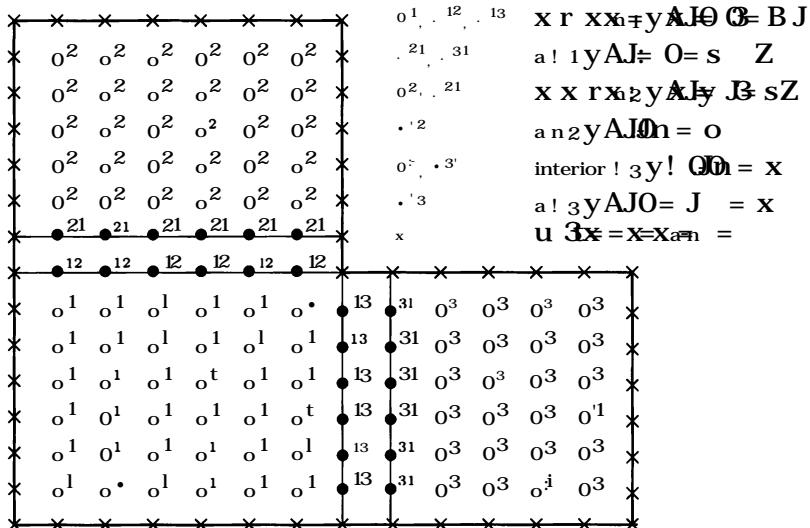


Figure 1152 The overlapping Schwarzf framework

a block linear system of the form

$$Au = \begin{bmatrix} A_1 & 0 & 0 & B_1 & 0 & C_1 & 0 \\ 0 & A_2 & 0 & 0 & B_2 & 0 & 0 \\ 0 & 0 & A_3 & 0 & 0 & 0 & C_2 \\ F_1 & 0 & 0 & Q_1 & D & 0 & 0 \\ 0 & F_2 & 0 & H & \tilde{Q}_4 & 0 & 0 \\ G_1 & 0 & 0 & 0 & 0 & Q_5 & E \\ 0 & 0 & G_2 & 0 & 0 & K & \tilde{Q}_5 \end{bmatrix} \begin{bmatrix} u_{o^1} \\ u_{o^2} \\ u_{o^3} \\ u_{\bullet^{12}} \\ u_{\bullet^{21}} \\ u_{\bullet^{13}} \\ u_{\bullet^{31}} \end{bmatrix} = \begin{bmatrix} /_{o^1} \\ /_{o^2} \\ /_{o^3} \\ /_{\bullet^{12}} \\ f_{\bullet^{21}} \\ /_{\bullet^{13}} \\ f_{\bullet^{31}} \end{bmatrix} = f.$$

In the multiplicative Schwarz approach we cycle through the subdomains improving the interior unknowns along the way. For example, fixing all but the interior Ω_1 unknowns, we solve

$$\begin{bmatrix} A_1 & B_1 & C_1 \\ F_1 & Q_4 & 0 \\ G_1 & 0 & Q_5 \end{bmatrix} \begin{bmatrix} u_{\bullet^1} \\ u_{\bullet^{12}} \\ u_{\bullet^{13}} \end{bmatrix} = \begin{bmatrix} f_{\bullet^1} \\ f_{\bullet^{12}} \\ f_{\bullet^{13}} \end{bmatrix} - \begin{bmatrix} 0 \\ Du_{\bullet^{21}} \\ Eu_{\bullet^{31}} \end{bmatrix}.$$

After updating u_{01} , u_{12} , and u_{13} we proceed to fix all but the interior Ω_2 unknowns and solve

$$\begin{bmatrix} A_2 & B_2 \\ F_2 & \tilde{Q}_4 \end{bmatrix} \begin{bmatrix} u_{\circ^2} \\ u_{\bullet^{21}} \end{bmatrix} = \begin{bmatrix} f_{\circ^2} \\ f_{\bullet^{21}} \end{bmatrix} - \begin{bmatrix} 0 \\ Hu_{\bullet^{12}} \end{bmatrix},$$

and update $u_{\circ,2}$ and $u_{\bullet,21}$. Finally, we fix all but the interior 13 unknowns and obtain improved versions by solving

$$\begin{bmatrix} A_3 & C_2 \\ G_2 & \tilde{Q}_5 \end{bmatrix} \begin{bmatrix} u_{\circ^3} \\ u_{\bullet^{31}} \end{bmatrix} = \begin{bmatrix} f_{\circ^3} \\ f_{\bullet^{31}} \end{bmatrix} - \begin{bmatrix} 0 \\ Ku_{\bullet^{13}} \end{bmatrix}.$$

This completes one cycle of multiplicative Schwarz. It is Gauss-Seidel-like in that the most recent values of the current solution are used in each of the three subdomain solves. In the additive Schwarz approach, no part of the solution vector \mathbf{u} is updated until after the last subdomain solve. This Jacobi-like approach has certain advantages from the standpoint of parallel computing.

For either the multiplicative or additive approach, it is possible to relate $u^{(k)}$ to $u^{(j)}$ via an expression of the form

$$u^{(\text{new})} = u^{(\text{old})} + M^{-1}(f - Au^{(\text{old})}),$$

which opens the door to a new family of preconditioning techniques. The geometry of the subdomains and the extent of their overlap critically affects efficiency. Simple geometries can clear a path to efficient subdomain solving. Overlap promotes the flow of information between the subdomains but leads to more complicated preconditioners. For an in-depth review of domain decomposition ideas, see Saad (IMSL, pp. 451-493).

Problems

P11.5.1 0.1 (11.5.2).

P11.5.3 nf8ro6..

$$A = \begin{bmatrix} 1 & 1 & 3 & 0 \\ 1 & 2 & 0 & 3 \\ 3 & 0 & 19 & -8 \\ 0 & 3 & -8 & 11 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 3 & -3 & 1 & 0 \\ 0 & 3 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 3 & 0 \\ 0 & 1 & -3 & 3 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$SM.apMiAIt \quad Ap + M8.0tut \cdot (Matareh TA.iMl63Ai6M-P) = \{ (4), (3, 2) \}.$$

P11.5.4 8. **P** | $(8.1.5.6)-(11.5.8)$ (*M09Section11.htm*)
 $\Rightarrow \text{Eq. 11-2a}$, $L = 1 = T$

P11.5.5 **lde8** $\mathbf{R}^{m \times m}$, $S^n((3, n^n, n^{\frac{n}{m}}) \times 3, \beta^n n^3(1^n)^m 3^{\frac{n}{m}})$

= WmX

t= a 11qpS - i(ci ,qi6,a 1 < i < m. wG O(m) 2(x+)k(+ =)2x.r

P11.5.6 nlddgg $\mathbf{R}^{m \times m}$, $n_{nn}, 3^n$, $(n-1)3^3, n^n$, $((3^n \times \beta^n) \cdot \mathcal{O}(m)) = 2(n + \beta)$

P11.5.7 38.1 8f8 48tdl.. 8.(11.5.10). *Iro.M,t : w..g p a -@Ex XO=-.JaJa = L-+O= J a = L Ja J. M.Q.7 qIt ro..tl XApS.+Si.0 ApS0M.dl.ApS.4Si. r. LJE=aa = L-nH=O= a J = QC... InixAlt pttaaAh 6Tc1,...,Gp a = Ga > O 1 - O = + aa = O- .L = J*

P11.5.8 m61d.8.82 d.8.1 I.f 1. 1.6..8 8.f6o.8.1d§14..1.m46rob.. .. 1.1.8
n46rob.. b t8o8.8. 68.84 .. .81- ...8t

$$Au = \begin{bmatrix} A_{11} & A_{12} & 0 \\ A_{21} & A_{22} & A_{23} \\ 0 & A_{31} & A_{33} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} = !$$

/Aer /e ogSle oAoA22R.sam,iggAMia ARnnaANao A33 3,s.A nmin i))MlbR.inA
lglInPla(k) PRC)Monlu(k+1) iRinnmAggl+P.CsgnP gPlainPiA M1Snl)MlaIsMHe

$$\begin{bmatrix} A_n & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} a_1^{(k)} \\ a_2^{(k)} \end{bmatrix} = \begin{bmatrix} : \\ A_{21} & A_{22} & J \end{bmatrix} \begin{bmatrix})t \\ U_a^{(k)} \end{bmatrix},$$

$$\begin{bmatrix} A_{22} & A_{23} \\ A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} a_1^{(k)} \\ a_3^{(k)} \end{bmatrix} = \begin{bmatrix} A_{21} & A_{22} & A_{23} \\ J & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix})t \\ L_t^{(k)} \\ U_3^{(k)} \end{bmatrix},$$

$$\begin{bmatrix} u_1^{(k)} \\ u_2^{(k+1)} \\ u_3^{(k+1)} \end{bmatrix} = \begin{bmatrix} U_1^{(k)} \\ U_2^{(k)} \\ U_3^{(k)} \end{bmatrix} + \begin{bmatrix} a \\ a_2^{(k)} \\ a_3^{(k)} \end{bmatrix}.$$

f iTEArMPaACinMRb(:C2 u(k+1) = u(k) + M-1(I - Au(k)). f STA)Ain1Mnm)eoPnRiA Lam+ iMoinAr

$$\begin{bmatrix} A_u & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} a_1^{(k)} \\ a_2^{(k)} \end{bmatrix} = \begin{bmatrix} : \\ A_{21} & A_{22} & J \end{bmatrix} \begin{bmatrix} u_1^{(k)} \\ U_2^{(k)} \\ U_3^{(k)} \end{bmatrix},$$

$$\begin{bmatrix} A_{22} & A_{23} \\ A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} a_1^{(k)} \\ a_3^{(k)} \end{bmatrix} = \begin{bmatrix} A_{21} & A_{22} & A_{23} \\ J & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} u_1^{(k)} \\ u_2^{(k)} \\ u_3^{(k)} \end{bmatrix},$$

$$\begin{bmatrix} u_1^{(k)} \\ u_2^{(k+1)} \\ U_3^{(k+1)} \end{bmatrix} = \begin{bmatrix} U_1^{(k)} \\ u_2^{(k)} \\ U_3^{(k)} \end{bmatrix} + \begin{bmatrix} 1 \\ a_2^{(k)} \\ a_3^{(k)} \end{bmatrix}.$$

IM8MnnM,Pa. AAMAASis.f ,nL.,))M(. dr I(T .

Notes and References for

P.... d.d8.48.48.0 robRt86. croRg,R4...8

- .4 3bAg,la(076)69D(2)Bu5 --u['3 1r "NIT,Z,,5 ,ATv
EM9Ma, (07z)(9 3. D28(u Ir D: 8t k12 -1)I Bu) D8)Du34(u(3 euk BD2<
'3 1r TMathematics of Finite Elements and Applications, 1984, f CT3aoACP-
hMA,2A+IMQITr ,T.
1,4 wiMnAp.4ai EiaRAg(7)(9D)12)R (Ir u2< D:)II 3 4 518u(1) B8u 211 25
)) EN I1384(TN(Conference on the Numerical Solution of Differential Equations, Dundee,
t .434ain,la f oTL)MPAM AMgfr. II MB4
1,L4Amiao4A9P,Aa,niiae n4, Lmsg(H)7)T 91(NK (Ir u2< '81r (1) I28
'2,)2dB I 212TN2Advances in Computer Methods for Partial Differential Equations, IM
BPamaAiAf,BelTrnAM, aPAM,R2A+wMsa,+ R2B.
0. yd0< + (074)€ 9D8(1" 3)213r (7) e2uBD< 1 all("NComputer Methods
in Applied Mechanics and Engineering ,95N5T,
hMalaas..M.M .lgisSo EMQAI Mt(H)7)(9D,(0)E21r]1(),k (Bu2< '315 7)
'3 5 D2t B-18 u1211B8u 20)28 u,)2dB F I u1("NuSparse Matrix Computations,
Mlwsm.o EMIM f AE.3aoACPMA,2A+IMB5J,r,51.
.4 Elsrg Mtae a4Eslan (07)(9)t 125u212r l1(NK (02< 4k1D: 8ur 4
(8)u '31r (7) I '8r 85128) u)u88)I138D TN2Sparse Matr Computations,
.4 4wsaariao E,,4l,A f AE.3-icACPMA,2A+ 1 MG55r,dr

IMia liAMiPA+)MAalRdPamAamaRcs,AM MnAaSis.f ,nIT .nAsMiaf,AT . Liie f ,L
h,3T. v = x = 2€,q L0_TEMPLATES . - Or &• Q < - +ne y > 0e+ t

- o. ydO A o.020 pDOp A > OX A On = 03= 3= = 10= wTA O A A A On > 02= p- an > c
 BIT 25. (77nDtU
 n4wAa.RJJ.r .hAalaePpPlaRari.rA Ria sttpACte8 ss.iAtT J. Comp. Phys. 182,
 ,Ndr TTM
 hiSA.talaaA.aE+ Rp iAtA i)).lbPiipA PaiAtA AalaePpPlaRata, IeAe
 n4wAa.RA 4E14AtA.. iae ri4 ,N.7.M3 sSi.tA3SS.lbPiipAaiAtl hAalaePpRlaA. pmA
 Ala)IripAiePAApmle SIAM J. Sci. Comput. 17, N5 N
 94Aml+iae i9siie ,N.T.Y.3S).lbRCipAaiAtAaAayaPAt1 wglabR hi.pRpPlaRataAf
 SIAM J. Sci. Comput. 18, N.Tr (7T.Y
 n4.4.ipA iae4.IaB,A,N T.Y.hi.ig.AghAalaPpPlaRap sSi.tA 3S).lbP.ipA hiA.tA6
 SIAM J. Sci. Comput. 18, d5dd.5Y
 24 4tge iaJ .434 sAipP Y.sSi.tA 3SS.lbRCipAAtAhAalaPpRlaAtpApl.f
 nRaRiPPlaAaR -sAISIAM J. Sci. Comput. 19, 7J,r71.M
 n4wAa.Pae n4 ii ,N.d.Y.3 sSi.tA 3SS.lbRCipAAtAhAalaPpRlaA.2latt.iAp.Pa
 ,PaAi.sttpACt.TSIAM J. Sci. Comput. 19, .7dr .., Y
 Bi.PII titSAapt)lgtaI C P g AalaeRpPlaRataPtaI ttAae
 ,44 .Imatla.AM 3PamgRae.u hiI, oN15.Y.hl,taliRi, hAalaeRpRlaA.tAla)IripAim
 ePapAi,algPlaTSIAM J. Numer. Anal. 20, 57Ir 5T7Y
 ,43eiCt ,(d..u.i.tpS hAalaePpRlaAaR IripAiePAApmle6SIAM J. Sci. Stat. Com-
 put. 6, I ,75Y
 s43tmSta4niaplI), iae h4sitgl , (d. Y.3eiSpPiA, talCRighAalaPpPlaRata.A.iPjRia
 keA=aRPAI.sttpACtBIT 29, d5r7J.v
 14a4; Al ae,N J.u.,a AlaelripAiePAAapt)A nApmlt hglalCRighAalaRpPlaA.ti
 Agt l. Ali),Ab 2laLA.CPpRnlpPaAtNumer. Math. 57, 1dr5(IY
 s43tmStaMiiapAsAgae.Mopl ,N I.Y.3 Ali)i.Ptla l. 3eiSpPiA mAStrnAI,A.p s.-si.At
 hl,talCRighAalaPpPlaRarA.CRpRhtPpRFA=aRPAI.sttpAitTSIAM J. Sci. Stat.
 Comput. 13, Nr 1.Y
 amRAalCSgApA) tBti apl.PeiPpRaeARTapl.pmaiie iaigt.AePae
 .434nAPA.RaBe .434iiia eA.BltP, NTTM3a.1pA.ipPiAsgpRlaApmlA ,PaAi. 9-IipPla
 sttpAitl. amRapmAlA aPAPRlRt stCiAp.Ra ,IipPl6Math. Comput. 31, N,d N71Y
 a434hiapAIAGNDJ.Y.3a faaliS,ApApL.P.ipPlaAaP-IA hltRpPEA=aP,PAAI.sttM
 pACT Math. Comput. 34, T5r.TY
 A4L.Paiae.M.M.i ,N.. Y.1aAC)gApAmlgAtBtP.ipPlat + PpRiPpAJ nAistTM
 J. Sci. Comput. 21, I,r.,u
 ,Ri+ PtA. prhRAalCSgApA apl.P.ipRiAipArt AmiiA
 n4wl,,mA.iae I; siie ,IJJ.Y.YnI gpRiAigAalaPpRlaAtp.sapAjl. f aiA.tA,wAe,r tT
 SIAM J. Sci. Comput. Zt. NTr (7JY
 .49gCiaoNd7Y SpiSRgBaigtRt TaliS,ApA iapl. R.ipPi6Math. Comput. .tN.NNdY
 halCSgApA i apl.P.ipRlaiiAgtl SAAeAiRtAaAaC.aB 2n,s.)S4I.T I...+ A,,
 -x.Ri,N.d Ma 1nf -TeayAaA(C)gApApmlrlaigP.ipPlaApmlA ,irA r attCiAp.Pa ,RaAi.
 sttpACt6Numer. Lin. Alg. 3 ,Nr.NIu
 -c,-c wiR4sEI . iae 34.4ipmAAJN.MB A..t l. hal.)gApApmlrlai, i apl.R.ipRla
 nApmlt6nApmlre amAl.P. BIT 41, .5V7Y
 f aaliS,ApAglab apl.R.ipPlaA JRtaIttAae
 .4 lleA.PriAaeEul gPpANd. Y.hAadaPpPlaRalaCSgApA BAta,PlAelaplAtMath
 Comput. 42, .. 77u
 h4AalaI t.M.4l,sS. iae .4 nAs.iap,(d..u.wglabR hAalaPpRlaPapmAlaelripAiePAAp
 nApml6SIAM J. Sci. Stat. Comput. 6, IIJ.I.TY
 o. ydO< o.020 pDwn = 2< 0= 0Ha = RaA = AnMAOnm = n=AOn = ydO< 0Xa = 0
 y= o-10g. Comput. Appl. Math. 12-13, 5 (dY
 o. ydO< o.020 pDly10= OAx= A= 2f A= nHa = RaA = A3AO= 0-N. Alg. Applic.
 14, NTF NJM

9AAcos((18+ar1SeSe 8srFos)a) (isAt+MJSs at +1mcos (AO

HB4sou Ay,1B He=1s1(ra ktl xt ar1Se b)srFosA keyArt .5 J. Comput. Math. 40 NINNN

.MI iae o. yd0< on2a. ,e= On= ly=ak<one n= n= X<On= = = O:a= 1000
y.. < Ona= n=<<-1=n1n02N Numer. Math. 78 Nd IJQi

IMwlgdmiae Ii Liie ,IJJIM,a pmAgipT lafApHra ,r tae EaplMA)MlbT .ipfa iAMU SIAM J. Matrix Anal. Applic. 24UN.I2Tu

2I.AMII tiAaplMt)iMfpgAgAapipTf lptn)MAalaeT pTala1pmlemiiAS)m 55Ag)AeU tAAr

.MIAIMiaPd iMamAglBhMAalaeT pTlaAe ipA .MielAppla BAaplM)IpAMif6
BIT 24 715r 752M

AMAmMiae lMMR.ANDDla BaplMR-fadAgApA aplMS-dje IL,r fMAalaeT
pT lafApHra ,r tae EaplMA)MlbT .ipfa iAMU SIAM J.

A ~~co~~set ~~S~~A ~~co~~seasnl ~~In~~ on one+os+in

TLStat 0.9. mil 1000+use QASn1:Sens Prv)mt+ +mer(m Cen+sent ProSSS1:SsU
et=5SIAM J. Sci. Stat. Comput. 10, J N7J.4

sMA!MA..A!N!.S!aM!A Alaa ~~AsCSAM~~ pyA saysM!+)gA+A.p .El+iR.EAal+)lRp- l,6
Numer. Math. 83, No 1, Jan 54

Myiar 3 JJ 4 The Schur Complement and its Applications, s) MFA..BAMg^{2A}+II MBu
 z. n a i 00a a+ 022pDOn• y1=jj Oo= A = 0 = 01+ n10 O= < a 1 nKeyM. < OT On =
 X= Onn= 0 = 5104 JNSci. Comput. 27, NIIf NIN 4

JMia liAMiRM+pmlCiRaeAalC)ltRpRJMie- rCAAE3C+A82,3,))M5iTr ,\\$. + Agg
T

a,x Ayi i.e a,hMipyA+3Ni S4.El+i- . EAr IC)ltRpqgRIMRpp6Acta Numerica 3 uMM / E.
 uw, ipl33em .IRMaHAN clue. .aLls3LA2t,I. Q,t9,ILes PaA.ßau.auant,TsiAM
J. Sci. Statist. Comput. [7Tr ..54]

EMSat, aM.ayi.'.. nHsMips, x Mlrrt' lui, Bl- rPlotS N3.1SuDomain Decomposition Methods for Partial Differential Equations s.3n, hsSeRallat, hvBhAg)imBi M

Methods for Partial Differential Equations, SIAM, Philadelphia, 1988; and Matrix Analysis and Applications, SIAM, Philadelphia, 1990.

E+IR, EAaI+)IIR, pRIAPYIOOSIAM J. Matrix Anal. Appl. 1, 175-194
 WMI - phyuelMtpie aMII) 3N.7S4Domain Decomposition- Parallel Multilevel Methods for
 Elliptic Partial Differential Equations. Aii+SMR orA - iINNAAIA +SMR orAM 5

Elliptic Partial Differential Equations, AI+SMRerA - 11MMPAI+SMRerA15,
 M s.e.. ls3N1 S.sl+I 2aliA.gi)R.r ElCiR.EAl+)ltRpRlAplet, 6 SIAM Review 40
 1 TrNi4

3. altAg.i.n o . Ln0< oyn ⑩ Domain Decomposition Methods: Theory and Algorithms, S)MRAf
BAM-2011 MB

JMR.tRr(pApMpApMgA)MAae p- laRMgA.ptc.iAt)MISgAate +lMAaHMigUAMRaig
DpRC - ipHr

h9M gga MMF,wbl,aAgI aae nM3sisaeAMNI S4,hMlaeRp-1AMtaeA,aRpA
sttP*C*Mt-*n*) pB+ pi U6SIAM L Matrix Agar Applie L LUr5NN4

ETNA 8, 176-4
 nMVAa-iDe nMsCi 3 JJJ54.3 llSstp lMAAl.e p-lamph nAClMlAcs MA+AlMiMrA
 D-MiMrA At lMIS & QM A S i G i L 4

s) iMthAitp scsi. At hMISgA SIAM J. Sci. Comput. 14
 nMal ta.h, A MfA , i.e nMbuLisaeArJ JS4LsSt)iaAhM3aleRp lsAvr 1ME- taMAp

gentle IMISgA6BIT 4[Tr.lu
0. yd< oe=nk<kn Oe= nJG> a DXAOna= n=H0= J=yo= OaOeo= Ox-ono ny

$e = n_0 = -ia - n_{\infty} = n_{\text{neon}}$ numer. Lin. Alg. Applic. , 0 5N4
 3.III, M- iA- ~~MA~~ E M A M I M T A. 3 JJS4 3A+Agt lf hAal.n- p L A M I r A, saig A R. A i.

,pmAReAitlaRipAo + BpAalRp- l- aragseA - Atpgppalp pyAMMAalaeRpRtAC

J.) a y < ETaea < 0 = 13L oann0 er nfaZik Dy0a := X> One n0 Hj n=0 00

y< yn=JTS DAN J. Sci. Comput. II.5.II.u
 .M.Mg.S iaeV.IA3N. S4.,abiap hMAal.e pRMr HeripAMieRAapApyle- RpmAMr „pAM

pHPh'6SIAM J. Sci. Comput. „N5 N5IJ4
I 2mit d LHM gAbPS MgrinAMic Ab SIAM J. S. C. + ss win NiZL4

9MMI-CAT(R) pmMAl p- pR1Me al pAbta tRtasttAPe

0. $yD = \frac{1}{2} \ln \left(\frac{1 + \sqrt{1 + 4y^2}}{1 - \sqrt{1 + 4y^2}} \right)$

e = n lo yaa = On= reθn= la Numer. Lin. Alg. 8 17r f/74
z. 1 = a xPpnJe 20D- = = nT = n = = On= reθn= lo yaa = On= NBH 45,
 INT 4

11.6 The Multigrid Framework

Let $A_{h,h,j}$ be a linear system that arises when an elliptic boundary value problem is discretized on a structured grid. The discrete Poisson problems that we discussed in §483 and §484 are examples. The superscript ' h ' is a reminder that the size of the system depends on the fineness of the grid, i.e., the spacing between gridpoints.

The multigrid idea exploits relationships between the "fine grid" solution u_h and its smaller, "coarse grid" analog $u_{\frac{h}{2}}$. Given a current approximate solution u , the overall framework involves recursive application of the following strategy:

Pre-smooth With $u_{S_{h,j}} = u$, perform $m_{P,1}$ steps of a suitable iterative method $u_{S_{h,j}} \leftarrow Gu_{S_{h,j}}$ to produce u_j , an error-smoothed version of u .

Step 1. Compute the current fine-grid residual $r_h = b_h - Ah_j$. This vector will be rich in certain eigenvector directions and nearly orthogonal to others.

Step 2. Map r_h to $\mathbf{r}_{\frac{h}{2}}$, a vector that defines what the fine-grid residual looks like on the coarse grid corresponding to $\frac{h}{2}$. This will involve an averaging process.

Step 3. Solve the much smaller coarse-grid correction system $A_{\frac{h}{2},\frac{h}{2}} \mathbf{z}_{\frac{h}{2}} = r_{\frac{h}{2}}$

Step 4. Map $\mathbf{z}_{\frac{h}{2}}$ to \mathbf{z}_h , a vector that defines what the correction looks like on the fine grid. This will involve interpolation.

Step 5. Update u to $u_{j+1} = u_j + \mathbf{z}_h$

Post-smooth With $u_{S_{h,j+1}} = u_j$, perform $m_{P,2}$ steps of a suitable iterative method $u_{S_{h,j+1}} \leftarrow Gu_{S_{h,j+1}}$ to produce u_{j+1} , an error-smoothed version of u_j .

Our plan is to discuss the key issues associated with this paradigm using the 1-dimensional model problem introduced in §483. The weighted Jacobi method is developed for the pre-smooth and post-smooth steps. Its properties clarify the eigenvector comment in Step 1. After defining the mappings $r_h \rightarrow \mathbf{r}_{\frac{h}{2}}$ and $\mathbf{z}_h \rightarrow \mathbf{z}_{\frac{h}{2}}$ associated with Steps 2 and 4, we explain why the Step 5 update results in an improved solution.

Recursion enters the picture through Step 3 as we can apply the same solution strategy to the similar, smaller system $A_{\frac{h}{2},\frac{h}{2}} \mathbf{z}_{\frac{h}{2}} = r_{\frac{h}{2}}$. It is through this recursion that we arrive at the overall multigrid framework: the $4h$ -grid problem helps solve the $2h$ -grid problem, the $8h$ -grid problem helps solve the $4h$ -grid problem, etc. Depending upon its implementation, the process can be used to either precondition or completely solve the top-level $A_{h,h,j}$ problem.

The tutorial by Briggs, Henson, and McCormick (2000) provides an excellent introduction to the multigrid framework that was originally proposed in Brandt (1977). For shorter introductions, see Strang (2007, pp. 571–585), Greenbaum (IMSL, pp. 183–197), Saad (IMSLA, pp. 407–450), and Demmel (ANLA, pp. 331–347).

$11h \text{ dlr } \Gamma \text{ Dwi.r tFwf.i.r or,r sFfir Do sFfir Ah } \%7 \text{ Qh}$

Consider the problem of finding a function $u(\mathbf{x})$ of $[\mathbf{Q}]$ that satisfies

$$\frac{d^2u(x)}{dx^2} = F(x), \quad u(Q) = u(l) = 0 \quad (11)$$

Our goal is to approximate the solution to (11.6.1) at $x = h, 2h, \dots, nh$ using the discretization strategy set forth in §4.8.3. Here and throughout this section,

$$n = 2^k - 1, \quad m = 2^k l - 1, \quad h = 1/2^k.$$

This leads to a linear system

$$A_h h = b_h \quad (11.6.2)$$

where $b_h \in \mathbb{R}^m$ and $A_h \in \mathbb{R}^{m \times m}$ is defined by

$$A_h = \frac{1}{h^2} \begin{bmatrix} 2 & -1 & & \cdots & & 0 \\ -1 & 2 & & & & \vdots \\ & \ddots & \ddots & & & \vdots \\ & & \ddots & \ddots & & \vdots \\ & & & \ddots & \ddots & -1 \\ 0 & 0 & \cdots & -1 & 2 \end{bmatrix}. \quad (11.6.3)$$

Note that A_h is a multiple of TDD, a matrix that we defined in (4.8.7). It has a completely known Schur decomposition

$$(Q^h)^T A^h Q^h = \Lambda^h = \text{diag}(\lambda^h), \quad (11.6.4)$$

where the vector of eigenvalues $\lambda^h \in \mathbb{R}^m$ is given by

$$\lambda_j = \frac{4}{h^2} \cdot \sin^2 \left(\frac{j\pi}{2(n+1)} \right), \quad j = 1:m, \quad (11.6.5)$$

and the orthogonal eigenvector matrix $Q^h = [q_1 \ I \ \dots \ I \ q_m]$ is prescribed by

$$q_j = \frac{1}{\sqrt{n+1}} \begin{bmatrix} \sin(\theta_j) \\ \vdots \\ \sin(m\theta_j) \end{bmatrix}, \quad \theta_j = \frac{j\pi}{n+1}. \quad (11.6.6)$$

The components of this vector involve samplings of the function $\sin(j\pi x)$. As j increases, this function is increasingly oscillatory, prompting us to split the eigenmodes in half. We regard q_j as a low-frequency eigenvector if $1 \leq j \leq m$ and as a high-frequency eigenvector if $j > m$.

To facilitate the divide-and-conquer derivations that follow, we identify some critical patterns associated with Q^h and A_h . If

$$P_h = \text{diag}(s_1, \dots, s_m), \quad S = \frac{1}{h^2} \begin{pmatrix} 2 & & & \\ & 2 & & \\ & & \ddots & \\ & & & 2 \end{pmatrix}, \quad (11.6.7)$$

$$C_h = \text{diag}(c_1, \dots, c_m), \quad c_j = \cos\left(\frac{j\pi}{2(n+1)}\right), \quad (11.6.8)$$

then

$$A_h = \frac{4}{h^2} \begin{bmatrix} 0 & 0 \\ 1/2 & 0 \\ 0 & \varepsilon_m C^h \varepsilon_m \end{bmatrix} \quad (11.6.9)$$

where e_m is the $m \times m$ exchange permutation. Regarding Q^h , it houses scaled copies of its $m \times m$ analog Q^M .

$$Q_{(222m)} = [Q_{IOI}^2 - Q_m^2]/\dots \quad (1161)$$

These results follow from the definitions (116) - (1168) and trigonometric identities

hsha hib et. 3t"b Ax kxbj sf ebf)§ -§ 1ef §b 4t fksb h§ f ektb

Critical to the multigrid framework is the role of the smoothing iteration. The term "smoother" is applied to an iterative method that is particularly successful at damping out the high-frequency eigenvector components of the error. To illustrate this part of the process, we introduce the weighted Jacobi method. If $L = \text{tril}(A, -1)$, $D = \text{diag}(a_{ii})$, and $U = \text{triu}(A, 1)$, then the iterates for this method are defined by

$$u_k = G(kl)_+ c,$$

where $c = wD^{-1}b$, $G = (1-wI)^{-1}wD^{-1}(L+U)$, and w is a free parameter that we assume satisfies $0 < w \leq 1$. Note that if $w = 1$, then the method reverts to the simple Jacobi iteration (11.22). Other iterations can be used, but the weighted Jacobi method is simple and adequately communicates the role of the smoother in multigrid.

If we apply the weighted Jacobi method to (1162), then it is easy to verify that the iteration matrix is given by

$$G^{h,\omega} = I_n - \frac{\omega h^2}{2} A^h. \quad (1161)$$

By using (1164) and (116) we see that its Schur decomposition is given by

$$(Q^h)^T G^{h,\omega} Q^h = \text{diag}(\tau^{h,\omega}), \quad \mathbf{f}_W = \mathbf{l} - 2\pi m^2 \left(\frac{\mathbf{j}_7}{2n+1} \right). \quad (116.12)$$

It follows that $(h_w) < 1$ because we assume $0 < w \leq 1$ to guarantee convergence. The explicit Schur decomposition enables us to track the error in each eigenvector direction given a starting vector u_8 .

$$\mathbf{u} - \mathbf{u}^h = \sum_{j=1}^n j \mathbf{q}_j \quad ; \quad (\mathbf{u} - \mathbf{u}^h) = (\mathbf{G}_w \mathbf{p}_w \mathbf{a} - \mathbf{u}^h) = \sum_{j=1}^n a_j (j \mathbf{h}_w \cdot \mathbf{q}_j)$$

Thus the component of the error in the direction of the eigenvector \mathbf{q}_j tends to zero like $4w^j$. These rates depend on w and vary with j . We now ask, is there a smart way to choose the value of w so that the error is rapidly diminished in each eigenvector direction?

Assume that $n \gg 1$ and consider (11.6.12). For small j we see that D is close to unity regardless of the value of w . On the other hand, we can move the "large j " eigenvalues toward the origin by choosing a smaller value of w . These qualitative observations suggest that we choose w to minimize

$$\mu(\omega) = \max\{ |\tau_{m+1}^{h,\omega}|, \dots, |\tau_n^{h,\omega}| \}.$$

In other words, w should be chosen to promote rapid damping in the direction of the high-frequency eigenvectors. Because the damping rates associated with the low-frequency eigenvectors are much less affected by the choice of w they are left out of the optimization. Since

$$-1 \leq \frac{\text{Tr } \mathbf{T}_w}{n} \leq 0 \leq \frac{\text{Tr } \mathbf{T}_w}{m}, \quad 0 \leq \frac{\text{Tr } \mathbf{T}_w}{n} \leq 1$$

it is easy to see that the optimum w should make \mathbf{T}_w and \mathbf{T}_w^{-1} equal in magnitude but opposite in sign, i.e.,

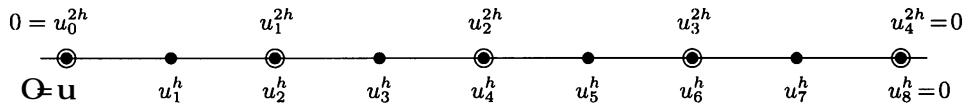
$$1 + 2w^{1/2} \left(\frac{n}{2(n+1)} \right) = - \left(1 + 2w^{-1/2} \left(\frac{(m+1)}{2(n+1)} \right) \right).$$

This is essentially solved by setting $w = 2/3$. With this choice, $\mu(2/3) = 1/3$ and so

$$\left(\begin{array}{c} \text{p-th iterate error in} \\ \text{high-frequency directions} \end{array} \right) \leq \left(\begin{array}{c} \text{Starting vector error in} \\ \text{high-frequency directions} \end{array} \right).$$

$$\text{nrNN}(0) \rightarrow (\mu(\mathbf{X}) - 1)\mathbf{I} - (\mu((\mu(\mathbf{A}))^{-1}/\mu))\mathbf{n}\mathbf{A}\mathbf{i}\mu(\mathbf{X})\mathbf{r} + \mathbf{I}\mathbf{X}(\mu)\mathbf{x}(\mathbf{A}\mathbf{r})$$

Suppose for some modest value of p we use the weighted Jacobi iteration to obtain an approximate solution \mathbf{u} to $\mathbf{A}\mathbf{u} = \mathbf{b}$. We can estimate its error by approximately solving $\mathbf{A}\mathbf{z} = \mathbf{m} - \mathbf{b} - \mathbf{A}\mathbf{u}$. From the discussion in the previous section we know that the residual $\mathbf{m} - \mathbf{A}(\mathbf{u} - \mathbf{A}\mathbf{u})$ resides mostly in the span of the low-frequency eigenvectors. Because \mathbf{m} is smooth, there is not much happening from one gridpoint to the next and it is well-approximated on the coarse grid. This suggests that we might get a good approximation to the error in \mathbf{u} by solving the coarse grid version of $\mathbf{A}\mathbf{z} = \mathbf{m}$. To that end we need to detail how vectors are transferred when we switch grids. Note that on the fine grid, gridpoint j is coarse gridpoint j :



To map values from the fine grid (with $n = 2^k - 1$ gridpoints) to the coarse grid (with $m = 2^{k-1} - 1$ gridpoints), we use an $m \times n$ restriction matrix R_h^{2h} . Similarly, to generate fine grid values from coarse grid values, we use an $n \times m$ prolongation matrix P_{2h}^h . Before these matrices are formally defined, we display the case when $n = 7$ and $m = 3$.

$$R_h^{2h} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 \end{bmatrix}, \quad P_{2h}^h = \frac{1}{2} \begin{bmatrix} 1 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \\ 0 & 0 & 1 \end{bmatrix}. \quad (116.13)$$

The intuition behind these choices is easy to see. The operation $\mathbf{R}_h^T \mathbf{R}_h$ takes a fine-grid vector of values and produces a coarse-grid vector of values using a weighted average around each even-indexed component:

$$\begin{bmatrix} u'' \\ u_h \\ u_{\bar{h}} \end{bmatrix} = \mathbf{R}_h \begin{bmatrix} u_t \\ u \\ u \\ u_1 \\ u_h \\ u_{\bar{h}} \\ u \\ u_g \end{bmatrix} = \begin{bmatrix} M + \dots + 1/4 \\ (u + 2u_1 + u_g)/4 \\ (u_g + 2u_1 + u_g)/4 \end{bmatrix}.$$

The prolongation matrix generates "missing" fine-grid values by averaging adjacent coarse grid values:

$$\begin{bmatrix} u_h \\ u_2 \\ u_{\bar{h}} \\ u_1 \\ u_g \\ u \\ u_h \end{bmatrix} = \mathbf{P}_h \begin{bmatrix} u'' \\ u_h \\ u_{\bar{h}} \end{bmatrix} = \begin{bmatrix} (u_h + u_{\bar{h}})/2 \\ u_{\bar{h}} \\ (u_h + u_{\bar{h}})/2 \\ u_h \\ (u_h + u_{\bar{h}})/2 \\ u_{\bar{h}} \\ (u_{\bar{h}} + u_h)/2 \end{bmatrix}.$$

The special end-conditions make sense because we are assuming that the solution to the model problem is zero at the endpoints.

For general $n = 2^k - 1$ and $m = 2^{k-1} - 1$, we define the matrices \mathbf{R}_{hEJmn} and \mathbf{P}_{hEJmn} by

$$\mathbf{R}_h^T = 4\mathbf{B}(22m), \quad \mathbf{P}_h = 2\mathbf{B}(22m) \quad (11.6.14)$$

where

$$\mathbf{B} = 4_{n-1}^{-1} \mathbf{A} \quad (11.6.15)$$

The connection between the even-indexed columns of this matrix and \mathbf{P}_h and \mathbf{R}_h is clear from the example

$$\mathbf{B} = \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 \end{bmatrix}, \quad (n = 7).$$

With the restriction and prolongation operators defined and letting $WJ(k, u)$ denote the k th iterate of the weighted Jacobi iteration applied to $A_{hh} - b_h$ with starting vector u_0 we can make precise the 2-grid multigrid framework

$$\begin{aligned}
 \text{Presmooth: } & u_1 = WJ(p_1, u_0), \\
 \text{Finegrid residual: } & r_h = b_h - A_{hh}u_1, \\
 \text{Restriction: } & r_{2h} = R_h^T r_h, \\
 \text{Coarsegrid correction: } & A_{2h}^{-1}r_{2h} = u_{2h}, \\
 \text{Prolongation: } & z_h = p_h^{-1}r_{2h}, \\
 \text{Update: } & u_h = u_1 + z_h, \\
 \text{Post-smooth: } & U_+ = WJ(p_2, u_h).
 \end{aligned} \tag{11.6.16}$$

By assembling the middle five equations, we see that

$$u_h = u_0 + P_h(A_{2h}^{-1})^{-1}R_h A_h(u_h - u_0)$$

and so

$$(u_h - u^h) = E_h(u_{p_1}^h - u^h) \tag{11.6.17}$$

where

$$E_h = I_n - P_{2h}^h(A^{2h})^{-1}R_h^{2h}A^h \tag{11.6.18}$$

can be thought of as a 2-grid error operator. Accounting for the damping in the weighted Jacobi smoothing steps, we have

$$(u_p^h - u^h) = (G^h)^p(u_c^h - u^h), \quad p \in \{p_1, p_2\},$$

where $G_h = G_h^{2h}$, the optimal-witeration matrix. From this we conclude that

$$(u_{++}^h - u^h) = (G^h)^{p_2}E^h(G^h)^{p_1}(u_c^h - u^h). \tag{11.6.19}$$

To appreciate how the components of the error diminish, we need to understand what E_h does to the eigenvectors q_1, \dots, q_n . The following lemma is critical to the analysis.

Lemma 11.6.1. If $n = 2^{k-1}$ and $m = 2^{k-1}-1$ then

$$(Q^h)^T P_{2h}^h Q^{2h} = \sqrt{2} \begin{bmatrix} C^h \\ 0 \\ -\mathcal{E}_m S^h \end{bmatrix}, \quad (Q^{2h})^T R_h^{2h} Q^h = \sqrt{\frac{1}{2}} \begin{bmatrix} C^h \\ 0 \\ -\mathcal{E}_m S^h \end{bmatrix}^T \tag{11.6.20}$$

where the diagonal matrices S and C are defined by (11.6.7) and (11.6.8).

From (11.6.4), (11.6.9), and (11.6.15) we have

$$(Q^h)^T B^h Q^h = 4I_n - h^2 \Lambda^h = 4 \begin{bmatrix} C^h & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & \mathcal{E}_m S^h \mathcal{E}_m \end{bmatrix} = D^h.$$

Define the index vector $\mathbf{idx} = 22\mathbf{a}_m$. Since $(Q^h)^T B^h = D^h Q^h \mathbf{r}$, it follows from (1161) that

$$(Q^h)^T B^h(:, \mathbf{idx}) = I^T Q^h \mathbf{idx} \mathbf{f} = D^h \begin{bmatrix} I_m \\ 0 \\ -\mathcal{E}_m \end{bmatrix} (Q^h \mathbf{f}).$$

Thus,

$$(Q^h)^T B^h(:, \mathbf{idx}) Q^{2h} = \frac{4}{\sqrt{2}} \begin{bmatrix} C^h & 0 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & \mathcal{E}_m S^h \mathcal{E}_m \end{bmatrix} \begin{bmatrix} I_m \\ 0 \\ -\mathcal{E}_m \end{bmatrix} = \frac{4}{\sqrt{2}} \begin{bmatrix} C^h \\ 0 \\ -\mathcal{E}_m S^h \end{bmatrix}.$$

The lemma follows since $P = B^h(:, \mathbf{idx})/2$ and $R = B^h(:, \mathbf{idx})^T/4$. \square

With these diagonal-like decompositions we can expose the structure of Eh .

Theorem 11.6.2. If $n = 2k + 1$ and $m = 2k + 1$, then

$$Eh = Q^h \begin{bmatrix} Sh & 0 & Ch \\ 0 & 1 & 0 \\ & 0 & Sh \end{bmatrix} Q^h. \quad (1162)$$

Proof. From (1161), it follows that

$$(Q^h)^T E Q^h = I_n = (Q^h)^T P_{IR}^{-1} (Q^h)^T A_h^{-1} Q^h = (Q^h)^T R^{-1} (Q^h)^T A_h^{-1} Q^h = (Q^h)^T A_h^{-1} Q^h.$$

The proof follows by substituting (1164), (1169), (1162), and

$$(Q^h)^T A_h^{-1} Q^h = \frac{1}{2h^2} (I_m, \dots)$$

into this equation and using trigonometric identities. \square

The block matrix (1162) has the form

$$\begin{bmatrix} S^h & 0 & C^h \mathcal{E}_m \\ 0 & 1 & 0 \\ \mathcal{E}_m S^h & 0 & \mathcal{E}_m C^h \mathcal{E}_m \end{bmatrix} = \begin{bmatrix} s^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & s & 0 & 0 & 0 & c \\ 0 & 0 & s^2 & 0 & c_3^2 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & s & 0 & c_2^2 & 0 \\ 0 & s & 0 & 0 & 0 & c \\ s^2 & 0 & 0 & 0 & 0 & c_2 \end{bmatrix}, \quad (n = 7),$$

from which it is easy to see that

$$\begin{aligned} Eh_{jj} &= s(jq + q_j + 1), & j = lm \\ Eh_{jl} &= 0, \\ Eh_{(j+1)l} &= c(jq + q_j + 1), & j = lm \end{aligned} \quad (1162)$$

This enables us to examine the eigenvector components in the error equation (11.6.19) because we also know from §11.6.2 that $Gh\eta_i = T^{\frac{2}{3}} \eta_i$ where $T = T_h^{\frac{2}{3}}$. Thus, if the initial error $h\eta$ has the eigenvector expansion

$$u_c^h - u^h = \underbrace{\sum_{j=1}^m \alpha_j q_j}_{\text{ltl gr'cefswl}} + \underbrace{\alpha_{m+1} q_{m+1}}_{\beta} + \underbrace{\sum_{j=1}^m \alpha_{n-j+1} q_{n-j+1}}_{\in \beta, \alpha}$$

and we execute (11.6.16), then the error in u_{++}^h is given by

$$u_{++}^h - u^h = \sum_{j=1}^m \tilde{\alpha}_j q_j + \tilde{\alpha}_{m+1} q_{m+1} + \sum_{j=1}^m \tilde{\alpha}_{n-j+1} q_{n-j+1},$$

where

$$\tilde{\alpha}_j = (\alpha_j + \dots + \alpha_{n-j+1}) C_j^2, \quad j = 1:m,$$

$$\tilde{\alpha}_{m+1} = \alpha_{m+1} \tau_{m+1}^{p_1+p_2},$$

$$\tilde{\alpha}_{n-j+1} = (\alpha_j \tau_j^{p_1} s_j^2 + \alpha_{n-j+1} \tau_{n-j+1}^{p_1} c_j^2) \tau_{n-j+1}^{p_2}, \quad j = 1:m$$

It is important to appreciate the damping factors in these expressions. By virtue of the weighted Jacobi iteration design $|\tau_{n-j+1}| \leq 1/3$ for $j = 1:m$ from the definition of s_j in (11.6.7), we also have $|s_j| \leq 1/2$. It follows from the above recipes that high-frequency error is nicely damped by fine-grid smoothing and that low-frequency error is attenuated by the coarse-grid operations. This interplay together with the fact that these bounds are independent of n are what make the multigrid framework so powerful.

hs hnmstb -i Kf rSb tttb 6Qesb 98D .eTf \$u Qt QST Sb

If the coarse-grid system in (11.6.16) is solved recursively, then we can encapsulate the overall process as follows given that $A_{h+1} := A_h$

```

function  $u_{++}^h = \text{ngV}(u, b, h)$ 
if  $h \geq h_{\text{max}}$ 
     $u_{++}^h = \text{WJ}(u, p_0)$  (for example)
else
     $u_1 = \text{WJ}(u, p_1)$ 
     $r_h = b - A_h u_1$ 
     $r_{h+1}^2 = R_h^{2h} r_h^2$ 
     $z_{h+1}^2 = \text{ngV}(r_{h+1}^2, A_h)$ 
     $u_1 = u_1 + P_{h+1} z_{h+1}^2$ 
     $u_{++}^h = \text{WJ}(u_1, P_0)$ 
end

```

Note that the base case ($h; : h_{\max}$) is defined by a "coarse enough" gridpoint-spacing parameter h_{\max} and that the solution of the (possibly small) linear system at that level can be obtained in various ways. Figure 11.61 depicts the flow of events called a V-cycle if $h_{\max} = 16h$. Five grids are used and the process starts by recursing for

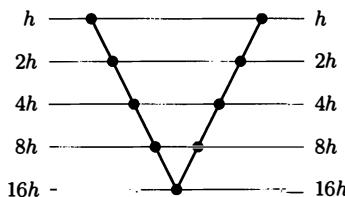


Figure 11.61 A V-cycle

times before the correction equation is solved. This is done on the $16h$ -grid. After that, the corrections are mapped upwards through four levels, eventually generating a solution to the top-level h -grid problem.

Examination of mgV reveals that a V-cycle involves $O(n)$ flops, a hint that the multigrid framework is incredibly efficient. The coefficient of n in the complexity assessment depends on the iteration parameters p_0 , P_1 and P_2 . However, the rate of error damping is independent of n , which means that these error-control parameters are not affected by the size of the problem.

The V-cycle that we illustrated is but one of several strategies for moving in between grids during the course of a multigrid solve. The pattern for full multigrid depicted in Figure 11.62. Here the coarse grid system is used to obtain a starting value

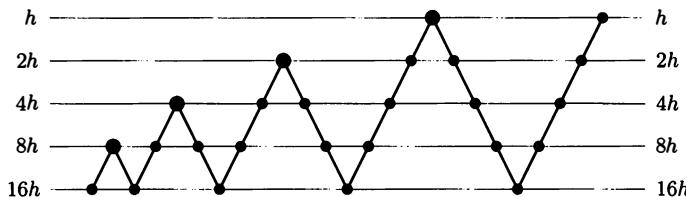


Figure 11.62 Full multigrid

for its fine-grid neighbor and then a V-cycle is performed to obtain an improvement. The process is repeated.

thinking about what happens

The multigrid framework is rich with options, some of which are not obvious for a simple, model-problem treatment. For general elliptic boundary value problems on complicated domains, there are several critical decisions that need to be made if the overall procedure is to be effective:

- Determine how to extract the coarse grid from the fine grid, e.g., every other grid point in each coordinate direction or every other gridpoint in just one coordinate direction

- Determine the right restriction and prolongation operators
- Determine the right smoother, e.g., (blocked) weighted Jacobi or Gauss-Seidel.
- Determine the number of pre-smoothing steps and post-smoothing steps
- Determine the depth and "shape" of the recursion, i.e., the number of participating grids and the order in which they are visited
- Determine a basic strategy, i.e., should bottom-level linear systems be solved exactly or approximately?

With so many implementation parameters, it is not surprising that the multigrid framework can be tuned to address a very broad range of problems.

Problems

P11.6.1 3.8.8 (or)=ED()(r))

P11.6.2 fe1., 2.8.82..o.2.2 ..8 o832828 d.8811.88.8. ..a8101

P11.6.3 ro.1. ()Nr)NkE- TEh)()•kNθ=)ETEh

P11.6.4 f.2 ..8 .8 ...8.8. 8>PJ =ED(hrE)NQD+CE=f • E-T=={E r)O E=EOP k=E(10)f)N)TEF PNh10 • (ED+P EDF KT•SEN()D O)NO =)ETEh1• (f T)NfP(O)=kO=EDE• nO =E- • ((O- O++=)o)r)eED#NO EO)Nr)b

Notes and References for §11.6

1.8 n.2..... . 868.8 8.1..... .82N.. 1.or

11P...2 (o(D))A(- kHtD)D= (kTnO• - (T• E=q• (ED-E)=)(O (E• r)O) Math. Comput. 31, 555r 5Jx

.LMia AbaAUUAtAmlbLInL1bAMlebbptAAe

.4 sLMia((D))Computational Science and Engineering aA,,At,At,AiCSM 1bAMAtAUUAtA34

IlMAla,eA)ppMAiL+AlabtK eAe

h4AttA, 1afo(a) (An Introduction to Multigrid Methods, a1gAtAm1amAtLAM.
a4 iaOSI tanto(a)) Multi-Grid Methods and Applications, sSMiAMLBAMM,na4
s44l aAlMC 1aB4D) Multigr d Methods, sf 3l hIS g1Lnlatbm,ieSm1h34
9.4wMiS 10(A(()) Multigr d Methods, nlar+ia sa1Aa1a1a1a1a1iMg.. r 454
a.n. wM lrb49Aatla, iae s4.4l aAlMCn44)' A Multigr d T torial, tAalaeAdp11asf 3l
hISUnap1latigA,m1b3:

r 4lppAaSAMr,tLAMUAA34samIg.(M)) Multigrid, 3iaeA+1hMAttlaela.

I4sm) IMi((()) Matri -Based Multigr d tAalaeAeLL, ll8 Mlar 2M.II MB4

IIUlrMaaSAItAe i SMAalaeILnL1bAMlebbptAAe, Aa1MtArMNS, ACAMiAtLmAt,L
tlUiAttLAQpmiapiSLIMAtAttAaaA6pmAa.RM1e ttLAC. tAA8

94PI ()((o)) x)OE=f AON• D=)= P QOP • (• =T)T• ED(r = (= P OEP R Extr AM Review
34 .dN7N 54

a46Amia iae w4. sC 1nr((o))))• + = TEP • - (• =T)T• EDA.)f (Ef D)(• ETkQ • E x)T(TP
(E• r)O• E(E=kE(P))(EIAO=N)HETNA 2 N,F Ndl.

w4hAA(())))(TD-EP OE)N• • = T(-) (E)EP • EDF KT• EEE L)O= •) x)T(TR=ET=)
)-T- (ECE)T-1(f • E=)IAM J. Sci. Comput. 31, 1dJ5r 1,05

amACIgLnrmAiaa SAAHAAeAd.rMlegAMIS,Ait4 AMAL1bAMLB1. algebraic
multigrid+ALmlen CAL1pmalat1eAMiS(AaAtlaaAMLi)g1aiL 1laALL latAA8

For more details on large sparse linear systems, see the book by Saad [1996]. In addition, the book *Iterative Methods for Sparse Linear Systems* by Saad [1992] and the book *Iterative Methods for Sparse Linear Systems, Second Edition* by Saad [2003] are excellent sources for iterative methods. The book *Finite Element Analysis Using High-Order Finite Elements* by S. P. Oden and J. C.一人 [1996] provides a detailed treatment of finite element analysis using high-order finite elements. The book *Computational Fluid Dynamics: The Basic Equations and Numerical Methods* by R. H. Hoogstraten [1988] provides a detailed treatment of computational fluid dynamics. The book *Computational Fluid Dynamics: A Practical Approach* by R. H. Hoogstraten [1995] provides a detailed treatment of computational fluid dynamics. The book *Computational Fluid Dynamics: A Practical Approach* by R. H. Hoogstraten [1995] provides a detailed treatment of computational fluid dynamics.

Chapter 12

Special Topics

elPe

elPI

M

elPn

elPo

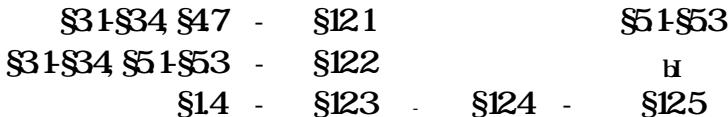
elPr

Prominent themes in this final chapter include data sparsity, lowrank approximation, exploitation of structure, the importance of representation, and largescale problems. We revisit (unsymmetric) Toeplitz systems in §12.1 and show how fast stable methods can be developed through a clever data-sparse representation. The idea extends to other types of structured matrices. Representation is also central to the $O(n)$ methods developed in §12.2 for matrices that have lowrank off-diagonal blocks.

The next three sections form a sequence. The Kronecker product section has general utility, but it is used very heavily in both §12.4 and §12.5 which together provide a brief introduction to the rapidly developing field of tensor computations.

Reading Path

Within this chapter, there are the following dependencies:



The schematic also hints at the minimum "prerequisites" for each topic.

12.1 Linear Systems with Displacement Structure

If $A \in \mathbb{R}^{m \times n}$ has rank r , then it has a (non-unique) product representation of the form UV^T where $U, V \in \mathbb{R}^{m \times r}$. Note that if $r < n$, then the product representation is much

more compact than the explicit representation that encodes each a_{ij} . In addition to the obvious storage economies, the product representation supports fast computation. If the product representation is fully utilized, then the n -by- n matrix-matrix product $AB = UV^T$ is $O(n^2r)$ instead of $O(n^3)$. Likewise, by applying the Sherman-Morrison-Woodbury formula, the solution to a linear system of the form $(I + UV^T)x = b$ is $O(nr + r^3)$ instead of $O(n^3)$. The message is simple in both cases: work with U and V and not their explicit product UV^T .

In this section we continue in this direction by discussing "lowrank" ways to represent Cauchy, Toeplitz, and Hankel matrices together with some of their generalizations. The data-sparse representation supports fast stable linear equation solving. The key idea is to turn explicit rank-1 updates that are at the heart of Gaussian elimination into equivalent, inexpensive updates of their representation. Our presentation is based on Golberg, Kailath, and Olshevsky (1995) and Gu (1998).

si mshsb p1e 3Ptf \$TSf b 9ttdb

If $F, G \in \mathbb{J}^{n \times n}$ and the Sylvester map

$$X = FX - XG \quad (12.11)$$

is nonsingular, then the $\{F, G\}$ -displacement rank of $A \in \mathbb{J}^{n \times n}$ is defined by

$$\text{rank}_F(A) = \text{rank}(FA - AG). \quad (12.12)$$

Recall from §7.63 that the Sylvester map is nonsingular provided $\text{gt}(F) \cup \text{gt}(G) = \{n\}$. Note that if $\text{rank}_F(A) = r$, then we can write

$$FA - AG = RS^T, \quad R, S \in \mathbb{R}^{n \times r}. \quad (12.13)$$

The matrices R and S are generators of A with respect to F and G , a term that makes sense since we can generate A (or part of A) by working with this equation. If $r < n$, then R and S define a data-sparse representation of A . Of course, for this representation to be of interest F and G must be sufficiently simple so that the reconstruction of A via (12.13) is cheap.

himhhib .tDfeK rI T\$htQx Tf\$eb

If $w_k \in \mathbb{R}^n$ and $\text{gt}(E_{Aa}) \text{ and } w_k = \mathbf{j}_k$ for all k and j , then the n -by- n matrix $A = (a_{ij})$ defined by

$$a_{kj} = \frac{1}{\omega_k - \lambda_j}$$

is a Cauchy matrix. Note that if

$$\Omega = \text{diag}(\omega_1, \dots, \omega_n), \quad A = \text{diag}(\lambda_1, \dots, \lambda_n),$$

then

$$[A - AA]_k = \frac{\omega_k}{\omega_k - \lambda_j} - \frac{\lambda_j}{\omega_k - \lambda_j} = 1$$

If $e \in \mathbb{R}^n$ is the vector of all 1's, then

$$A - AA = ee^T$$

and thus $\text{rank}(A) = 1$

More generally, if $R \in \mathbb{R}^{nr}$ and $S \in \mathbb{R}^{nr}$ have rank r , then any matrix A that satisfies

$$A - AA = RS^T \quad \{1214\}$$

is a Cauchy-like matrix. This just means that

$$a_{k3} = \frac{\frac{k}{3}}{k-3}$$

where

$$R^T = [r_1 I \dots I_m], \quad S^T = [1 I \dots I_n]$$

are column partitionings. Note that R and S are generators with respect to I_1 and A and that $O(r)$ fops are required to reconstruct a matrix entry a_{k3} from (1214).

1mclA T oA nW }7dA E+A jAL 7Ni -N7A

Suppose

$$A = \begin{bmatrix} g^T \\ f & B \end{bmatrix} \quad \text{a} \in \mathbb{R}^{r, f+g}, \quad R \in \mathbb{R}^{n \times r}, \quad S \in \mathbb{R}^{n \times r},$$

and assume $A \neq 0$. The first step in Gaussian elimination produces

$$A_i = B - \frac{1}{f} f g^T$$

and the factorization

$$A = \begin{bmatrix} & & & \circ \\ & & & \\ & & & \\ f & A & I_n & 1 \end{bmatrix} \begin{bmatrix} & & & g^T \\ & & & \\ & & & \\ & & & A_1 \end{bmatrix}.$$

Let us examine the structure of A_i given that A is a Cauchy matrix. If $n = 4$ and $a_{k3} = 1/(k-3)$, then

$$A_i = \begin{bmatrix} \frac{1}{\omega_2 - \lambda_2} & \frac{1}{\omega_2 - \lambda_3} & \frac{1}{\omega_2 - \lambda_4} \\ \frac{1}{\omega_3 - \lambda_2} & \frac{1}{\omega_3 - \lambda_3} & \frac{1}{\omega_3 - \lambda_4} \\ \frac{1}{\omega_4 - \lambda_2} & \frac{1}{\omega_4 - \lambda_3} & \frac{1}{\omega_4 - \lambda_4} \end{bmatrix} - \begin{bmatrix} \frac{\omega_1 - \lambda_1}{\omega_2 - \lambda_1} \\ \frac{\omega_1 - \lambda_1}{\omega_3 - \lambda_1} \\ \frac{\omega_1 - \lambda_1}{\omega_4 - \lambda_1} \end{bmatrix} \begin{bmatrix} \frac{1}{\omega_1 - \lambda_2} \\ \frac{1}{\omega_1 - \lambda_3} \\ \frac{1}{\omega_1 - \lambda_4} \end{bmatrix}^T.$$

If we choose to work with the explicit representation of A , then for general n this update requires $O(n^2)$ work even though it is highly structured and involves $O(n)$ data. And worse, all subsequent steps in the factorization process essentially deal with general matrices rendering an LU computation that is $O(n^3)$.

1Ah)A 9G+V2i obIA .}TA }IA }TolA Af I} -oA

The situation is much happier if we replace the explicit transition from A_0 to A_1 with a transition that involves updating data sparse representations. The key to developing a fast LU factorization for a Cauchy-like matrix is to recognize that rank-1 updates preserve displacement rank. Here is the result that makes it all possible.

Theorem 12.1.1. Suppose $A \in \mathbb{R}^{n \times n}$ satisfies

$$nA - AA = RS^T \quad (12.15)$$

where $R, S \in \mathbb{R}^{n \times r}$ and

$$\Omega = \text{diag}(\omega_1, \dots, \omega_n), \quad \Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$$

have no common diagonal entries. If

$$A = \begin{bmatrix} a & g^T \\ f & B \end{bmatrix}, \quad R = \begin{bmatrix} r_1^T \\ R_1 \end{bmatrix}, \quad S = \begin{bmatrix} s_1^T \\ S_1 \end{bmatrix}$$

are conformably partitioned $a = Q$ and

$$\Omega_1 = \text{diag}(\omega_2, \dots, \omega_n), \quad \Lambda_1 = \text{diag}(\lambda_2, \dots, \lambda_n),$$

then

$$\Omega_1 A_1 - A_1 \Lambda_1 = \tilde{R}_1 \tilde{S}_1^T \quad (12.16)$$

where

$$A_1 = B - \frac{1}{\alpha} g g^T, \quad \tilde{R}_1 = R_1 - \frac{1}{\alpha} f r_1^T, \quad \tilde{S}_1 = S_1 - \frac{1}{\alpha} g s_1^T.$$

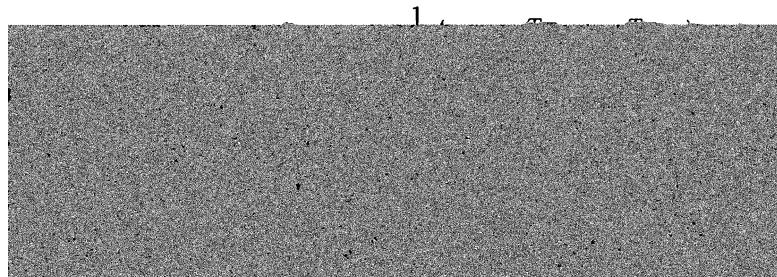
Proof. By comparing blocks in (12.15) we see that

$$(1.1) : (W - >)a = rfs_1 \quad (1.2) : g^T A_1 = Wg^T - rfS_1^T,$$

$$(2.1) : Qf = R_1 s_1 + > if, \quad (2.2) : ! B - BA_1 = R_1 S_1,$$

and so

$$\Omega_1 A_1 - A_1 \Lambda_1 = \Omega_1 \left(B - \frac{1}{\alpha} f g^T \right) - \left(B - \frac{1}{\alpha} f g^T \right) \Lambda_1$$



This confirms (12.16) and completes the proof of the theorem D

The theorem says that

$$\text{rank } (\mathbf{A})_r = \text{rank } (\mathbf{A}_1)_r.$$

This suggests that instead of updating \mathbf{A} explicitly to get \mathbf{A}_1 at a cost of $O(n^3)$ fops, we should update \mathbf{A} 's representation $\{\Omega, \mathbf{A}, \mathbf{R}, \mathbf{S}\}$ at a cost of $O(nr)$ fops to get \mathbf{A}_1 's representation $\{\tilde{\Omega}, \mathbf{A}_1, \tilde{\mathbf{R}}_1, \mathbf{S}_1\}$.

1hit 4A 2+A FAXX[A,]Ni- .(EGTAq) - 7GiaA

Based on Theorem 12.1.1 we can specify a fast LU procedure for Cauchy-like matrices. If \mathbf{A} satisfies (12.15) and has an LU factorization, then it can be computed using the function `LUdisp` defined as follows:

Algorithm 12.1.1 If $w \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n \times n}$ have no common components, $R, S \in \mathbb{R}^{n \times n}$, and $Q = AA^T = RST$ where $Q = \text{diag}(w_1, \dots, w_n)$ and $A = \text{diag}(A_1, \dots, A_n)$, then the following function computes the LU factorization $A = LU$.

```

function [L, U] = LUdisp(w, A, R, S, n)
    r = R(:, :), R1 = R(2:n, :)
    s = S(:, :), S1 = S(2:n, :)
    if n == 1
        L = 1
        U = r - (w - A)
    else
        a = (R * s) / (w - A)
        a = a
        f = a(2n)
        g = (S * r) ./ (w - A(2n))
        Ri = Ri - fr^T / a
        i = Si - gs / a
        [Li,Ui] = LUdisp(w(2n), A(2n), R1, i, n-1)
        L = [
            f a ; i
        ]
        U = [
            alpha g^T
            0 U1
        ]
    end

```

The nonrecursive version would have the following structure

Let $R^{(1)}$ and $S^{(1)}$ be the generators of $\mathbf{A} = \mathbf{A}(1)$ with respect to $\text{diag}(\mathbf{w})$ and $\text{diag}(\mathbf{A})$.

for $k = 1:n - 1$

Use $w(kn)$, $A(kn)$, $R^{(k)}$ and $S^{(k)}$ to compute the first row and column of

$$\mathbf{A}(k) = \begin{bmatrix} \alpha \\ f \end{bmatrix}.$$

$$L(k+1:n, k) = f/\alpha, \quad U(k, k) = \alpha, \quad U(k, k+1:n) = g^T$$

Determine the generators $R^{(k+1)}$ and $S^{(k+1)}$ of $\mathbf{A}(k+1) = B - fg^T/\alpha$ with respect to $\text{diag}(\mathbf{w}(kn))$ and $\text{diag}(\mathbf{A}(kn))$.

end

$$U(n, n) = R^{(n)} \cdot S^{(n)} / (\omega_n - \lambda_n)$$

A careful accounting reveals that $2nr^2$ fops are required

himthnb .Tf kQt'b

The procedure just developed has numerical difficulties if a small α shows up during the recursion. To guard against this we show how to incorporate a pivoting strategy. Suppose $\mathbf{A} \in \mathbb{R}^{n \times n}$ is a Cauchy-like matrix that satisfies the displacement equation

$$\mathbf{Q}\mathbf{A} - \mathbf{A}\mathbf{A} = \mathbf{R}\mathbf{S}\mathbf{T}$$

for diagonal matrices Ω and Λ and n -by- r matrices R and S . If P and Q are n -by- n permutations, then

$$(P\Omega P^T)(PAQ^T) - (PAQ^T)(Q\Lambda Q^T) = (PR)(QS)^T.$$

This shows that

$$\mathbf{A} = \mathbf{P}\mathbf{A}\mathbf{Q}^T$$

is a Cauchy-like matrix having generators

$$\mathbf{R} = \mathbf{P}\mathbf{R}, \quad \mathbf{S} = \mathbf{Q}\mathbf{S}$$

with respect to the diagonal matrices

$$\mathbf{f} = \mathbf{P}\mathbf{f}\mathbf{P}^T, \quad \tilde{\Lambda} = Q\Lambda Q^T.$$

Thus, it is easy to track row and column permutations in the displacement representation

$$A \rightarrow PAQ^T, \quad \equiv \quad \{\mathbf{Q}, \mathbf{A}, \mathbf{R}, \mathbf{S}\} \rightarrow \{\mathbf{P}, \mathbf{Q}^T, \mathbf{Q}\mathbf{A}\mathbf{Q}^T, \mathbf{P}\mathbf{R}, \mathbf{Q}\mathbf{S}\}.$$

By taking advantage of this, it is a simple matter to incorporate partial pivoting in LU and to endow the factorization $\mathbf{PA} = \mathbf{LU}$:

As before, if $\mathbf{w} \in \mathbb{R}^n$ and $\mathbf{A} \in \mathbb{R}^n$ have no common components, $\mathbf{R}, \mathbf{S} \in \mathbb{R}^{n \times r}$, and $\mathbf{Q}\mathbf{A} - \mathbf{A}\mathbf{A} = \mathbf{R}\mathbf{S}\mathbf{T}$, then the following function computes the LU-with-pivoting factorization $\mathbf{PA} = \mathbf{LU}$ where $\Omega = \text{diag}(\omega_1, \dots, \omega_n)$ and $\mathbf{A} = \text{diag}(\mathbf{a}_1, \dots, \mathbf{a}_n)$.

function $[L, U, P] = \text{ILUspPiv}(w, R, S, n)$

De ne $R_1 R_1 \bullet 1$ and S_1 b $R = \begin{bmatrix} \bullet & \end{bmatrix}$ and $S = \begin{bmatrix} \bullet \\ ; \end{bmatrix}$.

if $n = 1$

$L = 1$

$U = \text{rfsif}(W, 1)$

else

$a = (Rs) ./ (w, 1)$

Determine permutation $P \in \text{Inn}$ so that $|P_{11}|$ is maximal and
update $a = Pa$, $R = PR$, $w = Pw$

$a = a1$

$f = a(2n)$

$g = (Sr) ./ (W, (2n))$

$\tilde{R}_1 = R_1 - fr_1^T / \alpha$

$S_1 = S_1 - g1/a$

$[L, U, P] = \text{ILUspPiv}(w(2n), (2n), R1, S1, n-1)$

$L = \begin{bmatrix} P1/a & i1 \end{bmatrix}$

$U = \begin{bmatrix} \alpha & g^T \\ 0 & U_1 \end{bmatrix}$

$P = \begin{bmatrix} 1 & 0 \\ 0 & P_1 \end{bmatrix} P$

end

The processing of the recursive call is based on the fact that if

$$PA = \begin{bmatrix} A & g^T \\ f & B \end{bmatrix} = \begin{bmatrix} 1 & O \\ f/a & I_n \end{bmatrix} \begin{bmatrix} A & g \\ O & A1 \end{bmatrix}, \quad A_1 = B - \frac{1}{\alpha} fg^T,$$

and $P_1 A_1 = L_1 U_1$, then

$$\begin{bmatrix} 1 & 0 \\ 0 & P_1 \end{bmatrix} PA = \begin{bmatrix} 1 & 0 \\ P_1 f/\alpha & L_1 \end{bmatrix} \begin{bmatrix} \alpha & g^T \\ 0 & U_1 \end{bmatrix}.$$

For ILUspPiv implementation details and a proof of its stability, see Gu (1998).

1A1hA T d2G QEGTAq = Gi oA} IA + } T & EGTAq = Ga

Recall from §4.7 that a Toeplitz matrix is constant along each of its diagonals. For example, if $c \in \mathbb{R}^{n-1}$, $\tau \in \mathbb{R}$, and $r \in \mathbb{R}^{n-1}$ are given, then the matrix $T \in \mathbb{R}^{n \times n}$ defined by

$$t_{ij} = \begin{cases} c_{j-i} & \text{if } i > j, \\ \tau & \text{if } i = j, \\ r_{j-i} & \text{if } j > i, \end{cases}$$

is Toeplitz, e.g.,

$$T = \begin{bmatrix} \tau & r_1 & r_2 & r_3 & r_4 \\ c_1 & \tau & r_1 & r_2 & r_3 \\ c_2 & c_1 & \ddots & r_1 & r_2 \\ c_3 & c_2 & c_1 & \ddots & T_1 \\ c_4 & c_3 & c_2 & c_1 & \tau \end{bmatrix}.$$

To expose the low-displacement-rank structure of a Toeplitz matrix, we define matrices Z and Y , analogously to their $n = 5$ instances:

$$Z_\phi = \begin{bmatrix} 0 & 0 & 0 & 0 & \phi \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad Y_{\gamma, \delta} = \begin{bmatrix} \gamma & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & \delta \end{bmatrix}. \quad \{1217\}$$

It can be shown that

$$Z\Gamma - TZ = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & \times \end{bmatrix}, \quad \text{rank}_{\{Z_1, Z_{-1}\}}(T) \leq 2, \quad \{1218\}$$

$$Y\Gamma - TY = \begin{bmatrix} \times & \times & \times & \times \\ 0 & 0 & 0 & \times \\ 0 & 0 & 0 & \times \\ 0 & 0 & 0 & \times \\ \times & \times & \times & \times \end{bmatrix}, \quad \text{rank}_{\{Y_{00}, Y_{11}\}}(T) \leq 4. \quad \{1219\}$$

Furthermore, $A(Z_1) \cup A(Z_{-1}) = a$ and $A(Y_{00}) \cup A(Y_{11}) = w$.
A Hankel matrix is constant along its antidiagonals, e.g.,

$$H = \begin{bmatrix} c_4 & c_3 & c_2 & c_1 & \tau \\ c_3 & c_2 & c_1 & \tau & r_1 \\ c_2 & c_1 & \ddots & r_1 & r_2 \\ c_1 & \tau & r_1 & r_2 & r_3 \\ \tau & r_1 & r_2 & r_3 & r_4 \end{bmatrix}.$$

Note that if $H \in \mathbb{R}^{n \times n}$ is Hankel, then $\mathcal{E}_n H$ is Toeplitz, and so it is not surprising that

Hankel and Toeplitz matrices have similar displacement rank properties.

$$Z - Z_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & \times \\ \times & \times & \times & \times & \times \end{bmatrix}, \quad \text{rank}_{\{Z_1^T, Z_{-1}\}}(H) \leq 2, \quad (12.1.10)$$

$$Y_{00} - Y_u = \begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & \times \\ 0 & 0 & 0 & 0 & \times \\ \times & \times & \times & \times & \times \end{bmatrix}, \quad \text{rank}_{\{Y_{00}, Y_{11}\}}(H) \leq 4. \quad (12.1.11)$$

It follows from (12.1.10) and (12.1.11) that if $A = T + H$ is the sum of a Toeplitz matrix and a Hankel matrix, then $\text{rank}(A) \leq 4$.

The classes of Toeplitz, Hankel, and Toeplitz-plus-Hankel matrices can be expanded through the notion of low displacement rank. Analogous to how we defined Cauchy-like matrices in (12.1.4) we have the following a suming that $R \in \mathbb{M}^{nr}$, $S \in \mathbb{M}^{mr}$, and $r \ll n$

$$\left\{ \begin{array}{l} Z_1 A - AZ_1 = RS^T \\ Z A - AZ_1 = RS^T \\ Y_{00} A - AY_{11} = RS^T \end{array} \right\} \text{ means that } A \text{ is } \left\{ \begin{array}{l} \text{Toeplitz-like} \\ \text{Hankel-like} \\ \text{Toeplitz-plus-Hankel-like} \end{array} \right\}.$$

Our next task is to show that a linear system with any of these properties can be efficiently converted to a Cauchy-like system and solved with $O(nr)$ work.

Suppose $FA - AG = RS$, $A, F, G \in \mathbb{M}^{nr}$, $R, S \in \mathbb{M}^{mr}$, $r \ll n$,

and that F and G are diagonalizable.

$$X_F^{-1}FX_F = \text{diag}(w_1, \dots, w_r) = \Omega,$$

$$X_G^{-1}GX_G = \text{diag}(i_1, \dots, i_r) = \Lambda.$$

For clarity we assume that F and G have real eigenvalues. It follows from

$$(X_F^{-1}FX_F)(X_F^{-1}AX_G) - (X_F^{-1}AX_G)(X_G^{-1}GX_F) = (X_F^{-1}R)(X_G^TS)^T$$

that

$$\Omega \tilde{A} - \tilde{A} \Lambda = \tilde{R} \tilde{S}^T$$

where $\tilde{A} = X_F^{-1}AX_G$, $\tilde{R} = X_F^{-1}R$, and $\tilde{S} = X_G^TS$. Thus, \tilde{A} is Cauchy-like and we can go about solving the given linear system $\tilde{A}\tilde{x} = \tilde{b}$ as follows:

- tp1 Compute $\tilde{R} = X_F^{-1}R$, $S = X^T S$, $b = X^{-1}b$ and $A = X_F^{-1}AX$..
 - tp2 Use Algorithm 12.12 to compute $P\tilde{A} = LU$.
 - tp3 Use $P\tilde{A} = LU$ to solve $\tilde{A}\tilde{x} = \tilde{b}$.
 - tp . Compute $x = X \cdot$.

This will not be an attractive framework unless the matrices F and G have fast eigen systems, a concept introduced in §48. Fortunately, this is the case for the matrices Z_1 , Z_{-1} , Y_{00} and Y_{11} . For example,

$$\mathcal{S}_n^T Y_{00} \mathcal{S}_n = 2 \operatorname{diag} \left(\cos \left(\frac{\pi}{n+1} \right), \dots, \cos \left(\frac{n\pi}{n+1} \right) \right), \quad \{12112\}$$

$$C_n^T Y_{11} C_n = 2 \operatorname{diag} \left(1, \cos\left(\frac{\pi}{n}\right), \dots, \cos\left(\frac{(n-1)\pi}{n}\right) \right), \quad \{12113\}$$

where S_n is the sine transform (DST-I) matrix

$$[S_n]_{\mathbf{k}} = \sqrt{n^2} \cdot \sin\left(\frac{\mathbf{k}_1}{n+1}\right)$$

and C_n is the cosine transform (DCT-11) matrix

$$[c_n]_j - \sum_{n=1}^{\infty} \cos\left(\frac{(2k-1)(j-1)\pi}{2n}\right) q_j = \begin{cases} 1/\sqrt{2} & \text{if } j = 1, \\ 1 & \text{if } j > 1. \end{cases}$$

This allows products like $S_n R$ and $C_n^T S$ to be computed with $O(m \log n)$ fops. In short, Step 3 in the above framework is the most expensive step in the process and it involves $O(mr)$ work. See Gohberg, Kailath, and Olshevsky (1995) and Gu (1999) for details and related references.

Problems

P12.1.1 18.8 f.0aa7.. f.0aN f. n.8rob 11Z1X - XZ- 1= x. X-x0. ..R IuM.
 $\cdot uAi m \cdot X - XY[1] = XX \cdot X-x_2$

P12.1.2 88.8.8. . .8.84...1.8 .8.18. 811..8N1... .0a.10a

P12.1.3 f. If $T \in R \cap R_{\frac{1}{2}}$, then $(C \cup C) \cap D(b \in nAR, S \in R)^e = (rC_2 Z \Gamma - TZ_{-1} - RST, Z, T = 1 \in AR, S \in R^n, T \in ER^n, (Z \Gamma - TZ_{-1} - RST) \Gamma 2A2 2t \in Cb rC_2 D(b \in nA = i = Tc,)^2 x = T(1, \frac{1}{2})$.

P12.1.4 f. If $T \in RrR^{-1}$, then $A11T\alpha C(C)C \in D(b) = nAR, S \in Rr = (nC2)Y_0T - TY_0 = RST$.
 $Z, T = 11 = AR, S \in Rr \Rightarrow TE \in RrR^{-1} = Y_0T - TY_0 = RSTzwxG$.
 $(62 + 14 - 3)(4_29 + (-Tc)) \Rightarrow 2x = Td$.

P12.1.5 e8.1f.0a59 a

P12.1.6 n.8rob 11A E R_rR_T→Ai)Ax .1

$$= \prod_b b = 2^{12} = 4096$$

m A 1tLx0A 1fC .iaBAO C�pAl ASg 16ipM 1TapeniB AstAlf pmA ApAap 1ptf + MT
n=N n=Nd bO =N@Na I

Notes and References for

- fig. 8.8... r...8.4.R8.8.6 ..8. 8Jl. ...8.R... r. ...4..8. ...R48.** $w = x \bullet = VVx x =$
- F n. = $\boxed{x = a - a = a}$ ix $\boxed{(\cdot)}$. Fast Reliable Algorithms for Matrices with Structure, S, SI, $hS, \text{apmlahmgiaAgSmBc}$
- BcgtnAitB)ecTn11). Structured Matrices in Mathematics, Computer Science, and Engineering I and II, 3Is AlapACSIMI MtmAiipRBl, cIdJngh & 42w NMIStp, twowP
t1e- f.plw51 4tulC ppu5 - l a(tlayhuwPx1 liha.plyMiwApS4P 5Ap fAlt rtSalR(n1s1). Structured Matrices and Applications-The Geor Heinig Memorial Volume, wRMBmiItAM.sSMmarAM sRp- AM,iadM
- hiSAMt alaaBypAaAiAgISCAlaptp tpiSgAgiAMMpMIapiKIApMnRAl, JeAT
- ac5iR,jpmMS ar.iae Ic nlMfh(o). 4t.aJ1Atrixtpi oApya MEAll.ta ApSIIptAl x1rAi.MpawL J. Math. Anal. Appl. 68, 5.r ,ITv
- 9 AmIiae ac5imgip1as). 4t.aJ0 .tctpi 2ilr.irlt 5l 2Apyt0u5ApStlrixMpStApS ot0AitS 1Ail.ta wLin. Alg. Applic. 151, N. IITv
- ac5iPgip2 BaMltAe(s(,). 4t.aJ0A.tctpi 2ilr.irlt0 .utMlh ApSeJJ0.Ai.Mpaw SIAM Review 37, I,T-5d72
- ,c.lmSAMaM5imgiae B MmaitBt(h(,). 4 i CArara.Apx lrix.pA, Mpi NAli.A1N.IMi.p) 5l 1Ail.ta 2.iu t.aJ0.trixtpi 2ilr.irlt wDMath. Comput. 212, NT-, .T7v
- a.5imgipae B MmjtO (s(o)3 4tlaJ1A.tctprie2ilr.irlt eJJlMA.u iMNMOhpMc.AApStlrixMpSt ApS ot0AitS Ail.tawL Lin. Alg. Applic. 261, „.Jv „ AmaB(t(o) = 4tAill.ta 2.iu 2.uttellStl tl aJ0Actpi Iilr.irl tw11Lin. Alg. Applic. 218 I.,5J Nv
- nM (s(g). 42iA.0t ApSx/.tpi e0)Mliuca 5l 2ilr.irltS 2haitca M B.ptAl 2haitrixw SIAM J. Matrix Anal. Applic. 19, IT,15Jv
- scAmiaeM.ABiMiaI. P Ml ar.ypRiiae 9MnI(n1l0)Tr4e 2rJtI.wi e0)Mliuri5l MtJ1li3 2haitrixaM.flptAl x1rAilMpau SIAM J. Matr. Anal. Applic. 29, I,T NT7v
- EmtS,iaACNpBeA. asa)T pAapM A.pt -IiM16MIS,ACT
- IM..A2ia.M.Mirt. iae lc.c ACilat (s(α)-3 4tlaJ0Actp, NltMpS.ilMptl5l .MtJ0.i3 fftwi 2rAlta utilAi.Mpau TNA 2, „.7v
- Ic.I (s((g). 4,d2 xw1)Ml.i.rixa5l 2ilr.irltS I.ptAl ItAai 2rAlta NLM1trixw SIAM J. Matr. Anal. Applic. 20, I.,I7v
- .c lleMmrrA1.). 4xwi 2M0ri.MpMtJ0.i3reApS.Ar.(heI.t fftAaie2rAlta NLM1trixw SIAM J. Matr. Anal. Applic. 28, TI, Tdv
- IMmatRi2ppmSSgmaip BlentSgiaACAap. MAlaerRpPlAAMt
- .. lmSAMiae B c ,t2AitB(tα)-3 4.MrixJ1trlihM.1r0i.J 0..Ai.Mp2.iu 5t.iMla 5l 2ilr.irltS 1Aill.t8w 11 Linear Alg. Applic. 202, N 75.Iv
- nc95m,AMeEch,1,AiMts(α). 4N.JMitsAr.uhe0lyt NltMpSli.MptlaMl ot)r0All3tS 2M0rilMp M B eNMatNIM.0triaal SIAM J. Sci. Comput. 21, dd-NJv
- aMbiR,ipniae BcgtnAitBt(n11,). 4t.aJ .tctpi 2 ilrlt eJJlMA.u iMtlalitregl)MpMrxtil..3 gTrapa.MlcfAatS NltMpS.i.MptlaME-2ilAp) .hJt ApSMB .uAp .hJ tw1SIAM J. Matrix Anal. Applic. 26, TJ7-T5v

12.2 Structured-Rank Problems

Just as a sparse matrix has lots of zero entries, a structured rank matrix has lots of lowrank submatrices. For example, it could be that all off-diagonal blocks have unit rank. In this section we identify some important structured rank matrix problems and point to how they can be solved very quickly with data-sparse representations. To avoid complicated notation we adopt a small- n , proof-by-example style of exposition. Readers who prefer more detail and rigor should consult the definitive, two-volume treatise by Vandebril, Van Barel, and Meerbergen (2009).

$$1 \leq i_1 \leq n \quad 1 \leq j_1 \leq n \quad \text{and} \quad 1 \leq i_2 \leq n \quad 1 \leq j_2 \leq n$$

A matrix $A \in \mathbb{R}^{n \times n}$ is semiseparable if every block that does not "cross" the diagonal has unit rank or less. This means

$$j_2 \leq i_1 \text{ or } i_2 \leq j_1 \Rightarrow \text{rank}(A(i_1:i_2, j_1:j_2)) \leq 1. \quad (1221)$$

The rank-1 blocks of interest in a semiseparable matrix are wholly contained in either its upper triangular part or its lower triangular part, e.g.,

$$\begin{bmatrix} & & a_{11} & a_{12} & a_{13} \\ & & a_{21} & a_{22} & a_{23} \\ & & a_{31} & a_{32} & a_{33} \\ & & a_{41} & a_{42} & a_{43} \\ a_{11} & a_{12} & & & & \\ a_{21} & a_{22} & & & & \\ a_{31} & a_{32} & & & & \\ a_{41} & a_{42} & & & & \end{bmatrix}, \quad \begin{aligned} \text{rank}(A(1:3, 3:4)) &= 1, \\ \text{rank}(A(5:6, 1:2)) &= 1 \end{aligned}$$

Semiseparable matrices are data-sparse and enormous savings can be realized when their structure is exploited. For example, we will show that the factorizations $A = LU$ and $A = QR$ for semiseparable A require just $O(n)$ fops to compute and $O(n)$ fops to represent.

An important example of a semiseparable matrix is the inverse of a unit bidiagonal matrix. Given $r \in \mathbb{R}^{n-1}$ we define $B(r) \in \mathbb{R}^{n \times n}$ by

$$B(r) = \begin{bmatrix} 1 & -r_1 & 0 & 0 & 0 \\ 0 & 1 & -r_2 & 0 & 0 \\ 0 & 0 & 1 & -r_3 & 0 \\ 0 & 0 & 0 & 1 & -r_4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (1222)$$

Observe that any submatrix extracted from the upper triangular portion of

$$B(r)^{-1} = \begin{bmatrix} 1 & r_1 & r_{12} & r_{123} & r_{1234} \\ 0 & 1 & r_2 & r_{23} & r_{234} \\ 0 & 0 & 1 & r_3 & r_{34} \\ 0 & 0 & 0 & 1 & r_4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (1223)$$

has unit rank. If $x \in \mathbb{R}^n$ and $r = x(2n)/x(1:n-1)$ is defined, then

$$B(r)^T x = x_1 e_1.$$

Thus, the matrix $B(r)$ can (in principle) be used to introduce zeros into a vector.

hipigib 5Dt eee=Ett f P=bt f eTf=eb

Certain products of Givens rotations exhibit rank structure, but we focus on the key fact in more general terms. If $a, (., ', 8) \in I^{n-1}$ and

$$M_k = \text{diag}(I_{k-1}, \tilde{M}_k, I_{n-k-1}), \quad \tilde{M}_k = \begin{bmatrix} \alpha_k & \beta_k \\ \gamma_k & \delta_k \end{bmatrix},$$

for $k = 1 \dots n-1$, then the matrix $M = M_1 \dots M_{n-1}$ is fully illustrated by

$$a = MM_1M_2M_3M_4 = \begin{bmatrix} C1 & f1a2 & f1f2a3 & f1f2f3a4 & f1f2f3f4 \\ \gamma_1 & 802 & 8f2a3 & 8f2f3a4 & 8f2f3f4 \\ 0 & '2 & 803 & 8f3a4 & 8f3f4 \\ 0 & 0 & '3 & 804 & 8f4 \\ 0 & 0 & 0 & '4 & 04 \end{bmatrix}. \quad (1224)$$

It has the property that off-diagonal blocks have unit rank or less provided they do not "intersect" the diagonal. Quasiseparable matrices have this property and if A is such a matrix, then

$$j_2 < i_1 \text{ or } i_2 < j_1 \Rightarrow \text{rank}(A(i_1:i_2, j_1:j_2)) \leq 1. \quad (1225)$$

By comparing this with (1221), it is clear that the class of semiseparable matrices is a subset of the class of quasiseparable matrices.

hipiprb jlb 9-Ee=e=tf tf skleb

The MATLAB `tril` and `triu` notation is very handy when formulating a quasiseparable matrix computation. If $A \in \mathbb{R}^{m \times n}$, then a_{ij} is on its k th diagonal if $j = i + k$. The matrix $B = \text{tril}(A, k)$ is obtained from A by setting to zero all its entries above the k th diagonal while $B = \text{triu}(A, k)$ is obtained from A by setting to zero all its entries below the k th diagonal. If $k = 0$ then we simply write `tril(A)` and `triu(A)`. We also use the notation `diag(d)` to designate the diagonal matrix $\text{diag}(d_1, \dots, d_n)$ where $d \in \mathbb{R}^n$. Note that if $u, v, d, p, q \in \mathbb{R}^n$, then the matrix

$$A = \text{tril}(uv^T, -1) + \text{diag}(d) + \text{triu}(pq^T, 1) \quad (1226)$$

is quasiseparable, e.g.,

$$A = \begin{bmatrix} d & p1q2 & p1q3 & p1q4 & p1q5 \\ u1v1 & d2 & p2q3 & p2q4 & p2q5 \\ u2v1 & u3v2 & d3 & p3q4 & p3q5 \\ u3v1 & u4v2 & u4v3 & d4 & p4q5 \\ u4v1 & u5v2 & u5v3 & u5v4 & d5 \end{bmatrix}.$$

Should it be the case that $d = u \cdot \mathbf{R}^n = p \cdot a \cdot q$ then this matrix is semiseparable. The representation (1226) is referred to as the generator representation.

Not every qua iseparable matrix has a generator representation. For example, if $A = B(r)$ and r has nonzero entries, then it is impossible to find $u, v, d, p, q \in \mathbb{R}^n$ so that (1226) holds. To address this shortcoming we use the fact that

$$\begin{pmatrix} \text{Qua iseparable} \\ \text{Matrix} \end{pmatrix} \cdot * \begin{pmatrix} \text{Qua iseparable} \\ \text{Matrix} \end{pmatrix} = \begin{pmatrix} \text{Qua iseparable} \\ \text{Matrix} \end{pmatrix}, \quad (1227)$$

and embellish (1226) with a pair of inverse bidiagonal factors. It can be shown that if $A \in \mathbb{R}^{n \times n}$ is qua iseparable, then there exist $u, v, d, p, q \in \mathbb{R}^n$ and $t, r \in \mathbb{R}^{n-1}$ such that

$$A = t \cdot \text{tril}(uv, -1) \cdot * B(t)^{-1} T + \text{diag}(d) + \text{triu}(pqT, 1) \cdot * B(r)^{-1} S(u, v, t, d, p, q, r), \quad (1228)$$

e.g.,

$$A = \begin{bmatrix} d & P11qP & P11r2\beta & P11r234 & P11r2345 \\ U21M & d2 & P22\beta & P2234 & P22345 \\ U32M & U32M & da & p334 & p3345 \\ U432M & U432M & U433 & d4 & p445 \\ U5432M & U5432M & U5433 & U544 & ds \end{bmatrix}.$$

Wereferto(1228)as a quasiseparable presentation and it has a number of important specializations. If $d = u \cdot * v = p \cdot * q$ then A is semiseparable. If $t = r = 1_{n-1}$ then A is generator representable. If $u = q, v = p$, and $t = r$, then A is symmetric. The representation also supports the semiseparable plus diagonal structure. A matrix $S(u, v, t, d, p, q, r)$ has this form if d is arbitrary and $u \cdot * v = P \cdot * q$. Here are some inverse related facts that pertain to semiseparable, qua iseparable, and diagonal-plus-semiseparable matrices:

Fact 1. If A is nonsingular and tridiagonal, then A^{-1} is semiseparable. In addition, if the subdiagonal and superdiagonal entries are nonzero, then A^{-1} is generator representable.

Fact 2. If A is nonsingular and qua iseparable, then so is A^{-1} .

Fact 3. If $A = D + 8$ is nonsingular where D is diagonal and nonsingular and 8 is semiseparable, then $A^{-1} = n^{-1} + 8^{-1}$ where 8^{-1} is semiseparable.

Aspects of the first fact were encountered in §438

$$2 \cdot h(A)A^{-1}MN^{-1} - \{G\} + A^{-1}u \cdot * v \cdot F(T)T^T G \cdot f(NF) \cdot \{A\} \cdot \{d\} - \{G + d\} = \{A\} \cdot F(A) \cdot \{d\} - \{G + d\} = A$$

Lower and upper triangular matrices that are also semiseparable can be written as follows

$$\text{Lower semiseparable} = L = S(u, v, t, u \cdot * v, Q, Q) = \text{tril}(uv) \cdot * B(t)^{-1} \cdot r,$$

$$\text{Upper semiseparable} = U = S(Q, Q, p \cdot * q, p, q, r) = \text{triu}(pqT) \cdot * B(r)^{-1}.$$

Operations with matrices that have this structure can be organized very efficiently. Consider the matrix-vector product

$$y = (\text{triu}(pq^T) .* B(r)^{-1}) x \quad (12.29)$$

where $x, y, p, q \in \mathbb{R}^n$ and $r \in \mathbb{R}^{n-1}$. This calculation has the form

$$\begin{bmatrix} p_1 & P_{12} & P_{13} & P_{14} \\ 0 & 0 & p_2 & p_2 p_3 \\ 0 & 0 & 0 & p_3 p_4 \end{bmatrix} \begin{bmatrix} 1 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} y_3 \\ y_4 \end{bmatrix}$$

By grouping the q s with the x s and extracting the p s, we see that

$$\text{diag}(p_1 p_2 p_3 p_4) \begin{bmatrix} 1 & r_1 & r_1 r_2 & r_1 r_2 r_3 \\ 0 & 1 & r_2 & r_2 r_3 \\ 0 & 0 & 1 & r_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} Qx_1 \\ Qx_2 \\ Qx_3 \\ Qx_4 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

In other words, (12.29) is equivalent to

$$y = P .* \{B(r)^{-1}(Q * x)\}.$$

Given x , this is clearly an $O(n)$ computation since bidiagonal systemsolving is $O(n)$. Indeed, y can be computed with just $4n$ flops.

Note that if y is given in (12.29) and p and q have nonzero components, then we can solve for x equally fast: $x = (B(r)(y./p)) ./q$.

siuiuib 7\\$b IRb 3fQkT atQTkbkb tbu\\$ Tess t:PSb ht QTeb

Suppose $A = S(u, v, t, u.^*V, p, q, r)$ is an n -by- n semiseparable matrix that has an LU factorization. It turns out that both L and U are semiseparable and their respective representations can be computed with $O(n)$ work.

for k = n:-1:-1

Using A 's representation, determine T_k so that if $A = M_k A$, where

$$M_k = \text{diag}(I_{k-1}, \tilde{M}_k, I_{n-k-1}), \quad L_k = \begin{bmatrix} 1 & 0 \\ -T_k & 1 \end{bmatrix},$$

then $A(k+1:k,k)$ is zero

(12.20)

Compute the update $A = L_k A$ by updating A 's representation

end

$U = A$

Note that if $M = M_1 \cdots M_{n-1}$, then $MA = U$ and $M = B(\tau)$ with $\tau = [\tau_1, \dots, \tau_{n-1}]^T$. It follows that if $L = M^{-1}$, then L is semiseparable from (12.24) and $A = LU$. The challenge is to show that the updates $A = M_k A$ preserve semiseparability.

To see what is involved, suppose $n = 6$ and that we have computed 115 and M_4 so that

$$M_4 M_5 A = \left[\begin{array}{cccc|cc} x & x & x & x & x & x \\ x & x & x & x & x & x \\ A & A & A & \mu & \mu & \mu \\ A & A & A & \mu & \mu & \mu \\ \hline 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{array} \right] = S(u, v, t, u \cdot v, p, q, r)$$

is semiseparable. Note that the A block and the μ block are given by

$$\begin{bmatrix} \lambda & A & \lambda \\ \lambda & A & \lambda \end{bmatrix} = \begin{bmatrix} u_3 t_2 t_1 v_1 & u_3 t_2 v_2 & u_3 v_3 \\ u_4 t_3 t_2 t_1 v_1 & u_4 t_3 t_2 v_2 & u_4 t_3 v_3 \end{bmatrix},$$

$$\begin{bmatrix} \mu & \mu & \mu \\ \mu & \mu & \mu \end{bmatrix} = \begin{bmatrix} p_3 r_3 q_4 & p_3 r_3 r_4 q_5 & p_3 r_3 r_4 r_5 q_6 \\ p_4 q_4 & p_4 r_4 q_5 & p_4 r_4 r_5 q_6 \end{bmatrix}.$$

Thus, if

$$lh = \begin{bmatrix} 1 \\ -\tau_3 \end{bmatrix},$$

then

$$\tilde{M}_3 \begin{bmatrix} A & A & \lambda \\ A & A & \lambda \end{bmatrix} = \begin{bmatrix} u_3 t_2 t_1 v_1 & u_3 t_2 v_2 & u_3 v_3 \\ (u_4 t_3 - \tau_3 u_3) t_2 t_1 v_1 & (u_4 t_3 - \tau_3 u_3) t_2 v_2 & (u_4 t_3 - \tau_3 u_3) v_3 \end{bmatrix},$$

$$\tilde{M}_3 \begin{bmatrix} \mu & \mu & \mu \\ \mu & \mu & \mu \end{bmatrix} = \begin{bmatrix} p_3 r_3 q_4 & p_3 r_3 r_4 q_5 & p_3 r_3 r_4 r_5 q_6 \\ (p_4 - \tau_3 p_3 r_3) q_4 & (p_4 - \tau_3 p_3 r_3) r_4 q_5 & (p_4 - \tau_3 p_3 r_3) r_4 r_5 q_6 \end{bmatrix}.$$

If $u_3 = Q\tau_3 = u_4 t_3 / u_3$, and we perform the updates

$$u_4 = 0, \quad p_4 = p_4 - \tau_3 p_3 r_3,$$

then

$$M_3 M_4 M_5 A = \left[\begin{array}{cccc|cc} x & x & x & x & x & x \\ x & x & x & x & x & x \\ A & A & A & \mu & \mu & \mu \\ 0 & 0 & 0 & f & f & f \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \end{array} \right] = S(u, v, t, u \cdot v, p, q, r)$$

is still semiseparable. (The tildes designate updated entries.) Picking up the pattern from this example, we obtain the following O(n) method for computing the LU factorization of a semiseparable matrix.

Algorithm 12.2.1 Assume that $uv^*pq^* \in \mathbb{R}^n$ with $U^*V = p^*q$ and that $t, r \in \mathbb{R}^{n-1}$. If $A = \begin{bmatrix} S & uv^* \\ v^*p & q \end{bmatrix}$ has an LU factorization, then the following algorithm computes $p \in \mathbb{R}^n$ and $T \in \mathbb{R}^{n-1}$, so that if $L = Br^{-1}T$ and $U = \text{tri}(pq^*)B^{-1}$, then $A = LU$.

```

for k = n-1:-1
    Tk = tk / uk
    Pk = Pk - Pkk
end
f1 = p1

```

This algorithm requires about $5n^2$ ops. Given our remarks in the previous section about triangular semiseparable matrices, we see that a semiseparable system $\bar{A}\bar{x} = b$ can be solved with O(n) work: $A = LU$, $Ly = b$, $Ux = y$. Note that the vectors T and p in algorithm 12.2.1 are given by

$$T = (u(2n) \cdot *t) / u(n-1)$$

and

$$p = \begin{bmatrix} p_1 \\ p(2n) - p(n-1) \cdot *T^*r \end{bmatrix}.$$

Pivoting can be incorporated in Algorithm 12.2.1 to ensure that $|k| \leq 1$ for $k = n-1:-1$. At the beginning of step k , if $|k| < |k+1|$, then rows k and $k+1$ are interchanged. The swapping is orchestrated by updating the quasireducible representation of the current \bar{A} . The end result is an O(n) reduction of the form $M \cdots M_k \bar{A} = U$, where U is upper triangular and quasireducible and $M = \text{diag}(h_1, M_2, \dots, M_k)$ with

$$\tilde{P}_k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \text{ or } \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

See Vandebril, Van Barel, and Meerbergen (2008, pp. 15–17) for further details and also how to perform the same tasks when A is quasireducible.

sililnb 78 01f 8e s 8ff k8 98 Ex8 8tf 1kb

The QR factorization of a semiseparable matrix is also an O(n) computation. To motivate the algorithm we step through a simple special case that showcases the idea of a structured rank Givens update. Along the way we will discover yet another strategy that can be used to represent a semiseparable matrix.

Assume $A \in \mathbb{R}^{n \times n}$ is a lower triangular semiseparable matrix and that $a \in \mathbb{R}^n$ is its first column. We can reduce this column to a multiple of e_1 with a sequence of

$n - 1$ Givens rotations, eg

$$\begin{bmatrix} s_1 & 0 & 0 \\ s_1 & c_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & c_2 & s_2 \\ 0 & -s_2 & c_2 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & c_3 & s_3 \\ 0 & -s_3 & c_3 \end{bmatrix} = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \\ \bullet \end{bmatrix}$$

By moving the rotations to the right-hand side we see that

$$A(:,1) = \begin{bmatrix} \bullet \\ \bullet \\ \bullet \\ \bullet \end{bmatrix} = v_1 \begin{bmatrix} c_{11} \\ s_{21} \\ s_{31} \\ s_{41} \end{bmatrix}$$

Because this is the first column of a semiseparable matrix, it is not hard to show that there exist "weights" v_2, \dots, v_n so that

$$A_L = \begin{bmatrix} c_1 v_1 & 0 & 0 & 0 \\ c_2 s_1 v_1 & c_2 v_2 & 0 & 0 \\ c_3 s_2 s_1 v_1 & c_3 s_2 v_2 & c_3 v_3 & 0 \\ s_3 s_2 s_1 v_1 & s_3 s_2 v_2 & s_3 v_3 & v_4 \end{bmatrix} = B(s)^{-T} .* \text{tril}(cv^T) \quad (1221)$$

where

$$v = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}, \quad c = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ 1 \end{bmatrix}, \quad s = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix}.$$

The encoding (1221) is an example of the Givens-vector representation for a triangular semiseparable matrix. It consists of a vector of cosines, a vector of sines, and a vector of weights. By "transposing" this idea, we can similarly represent an upper triangular semiseparable matrix. Thus, for a general semiseparable matrix A we may write

$$A = A_L + A_U,$$

where

$$A = \text{tril}(A) = B(s_L)^{-T} .* \text{tril}(c_L v_L^T),$$

$$A = \text{triu}(A, 1) = B(s)^{-1} \cdot \text{triu}(v, 1),$$

where c_L , s_L , and v (resp. c_U , s_U , and v) are the cosine, sine, and weight vectors associated with the lower (resp. upper) triangular part. For more details on the properties and utility of this representation, see Vandebril and Van Barel (2005).

$$\text{1mcA} \quad \text{TDa T.A} \quad \text{Sj.} \quad = \text{GQ} \cdot \text{GAjA} \} \text{Alol} \quad \text{G+G} \} = \} \text{A2Aq} \quad = \text{GpA}$$

The matrix Q in the QR factorization of a semiseparable matrix $A \in \mathbb{R}^{n \times n}$ has a very simple form. Indeed, it is a product of Givens rotations $Q = G_1 \cdots G_{n-1}$ where the

underlying cosine-sine pairs are precisely those that define Givens representation of A. To see this, consider how easy it is to compute the QR factorization of A:

$$\begin{array}{c}
 \left[\begin{array}{cccc} 1 & & & \\ 0 & & & \\ 0 & 0 & c_3 & s_3 \\ 0 & 0 & -s_3 & c_3 \end{array} \right] \left[\begin{array}{cccc} c_1 v_1 & 0 & 0 & 0 \\ c_2 s_1 v_1 & c_2 v_2 & 0 & 0 \\ c_3 s_2 s_1 v_1 & c_3 s_2 v_2 & c_3 v_3 & 0 \\ s_3 s_2 s_1 v_1 & s_3 s_2 v_2 & s_3 v_3 & c_3 v_4 \end{array} \right] = \left[\begin{array}{cccc} c_1 v_1 & 0 & 0 & 0 \\ c_2 s_1 v_1 & c_2 v_2 & 0 & 0 \\ s_2 s_1 v_1 & s_2 v_2 & v_3 & s_3 v_4 \\ 0 & 0 & 0 & c_3 v_4 \end{array} \right], \\
 \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & c_2 & s_2 & 0 \\ 0 & -s_2 & c_2 & 0 \\ 0 & 0 & 0 & 1 \end{array} \right] \left[\begin{array}{cccc} c_1 v_1 & 0 & 0 & 0 \\ c_2 s_1 v_1 & c_2 v_2 & 0 & 0 \\ s_2 s_1 v_1 & s_2 v_2 & v_3 & s_3 v_4 \\ 0 & 0 & 0 & c_3 v_4 \end{array} \right] = \left[\begin{array}{cccc} 1 & 0 & 0 & 0 \\ v_2 & s_2 v_3 & s_2 v_4 & \vdots \\ 0 & 0 & c_2 v_3 & c_2 s_3 v_4 \\ 0 & 0 & 0 & c_3 v_4 \end{array} \right], \\
 \left[\begin{array}{cccc} c_1 & s_1 & 0 & 0 \\ -s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array} \right] \left[\begin{array}{cccc} c_1 v_1 & 0 & 0 & 0 \\ s_1 v_1 & v_2 & s_2 v_3 & s_2 s_3 v_4 \\ 0 & 0 & c_2 v_3 & c_2 s_3 v_4 \\ 0 & 0 & 0 & c_3 v_4 \end{array} \right] = \left[\begin{array}{cccc} v_1 & s_1 v_2 & s_1 s_2 v_3 & s_1 s_2 s_3 v_4 \\ 0 & c_1 v_2 & c_1 s_2 v_3 & c_1 s_2 s_3 v_4 \\ 0 & 0 & c_2 v_3 & c_2 s_3 v_4 \\ 0 & 0 & 0 & c_3 v_4 \end{array} \right].
 \end{array}$$

In general, if $\text{tril}(A) = B(s) - T_{\text{Rtril}}(v)$ is a Givens vector representation and

$$Q^T = G_1 \cdots G_{n-1} \quad (122.12)$$

where

$$G_k = \text{diag}(I_{k-1}, \tilde{G}_k, I_{n-k-1}), \quad \tilde{G}_k = \begin{bmatrix} c_k & s_k \\ -s_k & c_k \end{bmatrix}, \quad (122.13)$$

if $k = 1:n-1$, then

$$Q^T \text{tril}(A) = R_L = \text{triu}((D_n c)v^T) . * B(s)^{-1}. \quad (122.14)$$

(Recall that V_n is the downshift permutation, see §13.) Since Q^T is upper Hessenberg it follows that

$$Q^T \text{triu}(A, 1) = R_U$$

is also upper triangular. Thus

$$Q^T A = Q^T (A + A_{\perp}) = R_L + R_{\perp} = R$$

is the QR factorization of A. Unfortunately, this is not a useful O(n) representation of R from the standpoint of solving $Ax = b$ because the summation gets in the way when we try to solve $(R_L + R_{\perp})x = Q^T b$.

Fortunately, there is a harder way to encode R. Assume for clarity that A has a generator representation

$$A = \text{tril}(uv^T) + \text{triu}(pq^T), \quad (122.15)$$

where $u, v, p, q \in \mathbb{R}^n$ and $u \cdot v = p \cdot q$. We show that R is the upper triangular portion of a rank-2 matrix, i.e.,

$$R = \text{triu}(fg^T + hq^T), \quad f, g, h \in \mathbb{R}^n. \quad (122.16)$$

This means that any submatrix extracted from the upper triangular part of R has rank two or less.

From (122.15) we see that the first column of A is a multiple of u . It follows that the Givens rotations that define Q in (122.12) can be determined from this vector:

$$G_1 \cdots G_{n-1} u = \begin{bmatrix} \tilde{u}_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Suppose $n = 6$ and that we have computed G_5 , G_4 and G_3 so that $A\tilde{\beta} = G_3G_4G_5A$ has the form

$$A\tilde{\beta} = \begin{bmatrix} \mathbf{U}\mathbf{M} & p_1q_2 & p_1q_3 & p_1q_4 & p_1q_5 & p_1q_6 \\ \mathbf{U}\mathbf{M} & \mathbf{U}\mathbf{M} & p_2q_3 & p_2q_4 & p_2q_5 & p_2q_6 \\ \mathbf{U}\mathbf{M} & \mathbf{U}\mathbf{M} & \mathbf{U}\mathbf{M} & \tilde{f}_3g_4 + \tilde{h}_3 & \tilde{f}_3g_5 + \tilde{h}_5 & \tilde{f}_3g_6 + \tilde{h}_6 \\ 0 & 0 & 0 & f_4g_4 + h_4q_4 & f_4g_5 + h_4q_5 & f_4g_6 + h_4q_6 \\ 0 & 0 & 0 & 0 & f_5g_5 + h_5q_5 & f_5g_6 + h_5q_6 \\ 0 & 0 & 0 & 0 & 0 & f_6g_6 + h_6q_6 \end{bmatrix}.$$

Next, we compute the cosine-sine pair $\{\tilde{\alpha}, \tilde{\beta}\}$ so that

$$\begin{bmatrix} \tilde{\alpha} & \tilde{\beta} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \tilde{\alpha} & \tilde{\beta} \\ -\tilde{\beta} & \tilde{\alpha} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \tilde{\beta} \end{bmatrix} = \begin{bmatrix} \tilde{\alpha} & 0 \\ 0 & \tilde{\beta} \end{bmatrix}.$$

Since

$$\begin{bmatrix} c_2 & s_2 \\ -s_2 & c_2 \end{bmatrix} \begin{bmatrix} p_2q_j \\ \tilde{f}_3g_j + \tilde{h}_3q_j \end{bmatrix} = \begin{bmatrix} c_2p_2 + s_2\tilde{h}_3 \\ -s_2p_2 + c_2\tilde{h}_3 \end{bmatrix} q_j + \begin{bmatrix} s_2\tilde{f}_3 \\ c_2\tilde{f}_3 \end{bmatrix} g_j,$$

from $j = 36$ it follows that $A\tilde{\beta} = G_2\tilde{\alpha}G_3\tilde{\beta} = \text{diag}(1, G_2\tilde{\alpha}G_3\tilde{\beta})$ has the form

$$A\tilde{\beta} = \begin{bmatrix} \mathbf{U}\mathbf{M} & p_1q_2 & p_1q_3 & p_1q_4 & p_1q_5 & p_1q_6 \\ \mathbf{U}\mathbf{M} & \mathbf{U}\mathbf{M} & \mathbf{U}\mathbf{M} & \mathbf{U}\mathbf{M} & \mathbf{U}\mathbf{M} & \mathbf{U}\mathbf{M} \\ 0 & 0 & f_3g_3 + h_3q_3 & f_3g_4 + h_3q_4 & f_3g_5 + h_3q_5 & f_3g_6 + h_3q_6 \\ 0 & 0 & 0 & f_4g_4 + h_4q_4 & f_4g_5 + h_4q_5 & f_4g_6 + h_4q_6 \\ 0 & 0 & 0 & 0 & f_5g_5 + h_5q_5 & f_5g_6 + h_5q_6 \\ 0 & 0 & 0 & 0 & 0 & f_6g_6 + h_6q_6 \end{bmatrix}.$$

where

$$\tilde{f}_2 = s_2 \tilde{f}_3, \quad f_3 = c_2 \tilde{f}_3, \quad \tilde{h}_2 = c_2 p_2 + s_2 \tilde{h}_3, \quad h_3 = -s_2 p_2 + c_2 \tilde{h}_3.$$

By considering the transition from $A^{(3)}$ to $A^{(2)}$ via the Givens rotation G_2 , we conclude that $[A^{(2)}]_{22} = \tilde{u}_2 v_2$. Since this must equal $\tilde{f}_2 g_2 + \tilde{h}_2 q_2$ we have

$$g_2 = \frac{\tilde{u}_2 v_2 - \tilde{h}_2 q_2}{\tilde{f}_2}.$$

By extrapolating from this example and making certain assumptions to guard against division by zero, we obtain the following QR factorization procedure:

Algorithm 122.2.2 Suppose U, v, p and q are n -vectors that satisfy $U \cdot v = p \cdot q$ and $U \perp Q$. If $A = \text{tril}(U^T) + \text{triu}(pq^T, 1)$, then this algorithm computes cosine-sine pairs $\{C_k, S_k\}, \dots, \{C_l, S_l\}$ and vectors $f, g \in \mathbb{R}^n$ so that if Q is defined by (122.12) and (122.13), then $QA = R = \text{triu}(fg^T + hq^T)$.

$$\tilde{u}_n = u_n, \quad \tilde{f}_n = u_n, \quad g_n = v_n, \quad h_n = 0$$

for $k = n-1 \dots 1$

Determine G_k and S_k so that $\begin{bmatrix} C_k & S_k \\ -S_k & C_k \end{bmatrix} \begin{bmatrix} U_k \\ U_{k+1} \end{bmatrix} = \begin{bmatrix} U_k \\ 0 \end{bmatrix}$.

$$A = S_k G_k, \quad f_{k+1} = G_k f_k$$

$$\begin{bmatrix} h_{k+1} \\ \vdots \\ h_1 \end{bmatrix} = \begin{bmatrix} \ddots & \ddots \\ \ddots & \ddots \end{bmatrix} \begin{bmatrix} h_k \\ \vdots \\ h_1 \end{bmatrix}$$

$$g_k = (U_k^T h_k) / A$$

end

$J = 1$

Regarding the condition that $U_h \perp Q$ it is easy to show by induction that

$$ik = Sk \cdot Sh$$

The S_k are nonzero because $\|U_k\|_2 = \|U_{k:n}\|_2 = 0$. This algorithm requires $O(n)$ fops and $O(n)$ storage. We stress that there are better ways to implement the QR factorization of a semiseparable matrix than Algorithm 122.2. See VanCamp, Magonard, and Van Barel (2004). Our goal, as stated above, is to suggest how a structured rank matrix factorization can be organized around Givens rotations. Equally efficient QR factorizations for quasireversible and semireversible plus diagonal matrices are also possible.

We mention that an n -by- n system of the form $\text{triu}(fg^T + hq^T)x = y$ can be solved in $O(n)$ fops. An induction argument based on the partitioning

$$\begin{bmatrix} f_k G_k; H_k & f_{k+1} \dots !^T \\ ; \dots ; & ! \end{bmatrix} \begin{bmatrix} k \\ : \end{bmatrix} = \begin{bmatrix} y_k \\ \tilde{y} \end{bmatrix}$$

where all the "tilde" vectors belong to \mathbb{R}^{n-k} , shows why. If $\tilde{x}, \alpha = T\tilde{x}$, and $T\tilde{x}$ are available, then X and the updates $\alpha = \alpha + G_k X$ and $\{ = \{ + G_k X$ require $O(1)$ fops.

$$2hAqA \quad S - o\bar{A} \quad .1 \quad T(L = Ni \cdot N = dA \cdot 3+ + oA)$$

We briefly mention several other rank structures that arise in applications. Fat LU and QR procedures exist in each case.

If p and q are nonnegative integers, then a matrix A is $\{p, q\}$ -semiseparable if

$$\begin{aligned} h < i_1 + p &= \text{rank}(A[i_1:i_2, j_1:j_2]) \leq p \\ i_2 > 1 + q &= \text{rank}(A[i_1:i_2, j_1:j_2]) \leq q \end{aligned}$$

For example, if A is $\{2, 3\}$ -semiseparable, then

$$A = \left[\begin{array}{ccccccc} \times & \times & \times & \times & \times & \times & \times \\ a_{21} & a_{22} & a_{23} & \times & \times & \times & \times \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} & a_{37} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} & a_{47} \\ \times & \times & \times & a_{54} & a_{55} & a_{56} & a_{51} \\ \times & \times & \times & a_{64} & a_{65} & a_{66} & a_{61} \\ \times & \times & \times & a_{14} & a_{15} & a_{16} & a_{11} \end{array} \right] = \begin{matrix} \text{rank}(A[24:13]) \leq 2 \\ \text{rank}(A[37,47]) \leq 3 \end{matrix}$$

In general, A is $\{p, q\}$ -generally representable if we have $U, V \in \mathbb{R}^{n \times p}$ and $P, Q \in \mathbb{R}^{n \times q}$ such that

$$\begin{aligned} \text{tril}(A, p-1) &= \text{tril}(UV^T, p-1), \\ \text{triu}(A, -q+1) &= \text{triu}(PQ^T, -q+1). \end{aligned}$$

If such a matrix is nonsingular, then A^{-1} has lower bandwidth p and upper bandwidth q . If the $\{p, q\}$ -semiseparable definition is modified so that the rank- p blocks come from $\text{tril}(A)$ and the rank- q blocks come from $\text{triu}(A)$, then A belongs to the class of extended $\{p, q\}$ -separable matrices. If the $\{p, q\}$ -semiseparable definition is modified so that the rank- p blocks come from $\text{tril}(A, -1)$ and the rank- q come from $\text{triu}(A, 1)$, then A belongs to the class of extended $\{p, q\}$ -quasiseparable matrices. A sequentially semiseparable matrix is a block matrix that has the following form:

$$A = \left[\begin{array}{ccccc} D_1 & P_1 Q_1 & P_1 R_2 Q_2 & P_1 & Q_1 \\ U_2 V_1^T & D_2 & P_2 Q_2 & P_2 R_3 Q_3 & \\ U_3 T_2 V_1^T & U_3 V_2^T & D_3 & P_3 Q_3 & \\ U_4 T_3 T_2 V_1^T & U_4 T_3 V_2^T & U_4 V_3^T & D_4 & \end{array} \right]. \quad \{122.17\}$$

See Dewilde and van der Veen (1997) and Chandra et al. (2005). The blocks can be rectangular so least squares problems with this structure can be handled.

Matrices with hierarchical rank structure are based on lowrank patterns that emerge through recursive 2by2 blockings. (With one level of recursion we would have 2by2 block matrix whose diagonal blocks are 2by2 block matrices.) Various connections may exist between the lowrank representations of the off-diagonal blocks. The important class of hierarchically semiseparable matrices has a particularly rich and exploitable structure, see Xia (2012).

1t.lUA lOl mD} = }ABoA 6Gf dI }ENoAa= A31-A }IA To iE GzN oA

Fast versions of various two-sided, eigenvalue-related decompositions also exist. For example, if $A \in \mathbb{R}^{n \times n}$ is symmetric and diagonal-plus-semiseparable, then it is possible to compute the tridiagonalization $Q^T A Q = T$ in $O(n^2)$ fops. The orthogonal matrix Q is a product of Givens rotations each of which participate in a highly-structured update. See Meerbergen, Chandra, and Van Huffel (2001).

There are also interesting methods for general matrix problems that involve the introduction of semiseparable structures during the solution process. Van Barel, Van berghen, and van Dooren (2010) approach the product SVD problem through conversion to a semiseparable structure. For example, to compute the SVD of $A = A_1 A_2$ orthogonal matrices U_1 , U_2 and U_3 are first computed so that $(U_1 A_1 U_2)(U_2 A_2 U_3) = T$ is upper triangular and semiseparable. Verberghen, Vandebril, and Van Barel (2008) have shown how to compute orthogonal $Q \in \mathbb{R}^{n \times n}$ so that $Q^T B Z = R$ is upper triangular and $Q^T A Z = L$ has the property that $\text{tril}(L)$ is semiseparable. A procedure for reducing the equivalent pencil $L - \lambda R$ to generalized Schur form is also developed.

highinhsb b1"Stf t f1\$eb k b ttb 6x f ek" ktrb R33\$b IS ee\$:\$ "b ht f xTeb

We close with an eigenvalue problem that has quasimseparable structure. Suppose $H \in \mathbb{R}^{n \times n}$ is an upper Hessenberg matrix that is also orthogonal. Our goal is to compute $\sigma(H)$. Note that each eigenvalue is on the unit circle. Without loss of generality we may assume that the subdiagonal entries are nonzero.

If n is odd, then it must have a real eigenvalue because the eigenvalues of a real matrix come in complex conjugate pairs. In this case it is possible to deflate the problem by carefully working with the eigenvector equation $H\mathbf{x} = \mathbf{x}$ (or $H\mathbf{x} = -\mathbf{x}$). Thus, we may assume that n is even.

For $1 \leq k \leq n-1$, define the reflection $G_k \in \mathbb{R}^{n \times n}$ by

$$G_k = G(\phi_k) = \text{diag}(-1, R(\phi_k), I_{n-k-1})$$

where

$$R(\phi_k) = \begin{bmatrix} -\cos(\phi_k) & \sin(\phi_k) \\ \sin(\phi_k) & \cos(\phi_k) \end{bmatrix}, \quad 0 < \phi_k < \pi$$

These transformations can be used to represent the QR factorization of H . Indeed, after the Givens process described in §5.2.6 we can compute G_1, \dots, G_{n-1} so that

$$G_{n-1} \cdots G_1 H = G_n = \text{diag}(1, \dots, 1, -c_n).$$

The matrix G_n is the “ R ” matrix. It is diagonal because an orthogonal upper triangular matrix must be diagonal. Since the determinant of a matrix is the product of its eigenvalues, the value of c_n is either $+1$ or -1 . If $c_n = -1$, then $\det(H) = -1$, which in turn implies that H has a real eigenvalue and we can deflate the problem. Thus, we may assume that

$$H = G_1 \cdots G_n, \quad G_n = \text{diag}(1, \dots, 1, -1), \quad n = 2. \quad (12.2.18)$$

and that our goal is to compute

$$\sigma(H) = \{\cos(\theta_1) \pm i \sin(\theta_1), \dots, \cos(\theta_m) \pm i \sin(\theta_m)\}. \quad (12.2.19)$$

Note that (12.2.4) and (12.2.18) tell us that H is quasireversible.

Ammar, Gragg and Reichel (1986) propose an interesting $O(n^2)$ method that computes the required eigenvalues by setting up a pair of $m \times m$ bidagonal SVD problems. Three facts are required:

Fact 1. H is similar to $f = H_0 + He$ where

$$\begin{aligned} H_0 &= G_1 G_3 \cdots G_{n-1} = \text{diag}(R(<1), R(<3), \dots, R(<n-1)), \\ H_e &= G_2 G_4 \cdots G_n = \text{diag}(1, R(<2), R(<4), \dots, R(<n-2), -1). \end{aligned}$$

Fact 2. The matrices

$$C = \frac{H_0 + H_e}{2}, \quad S = \frac{H_0 - H_e}{2}$$

are symmetric and tridiagonal. Moreover, their eigenvalues are given by

- . $(C) = \{\pm \cos(f_i/2), \dots, \pm \cos(Q_m/2)\},$
- . $(S) = \{\pm \sin(Q/2), \dots, \pm \sin(Q_m/2)\}.$

Fact 3. If

$$\begin{aligned} QD &= \text{diag}(R(<1/2), R(<3/2), \dots, R(<n-1/2)), \\ QE &= \text{diag}(1, R(<2/2), R(<4/2), \dots, R(<n-2/2), -1), \end{aligned}$$

then perform shuffles of the matrices

$$C^{(1)} = Q_o C Q_e, \quad S^{(1)} = Q_o S Q_e$$

expose a pair of $m \times m$ bidiagonal matrices B_C and B_S with the property that

$$\begin{aligned} \sigma(B_C) &= \{\cos(Q/2), \dots, \cos(Q_m/2)\}, \\ \sigma(B_S) &= \{\sin(Q/2), \dots, \sin(Q_m/2)\}. \end{aligned}$$

Once the bidiagonal matrices B_C and B_S are set up (which involves $O(n)$ work), then their singular values can be computed via Golub-Kahan SVD algorithm. The angle k can be accurately determined from $\sin((k/2))$ if $0 < k < \pi/2$ and from $\cos((k/2))$ otherwise. See Ammar, Gragg and Reichel (1986) for more details.

Problems

P12.2.1 **7.8.8.... .8.8 ..1 ..8b** $B(r)^{-1}$ matc.atJAIA.0t1

P12.2.2 **3.8.8 A mI.mtASiMiSpSe lagt mA = S(u, t, v, d, p, r, q) 5l AJJIMJl.Ait0huMattIt.CMlav, v, t, d, p, r, ApSq.**

P12.2.3 **P8rob . s...m.8r.r.8. .8 8b84.8.mb .84.8. .8..41** $y = Ax + mAMx$
 $A = S(u, v, t, d, p, q, r).$

P12.2.4 **78>8.8 f.0a0aP8.8... 8u, v, t, d, p, q, ApSr aMuAi M= S(u, v, t, d, p, q, r).**

P12.2.5 **... .8.8 S(u, v, t, d, v, u, t) mAhctil.. JMaMi.I8trnltmitApSatcmatJAIA.0t, 2uM. CuA**C**a.uM0tayh.A iMlmatc.atJAIA.0t ApS).It ApA0)Ml.iuc ,LMc.r.i.t) .ia .Nw.atJAIA.0t ltJltatpiAi.Mt3**

P12.2.6 e8.1 .8 ..88 >.4..1, QN0 9a

P12.2.7 88.8.8.. 11.8.8. N.8.1.. 8...8. .1..... 8. y + mA M A p6nA M lb
) Gelirf :., f ^r_W, B ^r_N, ^T₄ ^{Up}_{Wly} E ^r_n ^m r E ^r_{n-1}. y + mA M A p6nA M lb

P12.2.8 e8.1 f.ONOFFP

P12.2.9 3.8.8 f.0a0P5

P12.2.10 1...8 1.. 1.. 1.1 ..841b ..84d ..8 N.4. 1.. N.8 ≥1. ...81.. N. 48...1.
 1....8. 1.f.0a0a.R5.88 ..84d..8 N.4. 1.. N.8 ≥1. ...81.. N. 48...1.
 y G ra + mAMAR^{N.m.}

P12.2.11 re4.. .8 ..8rob. ..

$$A = \begin{bmatrix} A_1 & B_1^T & 0 & 0 \\ B_1 & A_2 & B_2^T & 0 \\ 0 & B_2 & A_3 & B_3^T \\ 0 & 0 & B_3 & A_4 \end{bmatrix} = \frac{1}{4} \begin{bmatrix} V_1 V_1^T & V_1 U_2^T & V_1 U_3^T & V_1 U_4^T \\ U_2 V_1^T & U_2 V_2^T & V_2 U_3^T & V_2 U_4^T \\ U_3 V_1^T & U_3 V_2^T & U_3 V_3^T & V_3 U_4^T \\ U_4 V_1^T & U_4 V_2^T & U_4 V_3^T & U_4 V_4^T \end{bmatrix}$$

$\Rightarrow \mathbf{v} = \mathbf{U}_1, \dots, \mathbf{U}_4$

P12.2.12 n.. .8.8 a, b, /, g E \mathbf{R}^n , $m^n - l^n$ $\mathbf{R}^n + fg\mathbf{f}$. **İtlatlar IgİMVi.** .iAaE pA,
 $(p^{n_p} n_p^{n_p})^{n^n}$ n^n $n \in \mathbb{N}^n$ + y G r a M S.1 i A a y E \mathbf{R}^n , $(p^{n_p} n_p^{n_p})_{-l^n}$ $E \mathbf{R}^n$ p^{n^n}
y. f a l i A a y, d E \mathbf{R}^n , $(p^{n_p} n_p^{n_p})_{-l^n}$ p^{-l^n} y 3 f ij r.a+m 2 M t.İ C a e .m i p
r Glrif : iae A + r i M a L a t l a r IgİMVi

P12.2.13 e8.11..8 f.N8>4.1. 0N05M0..8 4I8 n G dv

$$\mathbf{P12.2.14} \quad \mathbf{n.8ra} \mathbf{brog} \mathbf{48..e.8} \mathbf{81.8.} \quad \mathbf{..8.81.} \quad \mathbf{8..8-8...} \quad \mathbf{..lb} \quad \mathbf{E} \mathbf{R}^{n \times n} \rightarrow \mathbf{R}^n \}_{\mathbf{p}}^{n,n} \quad \mathbf{n}(n)$$

Notes and References for §12.2

fe8... ...8.. 48.48.c..1..41..8. ...d ..1.cb 48....118..R .88

7ae.. 8.1.RNa eP..8.R caN11.8....1 f0075NMatrix Computations and Semiseparable Matrices, Vol. I Linear Systems, 91matLSB latr aliAMt hMAtwigo1C1M1Ax

lv BiaeASM BiawiM iaJ 2vI.pMlaiMefJJ dMatrix Computations and Semiseparable Matrices, Vol. II Eigenvalue and Singular Value Methods, 9imatLSB latr a1AMt1pMAttwgp1CIMA. JEv

$\frac{3t_w}{x S} = \frac{IKX}{x} \cdot \frac{x}{x} \cdot \frac{x}{x} \cdot \frac{x}{x} \cdot \frac{x}{x} \cdot \frac{x}{x} = \frac{IKX}{S}$

m- $T_k = x \neq -T$ c- $x \neq x_0 \neq a$ x=23 a wso= $=x_0$ x m x = -0x a $\neq x_0$
 $=\alpha$ x V x == x= x=N_{alg} x=Alg Applic. 12 d5r dM

IaFAMAaakAaAMaAtpmA tlglplla g2aAiMipllatae,A. tcDiMASMSISU2CptpMIa-pIaMAiaBaagIeAr

f. $\text{ImSAMf} \text{Integ } \text{Equations Operator Theory}$ 8 TdUr dj,v
lr 9leAgCiae f. $\text{ImSAMf NT.x .aiAMt lldMiIg_iae .laAiMALiSgAb1p8e-LNbmi.M$

hvEA+ lgAE 3v9v = x - x & .0aTaime Var ing Systems and Computations, 5gl+ ANA
AC 1.1.1.1

eAC lawtpa. I.3.
 LvAmiaeM*B*iMiaeIv.I f J.B.M. iae LpiS*U*rIM1m*M*iaeAJ,hUI t,s*A*ASiMiSu2
 LtpACif ,laAi*N*d ipllat6SIAM J. Matr x Anal. Applic. 25, 5T515d.
 LvAmiaeM*A*C*M*reEA+ IUPA. I. h*U*tPS*Y*l a. 3v*9*biaE*A**M*AA iaJ E*v*am*2*p*A* JJ.M
 LICAp 3grIM1pm*V*L*I*clAap lig*g*AC ItASiMISSAT ipllat6SIAM J. Matrix Anal.
 Applic. 27, 5N579

II 5a bT. 4 uQnat 28t u6 5a nre Q2228 hC3 (Q1hr)Q= T (In ZS12h qhasU
 PL(xZef)(eonAe DaenZu(Qef).ii J. Comput. Appl. Math. 164, T5N T.Tv
 IV wAggiv9neA,iiauw lySAMh5lqMianiae B.u.yAitBt.IJJ.ii .3 .. DolMaB,hAMAtMC
 1)A3grIMy1 IMLgi larAttAaSAMrn VI.itA)iMiSqA,BiaeHMAgAIAAM. J. Matrix
 Anal. Applic. 31, T.Jr N.v
 9Pniae nui .IJNij.llSI.t. 3))MlbLi.A AylgAtBtal.1 i.1EafliaB,L.MIa.iMAiA.MRA
 hlt1n1AEA=smaMRaAtSIAM J. Matrix Anal. Applic. 31, Id.r II Jv
 JMelt-Itnldf +A.ylet.yi. Ab,l1 y1AMiMayMiaBla.iMAAtAAe
 LwvCMinv.MAet-Buae a ..BSItay .IJJ.5v ..aMle.aplla .1A.iMaynaij.M1At .1m
 3))glai.nla6Engin. Anal. Boundary Elements 27, „J.r.II v
 LAyiaeM.ABimiau iae lv higt.IJJ.W.3 .. r,B EAali)lt1nlaLlgjA1M11AMiMay1aigqt
 LAintA)iMiS1MAtAa1lat.6SIAM J. Matr x Anal. Applic. 28, 7J27II,
 LAyiaeM.ABimiau.I. PvLIa99P liuiae 9r-yi .IJJ.Tv.3 LI)ANif 3grIMn.nfMll A),n-
 LttAit lf nnaAMi.1llatu6SIAM J. Matr x Anal. Applic. 29, NJT NJv
 LAyiaeM.ABimiau.I. 9.P liuiae 9M1 .IJJ.Ti.3 .. VI 3qIMy1 1MAl)ia1la ni.M1
 aABOper. Thor Adv. Applic. 179, NNN,5v
 9.PniLuAmiaeM.ABimiau iae PvLuJNj... iqrIM1y.tl .1A.iMay1aiqAi ltA)
 MiSgAAMnA1Numer. Lin. Alg. Applic. 17, ..5r .T7v
 LXamiaeM.ABimiau.1geAv.I. iae 2uLi.IaeAMi .IJNjJ.a .yA21iAMn-MiaB.yA
 ,LEnirlai,wglaBf LamlM)qAaAt lf Ent-MAr9,80,1he9tFfSIAM J. Matr x Anal.
 Applic. 31, II 7Nf.Ji
 hxi niM.RattlaJNN.3 .. liael.1 Ae 3grIMy1 1MAIC)I.lar i .nA.iMay1aiqAi rF
 LA)iMiS1MAtAa1.1f i ni.M1bu6SIAM J. Matrix Anal. Applic. 32, NI.NITN
 99PRiJNj..a .yAAl)gAb1t lf LliA .1A.iMam1higII-1MAiM1b3grIMrtu6SIAM J.
 Matr x Anal. Applic. 33, 5dd,NJx
 1Ael .1lat .1 Mnelirlai,Relirlaijae.AttAa{ AM. iMAttAa.ligla. Aaet6M+iat A1rAaf
 iighA ita1rIgiM1A)MlaAel M1M1M1M1 .1)MlaAaAa MisB1M1a1MAAtAa.tAAe
 2un..MlaiM1Lu AyiaeMitAeM1lau illA AqJ.Nx... iae Li.SqM1e..lla lf E lirlraig
 hgt LACnLLA)iMiS1MAt hlenirlaigae wnefai, 1 M.u6IT 41, N,r N.Ti
 nvBia wiMAx BiaeASgme 2vnri MlaiMeIJ1 .3a ,Myrlrai, N1Ael a.1la. i
 ni.MnRallAi 1tA)iMiSgM.uBfSIAM J. Matrix Anal. Applic. 27, N7r N.Tv
 nvBia wiM 9uBia Aii). 2un..MlaiMeIJJ.tv ...yrlraig Ln1M1Rhiat1ii.lla 1al
 whB,LA+A1iMiSgM1aAf LA.1tA)iMiS1dRtBfNum. Lin. Alg. 12, dμ .Jv
 lu BiaeASMOuBia Ai).u nuBia wiMAiaea2u n.MlaiMeIJ1 ..yrlraig Ln.1,M1t
 hist1Mii.nla lf i LtiA.M1ani.M1bali ERirlai,,hg1t,LA.nt3)iMiSgAn.y ; AAylnaA
 lf.mENirlai,6Numer. Math. 102 TJ.r TI 7v
 I. 9neAgiajv.lySAMiae ,u .3i iraian .IJJ. „a .yA... MAel .1lf i VI.ntA)iMiSgA
 ni.M1M1 AttAaSAMe hnelirlaiqlM.tuf Ein. Alq Applic. 420, d7 NJNv
 lx BiaeASMOuBiaAi).u nvBiawi.A. Eae2unr .lai.en.IJJ. „a .yAAlaiAMrA1M1)AM2
 RAf.yAM.yrlraigL li jiMnlhia/ 1M.i.1lat 1 h1elirlaiqLAC ltA)i.iSgA1ERirlai,T
 I. i.6 Numer. Math. 104, IJ. .5.v
 hi)AM1aaAMaAy iiM1l1t.MIa.i.AiaB A1rAaiq1AMi.1ltaagl e3e
 IV BiaeASM1xBia wi.A iae 2vnit.Mlai.eRIJJ,iv.3 VI nH.yle1 Al.)I.lar .yALnarIgiM
 Bi,IAfiniLA.ltA)i.iSgA1MR.A1Numer. Math. 99, μ72 N.
 lu BiaeASM1u Bia wiMA2xn.MlaiMeIJJ.li .3a „)g1-1 VI igrl.1y. 1MLtiiA.M1-
 LAi ltA)iMiS1MAtAa1Num. Lin. Alg. 12, 7I. .7.d
 2xI..MlaiMe.9vBia Ai.). iae nvBia wiMAIJJ.1x .E lineAae AladAM1grlMn.yifM
 Ali).I.nar.yA9nrAaeAali)lt1.1lf Lt.iA..1E nirlaE,r),Ir,LA.1tA)iMiS1M1aAt6imer.
 Alg. 39, 5T. .5d
 I99leAqia .lySAMiae BuqyAitBt.IJJ.1x.lyA VI „A.i.1la 1A.yle 1MA.i1 niaVIi2
 t1tA)iMiS1M1aAf ia 3MSn.iMteA1Num. Lin. Alg. Applic. 404, 5J. „I,v
 I9BiaSAM1aluBiae M1hvBia wiMAJd .3 V.,n.A.movAela LAi 1/A)iMiSiAMnAt6
 J. Comput. Appl. Math. 218, d1 „Nv
 nxBiawiMAMBiaSAMr1atvhBiaEllMA1JNj.r tna1AintA)iMiS1MnaAtAli)I.A
 yALBE ilfAaAM1q.1b hMleftVll.RAa6. Comput. Appl. Math. 234, 5T.r f1dJv
 IMental tt1lf .yAlM.yrlrai,Ci.Mn1nraAaiigMISgA1tS1Adae

0CBQar{ En Ond{ at Tre)As ly,7i Br IsAe lheanAe InIsAhas us)el5
Pr c. IEEE Conference on Decision and Contr 1, N75(1.77M

7mAMta AT pAgTpAMiMAMMaDpyl aT MtnlMpylAigAaiig MISgATcaa,IeT are
hOMSAMgaATAMhia f N., r glSgAlaiAMrAhApAyAvl 3grMnpMlaPpMtIpMPaAt
PpmCAAtlgtlM2MCinipMndAtAM J. N mer. Anal. 12, INr5v
3xwlAtA..AMtpaaAMxA f (. 4r a LplMKAAdAadAhlgtaCPigMlcPpMtAttAaSAMr
nipMPASATAM J. Matr. Anal. Applic. 16, N,5r NJ..4
wxlyamlMtpSwlAtA..AMtpAAAMtSA eAMIJ.ur ,pyhAMpIMStyAMMlaPpMt
9PrAaii,hAMISUASATAM J. Matri. Anal. Applic. 21, dJ,r dl,Y
nv .Hx I - I.PM,wHae P.c. A f J.B.M spisgAcineAe AladAegMcpypMpmA
lanpiMPPrAaSMIS,SIAM J. Matr. Anal. Applic. 25, 5dr, J,M
IMspA.iMp6.IJ.7.Mr 3a 9MMMI. BalignpMAttAasIMh 3,rlMpc.6SIAM J. Matr
Anal. Applic. 28, Jr 7Tu
lxMBViTe iEvsApBcaf J.J.cr 91aPACs,ACAcipi hAnIgpTmaPe3grlMPpM
pmAcpMOPrAaiighMISg6SIAM J. Matr. Anal. Applic. 28, 7151755M
JMi aPAPapMle apMmPMISgAfAAAi pBTatn@ S)u5,(5.M

12.3 Kronecker Product Computations

The Kronecker product (KP) has a rich algebra that supports a wide range of fast, practical algorithms. It also provides a bridge between matrix computations and tensor computations. This section is a compendium of its most important properties from that point of view. Recall that we introduced the KP in §136 and identified a few of its properties in §137 and §138. Our discussion of fast transforms in §14 and the 2-dimensional Poisson problem in §484 made heavy use of the operation

himmlb uteTfb .x kESx f T\$eb

Kronecker product computations are structured block matrix computations. Basic properties are given in §136–§138, including

$$\begin{array}{lll} \text{T transpose} & (\mathbf{B} \otimes \mathbf{C})^T & = \mathbf{B}^T \otimes \mathbf{C}^T, \\ \text{Inverse} & (\mathbf{B} \otimes \mathbf{C})^{-1} & = \mathbf{B}^{-1} \otimes \mathbf{C}^{-1}, \\ \text{Product:} & (\mathbf{B} \otimes \mathbf{C})(\mathbf{D} \otimes \mathbf{F}) & = \mathbf{B}\mathbf{D} \otimes \mathbf{C}\mathbf{F}, \\ \text{Associativity:} & \mathbf{B} \otimes (\mathbf{C} \otimes \mathbf{D}) & = (\mathbf{B} \otimes \mathbf{C}) \otimes \mathbf{D} \end{array}$$

Recall that $\mathbf{B} \otimes \mathbf{C} = \mathbf{C} \otimes \mathbf{B}$, but if $\mathbf{B} \in \mathbb{R}^{m_1 \times n_1}$ and $\mathbf{C} \in \mathbb{R}^{m_2 \times n_2}$, then

$$\mathbf{P}(\mathbf{B} \otimes \mathbf{C})^T = \mathbf{C} \otimes \mathbf{B} \quad (1231)$$

where $\mathbf{P} = \mathbf{P}_{m_1 m_2}$ and $\mathbf{Q} = \mathbf{P}_{n_1 n_2}$ are perfect shuffle permutations see §1211. Regarding the Kronecker product of structured matrices, if \mathbf{B} is sparse, then $\mathbf{B} \otimes \mathbf{C}$ has the same sparsity pattern at the block level. If \mathbf{B} and \mathbf{C} are permutation matrices, then $\mathbf{B} \otimes \mathbf{C}$ is also a permutation matrix. Indeed, if \mathbf{p} and \mathbf{q} are permutations of $\mathbf{1}_m$ and $\mathbf{1}_n$ then

$$\mathbf{I}_m(\mathbf{p}, :) \otimes \mathbf{I}_n(\mathbf{q}, :) = \mathbf{I}_{mn}(\mathbf{w}, :), \quad \mathbf{w} = (\mathbf{I}_m \otimes \mathbf{I}_n)(\mathbf{p} \otimes \mathbf{q}) \quad (1232)$$

We also have

$$\begin{aligned} (\text{orthogonal}) \circledast (\text{orthogonal}) &= (\text{orthogonal}), \\ (\text{stochastic}) \circledast (\text{stochastic}) &= (\text{stochastic}), \\ (\text{symmetric}) \circledast (\text{symmetric}) &= (\text{symmetric}). \end{aligned}$$

The inheritance of positive definiteness follows from

$$\begin{aligned} B = \\ \Rightarrow B \otimes C = G_B G_B^T \otimes G_C G_C^T = (G_B \otimes G_C)(G_B \otimes G_C)^T. \end{aligned}$$

$$\left. \begin{aligned} P_B B &= L_B U_B \\ P_C C &= L_C U_C \end{aligned} \right\} \Rightarrow (P_B \otimes P_C)(B \otimes C) = (L_B \otimes L_C)(U_B \otimes U_C),$$

$$\left. \begin{aligned} Q_B^H B Q_B &= T_B \\ Q_C^H C Q_C &= T_C \end{aligned} \right\} \Rightarrow (Q_B \otimes Q_C)^H (B \otimes C) (Q_B \otimes Q_C) = T_B \otimes T_C,$$

$$\Sigma_C$$

$$\det(B \otimes C) = \det(B^n) \cdot \det(C), \quad B \in \mathbb{R}^{m \times m}, C \in \mathbb{R}^{n \times n},$$

$$\text{tr}(B \otimes C) = \text{tr}(B) \cdot \text{tr}(C),$$

$$\|B \otimes C\|_F = \|B\|_F \cdot \|C\|_F$$

$$\|B \otimes C\|_F = \|B\|_F \|C\|_F$$

See Horn and Johnson (TMA) for additional KP facts

hiimhib 7esb 7atfki uTt"eb .x kt Dff b

We can think of the Kronecker product of two matrices $B = (B_{ij})$ and $C = (C_{ij})$ as the systematic layout of all possible products $b_{ij}C_{ij}$, e.g.,

$$\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \otimes \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} = \begin{bmatrix} b_{11}c_{11} & b_{11}c_{12} & b_{12}c_{11} & b_{12}c_{12} \\ b_{11}c_{21} & b_{11}c_{22} & b_{12}c_{21} & b_{12}c_{22} \\ \hline b_{21}c_{11} & b_{21}c_{12} & b_{22}c_{11} & b_{22}c_{12} \\ b_{21}c_{21} & b_{21}c_{22} & b_{22}c_{21} & b_{22}c_{22} \end{bmatrix}.$$

However, the Kronecker product of two block matrices $B = (B_{ij})$ and $C = (C_{ij})$ is not the corresponding layout of all possible block-level Kronecker products $B_{ij} \otimes C_{ij}$.

$$\begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \otimes \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} B_{11}C_{11} & B_{11}C_{12} & B_{12}C_{11} & B_{12}C_{12} \\ B_{11}C_{21} & B_{11}C_{22} & B_{12}C_{21} & B_{12}C_{22} \\ \hline B_{21}C_{11} & B_{21}C_{12} & B_{22}C_{11} & B_{22}C_{12} \\ B_{21}C_{21} & B_{21}C_{22} & B_{22}C_{21} & B_{22}C_{22} \end{bmatrix}.$$

The matrix on the right is an example of the Tracy-Singh product. Formally, if we are given the blockings

$$B = \begin{bmatrix} B_{11} & \dots & B_{1N} \\ \vdots & \ddots & \vdots \\ B_{M_1,1} & \dots & B_{MN} \end{bmatrix} \quad C = \begin{bmatrix} C_{11} & \dots & C_{1N} \\ \vdots & \ddots & \vdots \\ C_{M_2,1} & \dots & C_{MN} \end{bmatrix} \quad (1235)$$

with $B_{ij} \in \mathbb{R}^{m_i \times n_j}$ and $C_{ij} \in \mathbb{R}^{n_i \times m_j}$, then the Tracy-Singh product is an $M_1 \times N_2$ block matrix $B \otimes C$ whose (i,j) block is given by

$$[B \otimes C]_{ij} = \begin{bmatrix} B_{ij} \otimes C_{11} & \dots & B_{ij} \otimes C_{1N} \\ \vdots & \ddots & \vdots \\ B_{ij} \otimes C_{M_2,1} & \dots & B_{ij} \otimes C_{MN} \end{bmatrix}.$$

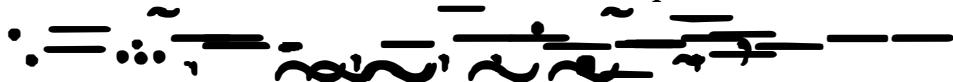
See Tracy and Singh (1972). Given (1235), it can be shown using (1231) that

$$B \otimes C = P(B \otimes Q)Q^T \quad (1236)$$

where

$$P = (\mathbf{M}\mathbf{M}^T)^{-1} \mathbf{m}_2 \mathbf{m}_1^T / (\mathbf{M}^T \mathbf{m}_1 \mathbf{M} \mathbf{m}_2)^{-1} \quad (1237)$$

$$Q = (\mathbf{JNINR} @ \mathbf{n2nl}) (\mathbf{/Nt} @ \mathbf{n1.Nt2}). \quad (1238)$$



There are two submatrices of $B \otimes C$ that are particularly important. The Hadamard product is a pointwise product:

$$B \underset{\text{HAD}}{\otimes} C = B.*C.$$

Thus if $B \in \mathbb{R}^{m \times n}$ and $C \in \mathbb{R}^{n \times n}$, then

$$\left[\begin{array}{cc} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{array} \right] \otimes_{\text{HAD}} \left[\begin{array}{cc} c_{11} & c_{12} \\ c_{21} & c_{22} \\ c_{31} & c_{32} \end{array} \right] = \left[\begin{array}{cc} b_{11}c_{11} & b_{11}c_{12} \\ b_{21}c_{21} & b_{21}c_{22} \\ b_{31}c_{31} & b_{31}c_{32} \end{array} \right].$$

The block analog of this is the Khatri-Rao Product. If $B = (B_{ij})$ and $C = (C_{ij})$ are each $m \times n$ block matrices, then

$$\mathbf{B} \underset{\text{KR}}{\circledast} C = (A_{ij}), \quad A_{ij} = B_{ij} \otimes C_{ij},$$

eg,

$$\left[\begin{array}{cc} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{array} \right] \otimes_{\mathbf{KR}} \left[\begin{array}{cc} C_{11} & C_{12} \\ C_{21} & C_{22} \\ C_{31} & C_{32} \end{array} \right] = \left[\begin{array}{cc} B_{11} \otimes C_{11} & B_{12} \otimes C_{12} \\ B_{21} \otimes C_{21} & B_{22} \otimes C_{22} \\ B_{31} \otimes C_{31} & B_{32} \otimes C_{32} \end{array} \right].$$

A particularly important instance of the Khatri-Rao product is based on column partitions:

$$[b_1 I \cdots I_{b_n}]_{\star_R}^{\star_R} [G_1 \cdots G_n] = [b_1^{\star_R} G_1 \cdots I_{b_n}^{\star_R} G_n].$$

For more details on the Khatri-Rao product, see Smilde, Bro, and Geladi (2004).

1A1 mA TFeA)oA }IA RgD foASf o7[G.:+A

In Kronecker product work, matrices are sometimes regarded as vectors and vectors are sometimes turned into matrices. To be precise about these reshapings, we remind the reader about the `vec` and `reshape` operations defined in §13.7. If $X \in \mathbb{R}^{m \times n}$, then `vec(X)` is an $n \times 1$ vector obtained by "stacking" X 's columns.

$$\text{vec}(\mathbf{X}) \in \begin{bmatrix} \mathbf{X}(:, 1) \\ \vdots \\ \mathbf{X}(:, n) \end{bmatrix}.$$

If $B \in \mathbb{R}^{m_1 \times n_1}$, $C \in \mathbb{R}^{m_2 \times n_2}$, and $X \in \mathbb{R}^{n_1 \times m_2}$, then

$$y = e \mathbf{x} B^T \quad \text{c : } \text{vec}(Y) = (B \circledast C) \cdot \text{vec}(X). \quad (1239)$$

Note that the matrix equation

$$F_1 X G f + \dots + F_p X C; = c \quad (12310)$$

is equivalent to

$$(G_1 \circledast F_1 + \dots + G_p \circledast F_p) \text{vec}(X) = \text{vec}(C). \quad (12311)$$

See Lancaster (1970), Vetter (1975), and also our discussion about block diagonalization in §7.6.3.

The `reshape` operation takes a vector and turns it into a matrix. If $a \in \mathbb{1}^{mn}$, then

$$A = \text{reshape}(a, m, n) \quad \text{c : } \text{vec}(A) = a$$

Thus, if $U \in \mathbb{R}^m$ and $v \in \mathbb{1}^n$, then $\text{reshape}(v \circledast U, m, n) = U^T$

i,uArg FF gwf 1Ag I2H SVd9g!Tg l1 !9+ 9A+!g

There is an important connection between matrix transposition and perfect shuffle permutations. In particular, if $A \in \mathbb{R}^{q \times r}$, then

$$\text{vec}(A^T) = P_{r,q} \text{vec}(A). \quad (12312)$$

This formulation of matrix transposition provides a handy way to reason about large scale, multipass transposition algorithms that are required when $A \in \mathbb{R}^{q \times r}$ is too large to fit in fast memory. In this situation the transposition must proceed in stages and the overall process corresponds to a factorization of $P_{r,q}$. For example, if

$$P_{r,q} = \Gamma_t \cdots \Gamma_1 \quad (12313)$$

where each Γ_k is a "data motion friendly" permutation, then $B = A^T$ can be computed with t passes through the data:

```

a = vec(A)
for k = 1:t
    a = lka
end
B = reshape(a, qr)

```

The idea is to choose a factorization (12313) so that the data motion behind the operation k th pass, i.e., $\Gamma_k a$, is in harmony with the architecture of the underlying memory hierarchy, i.e., blocks that can fit in cache, etc.

As an illustration, suppose we want to assign A^T to B where



We assume that A is stored by column which means that the A_i are not contiguous in memory. To complete the story, suppose each block comfortably fits in cache but that A cannot. Here is a 2 pass factorization of r_{qq}

$$\mathcal{P}_{q,rq} = \Gamma_2 \Gamma_1 = (I_r \otimes \mathcal{P}_{q,q}) (\mathcal{P}_{r,q} \otimes I_q).$$

If $a = r_1 \cdot \text{vec}(A)$, then

$$\text{reshape}(a, q, rq) = [A_1 \ \dots \ A_r].$$

In other words, after the first pass through the data we have computed the block transpose of A . (The A_i are now contiguous in memory.) To complete the overall task, we must transpose each of these blocks. If $b = r_2 a$, then

$$B = \text{reshape}(b, q, rq) = [A \ \dots \ A].$$

See Van Loan (FFT) for more details about perfect shuffle factorizations and multi-pass matrix transposition algorithms.

himmlnb 7eSbtX kSfd Sxb.x ktDf f br-pb

Suppose $A \in \mathbb{R}^{m \times n}$ is given with $m = m_1 m_2$ and $n = n_1 n_2$. For these integer factorizations the nearest Kronecker product (NKP) problem involves minimizing

$$(B, C) = \|IA - B \otimes C\|_F \quad (12.3.14)$$

where $B \in \mathbb{R}^{m_1 \times n_1}$ and $C \in \mathbb{R}^{m_2 \times n_2}$. Van Loan and Pitsianis (1992) show how to solve the NKP problem using the singular value decomposition of a permuted version of A . A small example communicates the main idea. Suppose $m_1 = 3$ and $n_1 = m_2 = n_2 = 2$. By carefully thinking about the sum of squares that define $\|\cdot\|_F$, we see that

$$\begin{aligned} (B, C) &= \left\| \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \\ a_{51} & a_{52} & a_{53} & a_{54} \\ a_{61} & a_{62} & a_{63} & a_{64} \end{bmatrix} - \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{bmatrix} \otimes \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \right\|_F \\ &= \left\| \begin{bmatrix} a_{11} & a_{21} & a_{12} & a_{22} \\ a_{31} & a_{41} & a_{32} & a_{42} \\ a_{51} & a_{61} & a_{52} & a_{62} \\ a_{13} & a_{23} & a_{14} & a_{24} \\ a_{33} & a_{43} & a_{34} & a_{44} \\ a_{53} & a_{63} & a_{54} & a_{64} \end{bmatrix} - \begin{bmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{12} \\ b_{22} \\ b_{32} \end{bmatrix} \begin{bmatrix} c_{11} & c_{21} & c_{12} & c_{22} \end{bmatrix} \right\|_F. \end{aligned}$$

Denote the preceding 6by-4 matrix by $'(A)$ and observe that

$$'(A) = \begin{bmatrix} \text{vec}(A_{11})^T \\ \text{vec}(A_{21})^T \\ \text{vec}(A_{31})^T \\ \text{vec}(A_{12})^T \\ \text{vec}(A_{22})^T \\ \text{vec}(A_{32})^T \end{bmatrix}.$$

It follows that

$$(B, C) = \mathbf{1}'(A) - \text{vec}(B)\text{vec}(C)^T \|_F$$

and so the act of minimizing $\|B\|_F + \|C\|_F$ is equivalent to finding a nearest rank-1 matrix to $'(A)$. This problem has a simple SVD solution. Referring to Theorem 248 if

$$U^T \mathcal{R}(A) V = \Sigma \quad (123.15)$$

is the SVD of $'(A)$, then the optimizing B and C are defined by

$$\text{vec}(B_{\text{opt}}) = . \mathbf{i} U(:, 1), \quad \text{vec}(C_{\text{opt}}) = . \mathbf{i} V(:, 1).$$

The scalings are arbitrary. Indeed, if B_{opt} and C_{opt} solve the NKP problem and $\Pi = Q$ then ΠB_{opt} and $(1/\sqrt{\lambda}) C_{\text{opt}}$ are also optimal.

In general, if

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1,n_1} \\ \vdots & \ddots & \vdots \\ A_{m_1,1} & \cdots & A_{m_1,n_1} \end{bmatrix} \quad (123.16)$$

where each $A_{ij} \in \mathbb{R}^{m_2 \times n_2}$, then $\mathcal{R}(A) \in \mathbb{R}^{m_1 n_1 \times m_2 n_2}$ is defined by

$$'(A) = \begin{bmatrix} \tilde{A}_1 \\ \vdots \\ \tilde{A}_{n_1} \end{bmatrix}, \quad \tilde{A}_j = \begin{bmatrix} \text{vec}(A_{1j})^T \\ \vdots \\ \text{vec}(A_{m_1 j})^T \end{bmatrix}$$

The SVD of $'(A)$ can be "reshaped" into a special SYD-like expansion for A .

Theorem 12.3.1 (Kronecker Product SVD). If $A \in \mathbb{R}^{m_1 n_1 \times m_2 n_2}$ is factored according to (123.16) and

$$'(A) = \mathbf{U} \mathbf{E} \mathbf{V}^T = \bigcup_{k=1}^r \mathbf{U}_k \mathbf{U}_k^T \mathbf{V}_k \mathbf{V}_k^T \quad (123.17)$$

is the SVD of $'(A)$ with $\mathbf{U}_k = U(:, k)$, $\mathbf{V}_k = V(:, k)$, and $\mathbf{U}_k^T = E(k, k)$, then

$$A = \bigcup_{k=1}^r \mathbf{U}_k \mathbf{U}_k^T \mathbf{V}_k \mathbf{V}_k^T \quad (123.18)$$

where $U_k = \text{reshape}(u_k, m_1, n_1)$ and $V_k = \text{reshape}(v_k, m_2, n_2)$.

Pr 6 In light of (123 18), we must show that

$$A_{ij} \leq \sum_{k=1}^r k U_{kj} V_{ik}$$

But this follows immediately from (123 17) which says that

$$\text{vec}(A_{ij})^T \leq \sum_{k=1}^r k U_{kj} V_{ik}$$

for all i and j . \square

The integer r in the theorem is the Kronecker product rank of A given the blocking (123 16). Note that if $f \leq r$, then

$$A_{\tilde{r}} = \sum_{k=1}^{\tilde{r}} \sigma_k U_k \otimes V_k \quad (123 19)$$

is the closest matrix to A (in the Frobenius norm) that is the sum of f Kronecker products. If A is large and sparse and f is small, then the Lanczos SVD iteration can effectively be used to compute the required singular values and vectors of (A) . See §104.

tmmicb .kle Qx1tsb kt.b .x kf PSeb

If A is structured, then it is sometimes the case that the B and C matrices that solve the NKP problems are similarly structured. For example, if A is symmetric and positive definite, then the same can be said of B_{opt} and C_{opt} (if properly normalized). Likewise, if A is nonnegative, then the optimal B and C can be chosen to be nonnegative. These and other structured NKP problems are discussed in Van Loan and Pitsianis (1992).

We mention that a problem like

$$\min_{\substack{\text{3D Toeplitz}}} \|A - B \otimes C\|_F, \quad B \in \mathbb{R}^{m \times m}, C \in \mathbb{R}^{n \times n},$$

turns into a constrained nearest rank-1 problem of the form

$$\min_{\substack{[3D] \\ I \in \mathbb{R}^{m \times n}}} \|A - b\|_F$$

where the nullspaces of F^T and G^T define the vector space of m -by- m and n -by- n Toeplitz matrices respectively. This problem can be solved by computing QR factorizations of F and G followed by a reduced-dimension SVD.

$$2\min_{\mathbf{A} \in \mathbb{R}^{m \times m}} \| \mathbf{A} - \mathbf{A} \mathbf{A}^T \|_F = \alpha - \alpha \mu + \mu$$

Suppose $\mathbf{A} \in \mathbb{R}^{m^2 \times m^2}$ and that we want to find $\mathbf{X} \in \mathbb{R}^{m \times m}$ so that

$$\| \mathbf{A} - \mathbf{X} \otimes \mathbf{X}^T \|_F$$

is minimized. Proceeding as we did with the NKP problem, we can reshape this into a nearest symmetric rank-1 problem

$$\| \mathbf{A} - \mathbf{X} \otimes \mathbf{X}^T \|_F = \| \text{vec}(\mathbf{A}) - \text{vec}(\mathbf{X}) \cdot \text{vec}(\mathbf{X})^T \|_F. \quad (1232)$$

It turns out that the solution \mathbf{X}_{opt} is a reshaping of an eigenvector associated with the symmetric part of (\mathbf{A}) .

Lemma 12.3.2. Suppose $M \in \mathbb{R}^{n \times n}$ and that $\mathbf{Q}^T \mathbf{T} \mathbf{Q} = \text{diag}(\alpha_1, \dots, \alpha_n)$ is a Schur decomposition of $\mathbf{T} = (M + e) \mathbf{T}/2$. If

$$|\alpha_k| = \max\{|\alpha_1|, \dots, |\alpha_n|\}$$

then the solution to the problem

$$\begin{aligned} \min_{\substack{\mathbf{Z} = \mathbf{Z}^T \\ \text{rank}(\mathbf{Z}) = 0}} & \| M - \mathbf{Z} \|_F \\ & \end{aligned}$$

is given by $\mathbf{Z}_{\text{opt}} = \mathbf{Q} \mathbf{Q}_k^T \mathbf{Q}$, where $\mathbf{Q}_k = \mathbf{Q}(:, k)$.

Proof. See Pl 2311. \square

timmmib .k3Df 1 "bf esbk\\$ t. Sef baB1sdslB1al

Suppose $\mathbf{A} \in \mathbb{R}^{n \times n}$, $n = m^2$ and that we wish to find $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{m \times m}$ so that

$$\| (\mathbf{X}, \mathbf{Y}) - \mathbf{I} \mathbf{A} - (\mathbf{X} \odot \mathbf{Y} - \mathbf{Y} \odot \mathbf{X}) \|_F$$

is minimized. It can be shown that

$$\| (\mathbf{X}, \mathbf{Y}) - \mathbf{I} \mathbf{A} - (\mathbf{X} \odot \mathbf{Y} - \mathbf{Y} \odot \mathbf{X}) \|_F = \| \text{vec}(\mathbf{X}) \cdot \text{vec}(\mathbf{Y})^T - \text{vec}(\mathbf{Y}) \cdot \text{vec}(\mathbf{X})^T \|_F. \quad (1232)$$

The optimizing \mathbf{X} and \mathbf{Y} can be determined by exploiting the following lemma

Lemma 12.3.3. Suppose $M \in \mathbb{R}^{n \times n}$ with skewsymmetric part $\mathbf{S} = (M - M^T)/2$. If

$$\mathbf{S}[u \ 1v] = [u \ 1v] \begin{bmatrix} \mathbf{Z} & \mathbf{J} \end{bmatrix} \quad u, v \in \mathbb{R}^n,$$

with $\mu = \|\mathbf{S}\|_F$, $\|u\|_2 = \|v\|_2 = 1$, and $u^T v = 0$, then $\mathbf{Z}_{\text{opt}} = \mu(u^T - v^T)$ minimizes $\| M - Z \|_F$ over all rank-2 skewsymmetric matrices $Z \in \mathbb{R}^{n \times n}$.

Proof. See Pl 2312. \square

2hltQyA L1bA ..1 b+A 2AN|A qN 3mBoAt= .dk oAAa7IN i+A

The Kronecker product of three or more matrices results in a matrix that has a recursive block structure. For example,

$$B \odot C \odot D = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \otimes \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{bmatrix} \otimes \begin{bmatrix} d_{11} & d_{12} & d_{13} \\ d_{21} & d_{22} & d_{23} \\ d_{31} & d_{32} & d_{33} \end{bmatrix}$$

is a 2 by 2 block matrix whose entries are 4 by 4 block matrices whose entries are 3 by 3 matrices.

A Kronecker product can be regarded as a data-sparse representation. If $A = B_1 \odot B_2$ and each B -matrix is m by n , then $2mn$ numbers are used to encode a matrix that has Mn^2 entries. The data sparsity is more dramatic for multiple Kronecker products. If $A = B_1 \odot \dots \odot B_p$ and $B_i \in \mathbb{R}^{m_i \times n_i}$, then $p m_i n_i$ numbers fully describe A , a matrix with $M_p N_p$ entries.

Order of operation can be important when a multiple Kronecker product is involved and the participating matrices vary in dimension. Suppose $B_i \in \mathbb{R}^{m_i \times n_i}$,

$$y = (B_1 \otimes \dots \otimes B_p)x \quad x \in \mathbb{R}^{N_p}$$

$$\text{reshape}(y, M_p/M_i, M_i) = (B_{i+1} \otimes \dots \otimes B_p) \cdot \text{reshape}(x, N_p/N_i, N_i) \cdot (B_1 \otimes \dots \otimes B_i)^T.$$

$$\text{P12.3.8 . N1.8 } \in \mathbb{R}^{mn \times mn}, \quad , \quad B \in \mathbb{R}^{n \times m}, \quad n \cdot n \cdot 3n \rightarrow 5^n X \in \mathbb{R}^{n \times n}, \quad n \cdot 5 \cdot 5$$

$$t_0(X) \cup \prod_{i=1}^m -B_i X \in \mathbb{F}$$

2.) a2.2etK Xin02ek Z E R^{m n × m n}, , C E R^{n × n}, n = n. n. 3n → X.5' X E R^{m × m}, n = 5,5
 tc(X) U ||A- X®C ||F

$\frac{1}{2} \mathbf{b}^T \mathbf{T} \mathbf{b} \leq \lambda \mathbf{x}^T \mathbf{A} \mathbf{x}$

P12.3.9 f.. 2.8.8.8.. or.8 84 d.g.4. 282.8 ..2.b = I_n[®]/ , + / , ®I_n]CAEA
 „ T_{→A})Ax 1 (4.8.7).

P12.3.10 $\text{re} \geq R^{mn \times mn}$, $\rightarrow^n((3^3, 5(3,3, n), C \in R^{n \times n}, (n, \rightarrow^n, 5(3^3, 5(3,3, n), \bullet$

P12.3.11 3.8.8 18... 1232 H̄ 'Ö p"

M axT1 M - 2xTx a²

$$+^{\prime \prime}(\bar{U}T_1\left(M_{+}M_{-}\right) /2$$

P12.3.12 3.8.8 18... 1233 H̄ 'Öf"

$$\mathbf{M}^T (\mathbf{x}\mathbf{T} - \mathbf{y}\mathbf{T})\mathbf{I} > \mathbf{M}\mathbf{I} + 2\mathbf{k}\mathbf{1}\mathbf{1}^T\mathbf{y} - 2\mathbf{x}^T\mathbf{y}^2 - 4x^T S y$$

$$+ m\Delta M(M_1^+ M_1^-)/20 - (e^+ + e^-)(p_1^+ + p_1^-)2s.$$

P12.3.13 **fig.8.14****b** $S \in \mathbf{R}^{n \times n}$, 5, "symmetric vector operation ltr gUt eAQaAe St

$$\mathbf{s}^\top \mathbf{U} \begin{bmatrix} & & \mathbf{s}_{13} \\ \vdots & \vdots & \mathbf{s}_{23} \\ \mathbf{s}_{31} & \mathbf{s}_{32} & \mathbf{s}_{33} \end{bmatrix} = \mathbf{svec}(S)^\top \begin{bmatrix} \mathbf{s}_{11} & \mathbf{Y} & \mathbf{s}_{21} & \sqrt{2}\mathbf{s}_{31} & \mathbf{s}_{22} & \sqrt{2}\mathbf{s}_{32} & \mathbf{s}_{33} \end{bmatrix}^T.$$

JMttCCApXKE R^{n × n}, , ,(135,(,(B C E R^{n × n}, (, " symmetr c Kr necker pr duct lteAQaASd

$$(\text{Bs } M) \cdot \text{svec}(X) = \text{svec}((\mathbf{1} \otimes \text{BT}) + (\mathbf{B} \otimes \text{CT})).$$

P12.3.14 1.8 bi-alter ate pr duct lteAQsAe St

$$B \circledast_{B^{\perp}} C = CB^{\perp}C + C B^{\perp}B).$$

'oB=I, C U! ThCA) (=II= n(AX+XI T_i (200). 3." (C.M M("S + . ("R- " +"

P12.3.15 N1.8 $\in \mathbb{R}^q$, , $g_i \in \mathcal{R}$ i $t = i G_{1:m}, -M\tau M, 1 \in \mathcal{N}$ $M, 12, \tau \in ()P$, (, ,

$$P \left(f \otimes \begin{bmatrix} g_1 \\ \vdots \\ g_m \end{bmatrix} \right) = \begin{bmatrix} f \otimes g_1 \\ \vdots \\ f \otimes g_m \end{bmatrix}.$$

$\in \tau$ „ $\neg(N, \{123\})$ “ +” B0 C 2EA2ADIE(

Notes and References for §12.3

1.8 ...8. 8~~2~~.8 or.8 84d~~8~~.42 . 4...1 rob.121... .8.28..8 4.00~~8~~.82 8. -8..427
1.1.4...8. 1.

P 18.8.8 fe3. 881R. .n 78...8(1983). a) +1(x-(k-1) -(k f(9k+ n
 Lin. Mult. Alg. 14, 132)

a(" , "T "((r . , (" -) "T "+ " + "((." T (" "O

- 1a Gslety, E. A. *lineQen Pntls lsse(2a) ra Uann Iemsn* (Ex SIAM J. Numer. Anal. 5, 11v, 52
 94a4wMA.ANT d2 .5MlaAaBAMlesapt aeI opM 1h o,as,st lasttpA. a2AlMt tEEE ns.
 Cir uits Syst. 25 TTI vN2
 34 Momod N. dM Kr necker Pr ducts and Matrix Calculus with Applications, 9ggdM+le . Am1am
 AtPAmrgoaa
 14 EoiN.dN .5MlaAaBAMlesapt es2s AggrASMIcns. Comput. c-90 NAM2
 .2B 2AaeAMlaesu2sAoga(d.12.a2ABAmAM.spopPlap.Pla2ABAaSA.opl. oae5M
 aAaBAMlel apt8r Ai 1A.6n. Multilin. Alg. ITT r ldd4
 .2B 2AaeAMtlaeae sur sA0M.dn BAroaeBAAzSA.oplMt Iop.RaAt PplhA ItAt Pa
 9IS loat oae II, plicM lopPlapmatuadian J. of Stat. 7, 7.vdN2
 A2Boa,loa dJJn2 amAr SnCSRpH MlaAaBAMlesapt J. Comput. and Appl. Math. 129,
 d1NJJ2
- 1af AMAaAaAaA MaApiaPlst5h, RAI)A.opPlaPaagl eAr
- A2r 2 iae 254 PpMn. Tr Gener lized Inverse of Matrices and Applications, l2a a1gAtbae
 slat. 2A. IMB2
- E 4sAt oae r 4s 1a2 cN.Tn4 .3 2A+ I opM 1hMlelpoae f pt3SSigaopl1lAphoMpnplAe
 IopMnaAtistica Neerlandica 26 N5v.(Tu
- h23r Aronees2IPpMnd.n2 .5MlaAaBAMlesaptua1poVtOpMnaAtus1raogMlaAtt1lar
 3SSnapPlatdM Review 91, d7vN52
- 94sASAMtP.I -2oar c. .5n2 sl.A „p2rlao. I opMRaMtpMI apStespMla5MlaAaBAM
 hMlel ah,pPlaplat Austr I. J. Combin. 7, I (5/II,u
- a4 EAosaAt oae9u sASAMtn a2AMla5MlaAaBAMlel ap. Combin. Theor., Ser. A
 66 NIVN5;
- ,4BoeAaSAMr2Asuote cN.7n2sA.PA alRho .1ar.tt SIAM Review 98 ITd2
 a2.liiAMptdJJ2 Numerical Methods for Bif urations of Dynamical Equilibria, sf 3I hsSp aopPlat.
 hmieA,Smidc2
- 32s.1,eA. r 8Mlneh2AgoelJJ,4 Multiway Analysis, l2a aPgAtAmRa2AaSpM2
 lMScoaBrMllap25h alaaAap1lal stgiAtpAM,pSSaopl1latAAe
- huoaapAMNTJn.9bSgnaslgI pP1lf,1aAoVpMPhl opnlatutAM Review 12 „v.772
 a49BAppAMr4.BAaplMtpMI apsMAtgl1lal ,1aAoVpMPhl opnlatuton. Alg. Applic.
 10 NdNvN ddu
- f tti AtlanopAenpm A2A1AapS,A.Aapopl1f 5h l)AMop1lalMAtltAtPAt
 .2A 4baeMAbae.u5caAcN.Tn1.5MlaAaBAMpAaAl)spA. f.SgA.Aap1lAeae.AaA.ogn.Ae
 sSAapMta1ssoc. Comput. Mach. 17, I 7JF7d4
- B 2AMATm2 samAMM.12.91aRAap1lSspA.I can)s,opRlf aAtl. hlesapt + PpSSAM
 aopPlat1sgpneAat1lac2SSMbl1op1lalath. Comput. 27, ...v7J,2
- A2Aw1IMNTn2.91aRAap1lSspA.I oal)I gopPlfaAatl. hlesapt ACM' ns. Math Sof w.
 5 N5Nd14
- h22 wh1ea22 EtBtAa(.7n2.91aPApAaplMehoo,AgcanSI,ponla aAtMhlesapt.6
 ACM' ns. Math Sof w. 22, Nv152
- h2uwlnbaea2r 2 EtBMA72.3rlMnp2T.58 a2A5T3a,3h3A5LSotAelSMot pmA
 Al.SI pAMba1S1gop1lf aAtlMhlesapt6CM' ns. Math Sof w. 22, I,v1.2
- a.2 spAAS..T2Matrix Calculus and Kr necker Pr duct with Applications and C++ Pr gr ms,
 alM,esanAaph1S,nt2nafarcsLAu
- I2.lmpoaAd JJ12.r AoPaAos5MlaAaBAMlesaSAMopPlatin. Alg. Applic. 417, 5Tv5N2
- a2A5h ittanopAe-1ppmi p .o)IM1pf.p gnaApMoat,M.tAABoa,loa c.a m + Ag8
- A.. .I car..ur 2Imata. oae r uau l2a1M2.I.s,pGAA3rAS.oae haoggAhlMrMoCM
 .lar.6J. Super comput. 5 N1INT2
- .u.Miaopl uAlaaA. oae r al gn.1A.d. I n2. r AaI.tPjA 3rl. Ppmtaep2A lgM pmA
 aAat1Mlel apSEE' ns. Signal Pr cess. 40, I,1 Nv5J2
- 2 .MoaopoAl14AMer 2,1mAAMn.I n4amAAtlMhlel apII op2A.op1adMlMrMo.nar
 ,carI orAM ..atoae,p2AMP Esh,SAMop1l6EEE SP Magazine, January, , v,d4
- lMoental tt11 pmM1, M 5h o)SMlbnoPla2oicM1Apt1l1op1lalAAT

- lr 5a Uta dt 4P B-(2)Q_s(n). 4eJJLMR68A7mMp. -lMpt.ytl NIM r.76Bp Linear Al-
gebr for Large Scale and Real Time Applications, ncsnllaAa ise .c.c .l,IS (tSaPS-1.2tl
Nr.1mA76Mp&StLst.u7S n(7M&-S
.PxPeSdtS oP.P ,M.AyS ApS f PtP5At 5tp (s((o))=4ftM.oAty xa7m8A76MM. 26)u&Stl 27A7ma5
7m.aIEEE T ns. Signal Process. 45. 7T5r .Td
lcE c2+ iB iae wBiaBAAe((.). 4tpaMl NIMSr.7fAama eJJLMRm8A7mMp&b7tllAx6YtlaSL
IEEE T ns. Signal Pr cess. 44. 57.J v
9v 5i. iae.c.c 2irt (s((g)). 4-lMtt.ytl NIMSr.7AtS 2Pt eJJLMR68A7mMp&a8A)t ota7MlA6MpSL
Lin. Alg. Applic. 284. NTN.Iv
f c2irt iae ch.QAiMt s((g). 4ota7Ml6p)D8Ata tt)lAstS .h 2JA76A1b&Ad6Ap7f1rlSLSIAM
J. Sci. Comput. 1g N75Ndl v
9v5i.. ise .c.c 2irt (n11). 4J768A1 -lMpt.ytl NIMSr.7eJJLMRm8A7mMp&f1M.y .MtJ1m75
4A7l6.tz SLSIAM J. Matr Anal. Applic. 22. ...NTIV
.c.c 2irtUlc5c2ruiae,c AMVlArA. 4-lMpt.ytl NIMSr.7eJJLMRm8A7mMp&D8A)t ota7Ml
76Mp .67tR6It fMNtSAhMtS676M SLSIAM J. Matr Anal. Applic. 25. dI.r dNx
3c2ciari1,A ise av9nsxA+iMn1.). 4e -lMtt.ytl NIMSr.7eJJLMRm8A7tNlt.MpS676Mp&l
2e,aSL Num. Lin. Alg. 11. TI5TIV
9catMxttmanBli((.)c 4-lMpt.ytl reNIMSr.7eJJLMR68A7mMp&a 2M8t rp.7mMte t1A7ASlm.taSL**
Lin. Alg. Applic. 97g. ,I5r ,5Tx
,c hAMVlArA 4-lMtt.ytl NIMSr.7eJJLMR68A7mMp&a u8A)t ota7MlA7mMp&7uep76reb t.7mIt
fMrpSAhMpS676Mp&Num. Lin. Alg. 19., IIv
av .iaOSI tamwc25mlMltBniae 9c9atMxttma1Bli,)1 42mtlAl+u6.A1-lMpt.ytl nt paMls
NIMSr.7eJJLMRm8A76Mp&w. Numer. Math. N5NN N.7v
BcUtyAitBtu,tAU AeAixie 9catMxtthalB(n11.)c 4tpaMlM Jtl7mtM.lr1761tIt1 MtJ1m7ApS
ot1A7tS 8A7lm.taSA. Lin. Alg. Applic. 412. NINv
9MatBliAase Aci,Ilxtlt (n10), 42.A1A1t 1MS1mt)M.otA1 ClAJua,amI) -lMpt.ytl 4r17mJms
A76M&L Rep. of the 24th International Conference on Machine Learning AlMi.,Urtu
9vAtB inass). 4-lMpt.ytl ClAJuaS&AGr ph Algor thms in the Language of Linear Algebr , f c
5ASsAm 9vnUSA Ml S2De1 Nr.16. A7mMp&S1Ast1J.6AS&eSs7o n1-
- xMlAa1AjaumMEN A1)Mlm7u85l bmtptAl aha7t8pS1tw7 a.NAlta JlM.1t8aS attr
- 2P 2rl.MM((.). 4268Jbt e1)Mlm7u8aAMr7 -lMtt.ytlNIMSr+76p7ut ftmptAlMS1SALin. Alg**
Applic. 297-8. 5.N5.du
- Ecac ItAx&Acaf ,xsluiae.c .itm1ltns((o). 4u8JLMtS NAlA1tBo 1t7uMS 5l IAl)t fttw7**
22rAlta NIM.1t8a Dp1M16P lMtt.ytl NIMSr.76LJ. Comput. Appl. Math. 78, 75r .Td
- 3cwiMM,Ja(g). 4x./mtt7 2Mbr7mMp&t7lAmptSfttw7 2rAlta NIM.bt8a .67. -lMpt.ytl NIMs5**
r.7 27lr.7rlt SLSIAM J. Matrix Anal. Applic. 19, Nr. N7Ju
- hcw,amyl,&ae acdEitiM(n11-1) 4f1M.y2lo 5l -lMtt.ytl 27lr.7rltS otJltatp7A7 mMp&dnin.**
Alg. Applic. 986. dG NJ.M
- 3cac wleisa3Biae 3c,1xlSlMtB(1117) 4.ut NdM.lra7taNlM.1t8 5l ll7uM)MpAb-lMpt.tl**
NIMSN.7a\$AM J. Sci. Comput. 25, N,d N75v
- Ac ecmdMx lse AcBria nlis (n11.). 42.m tS -lMtt.ytl NIMSr.72ha7t8aSASIAM J. Matr**
Anal. Applic. 29, N,d N.dv

12.4 Tensor Unfoldings and Contractions

An order-d tensor $A \in \mathbb{R}^{n_1 \times \dots \times n_d}$ is a real d-dimensional array $A[l:n_1, \dots, l:n_d]$ where the index range in the kth mode is from 1 to n_k . Low-order examples include scalars (order-0), vectors (order-1), and matrices (order-2). Order-3 tensors can be visualized as "Rubik cubes of data," although the dimensions do not have to be equal along each mode. For example, $A \in \mathbb{R}^{m \times n \times 3}$ might house the red, green, and blue pixel data for an m by n image, a "stacking" of three matrices. In many applications, a tensor is used to capture what a multivariate function looks like on a lattice of points, e.g., $A(i, j, k, \ell) = f(\mathbf{v}_i, \mathbf{x}_j, \mathbf{y}_k, \mathbf{z}_\ell)$. The function f could be the solution to a complicated partial differential equation or a general mapping from some high-dimensional space of input \mathbf{v} to a measurement that is acquired experimentally.

Because of their higher dimension, tensors are harder to reason about than matrices. Notation, which is always important, is critically important in tensor computations where vectors of subscripts and deeply nested summations are the rule. In this section we examine some basic tensor operations and develop a handy, matrix type of notation that can be used to describe them. Kronecker products are central.

Excellent background references include De Lathauwer (1997), Smilde, Bro, and Geladi (2004), and Kolda and Bader (2009).

himself R_t IPts t" etttb .kff x tf f Ttcb eb 2e SdT.Ttx Kbkkf b

To understand a tensor is

$$= \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \\ \hline & & & \\ & & & \\ & & & \\ & & & \\ \hline & & & \\ & & & \\ & & & \\ & & & \\ \hline & & & \end{bmatrix}.$$

This is an example of a *tensor contraction*. Tensor contractions are essentially re-

If $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ and $\mathbf{i} = (i_1, \dots, i_d)$ with $1 \leq i_k \leq n_k$ for $k = 1:d$, then

$$\mathcal{A}(\mathbf{i}) \equiv \mathcal{A}(i_1, \dots, i_k).$$

$$B = \mathcal{A}(L_1:R_1, \dots, L_d:R_d).$$

$$\Rightarrow \mathcal{B}(i_2) = \mathcal{A}(1, i_2, 2, 4),$$

$$\sum_{\mathbf{i}=1}^{\mathbf{n}} \equiv \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d}.$$

$$\|\mathcal{A}\|_F = \sqrt{\quad}$$

1) mA T oA)oA Sf o7 mA kA Tb := .7A

As with matrices, the $\text{vec}(\cdot)$ operator turns tensors into column vectors, eg,

$$\mathbf{A} \in \mathbb{R}^{2 \times 3 \times 2} \implies \text{vec}(\mathbf{A}) = \begin{bmatrix} \mathbf{A}(:, 1, 1) \\ \hline \mathbf{A}(:, 2, 1) \\ \hline \mathbf{A}(:, 3, 1) \\ \hline \mathbf{A}(:, 1, 2) \\ \hline \mathbf{A}(:, 2, 2) \\ \hline \mathbf{A}(:, 3, 2) \end{bmatrix} = \begin{bmatrix} a_{111} \\ a_{211} \\ a_{121} \\ a_{221} \\ a_{131} \\ a_{231} \\ a_{112} \\ a_{212} \\ a_{122} \\ a_{222} \\ a_{132} \\ a_{232} \end{bmatrix}.$$

Formally, if $\mathcal{A} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, then

$$\text{vec}(\mathcal{A}) = \begin{bmatrix} \text{vec}(\mathcal{A}^{(1)}) \\ \vdots \\ \text{vec}(\mathcal{A}^{(n_d)}) \end{bmatrix} \quad (1242)$$

where $\mathcal{A}^{(k)} \in \mathbb{R}^{n_1 \times \dots \times n_{d-1}}$ is defined by

$$\mathcal{A}^{(k)}(i_1, \dots, i_{d-1}) = \mathcal{A}(i_1, \dots, i_{d-1}, k) \quad (1243)$$

for $k=1:n_d$. Alternatively, if we define the integer-valued function cd by

$$\text{cd}(\mathbf{i}_{\text{n}}) = i_1 + (i_2 - 1)n_1 + (i_3 - 1)n_1n_2 + \dots + (i_d - 1)n_1 \dots n_{d-1}, \quad (1244)$$

then $a = \text{vec}(\mathcal{A})$ is specified by

$$a(\text{cd}(\mathbf{i}_{\text{n}})) = \mathcal{A}(\mathbf{i}), \quad 1 \leq \mathbf{i} \leq \mathbf{n}. \quad (1245)$$

= $\mathbf{n} \cdot \mathbf{u} \cdot \mathbf{p}^T, \dots, \tilde{\mathbf{p}}, \dots, \mathbf{q}, \mathbf{q}^T, \dots, \mathbf{7}$

If $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, then there are $6 = 3!$ possible transpositions identified by the notation $\mathbf{A}^{<[i \ j \ k]}$ where $[i \ j \ k]$ is a permutation of $[1 \ 2 \ 3]$:

$$\mathbf{l} = \left\{ \begin{array}{l} \mathbf{A}^{<[1 \ 2 \ 3]} \\ \mathbf{A}^{<[1 \ 3 \ 2]} \\ \mathbf{A}^{<[2 \ 1 \ 3]} \\ \mathbf{A}^{<[2 \ 3 \ 1]} \\ \mathbf{A}^{<[3 \ 1 \ 2]} \\ \mathbf{A}^{<[3 \ 2 \ 1]} \end{array} \right\} \implies \left\{ \begin{array}{l} \mathbf{bij} \\ \mathbf{blk} \\ \mathbf{bjik} \\ \mathbf{bki} \\ \mathbf{bki} \\ \mathbf{bji} \end{array} \right\} = a_{ijk}.$$

These transpositions can be defined using the perfect `shuf` e and the `vec` operator. For example, if $\mathbf{l} = \text{A}\langle \mathbf{P}[\mathbf{l}] \rangle$, then $\text{vec}(\mathbf{l}) = (\mathbf{P}_{\mathbf{n}}\mathbf{l}, \mathbf{n}_2^{\otimes} \mathbf{l}_{\mathbf{n}_3} \dots \mathbf{P}_{\mathbf{n}_d}\mathbf{l}_{\mathbf{n}_d})^T \text{vec}(\mathbf{A})$.

In general, if $\mathbf{A} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ and $\mathbf{p} = [1, \dots, p_d]$ is a permutation of the index vector \mathbf{l} , then $\mathbf{A}\langle \mathbf{P} \rangle \in \mathbb{R}^{n_d \times \dots \times n_1}$ is the \mathbf{p} -transpose of \mathbf{A} defined by

$$\mathbf{A}\langle \mathbf{P} \rangle(j_{p_1}, \dots, j_{p_d}) = \mathbf{A}(j_1, \dots, j_d) \quad 1 \leq j_k \leq n_k, \quad k = 1:d,$$

i.e.,

$$\mathbf{A}\langle \mathbf{P} \rangle(\mathbf{j}(\mathbf{p})) = \mathbf{A}(\mathbf{j}), \quad 1 : \mathbf{j} : : \mathbf{n}.$$

For additional tensor transposition discussion see Ragnarsson and Van Loan (2012).

$$\text{iA}(\mathbf{A}\mathbf{A}\mathbf{A}) - \mathbf{T} \cdot \text{cAql}\} \mathbf{2A} \quad \mathbf{A1} \times \mathbf{glmlf} + \mathbf{A}$$

Recall that a tensor unf lding is a matrix whose entries come from the tensor. Particularly important are the modal unfoldings. If $\mathbf{A} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ and $\mathbf{N} = \mathbf{n}_1 \dots \mathbf{n}_d$ then its mode- k unf lding is an n_k by (N/n_k) matrix whose columns are the mode- k fibers. To illustrate, here are the three modal unfoldings for $\mathbf{A} \in \mathbb{R}^{4 \times 3 \times 2}$:

$$\mathbf{A}(\mathbf{l}) = \begin{bmatrix} a_{111} & a_{121} & a_{131} & a_{112} & a_{122} & a_{132} \\ a_{211} & a_{221} & a_{231} & a_{212} & a_{222} & a_{232} \\ a_{311} & a_{321} & a_{331} & a_{312} & a_{322} & a_{332} \\ a_{411} & a_{421} & a_{431} & a_{412} & a_{422} & a_{432} \end{bmatrix}.$$

$$\mathbf{A}\langle \mathbf{Q} \rangle = \begin{bmatrix} a_{111} & a_{211} & a_{311} & a_{411} & a_{112} & a_{212} & a_{312} & a_{412} \\ a_{121} & a_{221} & a_{321} & a_{421} & a_{122} & a_{222} & a_{322} & a_{422} \\ a_{131} & a_{231} & a_{331} & a_{431} & a_{132} & a_{232} & a_{332} & a_{432} \end{bmatrix},$$

$$\mathbf{A}\langle \mathbf{QJ} \rangle = \begin{bmatrix} a_{111} & a_{211} & a_{311} & a_{411} & a_{121} & a_{221} & a_{321} & a_{421} & a_{131} & a_{231} & a_{331} & a_{431} \\ a_{112} & a_{212} & a_{312} & a_{412} & a_{122} & a_{222} & a_{322} & a_{422} & a_{132} & a_{232} & a_{332} & a_{432} \end{bmatrix}.$$

We choose to order the fibers left to right according to the "vec" ordering. To be precise, if $\mathbf{A} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, then its mode- k unf lding $\mathbf{A}(\mathbf{k})$ is completely defined by

$$\mathbf{A}(\mathbf{k})(\mathbf{i}_k, \text{cd}(\mathbf{k}, \mathbf{i})) = \mathbf{A}(\mathbf{i}) \quad (1246)$$

where $\mathbf{l}_k = [\mathbf{i}_1, \dots, \mathbf{i}_k, \mathbf{l}_{k+1}, \dots, \mathbf{l}_d]$ and $\mathbf{i} \cdot \mathbf{k} = [n_1, \dots, n_k, t, n_{k+1}, \dots, n_d]$. The rows of $\mathbf{A}(\mathbf{k})$ are associated with subtensors of \mathbf{A} . In particular, we can identify $\mathbf{A}(\mathbf{k})(\mathbf{q}, \cdot)$ with the order-($d-1$) tensor $\mathbf{A}(\mathbf{q})$ defined by $\mathbf{A}(\mathbf{q})(\mathbf{h}) = \mathbf{A}(\mathbf{k})(\mathbf{q} \text{cd}(\mathbf{k}), \mathbf{i} \cdot \mathbf{k})$.

himsmnb hk x\\$pos t\\$xtPb Rt eft1t "eb

In general, an unf lding for $\mathbf{A} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ is defined by choosing a set of row modes and a set of column modes. For example, if $\mathbf{A} \in \mathbb{R}^{2 \times 3 \times 2 \times 2 \times 3}$, $r = 13$ and $c = 45$ then

$$A_{r \times c} = \begin{bmatrix} (1) & (2) & (12) & (23) & (13) & (23) \\ an_{1,11} & an_{1,21} & an_{1,12} & an_{1,22} & an_{1,13} & an_{1,23} \\ a_{21,11} & a_{21,21} & a_{21,12} & a_{21,22} & a_{21,13} & a_{21,23} \\ a_{12,11} & a_{12,21} & a_{12,12} & a_{12,22} & a_{12,13} & a_{12,23} \\ a_{22,11} & a_{22,21} & a_{22,12} & a_{22,22} & a_{22,13} & a_{22,23} \\ a_{13,11} & a_{13,21} & a_{13,12} & a_{13,22} & a_{13,13} & a_{13,23} \\ a_{23,11} & a_{23,21} & a_{23,12} & a_{23,22} & a_{23,13} & a_{23,23} \\ an_{2,11} & an_{2,21} & au_{2,12} & au_{2,22} & au_{2,13} & au_{2,23} \\ a_{21,21} & a_{21,22} & a_{21,21} & a_{21,22} & a_{21,21} & a_{21,22} \\ a_{12,21} & a_{12,22} & a_{12,21} & a_{12,22} & a_{12,21} & a_{12,22} \\ a_{22,21} & a_{22,22} & a_{22,21} & a_{22,22} & a_{22,21} & a_{22,22} \\ a_{13,21} & a_{13,22} & a_{13,21} & a_{13,22} & a_{13,21} & a_{13,22} \\ a_{23,21} & a_{23,22} & a_{23,21} & a_{23,22} & a_{23,21} & a_{23,22} \end{bmatrix} \quad (124)$$

In general, let p be a permutation of $l:d$ and define the row and column modes by

$$r = p(l:e), \quad c = p(e+1:d),$$

where $0 \leq e \leq d$. This partitioning defines a matrix $A_{r \times c}$ that has r rows and c columns and whose entries are defined by

$$A_{r \times c}(cd(i, n(r)), cd(, ())) = A(i,). \quad (1248)$$

Important special cases include the modal unfoldings

$$r = [k], C = [1, \dots, k-1, k+1, \dots, d] \quad \text{or} \quad A_{r \times c} = A_{(k)}$$

and the vec operation

$$r = 1:d, C = [\emptyset] \quad \text{or} \quad A_{r \times c} = \text{vec}(A).$$

$\mathbb{R}^{g, r, \cdot}$

The outer product of tensor $B \in \mathbb{R}^{m_1 \times \dots \times m_f}$ with tensor $C \in \mathbb{R}^{n_1 \times \dots \times n_g}$ is the order $(f+g)$ tensor A defined by

$$A(i,) = B(i) \otimes C(), \quad 1 \leq i \leq m, 1 \leq v \leq n$$

Multiple outer products are similarly defined, e.g.,

$$A = B \otimes C \quad \text{or} \quad A(, , k) = B(i) \cdot C(j) \cdot V(k).$$

Note that if B and C are order 2 tensors (matrices), then

$$A = B \otimes C \quad * \quad A(i_1 i_2 j_1 h) = B(i_1 i_2) \cdot C(j_1 h)$$

and

$$A \otimes B = B \otimes A.$$

Thus, the Kronecker product of two matrices corresponds to their outer product as tensors.

1S)SqA }T i1ATb 1 :=A

Outer products between order-1 tensors (vectors) are particularly important. We say that $A \in \mathbb{R}^{n_1 \times \dots \times n_d}$ is a rank-1 tensor if there exist vectors $z^{(1)}, \dots, z^{(d)} \in \mathbb{R}^{k_1}, \dots, \mathbb{R}^{k_d}$ such that

$$A(i) = z^{(1)}(i_1) \cdots z^{(d)}(i_d), \quad 1 \leq i_1 \leq n_1$$

A small example clarifies the definition and reveals a Kronecker product connection:

$$A = \begin{bmatrix} & \\ & \end{bmatrix} \circ \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \circ \begin{bmatrix} & \\ & \end{bmatrix} = \begin{bmatrix} a_{111} \\ a_{211} \\ a_{121} \\ a_{221} \\ a_{131} \\ a_{231} \\ a_{112} \\ a_{212} \\ a_{122} \\ a_{222} \\ a_{132} \\ a_{232} \end{bmatrix} = \begin{bmatrix} UMM \\ UAM \\ UMW \\ UAW \\ UBW \\ UABW \\ UMMZ \\ UAMZ \\ UMWZ \\ UAWZ \\ UBWZ \\ UABWZ \end{bmatrix} = w \otimes v \otimes u.$$

The modal unfoldings of a rank-1 tensor are highly structured. For the above example we have

$$A_{(1)} = \begin{bmatrix} u_1 v_1 w_1 & u_1 v_2 w_1 & u_1 v_3 w_1 & u_1 v_1 w_2 & u_1 v_2 w_2 & u_1 v_3 w_2 \\ u_2 v_1 w_1 & u_2 v_2 w_1 & u_2 v_3 w_1 & u_2 v_1 w_2 & u_2 v_2 w_2 & u_2 v_3 w_2 \end{bmatrix} = u \otimes (w \otimes v)^T,$$

$$A_{(2)} = \begin{bmatrix} u_1 v_1 w_1 & u_2 v_1 w_1 & u_1 v_1 w_2 \\ u_1 v_2 w_1 & u_2 v_2 w_1 & u_1 v_2 w_2 \\ u_1 v_3 w_1 & u_2 v_3 w_1 & u_1 v_3 w_2 \end{bmatrix} = v \otimes (w \otimes u f),$$

$$A_{(3)} = \begin{bmatrix} u_1 v_1 w_1 & u_2 v_1 w_1 & u_1 v_2 w_1 & u_2 v_2 w_1 & u_1 v_3 w_1 & u_2 v_3 w_1 \\ u_1 v_1 w_2 & u_2 v_1 w_2 & u_1 v_2 w_2 & u_2 v_2 w_2 & u_1 v_3 w_2 & u_2 v_3 w_2 \end{bmatrix} = w \otimes (v \otimes u)^T.$$

In general, if $Z^{(k)} \in \mathbb{R}^{n_k \times k}$ for $k=1:d$ and

$$A = z^{(1)} \circ \dots \circ z^{(d)} \in \mathbb{R}^{n_1 \times \dots \times n_d},$$

then its modal unfoldings are rank-1 matrices:

$$A_{(k)} = Z^{(k)} \cdot (Z^{(1)} \otimes \dots \otimes Z^{(k-1)}) \otimes Z^{(k)}^T. \quad \{1249\}$$

For general unfoldingings of a rank 1 tensor, if p is a permutation of $l:d$, $\cdot = p(l:e)$, and $c = p(e+1:d)$, then

$$\mathcal{A}_{r \times c} = \left(z^{(p_e)} \circ \dots \circ z^{(p_1)} \right) \left(z^{(p_d)} \circ \dots \circ z^{(p_{e+1})} \right)^T. \quad (124.10)$$

Finally, we mention that any tensor can be expressed as a sum of rank 1 tensors

$$A \in \mathbb{R}^{n_1 \times \dots \times n_d} \quad \text{iff} \quad A = \sum_{i=1}^n A(i) l_n l(:, i) \circ \dots \circ l_n A(:, i_d).$$

An important §12.5 theme is to find more informative rank 1 summations than this!

hihsuib & le kb lkQ tfQ1lkcb tlnb ht QxTebDf Q13fTftQ1kb

Let us return to the notion of a tensor contraction introduced in §12.4.1. The first order of business is to show that a contraction between two tensors is essentially a matrix multiplication between a pair of suitably chosen unfoldings. This is a useful connection because it facilitates reasoning about high-performance implementation.

Consider the problem of computing

$$A(i,j,a_3 a_4 \{3\} 4 f \bar{5}) = \sum_{k=1}^{n_2} B(i,k,a_3 a_4) \cdot C(k,j,\{3\} 4 \{5\}) \quad (124.11)$$

where

$$\begin{aligned} A &= A(l:n_1 l:m_2 l:n_3 l:n_4 l:m_3 l:m_4 l:m_5), \\ B &= B(l:n_1 l:n_2 l:n_3 l:n_4), \\ C &= C(l:m_1 l:m_2 l:m_3 l:m_4 l:m_5), \end{aligned}$$

and $n_2 = m_1$. The index k is a contraction index. The example shows that in a contraction, the order of the output tensor can be (much) larger than the order of either input tensor, a fact that can prompt storage concerns. For example, if $n_1 = \dots = n_4 = r$ and $m_1 = \dots = m_5 = r$ in (124.11), then B and C are $O(r^9)$ while the output tensor A is $O(r^7)$.

The contraction (124.11) is a collection of related matrix-matrix multiplications. Indeed, at the slice level we have

$$A(\cdot, \cdot, \alpha_3, \alpha_4, \beta_3, \beta_4, \beta_5) = B(\cdot, \cdot, \alpha_3, \alpha_4) \cdot C(\cdot, \cdot, \beta_3, \beta_4, \beta_5).$$

Each A -slice is an n_1 -by- n_2 matrix obtained as a product of an n_1 -by- n_2 B -slice and an m_1 -by- m_2 C -slice.

The summation in a contraction can be over more than just a single mode. To illustrate, assume that

$$\begin{aligned} B &= B(l:m_1 l:m_2 l:t_1 l:t_2), \\ C &= C(l:t_1 l:t_2 l:n_1 l:m_2 l:n_3), \end{aligned}$$

and define $\mathcal{A} = \mathcal{A}(1:m_1, 1:m_2, 1:n_1, 1:n_2, 1:n_3)$ by

$$A(i_1 i_2 j_1 j_2 j_3) = \sum_{k_1=1}^{t_1} \sum_{k_2=1}^{t_2} B(i_1 i_2 k_1 k_2) \cdot C(k_1 k_2) j_1 j_2 j_3. \quad (124.12)$$

Note how "matrix like" this computation becomes with multiindex notation:

$$A(i,j) = \sum_{k=1}^t B(i,k) \cdot C(k,j), \quad \text{1 } i \text{ fam, 1 } j \text{ fan.} \quad (124.13)$$

A fringe benefit of this formulation is how nicely it connects to the following matrix-multiplication specification of A :

$$A[12]_x[34]_a = B[12]_x[34] \cdot C[2]_x[34]_{-1} a$$

The position of the contraction indices in the example (124.12) is convenient from the standpoint of factoring the overall operation as a product of two unfoldings. However, it is not necessary to have the contraction indices "on the right" in B and "on the left" in C to formulate the operation as a matrix multiplication. For example, suppose

$$B = B(l:t_2 l:m_1 l:i_1 l:m_2),$$

$$C = C(l:n_2 l:t_2 l:n_3 l:t_1 l:n_1),$$

and that we want to compute the tensor $A = A(l:m_1 l:m_2 l:n_1 l:n_2 l:n_3)$ defined by

$$A(i_2 j_1 i_1 j_2 j_3) = \sum_{k_1=1}^{t_1} \sum_{k_2=1}^{t_2} B(k_2 i_1 k_1, i_2) \cdot C(j_2 k_2 j_3 k_1, j_1).$$

It can be shown that this calculation is equivalent to

$$A[4]_x[352] = B[24]_x[31] \cdot C[42]_x[513].$$

Hidden behind these formulations are important implementation choices that define the overheads associated with memory access. Are the unfoldings explicitly set up? Are there any particularly good data structures that moderate the cost of data transfer? Etc. Because of their higher dimension, there are typically many more ways to organize a tensor contraction than there are to organize a matrix multiplication.

hc hshhyb ,esb hktfb .x kDf fb

A very simple but important family of contractions are the modal products. These contractions involve a tensor, a matrix, and a mode. In particular, if $S \in \mathbb{R}^{x_1 \times \dots \times x_d}$, $M \in \mathbb{R}^{m_k \times n_k}$, and $1 \leq k \leq d$, then A is the mode k product of S and M if

$$A_{(k)} = M \cdot S_{(k)}. \quad (124.14)$$

We denote this operation by

$$\mathcal{A} = \mathcal{S} \times_k M$$

and remark that

$$\mathcal{A}(\alpha_1, \dots, \alpha_{k-1}, i, \alpha_{k+1}, \dots, \alpha_d) = \sum_{j=1}^{\text{rk}} M(i,j) \cdot S(\alpha_1, \dots, \alpha_{k-1}, j, \alpha_{k+1}, \dots, \alpha_d)$$

and

$$\text{vec}(\mathcal{A}) = (I_{n_{k+1} \dots n_d} \otimes M \otimes I_{n_1 \dots n_{k-1}}) \cdot \text{vec}(S) \quad (124.15)$$

are equivalent formulations. Every mode k fiber in S is multiplied by the matrix M .

Using (124.15) and elementary facts about the Kronecker product, it is easy to show that

$$(S \times_k F) \times_j G = (S \times_j G) \times_k F, \quad (124.16)$$

$$(S \times_k F) \times_k G = S \times_k (FG), \quad (124.17)$$

assuming that all the dimensions match up.

hihs ghhb ThSb sDdf TSSxb .x kt Dff b

Suppose we are given an order-4 tensor $S \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4}$ and four matrices

$$M_1 \in \mathbb{R}^{m_1 \times n_1}, \quad M_2 \in \mathbb{R}^{m_2 \times n_2}, \quad M_3 \in \mathbb{R}^{m_3 \times n_3}, \quad M_4 \in \mathbb{R}^{m_4 \times n_4}.$$

The computation

$$A(i) = \sum_{j=1}^n S(j) \cdot M_1(i_1 j_1) \cdot M_2(i_2 j_2) \cdot M_3(i_3 j_3) \cdot M_4(i_4 j_4) \quad (124.18)$$

is equivalent to

$$\text{vec}(A) = (M_4 \circledast M_3 \circledast M_2 \circledast M_1) \text{vec}(S) \quad (124.19)$$

and is an order-4 example of a multilinear product. As can be seen in the following table, a multilinear product is a sequence of contractions, each being a modal product:

$a(0) = \text{vec}(S)$	$A(0) = S$
$a(1) = (I_{n_4} \circledast I_{n_3} \circledast I_{n_2} \circledast M_1) a(0)$	$A(1) = M_1 A(0) \quad (\text{Mode 1 product})$
$a(2) = (J_{n_4} \circledast I_{n_3} \circledast M_2 \circledast I_{n_1}) a(1)$	$A(2) = M_2 A(1) \quad (\text{Mode 2 product})$
$a(3) = (I_{n_4} \circledast M_3 \circledast I_{n_2} \circledast I_{n_1}) a(2)$	$A(3) = M_3 A(2) \quad (\text{Mode 3 product})$
$a(4) = (M_4 \circledast I_{n_3} \circledast I_{n_2} \circledast I_{n_1}) a(3)$	$A(4) = M_4 A(3) \quad (\text{Mode 4 product})$
$\text{vec}(A) = a(4)$	$A = A(4)$

The left column specifies what is going on in Kronecker product terms while the right column displays the four required modal products. The example shows that mode k operations can be sequenced.

$$A = Q \otimes M_1 \otimes M_2 \otimes M_3 \otimes M_4$$

and that their order is immaterial, e.g.,

$$A = Q \otimes M_4 \otimes M_1 \otimes M_2 \otimes M_3$$

This follows from (124.16).

Because they are used in §125 we summarize two key properties of the multilinear product in the following theorem.

Theorem 124.1. Suppose $S \in \mathbb{R}^{n_1 \times \dots \times n_d}$ and $M_k \in \mathbb{R}^{m_k \times n_k}$ for $k = 1:d$. If the tensor $A \in \mathbb{R}^{1 \times n_1 \times \dots \times n_d}$ is the multilinear product

$$A = S \otimes M_1 \otimes M_2 \otimes \dots \otimes M_d$$

then

$$A_{(k)} = M_k \cdot S_{(k)} \cdot (M_d \otimes \dots \otimes M_{k+1} \otimes M_{k-1} \otimes \dots \otimes M_1)^T.$$

If M_1, \dots, M_d are all nonsingular, then $S = A \otimes M_1^{-1} \otimes M_2^{-1} \dots \otimes M_d^{-1}$.

Proof. The proof involves equations (124.16) and (124.17) and the vec ordering of the mode k fibers in $A_{(k)}$. ³

1tnr iA = LV{jia :> N+A TGBA

We close with an example from Baumgartner et al. (2005) that highlights the importance of order of operations and what the space-time trade-off can look like when a sequence of contractions is involved. Suppose that A , B , C and V are N -by- N -by- N -by- N tensors and that S is defined as follows:

```

f r i = 14N
  s = 0
  f r k = 15N
    s = s + A(i1,ki,i2,k2) . B(i2,k3,k4,k5) . C(k5,k4,i4,k2) . V(ki,k5,k3,ks)
  end
  S(i) = s
end

```

Performing "as is," this is an $O(N^{10})$ calculation. On the other hand, if we can afford an additional pair of N -by- N -by- N -by- N arrays then work is reduced to $O(N^6)$. To see this, assume (for clarity) that we have a function $H = \text{Contract}(, 1)$ that computes the contraction

$$F(a_1, a_2, a_3, a_4) = \sum_{\beta_1=1}^N \sum_{\beta_2=1}^N X_{\alpha_1 \alpha_2 \beta_1 \beta_2} f_1 a_1 f_2 a_2 \cdot H(a_3 a_4 f_1 f_2),$$

a f notion: = Contract \mathcal{A} (, 1) that computes the contraction

$$: (o_1, o_2 o_3 o_4) = \sum_{i=1}^N \sum_{j=1}^N Q(o_1 i, o_2 / j) \cdot 1 (j / i, o_3 o_4),$$

and a f notion: = Contract \mathcal{B} (, 1) that computes the contraction

$$: (o_1 o_2 o_3 o_4) = \sum_{i=1}^N \sum_{j=1}^N Q(o_2 / i, o_4 / j) \cdot 1 (o_1 / i o_3 / j).$$

Each of these order-4 contractions requires $O(N^6)$ fops. By exploiting common subexpressions suggested by the parentheses in

$$((\mathcal{B}(i_2, k_3, k_4, k_5) \cdot \mathcal{D}(k_1, k_6, k_3, k_5)) \cdot \mathcal{C}(k_6, k_4, i_4, k_2)) \cdot \mathcal{A}(i_1, k_1, i_2, k_2),$$

we arrive at the following $O(N^6)$ specification of the tensor S:

$$\begin{aligned} &= \text{Contract } (\ ,) \\ &= \text{Contract}\mathcal{A}(\ , Q) \\ S &= \text{Contract}\mathcal{B}(\ , A) \end{aligned}$$

Of course, space-time trade-offs frequently arise in matrix computations. However, at the tensor level the stakes are typically higher and the number of options exponential. Systems that are able to chart automatically an optimal course of action subject to constraints that are imposed by the underlying computer system are therefore of interest. See Baumgartner et al. (2005).

Problems

P12.4.1 Pbd..ferob.f.01R 8.88...8 d ..1.selfed.fe fe8PfeN38...8. 8. 8 1.81..88
..1.. 8.1 1.. ..8 d1.fe 8obfe1l ...84dN

P12.4.2 3.8.8 1.1 1.8 .8 .fe1.8.0aPaQ5 f.0aPNP5 .fe. 181.8 .8 .fe1fe9NPNP5
.. f.0aPaP5

P12.4.3 P8rob .8...8 1.8.8 fe. 1.8 P18.8.1x...xnd? nn → n. ,o n,

P12.4.5 3.8.8 1l88.8. d01R

P12.4.6 ..ddg8.8 P E R^{n1 × ... × nd}, n, ψ, ψ^m p i : x̄ tx = V 2x #d. x̄ 4M_B x, = M, 12, () 1,, (,,

P12.4.7 ..ddg8.8 P E R^{n1 × ... × nd}, N n ... r. = x̄ = x̄ x̄ x = V 2x #d x x = : (xM,
, , , ,)f1M, -6)M, pf 9 x̄ M, 1-)M M, 12, ()1,, E R^{N × N}, n ψ, ψ^m
m p i 1x .2x x p 1̄ e 2 a

P12.4.8 ..ddg8.8 P E R^{n1 × ... × nd}, n, ψ, ψ^m , .n, ψ(n x (n → "k ElAtn lgg+ .mi.r it
MiaQ.AatlMd

P12.4.9 78 .18f.0aP75a.d84. > ..ro.fe.. 8 a = O p = t< 0O = lf w tl .mi. A i
.e .@n I/n - @n "I

P12.4.10 .ld d8.8P E R^{n1 × ... × nd}, n, ψ, ψ^m 1nψ, n, q a = O x p va v@d. = O ;M 12 ,
, r = x̄ .c=a p x̄ q p a >.

Notes and References for

fel.. .1.1f..1.1. 11N.1..1..1.1.1.6 .NN

T{q TdQfxt(o(D)/)x(E0-!)PQ=fE(-x(fxE))r)ON=##)=f@I(1)(CE1
)))+f- B)lqPIOEP!))- OP(()) Multiway Analysis.,pa a22tuApmAp2t,gm,aE9
a9.95,E .aE w9a9w.EAu((())T)#)E))P+(P=f(f PDEP)(- f CO(f PNsiAM Review.Nr
i,r JJ4

,u u2dI, tp., A, aa2Ala,, Enartup2_{vec} n = x + S_n n x 1 = a 1 n 12 n x x + S x k * S w = = px n x x 11.

a c m S x i S + S x = (4m+1)q PO E=PIEA - PENSAM J. Matr Anal. Appl. 33 Nr. 7.4

MATLAB O**J**i(E**v**e(**p****f****f****C****v**.**v**O**E****E****x****f****p****v**(**v****o****G**.**v****E****O****n****c****e****O****n****E****v****O****C****=****E****a****p****E****p****=****o****C****E****p****C****A****n****d****r**

) « DaG.E(= GB 1T CA(()) /) - (P)f(= ar BA) # -)q #)E = P) (=)A i(() - (P)f(= + !)P(P(If E(NCM T n . Math. Sof w, 51m5r7.59
w9aM w12aM.5, E ((D) /) - (G)E(A) # -)q)P + ((O(f PE = f(=)(O) =) OEP1OQP)P
)E = PNSIAM J. Sci. Comput 51hJ.115,v

ap2Ap.,2ar2t,An.2E l2ppnrp S2u,uC,aA2u tA,2 2,GSI...n,at.u2EntAI tt2ET

a9 n.aEut((()))x +(-)+)(EE(O(f(= !))A)+OEO#)E=P) f r)O)Nentif c Pr gr mming 11, IT #J.

Algebra Tools for Numerical Relativity, B., MJ , NJ 759

.9 w.lCr.uau2uB9812uE,w2uap,E3MnSuu B,Ap,SS2,.u AEAn,u.lPr.,u lMuuntau
s1 u.u s95utpaC,,u.ptu sMuuntpaA h.C uV9hI J 12,ne2alx hn..2ur lC.aIe .C u
OM EtSS.auaE 39snSmQ. ((())/)E(=)=f =Pf (=)))A)+OEQ O)O- !)P(0+A)
O) P)F x Ef (f(θE(+)=)+f =)I APP)NPr c IEEE, .5nIT7r I.I4

ap2CI.,nkt .a,ttnt A,CCI an.taE .p2Sl.aIC Ap2C nt.Mtn2,2AtuIAAUu2 A,CCI ~~8Ap~~
p.2 .p2m1ka :.u2E tt,2: .2at,ua...n,a .aE n .2tt En12u2as22

J. Chemometr. 14, N.J. 4

12.5 Tensor Decompositions and Iterations

Decompositions have three roles to play in matrix computations. They can be used to convert a given problem into an equivalent easy-to-solve problem, they can expose hidden relationships among the entries, and they can open the door to data sparse approximation. The role of tensor decompositions is similar and in this section we showcase a few important examples. The matrix SVD has a prominent role to play throughout. The goal is to approximate or represent a given tensor with an illuminating (hopefully short) sum of rank 1 tensors. Optimization problems arise that are multilinear in nature and lend themselves to the alternating least squares framework. These methods work by freezing all but one of the unknowns and improving the footer range variable with some tractable linear optimization strategy. Interesting matrix computations arise during this process and that is the focus of our discussion. For a much more complete survey of tensor decompositions, properties, and algorithms, see Kolda and Bader (2009). Our aim in these few pages is simply to give a snapshot of the "inner loop" linear algebra that is associated with a few of these methods and to build intuition for this increasingly important area of high dimensional scientific computing.

Heavy use is made of the Kronecker product and tensor unfolding. Thus, this section builds upon §12.3 and §12.4. We use order-3 tensors to drive the discussion, but periodically summarize what the theorems and algorithms look like for general-order tensors.

himihb 73b IT"esx c6x tSb u-pb

Let us think about the SVD of $A \in \mathbb{R}^{m \times n}$, not $\mathbb{R}^{n \times n}$.

$$A = \mathbf{U}\mathbf{E}\mathbf{T} = \sum_{i=1}^n \sigma_i u_i v_i^T, \quad (125.1)$$

but a $\mathbf{U}\mathbf{T}_A = \mathbf{E}\mathbf{T}$. The matrix \mathbf{U} structures the rows of \mathbf{T}_A so that they are orthogonal to each other and monotone decreasing in norm.

$$\mathbf{U}^T A = \begin{bmatrix} \sigma_1 v_1^T \\ \vdots \\ \sigma_n v_n^T \end{bmatrix}. \quad (125.2)$$

The optimality of this structure can be seen by considering the following problem

$$\max_{Q^T Q = I_r} \| \mathbf{Q} \mathbf{T}_A \|_F, \quad Q \in \mathbb{R}^{m \times r}. \quad (125.3)$$

It is easy to verify that the maximum value is $\sigma_1^2 + \dots + \sigma_r^2$ and that it can be attained by setting $Q = \mathbf{U}_{(:,1:r)}$. The left singular vector matrix does the best job from the standpoint of getting a "mask" as possible to the top of the transformed A . And that is what SVD does: it concentrates mask and supports an illuminating rank-1 expansion.

Now suppose $A \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and consider the following triplet of SVD's, one for each modal unfolding

$$U_1^T \mathcal{A}_{(1)} = \Sigma_1 V_1^T, \quad U_2^T \mathcal{A}_{(2)} = \Sigma_2 V_2^T, \quad U_3^T \mathcal{A}_{(3)} = \Sigma_3 V_3^T. \quad (125.4)$$

These define three independent modal products:

$$\mathcal{B}^{(1)} = \mathcal{A} \times_1 U_1, \quad \mathcal{B}^{(2)} = \mathcal{A} \times_2 U_2, \quad \mathcal{B}^{(3)} = \mathcal{A} \times_3 U_3. \quad (125.5)$$

Using Theorem 12.4.1, we have the following unfoldings:

$$\mathcal{B}_{(1)}^{(1)} = \Sigma_1 V_1^T (U_3 \otimes U_2)^T, \quad \mathcal{B}_{(2)}^{(2)} = \Sigma_2 V_2^T (U_3 \otimes U_1)^T, \quad \mathcal{B}_{(3)}^{(3)} = \Sigma_3 V_3^T (U_2 \otimes U_1)^T.$$

Note that each of these matrices has the same kind singular value "grading" that is displayed in (125.1). Recalling from §12.4.5 that the rows of an unfolding are subtensors, it is easy to show that

$$\begin{aligned} \|\mathcal{B}^{(1)}(\mathbf{i}, :, :) \|_F &= \sigma_i(\mathcal{A}_{(1)}), & i = 1:n_1, \\ \|\mathcal{B}^{(2)}(:, \mathbf{i}, :) \|_F &= \sigma_i(\mathcal{A}_{(2)}), & i = 1:n_2, \\ \|\mathcal{B}^{(3)}(:, :, \mathbf{i}) \|_F &= \sigma_i(\mathcal{A}_{(3)}), & \mathbf{i} = 1:n_3. \end{aligned}$$

If we assemble these three modal products into a single multilinear product, then we get

$$\mathbf{s} = \mathbf{A} \times_1 \mathbf{u}^1 \times_2 \mathbf{u}^2 \times_3 \mathbf{u}^3.$$

Because the U_i are orthogonal, we can apply Theorem 124.1 and get

$$\mathbf{A} = \mathcal{S} \times_1 U_1 \times_2 U_2 \times_3 U_3.$$

This is the higher-order SVD (HOSVD) developed by De Lathauwer, De Moor, and Vandebril (2000). We summarize some of its important properties in the following theorem.

Theorem 12.5.1 (HOSVD). If $\mathbf{A} \in \mathbb{R}^{n_1 \times \dots \times n_d}$ and

$$\mathbf{A}_{(k)} = U_k \Sigma_k V_k^T, \quad k=1:d$$

are the SVDs of its modal unfoldings, then its HOSVD is given by

$$\mathbf{A} = \mathcal{S} \times_1 U_1 \times_2 U_2 \cdots \times_d U_d \quad (1256)$$

where $\mathbf{s} = \mathbf{A} \times_1 \mathbf{u}^1 \times_2 U_2^T \cdots \times_d \mathbf{u}^d$. Therefore Equation (1256) is equivalent to

$$\mathbf{A} = \sum_{j=1}^n \mathcal{S}(j) \cdot U_1(:, j_1) \circ \cdots \circ U_d(:, j_d), \quad (1257)$$

$$\mathbf{A}(\mathbf{i}) = \sum_{j=1}^n \mathcal{S}(j) \cdot U_1(i_1, j_1) \cdots U_d(i_d, j_d), \quad (1258)$$

$$\text{vec}(\mathbf{A}) = (U_d \otimes \cdots \otimes U_1) \cdot \text{vec}(\mathcal{S}). \quad (1259)$$

Moreover,

$$\| \mathcal{S}_{(k)}(i, :) \|_F = \sigma_i(A_{(k)}), \quad i=1:\text{rank}(\mathbf{A}_{(k)}) \quad (12510)$$

for $k=1:d$.

Problem 12.5.1. We leave the verification of (1257)-(1259) to the reader. To establish (12510), note that

$$\begin{aligned} \mathbf{s}_{(k)} &= \mathbf{u}^1 A_{(k)} (U_d \otimes \cdots \otimes \mathbf{u}^k \otimes U_{k-1} \otimes \cdots \otimes U_1) \\ &= \Sigma_k \mathbf{V}^T (U_d \otimes \cdots \otimes U_{k+1} \circledast U_{k-1} \otimes \cdots \otimes U_1). \end{aligned}$$

It follows that the rows of $\mathbf{S}_{(k)}$ are mutually orthogonal and that the singular values of $A_{(k)}$ are the 2-norms of these rows. Q

In the HOSVD, the tensor \mathcal{S} is called the core tensor. Note that it is not diagonal. However, the inequalities (12510) tell us that, the values in \mathcal{S} tend to be smaller as "distance" from the $(1, 1, \dots, 1)$ entry increases.

16, σ_{*6} $p\}$ j v

If $\mathcal{A} \in \mathbb{R}^{n_1 \times \dots \times n_d}$, then its multilinear rank is a the vector of modal unfolding ranks

$$\text{rank}_*(\mathcal{A}) = [\text{rank}(\mathcal{A}_{(1)}), \dots, \text{rank}(\mathcal{A}_{(d)})].$$

Note that the summation upper bounds in the HOSVD can be replaced by $\text{rank}_*(\mathcal{A})$. For example, (1257) becomes

$$\mathcal{A} = \sum^{\text{wef 4blChal}} \mathcal{S}(\mathbf{j}) U_1(:, j_1) \circ \dots \circ U_d(:, j_d).$$

This suggests a path to lowrank approximation. If $\mathbf{r} \leq \text{rank}_*(\mathcal{A})$ with inequality in at least one component, then we can regard

$$\mathcal{A}^{(\mathbf{r})} = \sum^{\mathbf{r}} \mathcal{S}(\mathbf{j}) U_1(:, j_1) \circ \dots \circ U_d(:, j_d)$$

a a truncated HOSVD approximation to \mathcal{A} . It can be shown that

$$\text{vec } \mathcal{A}^{(\mathbf{r})} ((j,:) : \min_{1 \leq k \leq d} \sum^{\text{wef 4blChal}} \sigma_i(A_{(k)})^2). \quad (12511)$$

16, $\sigma_i \leq$

i

Suppose $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and assume that $\mathbf{r} \leq \text{rank}_*(\mathcal{A})$ with inequality in at least one component. Prompted by the optimality properties of the matrix SVD, let us consider the following optimization problem

$$\min_{\mathbf{x}} \|\mathcal{A} - \mathbf{X}\|_F \quad (12512)$$

such that

$$\mathbf{x} = \sum^{\mathbf{r}} \mathcal{S}(\mathbf{j}) \cdot U_1(:, j_1) \circ U_2(:, j_2) \circ U_3(:, j_3). \quad (12513)$$

We refer to this as the Tucker approximation problem. Unfortunately, the truncated HOSVD tensor $\mathcal{A}^{(\mathbf{r})}$ does not solve the Tucker approximation problem, prompting us to develop an appropriate optimization strategy.

To be clear, we are given \mathcal{A} and \mathbf{r} and seek a core tensor \mathcal{S} that is r_1 -by- r_2 -by- r_3 and matrices $U_1 \in \mathbb{R}^{n_1 \times r_1}$, $U_2 \in \mathbb{R}^{n_2 \times r_2}$, and $U_3 \in \mathbb{R}^{n_3 \times r_3}$ with orthonormal columns so that the tensor \mathbf{X} defined by (12513) solves (12512). Using Theorem 1241 we know that

$$\text{vec } \mathbf{X} \|\mathbf{F} = \|\text{vec}(\mathcal{A}) - (U_3 \odot U_2 \odot U_1) \cdot \text{vec}(\mathcal{S})\|_2.$$

Since $U_3 \odot U_2 \odot U_1$ has orthonormal columns, it follows that the "best" \mathcal{S} given any triplet $\{U_1, U_2, U_3\}$ is

$$\mathcal{S} = (U_3^T \odot U_2^T \odot U_1^T) \cdot \text{vec}(\mathcal{A}).$$

Thus, we can remove S from the search space and simply look for $U = U_3 \otimes U_2 \otimes U_1$ so that

$$\left\| \mathbf{I} - UU^T \right\| \cdot \text{vec}(\mathcal{A}) \leq \left\| \text{vec}(\mathcal{A}) \mathbf{I} - U^T \cdot \text{vec}(\mathcal{A}) \mathbf{I} \right\|$$

is minimized. In other words, determine U_1 , U_2 , and U_3 so that

$$\left\| (U_3^T \otimes U_2^T \otimes U_1^T) \cdot \text{vec}(\mathcal{A}) \mathbf{I} \right\|_{\mathbb{F}} = \begin{cases} \left\| U_1^T \cdot A_{(1)} \cdot (U_3 \otimes U_2) \mathbf{I} \right\|_{\mathbb{F}} \\ \left\| U_2^T \cdot A_{(2)} \cdot (U_3 \otimes U_1) \mathbf{I} \right\|_{\mathbb{F}} \\ \left\| U_3^T \cdot A_{(3)} \cdot (U_2 \otimes U_1) \mathbf{I} \right\|_{\mathbb{F}} \end{cases}$$

is maximized. By freezing any two of the three matrices $\{U_1, U_2, U_3\}$ we can improve the third by solving an optimization problem of the form (1253). This suggests the following strategy:

Repeat:

Maximize $\left\| U_1^T \cdot A_{(1)} \cdot (U_3 \otimes U_2) \mathbf{I} \right\|_{\mathbb{F}}$ with respect to U_1 by computing the SVD $A_{(1)} \cdot (U_3 \otimes U_2) = \tilde{U}_1 \Sigma_1 V$. Set $U_1 = \tilde{U}_1(:, 1:r_1)$.

Maximize $\left\| U_2^T \cdot A_{(2)} \cdot (U_3 \otimes U_1) \mathbf{I} \right\|_{\mathbb{F}}$ with respect to U_2 by computing the SVD $A_{(2)} \cdot (U_3 \otimes U_1) = \tilde{U}_2 \Sigma_2 V_2^T$. Set $U_2 = \tilde{U}_2(:, 1:r_2)$.

Maximize $\left\| U_3^T \cdot A_{(3)} \cdot (U_2 \otimes U_1) \mathbf{I} \right\|_{\mathbb{F}}$ with respect to U_3 by computing the SVD $A_{(3)} \cdot (U_2 \otimes U_1) = \tilde{U}_3 \Sigma_3 V_3^T$. Set $U_3 = \tilde{U}_3(:, 1:r_3)$.

This is an example of the alternating least squares framework. For ordered tensors there are optimizations to perform at each step.

Repeat:

for $k = 1:d$

Compute the SVD:

$$A_{(k)} (U_d \otimes \dots \otimes U_{k+1} \otimes U_{k-1} \otimes \dots \otimes U_1) = \tilde{U}_k \Sigma_k V_k^T.$$

$$U_k = \tilde{U}_k(:, 1:r_k)$$

end

This is essentially the Tucker framework. For implementation details concerning this nonlinear iteration, see De Lathouwer, De Moor, and Vandewalle (2000b), Smilde, Bro, and Geladi (2004, pp. 119–123), and Kolda and Bader (2009).

hxmihsb 7\\$b ..b n33 xkeTNQJTkLb .x kr\\$b

A nice attribute of the matrix SVD that is that the "core matrix" in the rank-1 expansion is diagonal. This is not true when we graduate to tensors and work with the

Tucker representation. However, there is an alternate way to extrapolate from the matrix SVD if we prefer "diagonalness" to orthogonality. Given $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and an integer r , we consider the problem

$$\min_{\mathcal{X}} \| \mathcal{A} - \mathcal{X} \|_F \quad (125.14)$$

such that

$$\mathcal{X} = \sum_{j=1}^r \mathbf{A}_j \cdot \mathbf{F}(j) \odot \mathbf{G}(j) \odot \mathbf{H}(j) \quad (125.15)$$

where $\mathbf{F} \in \mathbb{R}^{n_1 \times r}$, $\mathbf{G} \in \mathbb{R}^{n_2 \times r}$, and $\mathbf{H} \in \mathbb{R}^{n_3 \times r}$. This is an example of the CP approximation problem. We assume that the columns of \mathbf{F} , \mathbf{G} , and \mathbf{H} have unit 2-norm.

The modal loadings of the tensor (125.15) are neatly characterized through the Khatri-Rao product that we defined in §12.3.3. If

$$\mathbf{F} = [\mathbf{f}_1 \cdots \mathbf{f}_r], \quad \mathbf{G} = [\mathbf{g}_1 \cdots \mathbf{g}_r], \quad \mathbf{H} = [\mathbf{h}_1 \cdots \mathbf{h}_r],$$

then

$$\mathcal{X}_{(1)} = \sum_{j=1}^r \mathbf{i} \cdot \mathbf{f}_j \odot (\mathbf{h}_j \odot \mathbf{g}_j)^\top = \mathbf{F} \cdot \text{diag}(\mathbf{j}) \cdot (\mathbf{H}_0 \odot \mathbf{G})^\top,$$

$$\mathcal{X}_{(2)} = \sum_{j=1}^r \mathbf{i} \cdot \mathbf{g}_j \odot (\mathbf{h}_j \odot \mathbf{f}_j)^\top = \mathbf{G} \cdot \text{diag}(\mathbf{j}) \cdot (\mathbf{H}_0 \odot \mathbf{F})^\top,$$

$$\mathcal{X}_{(3)} = \sum_{j=1}^r \mathbf{A}_j \cdot \mathbf{h}_j \odot (\mathbf{g}_j \odot \mathbf{f}_j)^\top = \mathbf{H} \cdot \text{diag}(\mathbf{j}) \cdot (\mathbf{G}_0 \odot \mathbf{F})^\top.$$

These results follow from the previous section. For example,

$$\begin{aligned} \mathcal{X}_{(1)} &= \sum_{j=1}^r \mathbf{i} \cdot (\mathbf{f}_j \odot \mathbf{g}_j \odot \mathbf{h}_j)^\top = \sum_{j=1}^r \mathbf{i} \cdot \mathbf{f}_j \cdot (\mathbf{h}_j \odot \mathbf{g}_j)^\top \\ &= [\lambda_1 \mathbf{f}_1 \mid \cdots \mid \lambda_r \mathbf{f}_r] \left[\mathbf{h}_1 \otimes \mathbf{g}_1 \mid \cdots \mid \mathbf{h}_r \otimes \mathbf{g}_r \right]^T = \mathbf{F} \cdot \text{diag}(\lambda_j) \cdot (\mathbf{H} \odot \mathbf{G})^T. \end{aligned}$$

Noting that

$$\| \mathcal{A} - \mathcal{X} \|_F = \| \mathcal{A}_{(1)} - \mathcal{X}_{(1)} \|_F = \| \mathcal{A}_{(2)} - \mathcal{X}_{(2)} \|_F = \| \mathcal{A}_{(3)} - \mathcal{X}_{(3)} \|_F,$$

we see that the CP approximation problem can be solved by minimizing any one of the following expressions:

$$\| \mathcal{A}_{(1)} - \mathcal{X}_{(1)} \|_F = \| \mathcal{A}_{(1)} - \mathbf{F} \cdot \text{diag}(\mathbf{j}) \cdot (\mathbf{H}_0 \odot \mathbf{G})^\top \|_F, \quad (125.16)$$

$$\| \mathcal{A}_{(2)} - \mathcal{X}_{(2)} \|_F = \| \mathcal{A}_{(2)} - \mathbf{G} \cdot \text{diag}(\mathbf{j}) \cdot (\mathbf{H}_0 \odot \mathbf{F})^\top \|_F, \quad (125.17)$$

$$\| \mathcal{A}_{(3)} - \mathcal{X}_{(3)} \|_F = \| \mathcal{A}_{(3)} - \mathbf{H} \cdot \text{diag}(\mathbf{j}) \cdot (\mathbf{G}_0 \odot \mathbf{F})^\top \|_F \quad (125.18)$$

This is a multilinear least squares problem. However, observe that if we fix H , and G in (125.16), then $\|A(i) - X(i)\|_F$ is linear in F . Similar comments apply to (125.17) and (125.18) and we are led to the following alternating least squares minimization strategy:

Repeat:

$$\begin{aligned} \text{Let } \tilde{F} \text{ minimize } \|A(1) - \tilde{F} \cdot (H \otimes G)^T\|_F \quad \text{and for } j = 1:r \text{ set} \\ \lambda_j = \|\tilde{F}(:,j)\|_2 \quad \text{and} \quad F(:,j) = \tilde{F}(:,j)/\lambda_j. \end{aligned}$$

$$\begin{aligned} \text{Let } \tilde{G} \text{ minimize } \|A(2) - \tilde{G} \cdot (H \otimes F)^T\|_F \quad \text{and for } j = 1:r \text{ set} \\ \mathbf{i} = \|\tilde{G}(:,j)\|_2 \quad \text{and} \quad G(:,j) = \tilde{G}(:,j)/\lambda_j. \end{aligned}$$

$$\begin{aligned} \text{Let } \tilde{H} \text{ minimize } \|A(3) - \tilde{H} \cdot (G \otimes F)^T\|_F \quad \text{and for } j = 1:r \text{ set} \\ \mathbf{i} = \|\tilde{H}(:,j)\|_2 \quad \text{and} \quad H(:,j) = \tilde{H}(:,j)/\lambda_j. \end{aligned}$$

The update calculations for F , G , and H are highly structured linear least squares problems. The central calculations involve linear least square problems of the form

$$\min \|B \otimes C z - d\|_2 \tag{125.19}$$

where $B \in \mathbb{R}^{p_B \times q}$, $C \in \mathbb{R}^{p_C \times q}$, and $d \in \mathbb{R}^{p_B p_C}$. This is typically a "tall skinny" LS problem. If we form the Khatri-Rao product and use the QR factorization in the usual way, then $O(p_B p_C q^2)$ fops are required to compute z . On the other hand, the normal equation system corresponding to (125.19) is

$$((B^T B) \circledast (C^T C)) z = (B \odot C)^T d \tag{125.20}$$

which can be formed and solved via the Cholesky factorization in $O((p_B + p_C)q^2)$ fops.

For general tensors $A \in \mathbb{R}^{n_1 \times \dots \times n_d}$ there are d least squares problems to solve per pass. In particular, given A and r , the CP approximation problem involves finding matrices

$$F^{(k)} = [f_1^{(k)} \mid \dots \mid f_r^{(k)}] \in \mathbb{R}^{n_k \times r}, \quad k = 1:d$$

with unit 2-norm columns and a vector $\mathbf{x} \in \mathbb{R}^r$ so that if

$$\mathbf{x} = \sum_{j=1}^r A_j f_j \mathbf{l}_j \quad \text{for } j = 1:r \tag{125.21}$$

then $\|A - X\|_F$ is minimized. Noting that

$$X_{lk} = p^{lk} \text{diag}(\mathbf{x}) \left(p^{l_1} \otimes \dots \otimes p^{l_{k-1}} \otimes p^{l_k} \right)^T,$$

we obtain the following iteration

Repeat:

for $k = 1:d$

Minimize $\| A_{(k)} - \tilde{F}^{(k)} (F_1^d \otimes \dots \otimes F^{(k+1)} \otimes F^{(k-1)} \otimes \dots \otimes F^{(1)}) \|_F$
with respect to $\tilde{F}^{(k)}$.

for $j = 1:r$

$$\lambda_j = \| \tilde{F}_{(k)}(:,j) \|_2$$

$$F^{(k)}(:,j) = \tilde{F}_k(:,j)/\lambda_j$$

end

end

This is the CANDECOMP/PARAFAC framework. For implementation details about this nonlinear iteration, see Smilde, Bro, and Geladi (2004, pp. 113–119) and Kolda and Bader (2009).

hiigib 8 le T.b9Ntdb

The choice of r in the CP approximation problem brings us to the complicated issue of tensor rank. If

$$A = \sum_{j=1}^r \lambda_j f_j J^j \circ \dots \circ f_1$$

and no shorter sum of rank-1's exists, then we say that A is a rank- r tensor. Thus we see that in the CP approximation problem is a problem of finding the best rank- r approximation. Using the CP framework to discover the rank of a tensor is problematic because of the following complications.

Complication 1. The tensor rank problem is NP-hard. See and Hilla and Lim (2012).

Complication 2. The largest rank attainable for an $n_1 \times \dots \times n_d$ tensor is called the maximum rank. There is no simple formula like $\min\{n_1, \dots, n_d\}$. Indeed, maximum rank is known for only a handful of special cases.

Complication 3. If the set of rank- k tensors in $\mathbb{R}^{n_1 \times \dots \times n_d}$ has positive measure, then k is a typical rank. The space of $n_1 \times \dots \times n_d$ can have more than one typical rank. For example, the probability that a random 2-by-2-by-2 tensor has rank 2 is .79 while the probability that it has rank 3 is .21, assuming that the a_{ijk} are normally distributed with mean 0 and variance 1. See de Silva and Lim (2008) and Martin (2011) for detailed analysis of the 2-by-2-by-2 case.

Complication 4. The rank of a particular tensor over the real field may be different than its rank over the complex field.

Complication 5. There exist tensors that can be approximated with arbitrary precision by a tensor of lower rank. Such a tensor is said to be degenerate.

Complication 6. If

$$\mathbf{X}_r \in \sum_{j=1}^{r+1} \lambda_j U_1(:, j) \circ \cdots \circ U_d(:, j)$$

is the best rank- $(r+1)$ approximation of \mathcal{A} , then it does *not* follow that

$$\mathbf{X}_{r+1} = \sum_{j=1}^r \lambda_j \hat{U}_1(:, j) \circ \cdots \circ \hat{U}_d(:, j)$$

is the best rank- r approximation of \mathcal{A} . See Kolda (2009) for an example. Subtracting the best rank-1 approximation can even increase the rank! See Stegeman and Comon (2009).

See Kolda and Bader (2009) for references on tensor rank and its implications for computation. Examples that illuminate the subtleties associated with tensor rank can be found in the paper by de Silva and Lim (2008).

hchivnb & tekib uTt'Dftxb-t fDSezhrb-t TteTktfbn33 xkt fhb

The singular values of a matrix $A \in \mathbb{R}^{n_1 \times n_2}$ are the stationary values of

$$\psi_A(u, v) = \frac{u^T A v}{\|u\|_2 \|v\|_2} = \frac{\sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} A(i_1, i_2) u(i_1) v(i_2)}{\|u\|_2 \|v\|_2} \quad (1252)$$

and the associated stationary vectors are the corresponding singular vectors. This follows by looking at the gradient equation $\nabla \psi_A(u, v) = 0$. Indeed, if u and v are unit vectors, then this equation has the form

$$\nabla \psi_A(u, v) = \begin{bmatrix} Av - \psi_A(u, v)u \\ A^T u - \psi_A(u, v)v \end{bmatrix} = 0.$$

This variational characterization of matrix singular values and vectors extends to tensors, see Lim (2005). Suppose $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and define

$$\psi_{\mathcal{A}}(u_1, u_2, u_3) = \frac{\sum_{i=1}^n \mathcal{A}(i) \cdot u_1(i_1) u_2(i_2) u_3(i_3)}{\|u_1\|_2 \|u_2\|_2 \|u_3\|_2}$$

where $u_1 \in \mathbb{R}^{n_1}$, $u_2 \in \mathbb{R}^{n_2}$, and $u_3 \in \mathbb{R}^{n_3}$. It is easy to show that

$$\psi_{\mathcal{A}}(u_1, u_2, u_3) \approx \begin{cases} u_1^T \mathcal{A}_{(1)}(u_3 \otimes u_2) / (\|u_1\|_2 \|u_2\|_2 \|u_3\|_2), \\ u_2^T \mathcal{A}_{(2)}(u_3 \otimes u_1) / (\|u_1\|_2 \|u_2\|_2 \|u_3\|_2), \\ u_3^T \mathcal{A}_{(3)}(u_2 \otimes u_1) / (\|u_1\|_2 \|u_2\|_2 \|u_3\|_2). \end{cases}$$

If u_1 , u_2 , and u_3 are unit vectors, then the equation $\nabla \psi_A = 0$ is

$$\nabla \psi_{\mathcal{A}} = \begin{bmatrix} & & \otimes u_2) \\ & \bullet\bullet & \\ & \bullet\bullet & \otimes u_1) \\ A_{(3)}(u_2 \otimes u_1) \end{bmatrix} - \psi_{\mathcal{A}}(u_1, u_2, u_3) \begin{bmatrix} u \\ u \\ u_3 \end{bmatrix} = 0$$

If we can satisfy this equation, then we will call $\lambda_A(u_1, u_2, u_3)$ a singular value of the tensor A . If we take a componentwise approach to this nonlinear system we are led to the following iteration

Repeat:

$$\tilde{u}_1 = \mathcal{A}_{(1)}(u_3 \otimes u_2), \quad u_1 = \tilde{u}_1 / \| \tilde{u}_1 \|_2$$

$$\tilde{u}_2 = \mathcal{A}_{(2)}(u_3 \otimes u_1), \quad u_2 = \tilde{u}_2 / \| \tilde{u}_2 \|_2$$

$$\tilde{u}_3 = \mathcal{A}_{(3)}(u_2 \otimes u_1), \quad u_3 = \tilde{u}_3 / \| \tilde{u}_3 \|_2$$

$$_1^T\left(u_1,u_2,u_3\right)$$

This can be thought of as a higher-order power iteration. Upon comparison with the Tucker approximation problem with $\mathbf{U} = [\mathbf{I}, \mathbf{1}, \dots, \mathbf{1}]$, we see that it is a strategy for computing a nearest rank- \mathbf{I} tensor.

hihihTb uK .§ f xTf§ te kbbbT§f t f §zrb -tx Ttf Htf b nEE xkt ffb

If $C \in J_{\mathbb{N} \times \mathbb{N}}$ is symmetric, then its eigenvalues are the stationary values of

$$\phi_C(x) = \frac{x^T C x}{x^T x} = \frac{\sum_{i_1=1}^N \sum_{i_2=1}^N C(i_1, i_2) x(i_1) x(i_2)}{x^T x} \quad (12523)$$

and the corresponding stationary vectors are eigenvectors. This follows by setting the gradient of c to zero.

If we are to generalize this notion to tensors, then we need to define what we mean by a symmetric tensor. An order-d tensor $C \in \mathbb{R}^{x_1 \times x_2 \times \dots \times x_d}$ is symmetric if for any permutation p of $1:d$ we have

$$\mathcal{C}(\mathbf{i}) = \mathcal{C}(\mathbf{i(p)}), \quad 1 \leq \mathbf{i} \leq n.$$

For the case $d = 3$ this means $G_{jk} = G_{kj} = G_{ik} = G_{ki} = G_{ij} = G_{ji}$ for all $i, j,$ and k that satisfy $1 \leq i \leq N, 1 \leq j \leq N,$ and $1 \leq k \leq N.$

It is easy to generalize (12.523) to the case of symmetric tensors. If $\mathbf{C} \in \mathbb{M}^{N \times N}$ is symmetric and $\mathbf{X} \in \mathbb{M}^N$, then we define $\langle \mathbf{c} \rangle$ by

$$\text{c}(\mathbf{x}) = \frac{\sum_{i=1}^N c(i) \cdot x(i)}{\|\mathbf{x}\|_2^3} = \frac{\mathbf{x}^T \mathbf{C}(\mathbf{x})}{\|\mathbf{x}\|_2^3}. \quad (1252)$$

Note that if C is a symmetric tensor, then all its modal unfoldings are the same. The equation ' $\langle C(x) = 0$ ' with $\|x\|_2 = 1$ has the form

$$\langle C(x) = C_1(x \otimes x) - \langle C(x) \cdot x = 0$$

If this holds then we refer to $\langle C(x)$ as an eigenvalue of the tensor C , a concept introduced by Lim (2005) and Li (2005). An interesting framework for solving this nonlinear equation has been proposed by Kolda and Mayo (2012). It involves repetition of the operation sequence

$$\tilde{x} = C_{(1)}(x \otimes x) + \alpha x, \quad \lambda = \|\tilde{x}\|_2, \quad x = \tilde{x}/\lambda$$

where the shift parameter α is determined to ensure convexity and eventual convergence of the iteration. For further discussion of the symmetric tensor eigenvalue problem and various power iterations that can be used to solve it, see Zhang and Golub (2001) and Kofidis and Regalia (2002).

hihipib & tek b k & Q k denbs te kb 7 t1enbttb Qhs.Dx esb

In many applications, tensor decompositions and their approximations are used to discover things about a high-dimensional data set. In other settings, they are used to address the curse of dimensionality, i.e., the challenges associated with a computation that requires $O(n^d)$ work or storage. Whereas "bigness" is problematic in matrix computations, "big d" is typically the hallmark of a difficult large-scale tensor computation. For example, it is (currently) impossible to store explicitly an $n_1 \times \dots \times n_{100}$ tensor if $n_1 = \dots = n_{100} = 2$. In general, a solution framework for an order-d tensor problems suffers from the curse of dimensionality if the associated work and storage are exponential in d .

It is in this context that data-sparse tensor approximation is increasingly important. One way to build a high-order, data-sparse tensor is by connecting a set of low-order tensors with a relatively small set of contractions. This is the notion of a tensor network. In a tensor network, the nodes are low-order tensors and the edges are contractions. A special case that communicates the main idea is the tensor train (TT) representation, which we proceed to illustrate with an order-5 example. Given the low-order tensor "carriages"

- 91 ni xri,
- 92 ri x n2 x r2
- 93 r2 x n3 x r3
- 94 rax n4 x r4
- 95 r4 x ns

we define the order-5 tensor train, T , by

$$T(i) \in \sum_{k=1}^r 91(ii, ki)92(k1, i2, k2)93(k2, i3, k3)94(k3, i4, k4)95(k4, is). \quad (125.25)$$

The pattern is obvious from the example. The first and last carriages are matrices and all those in between are order-3 tensors. Adjacent carriages are connected by a single contraction. See Figure 125.1.

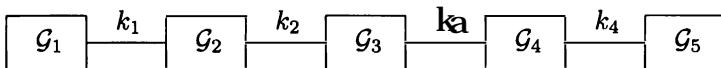


Figure 1251. The Order-5 tensor train (12.5.25)

To appreciate the data sparsity of an order- d tensor train $T \in \mathbb{R}^{n_1 \times \dots \times n_d}$ that is represented through its carriages, assume that $b_1 = \dots = b_d = r_1 = \dots = r_d = 2$. It follows that the TT-representation requires $O(d^2)$ memory locations, which is much less than the b^d storage required by the explicit representation.

We present a framework for approximating a given tensor with a data-sparse tensor train. The first order of business is to show that any tensor A has a TT representation. This can be verified by induction. For insight into the proof we consider an order-5 example. Suppose $A \in \mathbb{R}^{n_1 \times \dots \times n_5}$ is the result of a contraction between a tensor

$$B(i_1 i_2 k_2) = \sum_{k_1=1}^{r_1} 9\delta(i_1, k_1) Q(k_1, i_2 k_2)$$

and a tensor C as follows

$$A(i_1 i_2 i_3 i_4 i_5) = \sum_{k_2=1}^{r_2} B(i_1, i_2 k_2) Q(k_2 i_3 i_4 i_5).$$

If we can express C as a contraction of the form

$$Q(k_2 i_3 i_4 i_5) = \sum_{k_3=1}^{r_3} Q(k_2 i_3 k_3) C(k_3 i_4 i_5), \quad (12.5.26)$$

then

$$\begin{aligned} A(i_1 i_2 i_3 i_4 i_5) &= \sum_{k_2=1}^{r_2} \sum_{k_3=1}^{r_3} B(i_1, i_2 k_2) Q(k_2 i_3 k_3) C(k_3 i_4 i_5) \\ &= \sum_{k_3=1}^{r_3} \left(\sum_{k_2=1}^{r_2} B(i_1, i_2, k_2) G_3(k_2, i_3, k_3) \right) \tilde{C}(k_3, i_4, i_5) \\ &= \sum_{k_3=1}^{r_3} B(i_1 i_2 i_3 k_3) Q(k_3 i_4 i_5) \end{aligned}$$

where

$$\tilde{B}(i_1, i_2, i_3, k_3) = \sum_{k_1=1}^{r_1} \sum_{k_2=1}^{r_2} 9\delta(i_1, k_1) Q(k_1, i_2 k_2) Q(k_2 i_3 k_3).$$

The transition from writing A as a contraction of B and C to a contraction of B and \tilde{C} shows by example how to organize a formal proof that any tensor has a TT-representation. The only remaining issue concerns the "factorization" (12.5.26). It

tums out that the tensors $\mathbf{A}_{(1)}$ and \mathbf{C} can be determined by computing the SVD of the unfolding.

$$\mathbf{C} = \mathbf{U}_{\mathbf{B}} \mathbf{E}_3 \mathbf{M}_1$$

Indeed, if $\text{rank}(\mathbf{C}) = r_3$ and $\mathbf{C} = \mathbf{U}_{\mathbf{B}} \mathbf{E}_3 \mathbf{M}_1$ is the SVD with $\mathbf{E}_3 \in \mathbb{R}^{r_3 \times r_3}$, then it can be shown that (125.26) holds if we define $\mathcal{G}_3 \in \mathbb{R}^{r_2 \times n_3 \times r_3}$ and $\tilde{\mathbf{C}} \in \mathbb{R}^{r_3 \times n_4 \times n_5}$ by

$$\text{vec}(\mathbf{B}) = \text{vec}(\tilde{\mathbf{C}}), \quad (125.27)$$

$$\text{vec}(\mathbf{A}) = \text{vec}(\mathbf{E}_3 \mathbf{M}_1). \quad (125.28)$$

By extrapolating from this discussion we obtain the following procedure due to Oseledets and Tytyshnikov (2009) that computes the tensor train representation

$$\mathbf{A}(\mathbf{i}) = \sum_{\mathbf{k}(1:d-1)}^{\mathbf{r}(1:d-1)} \mathcal{G}_1(i_1, k_1) \mathcal{G}_2(k_1, i_2, k_2) \cdots \mathcal{G}_{d-1}(k_{d-2}, i_{d-1}, k_{d-1}) \mathcal{G}_d(k_{d-1}, i_d)$$

for any given $\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$:

M1 = $\mathbf{A}_{(1)}$

SVD: $\mathbf{M}_1 = \mathbf{U}_1 \mathbf{E}_1 \mathbf{M}_1$ where $\mathbf{E}_1 \in \mathbb{R}^{r_1 \times r_1}$ and $r_1 = \text{rank}(\mathbf{M}_1)$

for $k = 2:d-1$

$\mathbf{M}_k = \text{reshape}(\mathbf{E}_k \mathbf{M}_{k-1}, \mathbf{r}_k, \mathbf{l}_{k-1} \mathbf{n}_k \mathbf{n}_{k+1} \cdots \mathbf{n}_d)$ (125.29)

SVD: $\mathbf{M}_k = \mathbf{U}_k \mathbf{E}_k \mathbf{M}_k$ where $\mathbf{E}_k \in \mathbb{R}^{r_k \times r_k}$ and $r_k = \text{rank}(\mathbf{M}_k)$

Define $\mathcal{G}_k \in \mathbb{R}^{r_{k-1} \times n_k \times r_k}$ by $\text{vec}(\mathbf{M}_k) = \text{vec}(\mathbf{U}_k)$.

end

$\mathcal{G}_d = \mathbf{E}_{d-1} \mathbf{V}_{d-1}$

Like the HOSVD, it involves a sequence of SVDs performed on unfoldings.

In its current form (125.29) does not in general produce a data-sparse representation. For example, if $d=5$, $n_1 = \cdots = n_5 = n$, and M_1, \dots, M_4 have full rank, then $r_1 = n$, $r_2 = n^2$, $r_3 = n^2$, and $r_4 = n$. In this case the TT-representation requires the same $O(n^5)$ storage as the explicit representation.

To realize a data-sparse tensor train approximation, the matrices \mathbf{U}_k and \mathbf{M}_k are replaced with "thinner" counterparts that are intelligently chosen and cheap to compute. As a result, the r_k 's are replaced by (significantly smaller) f_k 's. The approximating tensor train involves fewer than $d(n_1 + \cdots + n_d) \cdot (\max_k f_k)$ numbers. This kind of approximation overcomes the curse of dimensionality assuming that $\max_k f_k$ does not depend on the modal dimensions. See Oseledets and Tytyshnikov (2009) for computational details, successful applications, and discussion about the low-rank approximations of M_1, \dots, M_{d-1} .

Problems

P12.5.1 ...8.8 a E Rⁿ¹ 123 f4 II 4II f E Rⁿ¹ a - gθ Rⁿ² o = • | | a = h®g®f .J ntCnaeCAe .mAM(Rⁿ³) (-),i dw" ,),(),(l · M(1,-,

P12.5.2 N... A E R^{1x2xn3} 3u=fJ oO0=(>no(h®g®f 3 00= 0= f (=f®goh E R^{n1xn2xn}: o = • a • eθ < = 30=F= Gex (x0bm 0

$$I(f, g, h) = \sum_{i=1}^n cxaG[Gqq] p(aG[Gqq])^3.$$

P12.5.3 .686 .6.68..d 8≥. .N8... 8≥.84.8. ..8.8.8. .6.. ..4dR 15

P12.5.4 fe8..8.18 Ptot73 .18..ot.1.8. .8.8. E R^{n₁ × ... × n_d ip)} -'R^{m₁.. n r}

$\sum_{k=1}^n R^{n_k} \prod_{j=1}^m R^{n_j m_j} A^{n_j m_j}_{(k)}$ = Q_kR_kE^{9U OE i(M-rCA(FAnA)=EC2A}

21 = matEAEQAI

P12.5.5 3.8.8 R01P51

P12.5.6 ..86 .6.R01RP5. RdMP15P8 8r....8.. 182.. ..ot.. || = (H ⊙ G ⊙ F). R

P12.5.7 k...b .8 .8.8. N. 68368.8.d..8..8. 81.688.8....8. R.0P10051

P12.5.8 P86...o.8.8.o .8 .. .2.8.... 9 .. 91.8.8.8

P12.5.9 ..d.8 .8 E R^{N × N × N × N p} R^{m. m. }_4 ip)}

$$\mathcal{A}(i_1, i_2, i_3, i_4) = \mathcal{A}(i_2, i_1, i_3, i_4) = \mathcal{A}(i_1, i_2, i_4, i_3) = \mathcal{A}(i_3, i_4, i_1, i_2).$$

$m(R^n) \in \mathbb{R}^{3 \times [2,4]} = (A;j) \text{ ntia} \quad \text{Str} \quad .i.SGAEm \quad r \text{ StSULAB} \quad .2ipA;j = A_{ij}$

iae A = A_{ij}.

P12.5.10 88.m08. 8.8.1 .8...8≥ 128...8.. ..8.8.18. .. 01P5836 ...

C.'-frE.H.DO -d'.y,fD.d,(E'E,DO-'ED-'.(5eP 9,'f-D,HEh4,fEG E,-44E, 4E,Dp
 -4(z' RDJ,H)-' R_dH--),D(44).

fo Pi fo-(e-5f'RkSP' RDD,J- fe84H-f411c AAI S CRD1E,4(5,p.Dp-e45'PP DE D
 E,DIKSIAM J. Matr. Anal. Appl. 21uN5vNTdx

3t.SSnar, LmA32E9A,Ih nh3l3.3An ABAtLnLAuLnASheAt8

rxwMf N,x h3l3.3A8 p.un.Se 3SSSln,atuQOnometr cs Intelligent Lab Syst. edu
 N,GNNx

lx.x 5,geof IJ N,MLmr,oSA.tu EAA,SlnLn,86SIAM J. Matr. Anal. Appl. 23, r ,e
 I,x

.x l, C.n .e lx wu,f IJlx.3 A..Ount. .. 3Ur,unLp, nLLnApA3r 33r I,E2UuN
 Comput. Stat. Data Analy. 50, NJT,N5F

,x E2 ,Lmhk=IJ7,x3 ,n.BSAlkAp2Aoa,anAEA-iS,tnLn,n.IhSLnQUn3Ur A4E
 sn,hUp,A,hf,LunEnor,aogIna,6SIAM J. Matr. Anal. Appl. 28 ,r r x
 xEAgAeAIIfxBsci,tLn..iu,ae 9x96 MlttaiBf r Qld.ABAEni2sn,,Unl2EhALn,
 ,ImuAAEn,AtriaMM,tt,nA,uLnAuSIAM J. Matr. Anal. Appl. 30 ,5r , x
 AxEkupn,ae AxBo,„o f Jdk .3 9oAlSn.ltOPApp,e,u A,iShLn,HLpr,aS 2s,u
 E+A,CS,tnLn,6SIAM J. Matr. Anal. Appl. 29, N,dN.dx

hiSAuA,aAAuAeLnAp atmaBnthaAUhE28

lx.x 5,Ueo=I,J,b.x.3 A,haLAuAb.OStApAttn,nUnLi. 9bLAatn; Lp2 ,Ur ,ko
 roB 3SSMlb,LnlnA,uA,u LmMLmr,,UaB aA,tu EA-,O,snLn,NSIAM J. Matr
 Anal. Appl. 24uT7ITTx

9nxoet,Aur f IJx.lmA w,ME2u,, LpA,SLnSSLn,aI.Si ,I I,LumSA.2u6. AMS
 19 „Tr „x

huA,C, u.x-x.UhSu „-x ,auwxI,huu.n = Qd.x.si i.2LunaQs,us .E si ii9r un
 laAt,uroaBuQO SIAM J. Matr. Anal. Appl. 30, Nr,r NT,x

BxAsnS .e ,x,-x,n = IJlx.a,tu uoaBae LpASStAeAtt,. Lp2v2sL ,kLr ..QOQEM
 n,Ln,ahu,gA,uQ SIAM J. Matr. Anal. Appl. 30, N,Q,NNTx

hxA,C,au9xI x.x LAMxIA,LmhkAMie 9xA,L,narf Qd.x.cAaAuhAB .E i On-U
 loQ „ nhghkci 3uodt6,in. Alg. Applic. 430, I,Tr 5JJTx

,x 9Senae wsci = IJNxhAup,u,dLnlnA,ui,e ,SLn.. SlnA,aenLn,tu Lp2v2sLU\$no
 SnaAloA B 3SOu,bn,oln, ol2t,uuQ SIAM J. Matr. Anal. Appl. 32, N,r IvN,Ox
 AxExouLnfaJNapA I,B. . I,i ,I,Si ,IAt,uuQ Multil. Alg. 59u,5 1,Ox
 3xsL)r.A.o ae hxA,,a f IJN,bspuALrnwAtLaQ N3SSu,ln,I.t f-a-uAy9 s,H
 I,ab,6,in. Alg. Applic. 433, NITN5Ox

A69xSgme ,xL,ix f JNII,tL t,luh,SSA.CMA 2hLmuuQOQNN5. x

l2AneAb eA arLAt,Mhar,SioighAte AnrA.ioghApM,hm,AuSn3AeUAnph,Ln2Lss
 S,uthlen,pyASgknr uA MA.AAt8

,x, -x,n = IJJ.. snarhSou,aB9ghAtai,ghAt Aut8 3 B,undLn,aSSou,-puQn,9 Eo
 nrt., LnfA99f,LAuLn,a\$MBfm,O A,,OhL,Ln,aoel .As n.IUSLns2asEOr ni
 huAAttnaJ N5Ix

,x Vrf IJ..x9nrA,i.S,t., orAgs,SAuti ..ALunA,uaQ Symbolic Comput. 40, N61Ner ,N
 ,x Vn=IJ7,x.loB oae9nrAai,Sh,3toshSAuti ..2punAt,uuLmAUgLnun,LAr 22,hs
 hUt,nEgoe LmASrIS.onASAMthAAnIE2 aA. Symbolic Comput. 41, N61N5r T6

,x Vrf IJT,9nrAaiU,At f ai.un.apt aAt,utuQ Math. Anal. Applic. 325uN75N5(x
 E A.uLkunmaEwsLhu.AutJN.J. xAmPh. SAu 9nr A.i.Sh,At a2as,uNun.8NQ6,e i6

lmAuAuA uarA. uaBN,SSu,ln,Ln,apAat,uSOu,bmioLn,8u,SUAiE O,k2ui9r p,EsL,
 t,SiALmA12A8

,x Edpm,hkAuWEAI..u cae9Bcae2kgSAIO.x,a LmAtpOsN ..e 1aB f պսկռն.
 3SSu,ln,opn,.. -nrmAu,MeA,mutuSIAM J. Mat. Anal. Applic. r N6rN5,r 6
 9x5, entae h8x1ArdufI r J,klmA,nrm2uM,deA,AuLAlm,e AinsnL,AE8ir2,2hu,
 oae9IAALnfAnUnLn,auShu,AAAnearLp2 99f,r AuaLn,A,9u9a-9. 3,UsLn-su
 sSAAape snr,luh,AAttnaB,Sx ITJ,FTNr 6

lx.-par oe x - 6,UhSf IJN,xoQ,A 30OM,bni,Ln, - nrue9ua2at,usuQ SIAM J. Mat
 Anal. and Applic. 23, 5v. Jx

IL tiki ot Plas X12iT b2(m .com)ot ot Gao wnpA(4h .cWp)ra(5 i4
 Str ctur d Matrices in Mathematics, Computer Science, and Engineering I, BxgtnAitBiaeu.y
 31s. hMEiceA80M55N5 v
 945= ecti8e h8. IArki,IJ.I.v.,a pmAAtpl8B, 3SSMIT c.ipnlarmAM,,MeASAM2
 st.HpMcAAstM6SIAM J. Matrix Anal. Applic. 23, d75div
 ,4EAnipmIAME 9EaeAig,AJJi.v.EnCAatnlaiApnla crpAMr ,MeASAM1
 AAtniae riaB,,l, II.v. I21AelApd88I gpongAmr 1SMly6 Alg. Applic. 391y5Nyv
 svir8Mtr18aeA,Bianli8,IJN1..w,IAbaAatM8est.CApM9A AecaryyXiv:1010.0707/2.
 s3.pUabn,sn z3k8C4ani6P ptiplen8aeceian Us2aFtmiroborasndest,nsrSIAM
 J. Matrix Anal. Applic. 32, N.2GNiv

BiMd121pla,1BACApmlEnAgta,)MrAee

,v 9ei8iae wsii ,IJ.J.v.3 2A.pla,MCtgj8aIApmleMAESIpnapp)wAtpl1pnnaAiM
 li8B,,r, II j 3SSMIT c.ipnlarmAM6SIAM J. Matrix Anal. Applic. 31, d75div
 w.s' i8e nv,,c ,IJ(J..VI 1c,2Apla I ipylet la _MC1gSaadat II 1pn1aMISSMIT n2
 .ipclat l. aAtM6SIAM J. Sci. Comput. 32, 5ylr 55.5v
 Iv OrmpAEl,ipmihAMy hv,3viaesBja-h5A,,IJJ..v.EcAMAapnig. ll.Apmpla
 3.rlMcpn1pmwAtplab,,F y IF ly8SSMIT nCipclatM6Numer. Algorithms 51.
 NF(ix)

.HM1t OV== +x x n xx x = -• = T x n == xx xx .xx x-x+r = = = = a = ,
 4T 4r x'Θ u'= + 2r,6 = 0= 25D, p0= o2(yJ0= - i000p< a =,Ep OKf= x (= 0
 c=r = = Oy, a= = OedOoHEx a =,a= a= x0s= 00= p009.N. Nat. Acad. Sci.
 , te(d5T N8Tx

n.EAnipyilAM J.Jlv.EAAICSltpd8t.nrmAM,MeASAMavgIABAM.thiMPEA=anpnlat
 iae r 8cslAaA6SIAM J. Mat. Anal. Applic. W.2 bOpLa
 & a Sa = • a 880= 0= 225D, OE= Xf= o(-a 8y & Ae - p0000=A Ep0= x oXa = =
 WWWIt= 880a=0=pya, A=0=ex xSIAM J. Mat. Anal. Applic. 30, uJT(Jd5.
 Ix9v5gAnae AvfMpchJ(J. .iApLMn1pmplMipAr,MatmnM4eAMatlMy6.in. Alg
 Applic. 435, 7iμlydx
 9v3AiMEulEIagtyiae lv.v 5lgfJNx .3 sAi,iSgASpngipnla3SSMIQAY.cppn&r
 AialanAigA8tM6AAICSltpcn8tQnometr cs, 7Td7v
 9x3AiMEulEIa,it av/. 5ley ixeMISIJN(.xAi iSgaAarIMAnlMcipd,188Al.r
 SgAEapyChemomet. Intell. Lab Syst. 106 iμv7.
 AApt8eav.x 5lgei,IJ(I.v,a aAtM6SiMtpn82laaAripciaAplMpcatlyMPlNxiNiX
 BiM1t pltgt,M.iarc8r mcry,ecCAatc8tMpcatAIrtAe
 svvamnpA.I. v.EAatnptipMrlMC1gipnMVIiapi. IA8M.iqn3pnMEISty6phys. Rev.
 Lett. 69, 11751d77v
 ax.iABS1tA8ew.2.5mlMl.rBjJdTv.aA1rlM+leIA8SSMET nippclamiplM IaA2
 pnlatnacy E c H ltc8tJ. Complexity 23, 7.TG7iu
 uBA,AeApiae 9v9atMpttp8cBliJ.H.v.wMAiBpAAsMFAEnA8tnlai,npt.l. Implr t)
 sBEc1l18 En1clt8t6SIAM J. Sci. Comput. 31, 5Tii 5Ty,v
 ax.iABS1tA8e s1malJJ. .3 2A. sAmAgMpmA8tMASMAApqyQ Fourier Anal.
 Applic. 15, TJ7 IVI
 vBA,AeApE vBsiiltpti8iy iae 9v9atMpttmanBliJ.v onnaA8tMHSMAAtlMMlr 2
 gH6 Computing 85, μ7,Ndv
 8vAgAeApiae 9vatMp tpacBEi(j.aa,AMltt3SSMIT ngipM1lpngAarnlai3MMitty6
 Lin. Alg. Applic. 432, TJdd.
 nv/MCAetAB NY. .c1MiM8L4Mf gFAA1SIIcpn18aAatM7Q AM J. Mat. Anal.
 Applic. 31, + 2+2R4
 013< pae a J30006=8.00E38 0020-UDpe,0= A= 8 s0= 0a y cGE0X6= 0,
 .(ba = 08= ep0= a = 0-R = = XsNAM J. Sci. Comput. 34, 37d5GT(5v

IMnatcr,pappmAIMAen.AatclaicptSAe

/vwAgBniale I v9V1,gA8BcCSIJ.I.v 21AMcASAMiplMgAight1.crmAEmCA8tr98
 Pmc. Nat. Acad. Sci. 99(16), NII7 12N
 /vwAt,Bn8e I x9V1pgAaBcC,SJ.Jy.v.3grIMnpnM21.AMnA3aigtctc8 .nEndatnlar.6
 SIAM J. Sci. Cor put. 26, IN51N.2

A-conjugate	63
A-norm	29
Aa en's method	180, 9
Absolute value notation	165
Additive Schwarz	162
Adjacency set	62
Af ne space	63
Algebraic multiplicity	35
Algorithm	13
Algorithmic detail	xii
Angles between subspaces	393
Antidiagonal	28
Approximate inverse preconditioners	501
Approximate Newton method	391
Approximation of a matrix function	523
Arnoldi process	595
implicit restarting	533
k-step decomposition	53
rational	38, 50
Arnoldi vectors	36
Augmented system method	36
Back substitution	107
Backward error analysis	100–1
Backward stable	16
Backward successive over-relaxation	619
Balancing	32
Band algorithms	1
Cholesky	180
Gaussian elimination	178–9
Hessenberg LU	19, 778
triangular systems	1
Band matrix	15
data structures and	17
inverse	123
LU factorization and	167
pivotting and	189, 21
profile Cholesky	16
Bandwidth	16
barrier	5
Bartels-Stewart algorithm	398–400
Bar ic solution f r least squares	22
Ba is	61
eigenvector	400
Bauer-Fike theorem	378
BICGSTab method	67
Biconjugate gradient method	645
Bidiagonalization	52
Golub-Kahan	215
Householder	215
Paige-Saunders	56
upper trianglnlarizing f rst	25
Bidiagonal matrix	15
Big-Oh notation	12
Binary powering	37
Bisection methods	
f r Toeplitz eigenproblem	26
f r tridiagonal eigenproblem	45
BLAS	12f
Block algorithms	196f
Cholesky	180, 178
cyclic reduction	47
data reuse and	47
diagonalization	523, 37
Gaussian elimination	144–6
Gauss-Seidel	65
Jacobi method f r eigenvalues	412
Jacobi method f r linear systems	613
Lanczos	169
LU	1820, 167
LU with pivoting	116
multiple right-hand-side triangular	109–10
QR f ctiorization	201
recursive QR factorization	21
SPIKE	178, 196
T idiagonal	196
unsymmetric Lanczos	56
Block-cyclic distribution layout	50
and parallel LU	16
Block diagonal dominance	57
Block distribution layout	50
Block Householder	229
Block matrices	215
data reuse and	5
diagonal dominance of	19
Block tridiagonal systems	196f
Bordered linear systems	22
Bunch-Kaufman algorithm	12
Bunch-Parlett pivoting	91
Cache	46
Cancellation	97
Cannon's algorithm	6
Canonical correlation problem	330
Cauchy-like matrix	680
conversion to	680
Cauchy-Schwarz inequality	5
Cayley transform	68, 245
CGNE	357
CGNR	63
Characteristic polynomial	638
generalized eigenproblem and	405
Chebyshev polynomials	616
Chebyshev semi-iterative method	612
Cholesky f ctiorization	13
band	189
block	180
downlating and	238–41
gaxpy version	121
matrix square root and	13
profile	184
recursive block	123
stability of	135
Cholesky reduction of A –> B	500
Chordal metric	407
Circulant systems	202

- Cla sical Gram-Schmidt, 254
 Coarse grid role in multigrid, 673
 Collatz-Wielandt f rmula, 373
 Colon notation, 6, 16
 Column
 deletion or addition in QR, 235- 8
 major order, 45
 ordering in QR factorization, 279- 80
 orientation, 5, 107- 8
 partitioning, 6
 pivoting, 276- 7
 weighting in lea t squares, 306- 7
 Communication costs, 52f
 Compact WY transf rmation, 244
 Companion matrix, 382- 3
 Complete orthogonal decomposition, 283
 Complete pivoting, 131- 3
 Complex
 Givens transformation, 243- 4
 Householder transf rmation, 243
 matrices, 13
 matrix multiplication, 29
 QR factorization, 256
 SVD, 80
 Complexity of matrix inversion, 174
 Componentwise bounds, 92
 Compressed column representation, 598- 9
 Computation/communication ratio, 53f
 Condition estimation, 140, 142- 3, 436
 Condition of
 eigenvalues, 359- 60
 invariant subspaces, 360- 1
 lea t squares problem, 265- 7
 linear systems, 87- 8
 multiple eigenvalues, 360
 rectangular matrix, 248
 similarity transf rmation, 354
 Conuent Vandermonde matrix, 206
 Conf rmal partition, 23
 Congruence transformation, 449
 Conjugate
 directions, 633
 transpose, 13
 Conjugate gradient method, 625f
 derivation and properties, 629- 30, 633
 Hestenes-Stiefel version, 634- 5
 Lanczos version, 632
 practical, 635- 6
 pre-conditioned, 651- 2
 Conjugate gradient squared method, 646
 Consistent norms, 71
 Constrained lea t squares, 313- 4
 Contour integral and $f(A)$, 528- 9
 Convergence. See under particular algorithm
 Courant-Fischer minimax theorem, 441
 CP approximation, 735- 8
 Craig's method, 637
 Crawf rd number, 499
 Cross product, 70
 CroB -validation, 308
 CS decomposition, 84- 5, 503- 6
 hyperbolic, 344
 subset selection and, 294
 thin version, 84
 CUR decomposition, 576
 Curse of dimensionality, 741
 Cuthill-McKee ordering, 602- 4
 Cyclic Jacobi method, 480- 1
 Cyclic reduction, 197- 8
 Data lea t squares, 325
 Data motion overhead, 53
 Data reuse, 46- 8
 Data sparse, 154
 Davidson method, 593- 4
 Decompositions and factorizations
 Arnoldi, 580
 bidiagonal, 5
 block diagonal, 397- 9
 Cholesky, 163
 companion matrix, 382
 complete orthogonal, 283
 CS (general), 85
 CS (thin), 84
 generalized real Schur, 407
 generalized Schur, 406- 7
 Hessenberg, 378f
 Hessenberg-triangular, 408- 9
 Jordan, 354
 f f ln165 6
 fY n114, 128
 QR, 247
 QR (thin version), 248
 real Schur, 376
 Schur, 351
 singular value, 76
 singular value (thin), 80
 symmetric Schur, 440
 tridiagonal, 458- 9
 Decoupling in eigenproblem, 349- 50
 Defective eigenvalue, 66, 353
 Def ating subspace, 404
 Def ation and
 bidiagonal f rm, 490
 Hessenberg triangular f rm, 409- 10
 QR algorithm, 385
 Denman-Beavers iteration, 539- 40
 Departure f om normality, 351
 Derogatory matrix, 383
 Determinant, 66, 348
 Gaussian elimination and, 114
 and singularity, 89
 Vandermonde matrix, 206
 Diagonal dominance, 154- 6, 615
 block, 197
 fY and, 156
 Diagonal matrix, 18
 Diagonal pivoting method, 191- 2
 Diagonal plus rank 1, 469- 71
 Diagonalizable, 67, 353
 Dif erentiation of matrices, 67
 Dimension, 64
 Direct methods, 598f
 Dirichlet end condition, 222
 Discrete cosine transf rm (DCT), 39
 Discrete Fourier transf rm (DFT), 33- 6
 circulant matrices and, 221- 2
 factorizations and, 41
 matrix, 34
 Discrete Poisson problem
 1-dimensional, 222- 4
 2-dimensional, 224- 31
 Discrete sine transf rm (DST), 39
 Displacement rank, 682
 Distance between subspaces, 82
 Distributed memory model, 57
 Divide-and-conquer algorithms
 cyclic reduction, 197- 8
 Stra sen, 30- 1
 tridiagonal eigenvalue, 471- 3

- Domain decomposition, 662–5
Dominant
 eigenvalue, 366
 invariant subspace, 368
Dot product, 4, 10
Dot product roundoff, 98
Double implicit shift, 388
Doubling formulae, 526
Downgrading Cholesky, 338–41
Drazin inverse, 356
Durbin's algorithm, 210
- Eckhart-Young theorem**, 79
Eigenproblem
 diagonal plus rank-1, 469–71
 generalized, 405f, 497f
 inverse, 473–4
 orthogonal Hessenberg matrix, 703–4
 symmetric, 439f
Toeplitz, 214–6
 unsymmetric, 347f
- Eigenstystem**
 fa t, 219
- Eigenvalue decompositions**
 Jordan, 354
 Schur, 351
- Eigenvalues**
 algebraic multiplicity, 353
 characteristic polynomial and, 348
 computing selected, 453
 defective, 66
 determinant and, 348
 dominant, 366
 generalized, 405
 geometric multiplicity, 353
 ordering in Schur form, 351, 396–7
 orthogonal Hessenberg, 703–4
 relative perturbation, 365
 repeated, 360
 sensitivity (symmetric case), 441–3
 sensitivity (unsymmetric case), 359–60
 singular values and, 355
 Sturm sequence and, 468
 symmetric tridiagonal, 467f
 trace, 348
 unstable, 363
- Eigenvector**, 67
 basis, 400
 dominant, 366
 left, 349
 matrix and condition, 354
 perturbation, 361–2
 right, 349
- Elementary Hermitian matrices.**
 See **Householder matrix**
- Elementary transformations.** See
Gauss transformations
- Equality constrained least squares**, 315–7
- Equilibration**, 139
- Equilibrium systems**, 192–3
- Equivalence of vector norms**, 69
- Error**
 absolute, 69
 damping in multigrid, 622–3
 relative, 70
 roundoff, 96–102
- Error analysis**
 backward, 100
 forward, 100
- Euclidean matrix norm.** See
- Frobenius matrix norm
Exchange permutation matrix, 20
Explicit shift in QR algorithm
 symmetric case, 461
 unsymmetric case, 385–8
Exponential of matrix, 530–6
- Factored form representation**, 237–8
Factorization. See **Decompositions and factorizations**
- Fa t methods**
 cosine transform, 36f
 eigensystem, 219, 228–31
 Fourier transform, 33f
 Givens QR, 245
 Poisson solver, 226–7
 sine transform, 36
- Field of values**, 349
- Fine grid role in multigrid**, 673
- Floating point**
 fl, 96
 fundamental axiom, 96
 maxims, 96–7
 normalized, 94
 numbers, 93
 storage of matrix, 97–8
- Flop**, 12
- Flopcounts**, 12, 16
 for square system methods, 298
- F-norm**, 71
- Forward error analysis**, 100
- Forward substitution**, 106
- Francis QR step**, 390
- Frechet derivative**, 521
- Frobenius matrix norm**, 71
- Frontal methods**, 610
- Full multigrid**, 678
- Function of matrix**, 513f
 eigenvectors and, 517–8
 Schur decomposition and, 518–20
 Taylor series and, 524–6
- Gauss-Jordan transformations**, 121
- Gauss-Radau rule**, 560–1
- Gauss rules**, 557–9
- Gauss-Seidel iteration**, 611–2
 block, 613
 Poisson equation and, 617
 positive definite systems and, 615
- Gauss transformations**, 112–3
- Gaussian elimination**, 111f
 banded version, 176–9
 block version, 144–5
 complete pivoting and, 131–2
 gaxpy version, 117
 outer product version, 116
 partial pivoting and, 127
 rook pivoting and, 133
 roundoff error and, 122–3
 tournament pivoting and, 150
- Gaxpy**, 5
 blocked, 25
- Gaxpy-rich algorithms**
 Cholesky, 164
 Gaussian elimination, 129–30
 LDL^T , 157–8
- Gaxpy vs. outer product**, 45
- Generalized eigenproblem**, 405f
- Generalized eigenvalues**, 405
 sensitivity, 407

- Generalized least squares, 356
 Generalized Schur decomposition, 467
 computation of, 523
 Generalized singular vectors, 52
 Generalized SVD, 390, 512
 constrained least squares and, 367
 Generalized Sylvester equation, 417
 Generator representation, 33
 Geometric multiplicity, 33
 Gershgorin theorem, 35, 42
 Ghost eigenvalues, 56
 givens, 20
 Givens QR, 223
 parallel, 25
 Givens rotations, 2942
 complex, 234
 fast, 25
 rank-revealing decompositions and, 282
 square-root free, 26
 Global memory, 5
 GMRES, 624
 m-step, 64
 preconditioned, 623
 Golub-Kahan
 bidiagonalization, 513
 SVD step, 49
 Gram-Schmidt
 classical, 24
 modified, 245
 Graph, 62
 Graphs and sparsity, 612
 Growth in Gaussian elimination, 102
 Haar wavelet transform, 4f
 factorization, 41
 Hadamard product, 70
 Hamiltonian matrix, 240
 eigenvalue problem, 451
 Hankel-like, 689
 Hermitian matrix, 8
 Hessenberg f rm, 15
 Arnoldi process and, 598
 Householder reduction to, 389
 inverse iteration and, 35
 properties, 312
 QR factorization and, 234
 QR iteration and, 356
 unreduced, 31
 Hessenberg QR step, 378
 Hessenberg systems, 19
 LU and, 19
 Hessenberg-triangular f rm, 489
 Hierarchical memory, 46
 Hierarchical rank structure, 72
 Higher-order SVD, 723
 truncated, 71
 Holder inequality, 9
 Horner algorithm, 367
 house, 25
 Householder
 bidiagonalization, 215
 tridiagonalization, 215
 Householder matrix, 218
 complex, 223
 operations with, 257
 Hyperbolic
 CS decomposition, 34
 rotations, 39
 transformations, 39
 Identity matrix, 9
 Ill-conditioned matrix, 88
 IEEE arithmetic, 91
 (ena) 13
 Implicit Q theorem
 symmetric matrix version, 40
 unsymmetric matrix version, 38
 Implicit symmetric QR step with
 Wilkinson Shift, 412
 Implicitly restarted Arnoldi
 method, 513
 Incomplete block preconditioners, 676
 Incomplete Cholesky, 376
 Indefinite least squares, 34
 Indefinite symmetric matrix, 19
 Indefinite systems, 63941
 Independence, 61
 Inertia of symmetric matrix, 48
 inf, 9
 Integrating $f(A)$, 578
 Interchange permutation, 126
 Interlacing property
 singular values, 437
 symmetric eigenvalues, 443
 Intersection
 nullspaces, 339
 subspaces, 31
 Invariant subspace
 approximate, 468
 dominant, 38
 perturbation of (symmetric case), 485
 perturbation of (unsymmetric case), 31
 Schur vectors and, 31
 Inverse, 9
 band matrices and, 123
 Inverse eigenvalue problems, 434
 Inverse error analysis. See
 Backward error analysis
 Inverse fast transform
 cosine, 228
 Fourier, 228
 sine, 228
 Inverse iteration
 generalized eigenproblem, 444
 symmetric case, 435
 unsymmetric case, 345
 Inverse low-rank perturbation, 65
 Inverse of matrix,
 perturbation of, 74
 Toeplitz case, 223
 Inverse orthogonal iteration, 34
 Inverse power method, 34
 Inverse scaling and squaring, 52
 Irreducible, 33
 Iteration matrix, 63
 Iterative improvement
 fixed precision and, 10
 least squares, 239, 222
 linear systems, 6140
 Iterative methods,
 Jacobi iteration for the SVD, 423
 Jacobi iteration for symmetric
 eigenproblem, 46
 classical, 498
 cyclic, 48
 error, 491
 parallel version, 423
 Jacobi method for linear systems,
 block version, 63

- diagonal dominance and, 65
 preconditioning with, 63
 Jacobi orthogonal correction method, 513
 Jacobi rotations, 47
 Jacobi-Davidson method, 515
 Jordan blocks, 402
 Jordan decomposition, 34
 computation of, 402
 matrix functions and, 54, 523
 Kaniel-Paige-Saad theory, 524
 Khatri-Rao product, 70
 Kogbetiantz algorithm, 36
 Kronecker product, 27, 708
 basic properties, 70
 multiple, 70
 SVD, 724
 Kronecker structure, 48
 Krylov
 matrix, 49
 subspaces, 38
 Krylov-Schur algorithm, 51
 Krylov subspace methods
 biconjugate gradients, 65
 CG (conjugate gradients), 62f
 CGNE (conjugate gradient normal equation error), 678
 CGNR (conjugate gradient normal equation residual), 678
 CGS (conjugate gradient squared), 66
 general linear systems and, 59
 GMRES (general minimum residual), 625
 MINRES (minimum residual), 634
 QMR (quasi-minimum residual), 647
 SYMMLQ, 69
 Krylov subspace methods f r
 general linear systems, 636–7, 627
 least squares, 612
 singular values, 548
 symmetric eigenproblem, 565, 527
 symmetric indefinite systems, 694
 symmetric positive definite systems, 625–39
 unsymmetric eigenproblem, 598
 Lagrange multipliers, 313
 Lanczos tridiagonalization, 56
 block version, 569
 complete reorthogonalization and, 515
 conjugate gradients and, 683
 convergence of, 524
 Gauss quadrature and, 56
 interior eigenvalues and, 553–4
 orthogonality loss, 51
 power method and, 545
 practical, 562f
 Ritz approximation and, 551–2
 roundoff and, 584
 selective orthogonalization and, 556
 s-step, 59
 termination of, 59
 unsymmetric, 59, 587
 Lanczos vectors, 59
 LDI, 168
 conjugate gradients and, 61
 with pivoting, 166
 Leading principal submatrix, 21
 Least squares methods, f opcounts f r, 23
 Least squares problem
 basic solution to, 22
 cross-validation and, 38
 equality constraints and, 315–7
 full rank, 356
 generalized, 34
 indefinite, 289
 iterative improvement, 289
 Khatri-Rao product and, 289
 minimum norm solution to, 289
 quadratic inequality constraint, 335
 rank deficient, 289
 residual vs. column independence, 256
 sensitivity of, 287
 solution set of, 28
 solution via Householder QR, 234
 sparse, 607–8, 642
 SVD and, 28
 Least squares solution using
 LSQR, 612
 modified Gram-Schmidt, 264–5
 normal equations, 453
 QR factorization, 254
 seminormal equations, 67
 SVD, 28
 Left eigenvector, 39
 Left-looking, 117
 Levels of linear algebra, 12
 Level 3 fraction, 19
 block Cholesky, 110
 block LU, 110
 Hessenberg reduction, 30
 Levinson algorithm, 21
 Linear equation sensitivity, 61
 Linear independence, 61
 Linearization, 415–6
 Load balancing, 50
 Local memory, 50
 Local program, 50
 Log of a matrix, 541–2
 Look-ahead, 27, 367
 Loop reordering, 9
 Loss of orthogonality
 Gram-Schmidt, 24
 Lanczos, 51
 Low-rank approximation
 randomized, 567
 SVD, 9
 LR iteration, 30
 LSMR, 62
 LSQR, 612
 LU factorization, 111
 band, 117
 block, 167
 Cauchy-like, 656
 determinant and, 14
 diagonal dominance and, 15
 differentiation of, 110
 existence of, 114
 gaxpy version, 117
 growth factor and, 111
 Hessenberg, 119
 mentality, 134
 outer product version, 116
 partial pivoting and, 128
 rectangular matrices and, 118
 roundoff and, 123
 semiseparable, 657
 sparse, 689
 Machine precision, 95
 Markov chain, 34
 Markowitz pivoting, 69

- MATLAB, xix
 Matrix functions, 513ff
 integrating, 527–8
 Jordan decomposition and, 514–5
 polynomial evaluation and, 526–7
 sensitivity of, 520–1
 Matrix multiplication, 2, 8ff
 blocked, 26
 Cannon's algorithm, 60–1
 distributed memory, 50ff
 dot product version, 10
 memory hierarchy and, 47
 outer product version, 11
 parallel, 49ff
 saxpy version, 11
 Strassen, 30–1
 tensor contractions and, 726–7
 Matrix norms, 71–3
 consistency, 71
 Frobenius, 71
 relations between, 72–3
 subordinate, 72
 Matrix-vector products, 33ff
 blocked, 25
 Memory hierarchy, 46
 Minimax theorem for
 singular values, 487
 symmetric eigenvalues, 441
 Minimum degree ordering, 604–5
 Minimum singular value, 78
 MINRES, 639–41
 Mixed packed format, 171
 Mixed precision, 140
 Modal product, 727–8
 Modal unfoldings, 723
 Modified Gram-Schmidt, 254–5
 and least squares, 264–5
 Modified LR algorithm, 392
 Moore-Penrose conditions, 290
 Multigrid, 670ff
 Multilinear product, 728–9
 Multiple eigenvalues,
 matrix functions and, 520
 unreduced Hessenberg matrices and, 382
 Multiple-right-hand-side problem, 108
 Multiplicative Schwarz, 664
 Multipliers in Gauss transformations, 112

 NaN, 95
 Nearness to
 Kronecker product, 714–5
 singularity, 88
 skew-hermitian, 449
 Nested-dissection ordering, 605–6
 Netlib, xix
 Neumann end condition, 222
 Newton method for Toeplitz eigenvalue, 215
 Newton-Schultz iteration, 538
 nnz, 599
 Node degree, 602
 Nondiagonality matrices, 383
 Nongeneric total least squares, 324
 Nonsingular matrix, 65
 Norm
 matrix, 71–3
 vector, 68
 Normal equations, 262–3, 268
 Normal matrix, 351
 departure from, 351
 Normwise-near preconditioners, 654

 null, 64
 Nullity theorem, 185
 Nullspace, 64
 intersection of, 328–9
 Numerical radius, 349
 Numerical range, 349
 Numerical rank
 least squares and, 291
 QR with column pivoting and, 278–9
 SVD and, 275–6

 off, 477
 Ordering eigenvalues, 396–7
 Ordering for sparse matrices
 Cuthill-McKee, 602–4
 minimum degree, 604–6
 nested dissection, 605–7
 Orthogonal
 complement, 65
 invariance, 75
 matrix, 66, 234
 Procrustes problem, 327–8
 projection, 82
 symplectic matrix, 420
 vectors, 65
 Orthogonal iteration
 symmetric, 454–5, 464–5
 unsymmetric, 367–8, 370–3
 Orthogonal matrix representations
 factored form, 237–8
 Givens rotations, 242
 WY block form, 238–9
 Orthogonality between subspaces, 65
 Orthonormal basis computation, 247
 Outer product, 7
 Gaussian elimination and, 115
 LDL^T and, 166
 sparse, 599–600
 between tensors, 724
 versus gaxpy, 45
 Overdetermined system, 260

 Packed format, 171
 Padé approximation, 530–1
 PageRank, 374
 Parallel computation
 divide and conquer eigensolver, 472–3
 Givens QR, 257
 Jacobi's eigenvalue method, 482–3
 LU, 144ff
 matrix multiplication, 49ff
 Parlett-Reid method, 187–8
 Parlett-Schur method, 519
 block version, 520
 Partitioning
 conformable, 23
 matrix, 5–6
 Pencils, equivalence of, 406
 Perfect shuffle permutation, 20, 460, 711–2
 Periodic end conditions, 222
 Permutation matrices, 19ff
 Perron-Frobenius theorem, 373
 Perron's theorem, 373
 Persymmetric matrix, 208
 Perturbation results
 eigenvalues (symmetric case), 441–3
 eigenvalues (unsymmetric case), 357–60
 eigenvectors (symmetric case), 445–6
 eigenvectors (unsymmetric case), 361–2
 generalized eigenvalue, 407

- invariant subspaces (symmetric ca e), 444–5
 invariant subspaces (unsymmetric ca e), 361
 lea t squares problem, 265–7
 linear equation problem, 82–92
 singular subspace pair, 488
 singular values, 487
 underdetermined systems, 301
Pipelining, 43
Pivoting
 Aa en's method and, 190
 Bunch-Kaufman, 192
 Bunch-Parlett, 191
 Cauchy-like and, 686–7
 column, 276–8
 complete, 131–2
 LU and, 125f
 Markowitz, 609
 partial, 127
 QR and, 279–80
 rook, 133
 symmetric matrices and, 165–6
 tournament, 150
Plane rotations. See Givens rotations
p-norms, 71
 minimization in, 260
Point, line, plane problems, 269–271
Pointwise operations, 3
Polar decomposition, 328, 540–1
Polynomial approximation and GMRES, 644
Polynomial eigenvalue problem, 414–7
Polynomial interpolation, Vandermonde
 systems and, 203–4
Polynomial preconditioner, 655–6
Positive definite systems, 159f
 Gauss-Seidel and, 615–6
 LDL^T and, 165f
 properties of, 159–61
 unsymmetric, 161–3
Positive matrix, 373
Positive semidefinite matrix, 159
Post-smoothing in multigrid, 675
Power method, 365f
 error estimation in, 367
 symmetric ca e, 451–2
Power series of matrix, 524
Powers of a matrix, 527
Preconditioned
 conjugate gradient method, 651–2, 656f
 GMRES, 652–3
Preconditioners, 598
 approximate inverse, 654–5
 domain decomposition, 662–5
 incomplete block, 660–1
 incomplete Cholesky, 657–60
 Jacobi and SSOR, 653
 normwise-near, 654
 polynomial, 655
 saddle point, 661
 Pre-smoothing role in multigrid, 675
Principal angles and vectors, 329–31
Principal square root, 539
Principal submatrix, 24
Probability vector, 373
Procrustes problem, 327–8
Product eigenvalue problem, 423–5
Product SVD problem, 507
Prof le, 602
 Cholesky, 184
 indices, 184, 602
Projections, 82
Prolongation matrix, 673
Pseudo-eigenvalue, 428
Pseudoinverse, 290, 296
Pseudospectra, 426f
 computing plots, 433–4
 matrix exponential and, 533–4
 properties, 431–3
Pseudospectral abscissa, 434–5
Pseudospectral radius, 434–5
QMR, 647
QR algorithm f r eigenvalues
 Hessenberg f rm and 377–8
 shifts and, 385f
 symmetric version, 456f
 tridiagonal f rm and, 460
 unsymmetric version, 391ff
 Wilkinson shift, 462–3
QR factorization, 246f
 block Householder, 250–1
 block recursive, 251
 classical Gram-Schmidt and, 254
 column pivoting and, 276–8
 complex, 256
 Givens computation of, 252–3
 Hessenberg matrices and, 253–4
 Householder computation of, 248–9
 lea t square problem and, 263–4
 modified Gram-Schmidt and, 254–5
 properties of, 246–7
 range space and, 247
 rank of matrix and, 274
 sparse, 606–8
 square systems and, 298–9
 thin version, 248
 tridiagonal matrix and, 460
 underdetermined systems and, 300
 updating, 335–8
Quadratic eigenvalue problem, 507–8
Quadratically constrained lea t squares, 314–5
Quasidinite matrix, 194
Qua iseparable matrix, 693
Quotient SVD, 507
QZ algorithm, 412–3
 step, 411–2
ran, 64
Randomization, 576–7
Range of a matrix,
 orthonormal basis and, 247
Rank of matrix, 64
 QR factorization and, 278–9
 SVD and, 275–6
Rank-deficient LS problem, 288f
 breakdown of QR method, 264
Rank-revealing decomposition, 280–3
Rank-structured matrices, 691f
Rayleigh quotient iteration, 453–4
 QR algorithm and, 464
 symmetric-definite pencils and, 501
R-bidiagonalization, 285
 , 13
Real Schur decomposition, 376–7
 generalized, 407
 ordering in, 396–7
Rectangular LU, 118
Recursive algorithms
 block Cholesky, 169
 Strassen, 30–1
Reducible, 373

- Regularized least squares, 307ff
 Regularized total least squares, 324
 Relative error, 69
 Relaxation parameter, 619–20
 Reorthogonalization
 complete, 564
 selective, 565
 Representation, 681–2
 generator, 693
 Givens, 697–8
 quasiseparable, 694
 28, 711
 and Kronecker product, 28
 Residuals vs. accuracy, 138
 Restarting
 Arnoldi method and, 581–2
 block Lanczos and, 569
 GMRES and, 644
 Restricted generalized SVD, 507
 Restricted total least squares, 324
 Restriction matrix, 673
 Riccati equation, 422–3
 Ridge regression, 307–8
 Riemann-Stieltjes integral, 556–7
 Right eigenvector, 349
 Right-looking, 117
 Ritz acceleration, orthogonal
 iteration and, 464–5
 Ritz approximation
 eigenvalues, 551–2
 singular values, 573
 Rook pivoting, 133
 Rotation of subspaces, 327–8
 Rotation plus rank-1 (ROPR), 332
 Rounding errors. See under particular
 algorithm
 Roundoff error analysis, 100
 dot product, 98–9
 Wilkinson quote, 99
 Row orientation, 5
 Row partition, 6
 Row scaling, 139
 Row weighting in LS problem, 304–5
 Saddle point preconditioners, 661
 Saxpy, 4, 11
 Scaling, linear systems and, 138–9
 Scaling and squaring for $\exp(A)$, 531
 Schur complement, 118–9, 663
 Schur decomposition, 67, 350–1
 generalized, 406–7
 matrix functions and, 523–4
 normal matrices and, 351
 real matrices and, 376–7
 symmetric matrices and, 440
 2-by-2 symmetric, 478
 Schur vectors, 351
 Secular equations, 313–4
 Selective reorthogonalization, 565–6
 Semidefinite systems, 167–8
 Semiseparable
 eigenvalue problem, 703–4
 LU factorization, 695–8
 matrix, 682
 plus diagonal, 694
 QR factorization, 698–701
 Sensitivity. See Perturbation results
 sep
 symmetric matrices and, 444
 unsymmetric matrices and, 360
 Shared-memory systems, 54–6
 Shared-memory traffic, 55–6
 Sherman-Morrison formula, 65
 Sherman-Morrison-Woodbury formula, 65
 Shifts in
 QZ algorithm, 411
 SVD algorithm, 489
 symmetric QR algorithm, 461–2
 unsymmetric QR algorithm, 385–90
 Sign function, 536–8
 Similar matrices, 67, 349
 Similarity transformation, 349
 condition of, 354
 nonunitary, 352–4
 Simpson's rule, 528
 Simultaneous diagonalization, 499
 Simultaneous iteration. See orthogonal iteration
 Sine of matrix, 526
 Singular matrix, 65
 Singular subspace pair, 488
 Singular value decomposition (SVD), 76–80
 algorithm for, 488–92
 constrained least squares and, 313–4
 generalized, 309–10
 geometry of, 77
 higher-order, 732–3
 Jacobi algorithm for, 492–3
 Lanczos method for, 571ff
 linear systems and, 87–8
 minimum-norm least squares solution, 288–9
 nullspace and, 78
 numerical rank and, 275–6
 perturbation of, 487–8
 principal angles and, 329–31
 projections and, 82
 pseudo-inverse and, 290
 rank of matrix and, 78
 ridge regression and, 307–8
 subset selection and, 293–6
 subspace intersection and, 331
 subspace rotation and, 327–8
 symmetric eigenproblem and, 486
 total least squares and, 321–2
 truncated, 291
 Singular values, 76
 condition and, 88
 eigenvalues and, 355
 interlacing property, 48
 minimax characterization, 487
 perturbation of, 487–8
 range and nullspace, 78
 rank and, 78
 smallest, 279–80
 Singular vectors, 76
 Skeel condition number, 91
 and iterative improvement, 140
 Skew-Hamiltonian matrix 420
 Skew-Hermitian matrix, 18
 Skew-symmetric matrix, 18
 span, 64
 Sparse factorization challenges
 Cholesky, 601
 LU, 609
 QR, 607
 Sparsity, 154
 graphs and, 601–2
 Spectral abscissa, 349
 Spectral radius, 349, 427, 614
 Spectrum of matrix, 348
 Speed-up, 53–4

- SPIKE framework, 199–201
 Splitting, 613
 Square root of a matrix, 163
 s-step Lanczos, 569
 Stable algorithm, 136
 Stable matrix, 436
 Steepest descent method, 626–7
 Stieltjes matrix, 658
 Strassen method, 30–1
 error analysis and, 101–2
 Strictly diagonally dominant, 155
 Stride, 45
 Structured rank, 691f
 types of, 702
 Sturm sequence property, 468–9
 Submatrix, 24
 Subnormal floating point number, 95
 Subordinate norm, 72
 Subset selection, 293–5
 using QR with column pivoting, 293
 Subspace, 64
 angles between, 329–31
 deflating, 414
 distance between, 82–3, 331
 dominant, 368
 intersection, 331
 invariant, 349
 nullspace intersection, 328–9
 orthogonal projections onto, 82
 rotation of, 327–8
 Successive over-relaxation (SOR), 619
 Sweep, 480
 Sylvester equation, 398
 generalized, 417
 Sylvester law of inertia, 448
 Sylvester map, 682
 Symmetric-definite eigenproblem, 497–501
 Symmetric eigenproblem, 439f
 sparse methods, 546f
 Symmetric indefinite methods
 Aaren, 188–90
 Diagonal pivoting, 191–2
 Parlett-Reid, 187–8
 Symmetric matrix, 18
 Symmetric pivoting, 165
 Symmetric positive definite systems, 163f
 Symmetric semidefinite properties, 167–8
 Symmetric successive over-relaxation,
 (SSOR), 620
 SYMLQ, 641
 Symplectic matrix, 29, 420
 symSchur, 478
- Taylor approximation of eA , 530
 Taylor series, matrix functions and, 515–7
 Tensor
 contractions, 726f
 eigenvalues, 740–1
 networks, 741
 notation, 721
 rank, 738–9
 rank 1, 725
 singular values, 739–40
 train, 741–3
 transpose, 722–3
 unfolding, 720
 Thin CS decomposition, 84
 Thin QR factorization, 248
 Thin SVD, 80
 Threshold Jacobi, 483
- Tikhonov regularization, 309
 Toeplitz-like matrix, 688
 Toeplitz matrix methods, classical, 208f
 Toroidal network, 58
 Total least squares, 320f
 geometry, 323–4
 Tournament pivoting, 150
 Trace, 348–9
 tr, 348
 Trace-min method, 595
 Tracy-Singh product, 709
 Transition probability matrix, 374
 Transpose, 2, 711–2
 French algorithm, 213
 Treppeniteration, 369
 Triangular matrices,
 multiplication between, 15
 unit, 110
 Triangular systems, 106–11
 band, 177–8
 nonsquare, 109–10
 roundoff and, 124–5
 semiseparable, 694–5
 Tridiagonalization,
 connection to bidiagonalization, 574
 Householder, 458–60
 Krylov subspaces and, 459–60
 Lanczos, 548–9
 Tridiagonal matrices, 15, 223–4
 QR algorithm and, 460–4
 Tridiagonal systems, 180–1
 Truncated
 higher-order SVD, 734
 SVD, 291
 total least squares, 324
 Tucker approximation problem, 734–5
- ULV decomposition, 282–3
 ULV updating, 341–3
 Underdetermined systems, 134, 299–301
 Undirected graph, 602
 Unfolding, 723–4
 Unit roundoff, 96
 Unit stride, 45
 Unit vector, 69
 Unitary matrix, 80
 Unreduced Hessenberg matrices, 381
 Unreduced tridiagonal matrices, 459
 Unstable eigenvalue, 363
 Unsymmetric
 eigenproblem, 347f
 Lanczos method, 584–7
 positive definite systems, 161–3
 Toeplitz systems, 216–7
 Updating
 Cholesky, 338–41
 QR factorization, 334–8
 ULV, 341–3
 UTV, 282
- Vandermonde systems, 203f
 confluent, 206
 V-cycle, 677–8
 vec, 28, 710–11
 for tensors, 722
- Vector
 computing, 43f
 loads and stores, 43
 norms, 68
 operations, 3–4, 44

- processing, 43
- Vectorization, tridiagonal system solving and, 181
- Weighted Jacobi iteration, 672–3
- Weighting least squares problems
 - column, 306–7
 - row, 304–5
- See also Scaling
- Well-conditioned matrix, 88
- Wielandt-Hofman theorem
 - eigenvalues, 442
 - singular values, 487
- Wilkinson shift, 462–3
- Work
 - least squares methods and, 293
 - linear system methods and, 298
 - SVD and, 493
- WY representation, 238–9
 - compact version, 244
- Yule-Walker problem, 201–10