

Constructing the PLU factorization

Set

$$A = \begin{bmatrix} a_{1,1} & \dots & a_{1,n} \\ \vdots & & \vdots \\ a_{n,1} & \dots & a_{n,n} \end{bmatrix}.$$

For $n = 1$

$$P = L = [1] \quad \text{and} \quad U = [a_{1,1}]$$

Linear equation solving

We want to solve

$$Ax = b$$

for a nonsingular $n \times n$ matrix A and an n -vector b

Gaussian elimination consists in computing a factorization

$$A = PLU$$

with P a permutation, L a unit lower triangular, and U an upper triangular

Allows to solve $Ax = b$ by solving the simpler equations

- ① $Pz = b$ (permute the entries of b)
- ② $Ly = z$ (forward substitution)
- ③ $Ux = y$ (backwards substitution)

Constructing the PLU factorization (cont.)

Let $n \geq 2$ and choose k such that $a_{k,1} \neq 0$

Gaussian elimination with partial pivoting (GEPP)

k such that $|a_{k,1}|$ is maximal

Swap rows 1 and k premultiplying by the permutation matrix

$$P_1 = \begin{bmatrix} 0 & 0 & \dots & 0 & 1 & 0 & \dots & 0 \\ 0 & 1 & & & 0 & & & \\ \vdots & & \ddots & & \vdots & & & \\ 0 & & & 1 & 0 & & & \\ 1 & 0 & \dots & 0 & 0 & & & \\ 0 & & & & & 1 & & \\ \vdots & & & & & & \ddots & \\ 0 & & & & & & & 1 \end{bmatrix}.$$

Consider the 2×2 -block

$$P_1^T A = \begin{bmatrix} a_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix}$$

Constructing the PLU factorization (cont.)

Then

$$\begin{bmatrix} a_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ L_{2,1} & \mathbb{1}_{n-1} \end{bmatrix} \begin{bmatrix} u_{1,1} & U_{1,2} \\ 0 & \tilde{A}_{2,2} \end{bmatrix}$$

with

$$u_1 = a_{1,1}, \quad L_{2,1} = a_{1,1}^{-1} A_{2,1}, \quad U_{1,2} = A_{1,2} \quad \text{and} \quad \tilde{A}_{2,2} = A_{2,2} - L_{2,1} U_{1,2}$$

The $(n-1) \times (n-1)$ matrix $\tilde{A}_{2,2}$ is the *Schur complement*

By the inductive hypothesis (case $n - 1$):

$$\tilde{A}_{2,2} = \tilde{P} \tilde{L} \tilde{U}$$



Complete pivoting

Gaussian elimination with complete pivoting (GECP): takes

 $|a_{k,l}|$ maximal among all the entries

↪ swaps the rows 1 and k , and the columns 1 and l

Gives a factorization

$$A = P_1 L U P_2$$

with P_1 and P_2 permutations

Can be more *numerically stable* but is also *more expensive* (in terms of speed)



Constructing the PLU factorization (cont.)

Then

$$\begin{aligned} P_1^T A &= \begin{bmatrix} 1 & 0 \\ L_{2,1} & \mathbb{1}_{n-1} \end{bmatrix} \begin{bmatrix} u_{1,1} & U_{1,2} \\ 0 & \tilde{P} \tilde{L} \tilde{U} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & \tilde{P} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \tilde{P}^T L_{2,1} & \tilde{L} \end{bmatrix} \begin{bmatrix} u_{1,1} & U_{1,2} \\ 0 & \tilde{U} \end{bmatrix} \end{aligned}$$

Hence $A = P L U$ with

$$P = P_1 \begin{bmatrix} 1 & \mathbf{0} \\ \mathbf{0} & \tilde{P} \end{bmatrix}, \quad L = \begin{bmatrix} 1 & \mathbf{0} \\ \tilde{P}^T L_{2,1} & \tilde{L} \end{bmatrix} \quad \text{and} \quad U = \begin{bmatrix} u_{1,1} & U_{1,2} \\ \mathbf{0} & \tilde{U} \end{bmatrix}.$$

$$\text{GEPP} \rightsquigarrow L = [l_{i,j}]_{i,j} \text{ with } |l_{i,j}| \leq 1 \text{ for all } i,j$$


The GEPP algorithm

for $i = 1, \dots, n - 1$

swap row k and row i of A and L for k such that

$$|a_{k,i}| = \max_{i \leq p \leq n} |a_{p,i}|$$

for $j = i + 1, \dots, n$ (compute column i of L)

$$l_{j,i} \leftarrow \frac{a_{j,i}}{a_{i,i}}$$

for $j = 1, \dots, n$ (compute row i of U)

$$u_{i,j} \leftarrow a_{i,j}$$

for $j, k = i + 1, \dots, n$ (update $A_{2,2}$)

$$u_{j,k} \leftarrow a_{j,k} - l_{j,i} u_{i,k}$$



Modified GEPP

Useful remark: the i -th column of A only used to compute the i -th column of L , and the i -th row of A only used to compute the i -th row of U

```

for  $i = 1, \dots, n-1$ 
    swap row  $k$  and row  $i$  of  $A$  and  $L$  for  $k$  such that
     $|a_{k,i}| = \max_{i \leq p \leq n} |a_{p,i}|$ 
    for  $j = i+1, \dots, n$ 
         $l_{j,i} \leftarrow \frac{a_{j,i}}{a_{i,i}}$  (replace by  $a_{j,i}$ )
    for  $j = 1, \dots, n$ 
         $u_{i,j} \leftarrow a_{i,j}$ 
    for  $j, k = i+1, \dots, n$ 
         $u_{j,k} \leftarrow a_{j,k} - l_{j,i} u_{i,k}$  (replace by  $a_{j,k}$ ,  $a_{j,i}$  and  $a_{i,k}$ )
    
```

Navigation icons

Complexity

The complexity (# flops) of GEPP on $n \times n$ matrices is

$$\sum_{i=1}^{n-1} \left(\sum_{j=i+1}^n 1 + \sum_{j=i+1}^n \sum_{k=i+1}^n 2 \right) = \sum_{i=1}^{n-1} ((n-i) + 2(n-i)^2) \\ = \left(\sum_{i=1}^{n-1} i \right) + 2 \left(\sum_{i=1}^{n-1} i^2 \right)$$

At this point, recall the asymptotic formulae for $k \geq 1$:

$$\sum_{i=1}^n i^k = \frac{n^{k+1}}{k+1} + O(n^k),$$

Hence this complexity is $\frac{2}{3}n^3 + O(n^2)$. Forward and backward substitutions have each a complexity of

$$n^2 + O(n)$$

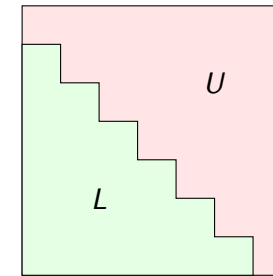
Hence GEPP solves the equation $Ax = b$ with

$$\frac{2}{3}n^3 + O(n^2) \text{ flops}$$

Navigation icons

Modified GEPP (cont.)

We need no extra space to store L and U !



Navigation icons

Floating point arithmetic

A floating point number is

$$f = \pm 0.d_1 d_2 \dots d_t \times \beta^e$$

$$\beta \geq 2, \quad 0 \leq d_i < \beta \text{ with } d_1 \neq 0 \quad \text{and} \quad L \leq e \leq U$$

$\beta \geq 2$ the base, L the underflow and U the overflow

The range is

$$\beta^{L-1} \leq |f| \leq \beta^U (1 - \beta^{-t}).$$

Floating point operations are defined by picking the closest floating point number:

$$a \odot b := \text{fl}(a \cdot b)$$

Machine epsilon: a bound for the relative error of roundoffs:

$$\frac{|a - \text{fl}(a)|}{|a|} \leq \varepsilon = \frac{\beta^{1-t}}{2}$$

Navigation icons

Single precision IEEE standard

s	e	q
1	8	23
sign	exponent	coefficient/mantissa

The *coded number* is

$$f = (-1)^s (1 + q) 2^{e-127}$$

Corresponds to $t = 24$, $\beta = 2$, $L = -126$ and $U = 127$

The maximal *relative error* is

$$2^{-24} \approx 6 \cdot 10^{-8}$$

and the *range* is

$$2^{-127} \approx 6 \cdot 10^{-38} \leq f \leq 2^{129} (1 - 2^{-24}) \approx 7 \cdot 10^{39}$$

Navigation icons

IEEE double precision standard

s	e	q
1	11	52
sign	exponent	coefficient/mantissa

The coded number is

$$f = (-1)^s (1 + q) 2^{e-1023}$$

Corresponds to $t = 53$, $\beta = 2$, $L = -1022$ and $U = 1026$

The maximal error is

$$2^{-53} \approx 6 \cdot 10^{-16}$$

and the range is

$$2^{-1022} \approx 2 \cdot 10^{-308} \leq f \leq 2^{1029} (1 - 2^{-24}) \approx 7 \cdot 10^{309}$$

Navigation icons

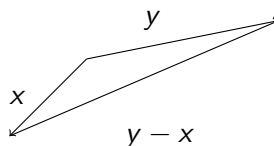
Vector and matrix norms

A *norm* on \mathbb{R}^n is a function

$$\|\cdot\|: \mathbb{R}^n \rightarrow \mathbb{R}$$

such that

- for every vector x we have that $\|x\| \geq 0$, and $\|x\| = 0$ if and only if $x = 0$
- for every vector x and scalar α we have that $\|\alpha x\| = |\alpha| \|x\|$
- for every vectors x, y we have that $\|x + y\| \leq \|x\| + \|y\|$ (*triangle inequality*).



Navigation icons

Vector and matrix norms (cont.)

Example: for $1 \leq p \leq +\infty$ the *p-norm* is defined as

$$\|x\|_p = \begin{cases} (\sum_{i=1}^n |x_i|^p)^{1/p} & \text{if } 1 \leq p < +\infty \\ \max_i |x_i| & \text{if } p = +\infty \end{cases}$$

Any two norms can be compared: there are $c_1, c_2 > 0$ such that

$$\|\cdot\|_1 \leq c_1 \|\cdot\|_2 \quad \text{and} \quad \|\cdot\|_2 \leq c_2 \|\cdot\|_1$$

Example:

$$\|\cdot\|_2 \leq \|\cdot\|_1 \leq n^{1/2} \|\cdot\|_2 \quad \text{and} \quad \|\cdot\|_\infty \leq \|\cdot\|_1 \leq n \|\cdot\|_\infty$$

Navigation icons

Vector and matrix norms (cont.)

A *matrix norm* is a norm on $\mathbb{R}^{n \times n}$ s.t. for all A, B we have that

$$\|AB\| \leq \|A\| \|B\|$$

Example: the *Frobenius norm*

$$\|A\|_F = \left(\sum_{i,j} |a_{i,j}|^2 \right)^{1/2}$$

Navigation icons

Perturbation theory

A matrix A is *well/badly (or ill) conditioned* if small changes in A can cause small/large changes in the solution of

$$Ax = b$$

Let x and $\hat{x} = x + \delta x$ solutions to

$$Ax = b \quad \text{and} \quad (A + \delta A)\hat{x} = b + \delta b$$

We have that

$$\begin{aligned} (A + \delta A)(x + \delta x) &= b + \delta b \\ - \frac{Ax}{\delta Ax + (A + \delta A)\delta x} &= \frac{b}{\delta b} \end{aligned}$$

Then

$$\delta x = A^{-1}(-\delta A \hat{x} + \delta b)$$

Navigation icons

Vector and matrix norms (cont.)

Definition: Given a norm on \mathbb{R}^n , the associated *operator norm* is

$$\|A\| = \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|}$$

It is a matrix norm

Properties:

- ① $\|A\|_\infty = \max_i \sum_j |a_{i,j}|$ (*maximum row sum*)
- ② $\|A\|_1 = \max_j \sum_i |a_{i,j}|$ (*maximum column sum*)
- ③ $\|A\|_2 = \lambda_{\max}(A^*A)^{1/2}$ with $A^* = \bar{A}^T$ and λ_{\max} the maximal eigenvalue
- ④ if Q and Q' are *orthogonal* then

$$\|QAQ'\|_2 = \|A\|_2 \quad \text{and} \quad \|QAQ'\|_F = \|A\|_F$$

In particular $\|Q\|_2 = 1$ and $\|Q\|_F = n^{1/2}$

Navigation icons

The condition number

Fix a norm $\|\cdot\|$. Then

$$\|\delta x\| \leq \|A^{-1}\| (\|\delta A\| \|\hat{x}\| + \|\delta b\|)$$

or equivalently

$$\frac{\|\delta x\|}{\|\hat{x}\|} \leq \kappa_{\|\cdot\|} \left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|A\| \|\hat{x}\|} \right) \quad (1)$$

with

$$\kappa_{\|\cdot\|} := \|A\| \|A^{-1}\|$$

the *condition number* of A with respect to $\|\cdot\|$

Navigation icons

Precision of approximations

Let $\lambda \in \mathbb{R}$ and $\hat{\lambda} = \lambda + \delta\lambda$ an approximation of λ with k correct digits in base $\beta \geq 2$. Then

$$\lambda = \beta^e \times d_1 \cdots d_k d_{k+1} \cdots \quad \text{and} \quad \hat{\lambda} = \beta^e \times d_1 \cdots d_k \tilde{d}_{k+1} \cdots$$

and so

$$\frac{|\delta\lambda|}{|\lambda|} \leq \beta^{-k}$$

or equivalently

$$-\log_{\beta} \left(\frac{|\delta\lambda|}{|\lambda|} \right) \geq k.$$

Roundoff with IEEE single precision and double precision give approximations with 24 and 53 correct bits, respectively:

$$-\log_2 \left(\frac{|\delta_{\text{single}} \lambda|}{|\lambda|} \right) \geq 24 \quad \text{and} \quad -\log_2 \left(\frac{|\delta_{\text{double}} \lambda|}{|\lambda|} \right) \geq 53.$$



Distance to the ill-posed problems

The condition number of the 2-norm has a geometrical interpretation as the inverse of its distance to the set of ill-posed problems:

$$\kappa_2(A) = \frac{1}{\text{distance}(A, \Sigma)} \quad (2)$$

with $\Sigma = \{A \mid \det(A) = 0\}$



Lost of precision

The inequality (1) translates into

$$-\log_{\beta} \frac{\|\delta x\|}{\|x\|} \geq -\log_{\beta} \kappa(A) - \log_{\beta} \left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|A\| \|\hat{x}\|} \right)$$

Warning: ill-conditioned matrices destroy the quality of your approximations!

For instance, for *exact* data truncated with IEEE single or double precision, the computed result of $Ax = b$ will be meaningless as soon

$$\kappa(A) > 2^{24} \approx 6 \cdot 10^8 \text{ (single precision)}$$

and

$$\kappa(A) > 2^{53} \approx 10^{16} \text{ (double precision)}$$



Error analysis in GEPP

We want to apply the two steps:

- ① analyse roundoff errors to show that the matrix

$$\hat{A}_{\text{GEPP}} := P_{\text{GEPP}} L_{\text{GEPP}} U_{\text{GEPP}}$$

has a small relative error (*backward analysis*)

- 2 apply perturbation theory to bound the error in the computed solution x_{GEP} of the equation

$$A_{\text{GEPP}} x = b$$



Error analysis in GEPP (cont.)

Rounding off the entries of A gives $\hat{A} = A + \delta A$ with

$$\frac{\|\delta A\|}{\|A\|} < \varepsilon \quad (\text{machine epsilon})$$

By perturbation theory, this error will be amplified to

$$\frac{\|\delta x\|}{\|x\|} < \kappa_{\|\cdot\|}(A) \varepsilon.$$

To keep this bound, for $\delta_{\text{GEPP}}A := A_{\text{GEPP}} - A$ we want

$$\frac{\|\delta_{\text{GEPP}} A\|}{\|A\|} \leq C_\varepsilon$$

with C as small as possible



The need of pivoting (cont.)

Set

$$A = LU = \begin{bmatrix} 1 & 0 \\ \eta^{-1} & 1 \end{bmatrix} \begin{bmatrix} \eta & 1 \\ 0 & 1 - \eta^{-1} \end{bmatrix}$$

Then

$$L_{\text{GEWP}} = \begin{bmatrix} 1 & 0 \\ \eta^{-1} & 1 \end{bmatrix} \quad \text{and} \quad U_{\text{GEWP}} = \begin{bmatrix} \eta & 1 \\ 0 & -\eta^{-1} \end{bmatrix}$$

and so

$$A_{\text{GEWP}} = L_{\text{GEWP}} U_{\text{GEWP}} = \begin{bmatrix} \eta & 1 \\ 1 & 0 \end{bmatrix},$$

is *not* close to A !

$$\frac{\|\delta A_{\text{GEWP}}\|_\infty}{\|A\|_\infty} = \frac{\left\| \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix} \right\|_\infty}{\left\| \begin{bmatrix} \eta & 1 \\ 1 & 1 \end{bmatrix} \right\|_\infty} = \frac{1}{2}$$

⇒ GEWP is not backward stable

The need of pivoting

Apply LU factorization without pivoting to the matrix

$$A = \begin{bmatrix} \eta & 1 \\ 1 & 1 \end{bmatrix}$$

with η a power of the base β that is smaller than ε , so that

$$1 \oplus \eta = \text{fl}(1 + \eta) = 1$$

For instance

$$\beta = 10, \quad \varepsilon = 0.5 \cdot 10^{-3} \quad \text{and} \quad \eta = 10^{-4}$$



The need of pivoting (cont.)

The solution of $A \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ is $x \approx \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

Solving

$$L_{\text{GEWP}} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

gives $y_1 = 1$ and $y_2 = 2 \ominus \eta^{-1} = -\eta^{-1}$. Then

$$U_{\text{GEWP}} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -\eta^{-1} \end{bmatrix}$$

gives $x_2 = \frac{-\eta^{-1}}{-\eta^{-1}} = 1$ and $x_1 = \frac{1 \ominus 1}{1 \ominus \eta} = 0$. Hence

$$\mathbf{x}_{\text{GEWP}} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

is *not* close to x



The need of pivoting (cont.)

The instability is also reflected in the conditions numbers:

$$\|A\|_{\infty} \approx 4 \quad \text{well-conditioned}$$

whereas

$$\|L\|_{\infty}, \|U\|_{\infty} \approx \eta^{-2} \quad \text{ill-conditioned}$$



Formal error analysis of GEPP (cont.)

Hence

$$\|A - A_{\text{GEPP}}\|_{\infty} \leq n \varepsilon \|L\|_{\infty} \|U\|_{\infty} \leq n^3 \varepsilon g_{\text{GEPP}} \|A\|_{\infty}$$

where

$$g_{\text{GEPP}} = \frac{\max_{i,j} |u_{i,j}|}{\max_{i,j} |a_{i,j}|} \quad \text{the pivot growth}$$

because

- $|l_{i,j}| \leq 1$ and so $\|L\|_{\infty} \leq n$
- $|u_{i,j}| \leq g_{\text{GEPP}} \|A\|_{\infty}$ and so $\|U\|_{\infty} \leq n g_{\text{GEPP}} \|A\|_{\infty}$

Thus

$$\frac{\|\delta_{\text{GEPP}} A\|_{\infty}}{\|A\|_{\infty}} \leq n^3 \varepsilon g_{\text{GEPP}} \quad (3)$$



Formal error analysis of GEPP

When the intermediate quantities are too large, the information in A can be easily lost

Suppose that A is already pivoted. Then

$$A = L_{\text{GEPP}} U_{\text{GEPP}} + E \quad \text{with } |E| \leq n \varepsilon |L| |U|$$

where

- $|E|$ the $n \times n$ matrix whose entries are the absolute values of those of E (and similarly for $|L|$ and $|U|$)
- ε the machine epsilon



Formal error analysis of GEPP (cont.)

In general $g_{\text{GEPP}} \leq 2^{n-1}$, and this bound can be attained:

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 \\ -1 & 1 & 0 & 1 \\ -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & -1 & 1 & 0 \\ -1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 8 \end{bmatrix}$$



Formal error analysis of GEPP (cont.)

This bound in (3) is too pessimistic in practice, since typically

$$\|L\|_\infty \|U\|_\infty \approx \|A\|_\infty$$

If this is the case, then

$$\frac{\|\delta_{\text{GEPP}} A\|}{\|A\|} \lesssim n \varepsilon$$

and GEPP would be stable

We thus say that GEPP is “backward stable in practice” (!?)

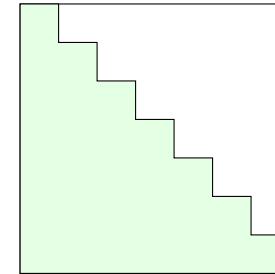


Symmetric matrices

An $n \times n$ matrix A is *symmetric* if

$$A^T = A$$

A symmetric matrix needs *half the space* to store its entries



We should be able to solve a symmetric problem

$$Ax = b$$

with $\approx \frac{n^3}{3}$ flops instead of $\approx \frac{2n^3}{3}$

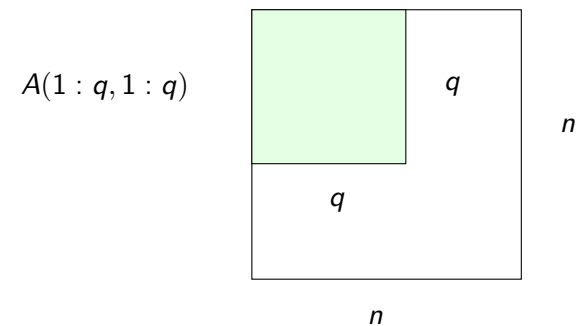


Matlab notation

For $p \leq q$ and $r \leq s$ set

$$A(p : q, r : s) = [a_{i,j}]_{p \leq i \leq q, r \leq j \leq s} \in \mathbb{R}^{(q-p+1) \times (s-r+1)}$$

For instance, the *leading* $q \times q$ -*principal submatrix* of A is



When does the LU factorization exist?

Let A be an arbitrary $n \times n$ matrix (not necessarily symmetric)

The following are equivalent (TFAE):

- 1 there are unique L unit lower triangular and U upper triangular such that $A = L U$
- 2 all leading principal submatrices of A are nonsingular

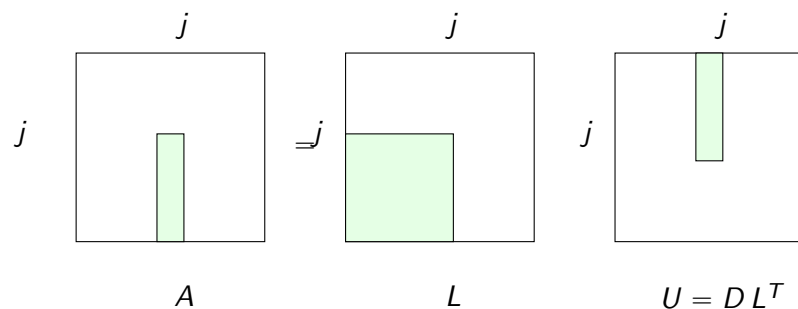
Navigation icons

Computing the LDLT factorization

For $j = 1, \dots, n$ set $A(j : n, j) \leftarrow L(j : n, 1 : j) v(1 : j)$ with

$$v = \begin{bmatrix} d_1 l_{j,1} \\ \vdots \\ d_{j-1} l_{j,j-1} \\ d_j \end{bmatrix}$$

as in the figure



Navigation icons

The LDLT factorization

When A is symmetric and has an LU factorization, the factors are connected:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \frac{c}{a} & 1 \end{bmatrix} \begin{bmatrix} a & 0 \\ 0 & d - \frac{bc}{a} \end{bmatrix} \begin{bmatrix} 1 & \frac{c}{a} \\ 0 & 1 \end{bmatrix}$$

In general symmetric case, set $d_i = u_{i,i}$ and write

$$U = D M$$

with $D = \text{diag}(d_1, \dots, d_n)$ and M unit upper triangular. Then

$$M = L^T$$

and so the LU factorization can be rewritten as

$$A = L D L^T$$

Navigation icons

Computing the LDLT factorization (cont.)

Hence the equations

$$d_j = a_{j,j} - \sum_{k=1}^{j-1} d_k \ell_{j,k}^2$$

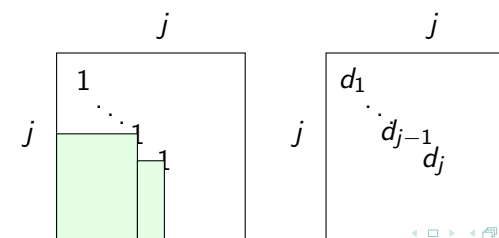
and

$$L(j+1 : n, j) = \frac{1}{d_j} (A(j+1 : n, j) - L(j+1 : n, 1 : j-1) v(1 : j-1))$$

gives the j th diagonal entry and the j th column

$$d_j \text{ and } L(j+1 : n, j)$$

from the previous diagonal entries and the $(j-1)$ th column of L



Navigation icons

The LDLT algorithm

Example

$$A = \begin{bmatrix} 1 & -1 & 2 \\ -1 & 5 & 2 \\ 2 & 2 & 17 \end{bmatrix}$$
$$L = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} \quad \text{and} \quad D = \begin{bmatrix} 1 \end{bmatrix}$$
$$[v] = -1, \quad L = \begin{bmatrix} & \\ & \\ 1 & \end{bmatrix} \quad \text{and} \quad D = \begin{bmatrix} & \\ & \\ & 4 \end{bmatrix}$$
$$v_1 = (-1) \cdot 1 = -1, \quad d_2 = 5 - (-1) \cdot (-1) = 4, \quad \ell_{3,2} = \frac{1}{4}(2 - 2 \cdot (-1))$$

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ↺ 🔍 ↻

The modified LDLT algorithm

↗ over-writing scheme



Example (cont.)

$$v = \begin{bmatrix} 2 \\ 4 \end{bmatrix} \quad \text{and} \quad D = \begin{bmatrix} & \\ & 9 \end{bmatrix}$$
$$v_1 = 2 \cdot 1 = 1, \quad v_2 = 1 \cdot 4 = 4, \quad d_3 = 17 - (2 \cdot 2 + 1 \cdot 4) = 9$$
$$\begin{bmatrix} 1 & -1 & 2 \\ -1 & 5 & 2 \\ 2 & 2 & 17 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 9 \end{bmatrix} \begin{bmatrix} 1 & -1 & 2 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$
$$A = \begin{bmatrix} 1 & -1 & 2 \\ -1 & 5 & 2 \\ 2 & 2 & 17 \end{bmatrix} \rightsquigarrow L \text{ \& } D = \begin{bmatrix} 1 & & \\ -1 & 4 & \\ 2 & 1 & 9 \end{bmatrix}$$

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ 🔍 ↺

The LU factorization of a symmetric matrix can be numerically unstable:

$$A = \begin{bmatrix} \eta & 1 \\ 1 & 1 \end{bmatrix}$$

with $0 < \eta < \varepsilon$ (machine epsilon)

A symmetric A is *positive definite (SPD)* if for all $x \in \mathbb{R}^n \setminus \{0\}$

$$x^T A x > 0$$

Important fact from LA:

$$A \in \mathbb{R}^{n \times n} \text{ symmetric} \iff A = Q^T \Lambda Q$$

with Q orthogonal and Λ diagonal: A is diagonalizable over the reals through an orthogonal similarity, and

$$A \text{ is SPD} \iff \Lambda = \text{diag}(\lambda_1, \dots, \lambda_n) \text{ with } \lambda_i > 0$$

The Cholesky factorization

An SPD $n \times n$ matrix is nonsingular, and moreover all its leading principal submatrices are nonsingular

\rightsquigarrow there is unit lower triangular L and a diagonal D such that

$$A = L D L^T \tag{4}$$

We have that $d_i > 0$ for all i and so we can write (4) as

$$A = G G^T$$

with $G = L \cdot \text{diag}(\lambda_1^{1/2}, \dots, \lambda_n^{1/2})$ (*Cholesky factorization*)

Computing the Cholesky factorization

Note that

$$a_{i,i} = \sum_{k=0}^j g_{j,k} g_{i,k}$$

and so

$$g_{j,j} g_{i,j} = a_{i,j} - \sum_{k=1}^{j-1} g_{j,k} g_{i,k}$$

→ we can compute the j -th column of G from the previous ones



The example revisited

Consider again

$$A = \begin{bmatrix} 1 & -1 & 2 \\ -1 & 5 & 2 \\ 2 & 2 & 17 \end{bmatrix}$$

For $j = 1$:

$$G = \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix}$$

and for $j = 2$:

$$G = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$

coming from the operations

$$g_{2,2} = (5 - (-1)^2)^{1/2} = 2 \quad \text{and} \quad g_{3,2} = \frac{1}{2} (2 - (-1) \cdot 2) = 2$$



The Cholesky algorithm

for $j = 1, \dots, n$

$$g_{j,j} \leftarrow (a_{j,j} - \sum_{k=1}^{j-1} g_{j,k}^2)^{1/2}$$

$$\text{for } i = j + 1, \dots, n$$

$$g_{i,j} \leftarrow \frac{1}{g_{j,j}} \left(a_{i,j} - \sum_{k=1}^{j-1} g_{i,k} g_{j,k} \right)$$

or alternatively:

for $j = 1, \dots, n$

$$a_{j,j} \leftarrow (a_{j,j} - \sum_{k=1}^{j-1} a_{j,k}^2)^{1/2}$$

for $i = j + 1, \dots, n$

$$a_{i,j} \leftarrow \frac{1}{g_{j,j}} \left(a_{i,j} - \sum_{k=1}^{j-1} a_{i,k} a_{j,k} \right)$$

Can be overwritten over A and does not need the auxiliary vector v



The example revisited (cont.)

For $j = 3$:

$$G = \begin{bmatrix} \\ \\ 3 \end{bmatrix}$$

coming from $g_{3,3} = (17 - (2^2 + 2^2))^{1/2} = 3$

Hence

$$\begin{bmatrix} 1 & -1 & 2 \\ -1 & 1 & 1 \\ 2 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 2 & 0 \\ 2 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 & -1 & 2 \\ 0 & 2 & 2 \\ 0 & 0 & 3 \end{bmatrix}$$

In the machine

$$A = \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & 2 & 1 & 1 \end{bmatrix} \rightsquigarrow G = \begin{bmatrix} 1 & & & \\ -1 & 2 & & \\ & 2 & 2 & 3 \end{bmatrix}$$



Complexity

The complexity of this algorithm is

$$\begin{aligned} \sum_{j=1}^n \left(2j - 1 + \sum_{i=j+1}^n (2j - 1) \right) &= \sum_{j=1}^n (2j - 1) (n - j + 1) \\ &= 2n \left(\sum_{j=1}^n j \right) - \sum_{j=1}^n j^2 + O(n^2) \\ &= 2n \left(\frac{n^2}{2} + O(n) \right) - \frac{n^3}{3} + O(n^2) \\ &= \frac{1}{3} n^3 + O(n^2) \end{aligned}$$

↪ half the complexity of the LU factorization

Is my A an SPD matrix?

Cholesky is the cheapest way of testing if a given symmetric $n \times n$ matrix is definite positive:

it will be the case if and only if the algorithm concludes!

Numerical stability

Pivoting is not necessary for the Cholesky algorithm to be numerically stable: the same analysis of GEPP shows that the Cholesky solution \hat{x} satisfies $(A + \delta A) \hat{x} = b$ with

$$|\delta A| \leq 3n\varepsilon |G| |G^T|$$

By the *Cauchy-Schwartz inequality*, for each i, j we have that

$$\begin{aligned} (|G||G^T|)_{i,j} &\leq \sum_{k=1}^n |g_{i,k}| |g_{j,k}| \\ &\leq \left(\sum_{k=1}^n g_{i,k}^2 \right)^{1/2} \left(\sum_{k=1}^n g_{j,k}^2 \right)^{1/2} = a_{i,i}^{1/2} a_{j,j}^{1/2} \leq \max_{i,j} |a_{i,j}| \end{aligned}$$

Hence $\|G\|_G^T\|_\infty \leq n\|A\|_\infty$ and so

$$\|\delta A\|_{\infty} \leq 3n^2 \varepsilon \|A\|_{\infty}$$

~> Cholesky algorithm is backward stable

Band matrices

A is a *band matrix* with *lower bandwith* b_L and *upper bandwith* b_U if

$$a_{i,j} = 0$$

whenever $i > j - b_L$ or $i < j + b_U$:

$$A = \begin{bmatrix} a_{1,1} & \cdots & a_{1,b_U+1} & & \\ \vdots & & & \ddots & \\ a_{b_L+1,1} & & & & a_{n-b_U,n} \\ & \ddots & & & \vdots \\ & & a_{n,n-b_L} & \cdots & a_{n,n} \end{bmatrix}.$$

GE on band matrices

\rightsquigarrow GEWP preserves the band structure:

where L is unit lower triangular with lower bandwidth b_L and U is upper triangular with upper bandwidth b_U

$$2 n b_L b_U + O(n(b_L + b_U)) \text{ flops}$$


Example (cont.)

$$L(4 : 4, 4) = [1] \quad \text{and} \quad U(4, 4 : 4) = [1]$$

Example

Example (cont.)

Example (cont.)

$$A = \begin{bmatrix} 2 & -1 & 0 & 0 \\ 4 & -1 & 3 & 0 \\ 0 & -1 & -2 & 1 \\ 0 & 0 & 3 & 4 \end{bmatrix} = L U = \begin{bmatrix} 1 & & & \\ 2 & 1 & & \\ 0 & -1 & 1 & \\ 0 & 0 & 3 & 1 \end{bmatrix} \begin{bmatrix} 2 & -1 & 0 & 0 \\ 1 & 3 & 0 \\ 1 & 1 \\ 1 \end{bmatrix}$$

GEPP on band matrices

GEPP can exploit band structure, but in a more involved way:

$$A = PLU$$

where U is banded with upper bandwith $b_L + b_U$ and L has at most $b_L + 1$ nonzero entries per column

Why?

At each step pivoting can only be done within the first b_L rows, and later permutations can reorder the earlier columns of L



Example (cont.)

We then swap the rows 2 and 3 of A to obtain

$$L(2:4, 2) = \begin{bmatrix} 1 \\ \frac{1}{2} \\ 0 \end{bmatrix}, \quad U(2, 2:4) = \begin{bmatrix} -1 & -2 & 1 \end{bmatrix}, \quad S_2 = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ 3 & 4 \end{bmatrix}$$

Swap the rows 3 and 4 of A :

$$L(3:4,3) = \begin{bmatrix} 1 \\ -1 \\ 6 \end{bmatrix}, \quad U(3,3:4) = \begin{bmatrix} 3 & 4 \end{bmatrix}, \quad S_3 = \begin{bmatrix} 1 \\ 6 \end{bmatrix}$$

Finally

$$U(4, 4 : 4) = \left[\frac{1}{6} \right]$$



Example

Let again

$$A = \begin{bmatrix} 2 & -1 & 0 & 0 \\ 4 & -1 & 3 & 0 \\ 0 & -1 & -2 & 1 \\ 0 & 0 & 3 & 4 \end{bmatrix}$$

and swap rows 1 and 2:

$$A = \begin{bmatrix} 4 & -1 & 3 & 0 \\ 2 & -1 & 0 & 0 \\ 0 & -1 & -2 & 1 \\ 0 & 0 & 3 & 4 \end{bmatrix}$$

Then

$$L(1:4,1) = \begin{bmatrix} 1 \\ \frac{1}{2} \\ 0 \\ 0 \end{bmatrix}, \quad U(1,1:4) = \begin{bmatrix} 4 & -1 & 3 & 0 \end{bmatrix}, \quad S_1 = \begin{bmatrix} \frac{-1}{2} & \frac{-3}{2} & 0 \\ -1 & -2 & 1 \\ 0 & 3 & 4 \end{bmatrix}$$



Example (cont.)

Hence

$$\begin{bmatrix} 2 & -1 & 0 & 0 \\ 4 & -1 & 3 & 0 \\ 0 & -1 & -2 & 1 \\ 0 & 0 & 3 & 4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & & & \\ 0 & 1 & & \\ 0 & 0 & 1 & \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{6} & 1 \end{bmatrix} \begin{bmatrix} 4 & -1 & 3 & 0 \\ & -1 & -2 & 1 \\ & & 3 & 4 \\ & & & \frac{1}{6} \end{bmatrix}$$

L has at most 2 nonzero elements per column, and U has upper bandwidth 2



Sparse matrices

GEPP does not preserve the sparse structure:

$$\begin{bmatrix} 1 & 0.1 & 0.1 & \dots & 0.1 \\ 0.1 & 1 & 0 & \dots & 0 \\ 0.1 & 0 & 1 & & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ 0.1 & 0 & 0 & \dots & 1 \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ 0.1 & 1 & & & \\ 0.1 & -0.01 & 1 & & \\ \vdots & \vdots & & \ddots & \\ 0.1 & -0.01 & \dots & \dots & 1 \end{bmatrix} \begin{bmatrix} 1 & 0.1 & 0.1 & \dots & 0.1 \\ 0.99 & -0.01 & \dots & \dots & -0.01 \\ & 0.99 & & & -0.01 \\ & & \ddots & & \vdots \\ & & & \ddots & 0.99 \end{bmatrix}$$



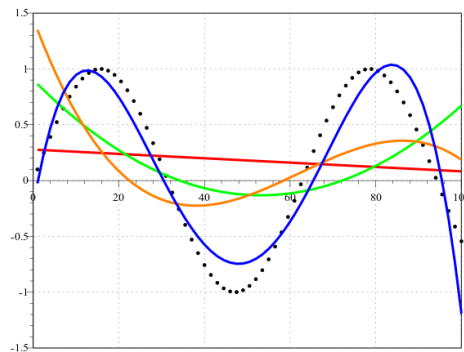
Navigation icons: back, forward, search, etc.

Example

Suppose we have pairs

$$(y_i, b_i) \in \mathbb{R}^2, \quad i = 1, \dots, m,$$

and we want to find the polynomial of degree d that best fits the b_i 's as a function of the y_i 's



Navigation icons: back, forward, search, etc.

The least square problem (LSP)

LSP: for an $m \times n$ matrix A and an m -vector b , find the n -vector x_{\min} minimizing the quantity

$$\|Ax - b\|_2$$

\rightsquigarrow gives the linear combination of the columns

$$Ax = \sum_{j=1}^n x_j \text{col}_j(A) \in \mathbb{R}^m$$

that best approaches (in the 2-norm) the m -vector b

- If $m = n$ and A is nonsingular then $Ax_{\min} = b$
- If $m > n$ then typically $Ax = b$ has no solution, and the minimum is positive

Central case: $m \geq n$ and $\text{rank}(A) = n$

Navigation icons: back, forward, search, etc.

Example (cont.)

Boils down to computing the polynomial

$$p(y) = \sum_{j=0}^d x_j y^j$$

minimizing the 2-norm of the *residual* $p(y_i) - b_i$, $i = 1, \dots, m$:

$$\sum_{i=1}^m (p(y_i) - b_i)^2$$

LSP for the data

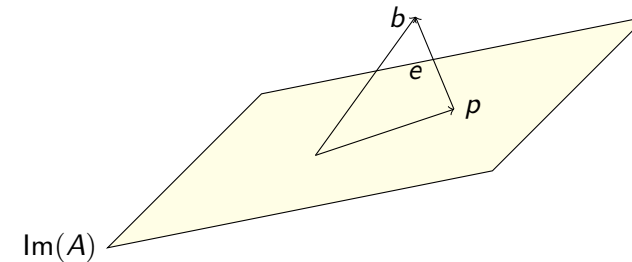
$$A = \begin{bmatrix} 1 & y_1 & \dots & y_1^d \\ \vdots & \vdots & & \vdots \\ 1 & y_m & \dots & y_m^d \end{bmatrix} \in \mathbb{R}^{m \times (d+1)} \quad \text{and} \quad b = \begin{bmatrix} b_1 \\ \vdots \\ b_m \end{bmatrix} \in \mathbb{R}^m$$

Navigation icons: back, forward, search, etc.

Three ways

- normal equations
- QR factorization
- SVD

LSP and its solution



- $p = Ax_{\min}$ the orthogonal projection of b into $\text{Im}(A)$
- $e = b - p$ the error vector

Normal equations

The m -vector e is perpendicular to $\text{Im}(A)$: for all $x \in \mathbb{R}^n$

$$0 = \langle Ax, e \rangle = (Ax)^T (b - Ax_{\min}) = x^T A^T (b - Ax_{\min})$$

Hence

$$\boxed{A^T A x_{\min} = A^T b} \quad (5)$$

The $n \times n$ matrix $A^T A$ is symmetric and positive definite

\Rightarrow it is nonsingular and x_{\min} is the only solution of (5)

By Pythagoras theorem, the 2-norm of the error is

$$\|e\|_2 = (\|b\|_2^2 - \|Ax\|_2^2)^{1/2}$$

Normal equations algorithm

We can solve the normal equations with Cholesky factorization:

Compute the lower triangular part of $C \leftarrow A^T A$

Compute $d \leftarrow A^T b$

Compute the Cholesky factorization $C = G G^T$

Solve $G y = d$ and $G^T x_{\min} = y$

Its *complexity* is

$$\left(m + \frac{n}{3}\right) n^2 + O(n^2) \text{ flops}$$

For $m \gg n$ the complexity $m n^2$ of computing the product $A^T A$ dominates

- uses standard algorithms
- reliable when A is far from rank deficient (but unstable otherwise)

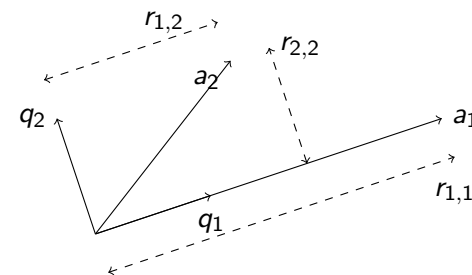
LSP and orthogonalization

The columns of A are independent ($\text{rank}(A) = n$) but not orthogonal

If this were the case, x_{\min} would be easy to find!

Gram-Schmidt orthogonalization

Set $a_j = \text{col}_j(A) \in \mathbb{R}^m$, $j = 1, \dots, n$



GS produces orthonormal m -vectors q_j , $j = 1, \dots, n$, such that

$$\text{span}(q_1, \dots, q_j) = \text{span}(a_1, \dots, a_j), \quad j = 1, \dots, n$$

Gram-Schmidt orthogonalization (cont.)

First step:

$$q_1 \leftarrow \frac{a_1}{\|a_1\|_2} \quad (\text{normalize})$$

GO step:

$$\tilde{a}_2 \leftarrow a_2 - \langle a_2, q_1 \rangle q_1 \quad (\text{orthogonalize})$$

$$q_2 \leftarrow \frac{\tilde{a}_2}{\|\tilde{a}_2\|_2} \quad (\text{normalize})$$

GO step:

$$\tilde{a}_3 \leftarrow a_3 - \langle a_3, q_1 \rangle q_1 - \langle a_3, q_2 \rangle q_2$$

$$q_3 \leftarrow \frac{\tilde{a}_3}{\|\tilde{a}_3\|_2}$$

The QR factorization

We can write the a_j 's in terms of the q_j 's:

$$a_1 = \|\tilde{a}_1\|_2 q_1$$

$$a_2 = \langle a_2, q_1 \rangle q_1 + \|\tilde{a}_2\|_2 q_2$$

$$a_3 = \langle a_3, q_1 \rangle q_1 + \langle a_3, q_2 \rangle q_2 + \|\tilde{a}_3\|_2 q_3$$

...

$$[a_1 \ a_2 \ a_3 \ \dots] = [q_1 \ q_2 \ q_3 \ \dots] \begin{bmatrix} r_{1,1} & r_{1,2} & r_{1,3} & \dots \\ 0 & r_{2,2} & r_{2,3} & \dots \\ 0 & 0 & r_{3,3} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

that is

$$A = QR$$

with

- Q orthogonal $m \times n$
- R upper triangular $n \times n$ matrix with positive diagonal entries

This factorization is unique!

Solving the LSP with the QR factorization

The QR factorization solves the normal equations, and so the LSP:

$$\begin{aligned} x_{\min} &= (A^T A)^{-1} A^T b \\ &= ((QR)^T Q R)^{-1} (Q R)^T b \\ &= (R^T R)^{-1} R^T Q b \\ &= R^{-1} Q^T b \end{aligned}$$

- The GS algorithm is not stable when the columns of A are close to rank deficient



Householder reflections (cont.)

Given $y \in \mathbb{R}^m$ there is a reflection that zeroes all but the first entry:

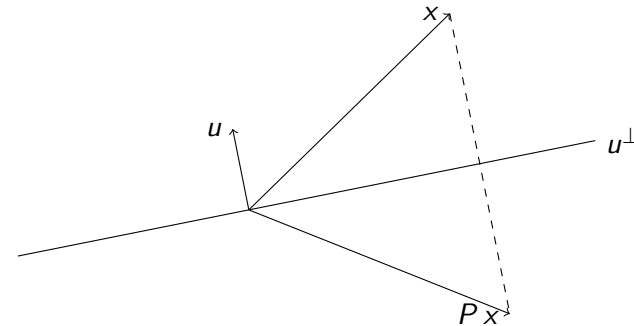
$$P y = \begin{bmatrix} c \\ 0 \\ \vdots \\ 0 \end{bmatrix} = c \mathbf{e}_1 \in \mathbb{R}^m$$

Since P is orthogonal

$$|c| = \|Py\|_2 = \|y\|_2$$



Householder reflections



A *Householder reflection* is

$$P = 1_m - 2uu^T \in \mathbb{R}^{m \times m}$$

for a unit vector $u \in \mathbb{R}^m$

It is symmetric ($P^T = P$) and orthogonal ($P^T P = \mathbb{1}_m$)



Householder reflections (cont.)

To compute u :

$$Py = (\mathbb{1}_m - 2uu^T)y = y - 2\langle u, y \rangle u = \pm \|y\|_2 e_1$$

thus

$$2\langle u, y \rangle u = y \pm \|y\|_2 e_1$$

Choose the sign so to avoid cancellations: u is a scalar multiple of

$$\tilde{u} = y \pm \|y\|_2 \mathbf{e}_1 = \begin{bmatrix} y_1 + \text{sign}(y_1) \|y\|_2 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

and so we set

$$u = \text{House}(y) := \frac{\tilde{u}}{\|\tilde{u}\|_2}$$



QR factorization with Householder reflections

Set $m = 4$ and $n = 3$:

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

- Choose P_1 such that

$$A_1 \leftarrow P_1 A = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix}$$

Navigation icons

QR factorization with Householder reflections (cont.)

$P_3 P_2 P_1 A = \tilde{R} (= A_4)$ is upper triangular. Hence

$$A = P_1^T P_2^T P_3^T \tilde{R} = Q R$$

with

- Q the first three columns of $P_1^T P_2^T P_3^T$
- R the first three rows of \tilde{R}

Navigation icons

QR factorization with Householder reflections (cont.)

- Choose $P_2 = \begin{matrix} & 1 & 2 \\ & 1 & 0 \\ 3 & 0 & P'_2 \end{matrix}$ such that

$$A_2 \leftarrow P_2 A_1 = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \end{bmatrix}$$

- Choose $P_3 = \begin{matrix} & 1 & 0 & 0 \\ & 0 & 1 & 0 \\ 2 & 0 & 0 & P'_3 \end{matrix}$ such that

$$A_3 \leftarrow P_3 A_2 = \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & 0 \end{bmatrix}$$

Navigation icons

Example

Let

$$A = \begin{bmatrix} 1 & -3 \\ 0 & 2 \\ -1 & -1 \end{bmatrix}$$

Set

$$\tilde{u}_1 = \begin{bmatrix} 1 + 2^{1/2} \\ 0 \\ -1 \end{bmatrix} \quad \text{and} \quad u_1 = \text{House} \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} = \frac{\tilde{u}_1}{\|\tilde{u}_1\|_2} = \begin{bmatrix} 0.92 \\ 0 \\ -0.38 \end{bmatrix}$$

Then

$$P_1 = \mathbb{I}_3 - 2 u_1 u_1^T = \begin{bmatrix} -0.71 & 0 & 0.71 \\ 0 & 1 & 0 \\ 0.71 & 0 & 0.71 \end{bmatrix}, \quad A_1 = P_1 A = \begin{bmatrix} -1.41 & 1.41 \\ 0 & 2 \\ 0 & -2.83 \end{bmatrix}$$

Navigation icons

Example (cont.)

Set then

$$\tilde{u}_2 = \begin{bmatrix} 2 + (2^2 + (-2.83)^2)^{1/2} \\ -2.83 \end{bmatrix} \quad \text{and} \quad u_2 = \text{House} \begin{bmatrix} 2 \\ -2.83 \end{bmatrix} = \frac{\tilde{u}_2}{\|\tilde{u}_2\|_2} = \begin{bmatrix} 0.89 \\ -0.46 \end{bmatrix}$$

Then

$$P_2 = \begin{bmatrix} 1 & 0 \\ 0 & \mathbb{1}_2 - 2 u_2 u_2^T \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -0.58 & 0.82 \\ 0 & 0.82 & 0.58 \end{bmatrix}$$

$$P_2 A_1 = \begin{bmatrix} -1.41 & 1.41 \\ 0 & -3.49 \\ 0 & 0 \end{bmatrix} = \tilde{R}$$

◀ ◻ ▶ ◀ 📄 ▶ ◀ ☰ ▶ ◀ ☰ ▶ ☰ ↺ 🔍 ↻

The algorithm

```
for  $i = 1$  to  $\min(m - 1, n)$ 
```

$$u_i \leftarrow \text{House}(A(i : m, i))$$

$$P_i \leftarrow \mathbb{1}_{m-i+1} - 2 u_i u_i^T$$

$$A_i(i : m, i : n) \leftarrow P'_i A(i : m, i : n)$$

- we do not really need P'_i but just the multiplication

$$(\mathbb{1}_{m-i+1} - 2 u_i u_i^T) A(i:m, i:n) = A(i:m, i:n) - 2 u_i (u_i^T A(i:m, i:n))$$

- each P_i can be “stored” as u_i
- Q can be stored as $P_1 \cdots P_{n-1}$

◀ ◻ ▶ ◀ 📄 ▶ ◀ ☰ ▶ ◀ ☰ ▶ ☰ ↺ 🔍 ↻

Example (cont.)

Finally

$$A = P_1^T P_2^T \tilde{R} = \tilde{Q} \tilde{R} = \begin{bmatrix} -0.71 & 0.58 & 0.41 \\ 0 & -0.58 & 0.82 \\ 0.71 & 0.58 & 0.41 \end{bmatrix} \begin{bmatrix} -1.41 & 1.41 \\ 0 & -3.49 \\ 0 & 0 \end{bmatrix}$$

$$= QR = \begin{bmatrix} -0.71 & 0.58 \\ 0 & -0.58 \\ 0.71 & 0.58 \end{bmatrix} \begin{bmatrix} -1.41 & 1.41 \\ 0 & -3.49 \end{bmatrix}$$

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ↺ 🔍 ↻

The algorithm (cont.)

The complexity of this algorithm is

$$2 n^2 m - \frac{2}{3} n^2 \text{ flops}$$

Compared with solving the normal equations via Cholesky's algorithm:

- twice the complexity (for $m \gg n$)
- more numerically stable

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡

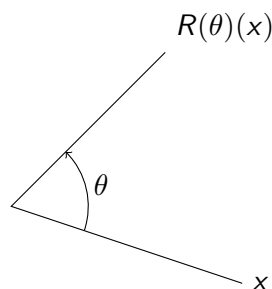
Plane rotations

A *rotation* on the plane with angle θ is the linear map

$$R(\theta): \mathbb{R}^2 \rightarrow \mathbb{R}^2$$

given by the orthogonal matrix

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$



Navigation icons

Plane rotations

A *Givens rotation*: matrix of a rotation on the (i, j) -plane of \mathbb{R}^m

$$R(i, j, \theta) = \begin{matrix} & i & j & \\ \begin{matrix} i \\ j \end{matrix} & \begin{bmatrix} \mathbb{1}_{i-1} & & & \\ & \cos(\theta) & -\sin(\theta) & \\ & \sin(\theta) & \cos(\theta) & \\ & & & \mathbb{1}_{n-j-1} \end{bmatrix} \end{matrix}$$

Navigation icons

QR factorization with Givens rotations

The QR factorization can be computed with Givens rotations similarly as with Householder reflections:

Given $x \in \mathbb{R}^m$ and $i > j$, we can zero x_j by choosing θ such that

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x_i \\ x_j \end{bmatrix} = \begin{bmatrix} (x_i^2 + x_j^2)^{1/2} \\ 0 \end{bmatrix}$$

or equivalently

$$\cos(\theta) = \frac{x_i}{(x_i^2 + x_j^2)^{1/2}} \quad \text{and} \quad \sin(\theta) = \frac{-x_j}{(x_i^2 + x_j^2)^{1/2}}$$

inverse trigonometric functions are not needed!

Navigation icons

Example

Let

$$A = \begin{bmatrix} 1 & -3 \\ 0 & 2 \\ -1 & -1 \end{bmatrix}$$

Setting

$$R_1 = \begin{bmatrix} 0.71 & 0 & -0.71 \\ 0 & 1 & 0 \\ 0.71 & 0 & 0.71 \end{bmatrix}$$

then

$$A_1 = R_1 A = \begin{bmatrix} 1.41 & -1.41 \\ 0 & 2 \\ 0 & -2.82 \end{bmatrix}$$

Navigation icons

Example (cont.)

Then set

$$R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.58 & -0.82 \\ 0 & 0.82 & 0.58 \end{bmatrix}$$

so that

$$A_2 = R_2 A_1 = \begin{bmatrix} 1.41 & -1.41 \\ 0 & 3.47 \\ 0 & 0 \end{bmatrix} = \tilde{R}$$

We conclude that $A = R_1^T R_2^T \tilde{R} = Q R$ with

$$Q = \begin{bmatrix} 0.71 & -0.58 \\ 0 & 0.58 \\ 0.71 & -0.58 \end{bmatrix} \quad \text{and} \quad R = \begin{bmatrix} 1.41 & -1.41 \\ 0 & 2 \end{bmatrix}$$



The condition number of a rectangular matrix

For a $n \times n$ matrix A we have that

$$\|A\|_2 = \lambda_{\max}(A^T A)^{1/2} \quad (\lambda_{\max} \text{ the largest eigenvalue})$$

and so

$$\kappa_2(A) = \left(\frac{\lambda_{\max}(A^T A)}{\lambda_{\min}(A^T A)} \right)^{1/2}$$

When A is $m \times n$, we define its *condition number* as

$$\kappa_2(A) := \kappa_2(A^T A)^{1/2} = \left(\frac{\lambda_{\max}(A^T A)}{\lambda_{\min}(A^T A)} \right)^{1/2}$$

It measures how far is A from being *rank deficient*



Complexity

The complexity of the QR factorization using Givens rotations is

$$3 m n^2 + O(m n)$$

It is useful in special situations, e.g. for Hessenberg matrices

$$A = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}$$



Numerical aspects

The *forward error analysis* of the LSP is controlled by $\kappa_2(A)$

QR factorization via Householder reflections or Givens rotations is *backwards stable*: if

$$A = QR$$

and $Q + \delta Q$ and $R + \delta R$ are the computed factors, then

$$A + \delta A = (Q + \delta Q)(R + \delta R)$$

where the relative error is bounded by

$$\frac{\|\delta A\|_2}{\|A\|_2} \leq O(n\varepsilon)$$

with ε the machine epsilon



QR factorization *versus* normal equations

Hence when $m \gg n$, the QR factorization via Householder or Givens solves the LSP with $\approx 2n^2m$ flops or $\approx 3n^2m$ flops respectively, and a loss of precision of

$$\approx \log_b \kappa_2(A) \text{ digits}$$

On the other hand, solving the normal equations via Cholesky solves the LSP with $\approx n^2m$ flops and a loss of precision of

$$\approx \log_b \kappa_2(A^T A) = 2 \log_b \kappa_2(A) \text{ digits}$$

Normal equations is the method of choice to solve the LSP when A is well-conditioned. If A is badly conditioned, then we might prefer applying the QR factorization via Householder or Givens, or the SVD (*to be discussed later*)



Example

For some medical research on the effect of a drug on sugar levels in blood, we take the following data from patients:

- initial blood level (sugar)
- amount of drug
- weight on day i , $i = 1, \dots, 7$
- final blood level

To correlate the final sugar level in terms of the rest of the data, we consider the LSP

$$\begin{bmatrix} \text{patient}_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \text{patient}_m \end{bmatrix} A \begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_9 \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ b_9 \end{bmatrix}$$



Rank deficient LSP

Let A be an $m \times n$ matrix and set $r = \text{rank}(A)$

It is *rank deficient* when

$$r < n$$

Typical in data analysis, when the variables have not been chosen wisely

If $r < n$ then the solution of the LSP is not unique: for all $x \in \text{Ker}(A) \simeq \mathbb{R}^{n-r}$ we have that

$$A(x_{\min} + x) = Ax_{\min} + Ax = Ax_{\min}$$

also solves the LSP



Example (cont.)

The solution x_{\min} would be the used as the predictor, since

$$b_i \approx \text{patient}_i \cdot x_{\min}$$

The matrix A is probably close to rank 3: the experiment is not well-designed, and columns 3, \dots , 9 should be identified from our knowledge of the problem

Otherwise, we might take x_{\min} very large, and a patient changing weight during the week would receive an unrealistic prediction



Rank deficient LSP via QR factorization

Suppose that $r < n$ and set

$$A = QR = Q \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & 0 \end{bmatrix} \begin{matrix} r \\ n-r \end{matrix}$$

With round off, we hope to compute

$$R = \begin{bmatrix} R_{1,1} & R_{1,2} \\ 0 & R_{2,2} \end{bmatrix}$$

with $R_{2,2}$ small (e.g. of size $\approx \varepsilon \|A\|_2$)

\rightsquigarrow we set $R_{2,2} = 0$ and then minimize $\|Ax - b\|_2$ as follows:

complete Q to an orthogonal $m \times m$ matrix $[Q \tilde{Q}]$, so that

$$\|Ax - b\|_2^2 = \left\| \begin{bmatrix} Q^T \\ \tilde{Q}^T \end{bmatrix} (Ax - b) \right\|_2^2 = \|Rx - Q^T b\|_2^2 + \|\tilde{Q}^T b\|_2^2$$

(multiplying by an orthogonal $m \times m$ matrix does not change the 2-norm)

Rank deficient LSP via QR factorization (cont.)

In general, this method is *not completely reliable* because R might be close to rank deficient even if $R_{2,2}$ is not small

Instead we apply QR factorization with pivoting

$$AP = QR$$

with P is a permutation $m \times m$ matrix:

at step i , choose the largest column j of A with $i \leq j \leq n$ and compute the Householder reflection that zeroes the entries $i+1, \dots, m$ in the i -th column

\rightsquigarrow attempts to keep $R_{1,1}$ well-conditioned and $R_{2,2}$ small

Rank deficient LSP via QR factorization (cont.)

Write $Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}$ and $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \begin{matrix} r \\ n-r \end{matrix}$. Then

$$\|Ax - b\|_2^2 = \|R_{1,1} x_1 + R_{1,2} x_2 - Q_1^T b\|_2^2 + \|Q_2^T b\|_2^2 + \|\tilde{Q}^T b\|_2^2$$

This expression is minimized by

$$x = \begin{bmatrix} R_{1,1}^{-1} (Q_1^T b - R_{1,2} x_2) \\ x_2 \end{bmatrix}$$

for any $(n-r)$ -vector x_2

The typical choice is

$$x_2 = 0$$

Singular values and singular vectors

The *singular value decomposition (SVD)* extends this factorization to any matrix, even outside the square case.

The *key ingredient* is

decoupling Q and Q^T

Let A be an $m \times n$ matrix with $m \geq n$. There are two sets of *singular vectors*

$$u_1, \dots, u_m \in \mathbb{R}^m \text{ (left)} \quad \text{and} \quad v_1, \dots, v_n \in \mathbb{R}^n \text{ (right)}$$

Both sets form orthogonal bases, and are connected by the relation

$$A v_i = \sigma_i u_i, \quad i = 1, \dots, n,$$

for the *singular values* $\sigma_1, \dots, \sigma_n \geq 0$

The SVD

In matrix form: both

$$U = \begin{bmatrix} u_1 & \cdots & u_m \end{bmatrix} \in \mathbb{R}^{m \times m} \quad \text{and} \quad V = \begin{bmatrix} v_1 & \cdots & v_n \end{bmatrix}$$

are *orthogonal* square matrices:

$$U^{-1} = U^T \quad \text{and} \quad V^{-1} = V^T.$$

Set

$$\Sigma = \begin{bmatrix} & & n \\ \sigma_1 & & \\ & \ddots & \\ & & \sigma \\ & 0 & m-n \end{bmatrix}$$

The (*full*) SVD is the factorization $A V = U \Sigma$ or equivalently

$$A \equiv U \Sigma V^T$$



Decomposition into matrices of rank 1

Set $r = \text{rank}(A)$. Then

- u_1, \dots, u_r is a basis of $\text{Im}(A)$
- v_{r+1}, \dots, v_n is a basis of $\text{Ker}(A)$

The column-row multiplication of $U\Sigma$ and V^T separates A into r pieces of rank 1:

$$A = \sum_{i=1}^r \sigma_i u_i v_i^T$$

In the example:

$$\begin{bmatrix} 3 & 0 \\ 4 & 5 \end{bmatrix} = 6.71 \begin{bmatrix} 0.32 \\ 0.95 \end{bmatrix} \begin{bmatrix} 0.71 & 0.71 \end{bmatrix} + 2.24 \begin{bmatrix} -0.95 \\ 0.32 \end{bmatrix} \begin{bmatrix} -0.71 & 0.71 \end{bmatrix}$$

We have that

$$\sigma_1 = 6.71 > \sigma_2 = 2.24$$

\rightsquigarrow the first piece is “more representative” of A

Example

$$A = \begin{bmatrix} 3 & 0 \\ 4 & 5 \end{bmatrix}$$

$$= U \Sigma V^T = \begin{bmatrix} 0.32 & -0.95 \\ 0.95 & 0.32 \end{bmatrix} \begin{bmatrix} 6.71 & 0 \\ 0 & 2.24 \end{bmatrix} \begin{bmatrix} 0.71 & 0.71 \\ -0.71 & 0.71 \end{bmatrix}$$

The thin and the reduced SVD's

The *thin SVD* avoids the 0's in the lower part of Σ and uses a diagonal matrix for the singular values:

$$A = U_n \Sigma_n V^T$$

with

$$U_n = \begin{bmatrix} u_1 & \cdots & u_n \end{bmatrix} \quad m \times n\text{-orthogonal}$$

$$\Sigma_n = \text{diag}(\sigma_1, \dots, \sigma_n) \quad n \times n\text{-diagonal}$$

The *reduced SVD* keeps only the nonzero singular values to remove the parts that are going to produce zeros for sure:

$$A = U_r \Sigma_r V_r^T$$

with

$$U_r = \begin{bmatrix} u_1 & \cdots & u_r \end{bmatrix} \quad m \times r\text{-orthogonal}$$

$$\Sigma_r = \text{diag}(\sigma_1, \dots, \sigma_r) \quad r \times r\text{-diagonal}$$

$$V_r^T = \begin{bmatrix} v_1 & \cdots & v_r \end{bmatrix} \quad n \times r\text{-orthogonal}$$



Reducing to SPD matrices

To identify the singular values and vectors, we can consider the SPD matrices

$$A^T A = (V \Sigma^T U^T) (U \Sigma V^T) = V \Sigma^T \Sigma V^T \in \mathbb{R}^{n \times n}$$

$$A A^T = (U \Sigma V^T) (V \Sigma^T U^T) = U \Sigma \Sigma^T U^T \in \mathbb{R}^{m \times m}$$

Then

- V contains the orthonormal eigenvectors of $A^T A$
- U contains the orthonormal eigenvectors of $A A^T$
- $\sigma_1, \dots, \sigma_r$ are the nonzero eigenvalues of both $A^T A$ and $A A^T$

Navigation icons

Existence and computation of the SVD (cont.)

For the full SVD, take

$$v_{r+1}, \dots, v_n \in \mathbb{R}^n \quad \text{and} \quad u_{r+1}, \dots, u_m \in \mathbb{R}^m$$

completing v_1, \dots, v_r and u_1, \dots, u_r to orthonormal bases. Then

$$A = U \Sigma V^T$$

with $U = [u_1 \ \dots \ u_m]$, $V = [v_1 \ \dots \ v_n]$ and

$$\Sigma = \begin{bmatrix} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_r & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{bmatrix}$$

Navigation icons

Existence and computation of the SVD

Consider the diagonalization

$$A^T A = Q \Lambda Q^T$$

and let v_1, \dots, v_n be the columns of Q , ordered so that v_1, \dots, v_r correspond to the nonzero eigenvalues $\lambda_1 \geq \dots \geq \lambda_r > 0$

Set

$$\sigma_k = \lambda_k^{1/2} \quad \text{and} \quad u_k = \sigma_k^{-1} A v_k, \quad k = 1, \dots, r$$

The u_k 's are orthonormal:

$$\begin{aligned} \langle u_j, u_k \rangle &= u_j^T u_k = (\sigma_j^{-1} A v_j)^T (\sigma_k^{-1} A v_k) \\ &= \sigma_j^{-1} \sigma_k^{-1} v_j^T A^T A v_k = v_j^T v_k = \langle v_j, v_k \rangle = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases} \end{aligned}$$

\rightsquigarrow the reduced SVD

$$A = [u_1 \ \dots \ u_r] \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix} \begin{bmatrix} v_1^T \\ \vdots \\ v_r^T \end{bmatrix}$$

Navigation icons

Example (cont.)

Set as before $A = \begin{bmatrix} 3 & 0 \\ 4 & 5 \end{bmatrix}$. Then

$$A^T A = \begin{bmatrix} 25 & 20 \\ 20 & 25 \end{bmatrix} \quad \text{and} \quad A A^T = \begin{bmatrix} 9 & 12 \\ 12 & 41 \end{bmatrix}$$

The eigenvalues and eigenvectors of $A^T A$ are

$$\begin{bmatrix} 25 & 20 \\ 20 & 25 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = 45 \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 25 & 20 \\ 20 & 25 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = 5 \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

Hence the right singular vectors are

$$v_1 = \frac{1}{2^{1/2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.71 \\ 0.71 \end{bmatrix} \quad \text{and} \quad v_2 = \frac{1}{2^{1/2}} \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} -0.71 \\ 0.71 \end{bmatrix}$$

and the singular values are

$$\sigma_1 = 45^{1/2} = 6.71 \quad \text{and} \quad \sigma_2 = 5^{1/2} = 2.24$$

Navigation icons

Example (cont.)

Moreover, the left singular vectors are

$$u_1 = \sigma_1^{-1} A v_1 = \frac{1}{10^{1/2}} \begin{bmatrix} 1 \\ 3 \end{bmatrix} = \begin{bmatrix} 0.32 \\ 0.95 \end{bmatrix}$$

$$u_2 = \sigma_2^{-1} A v_2 = \frac{1}{10^{1/2}} \begin{bmatrix} -3 \\ 1 \end{bmatrix} = \begin{bmatrix} -0.95 \\ 0.32 \end{bmatrix}$$

Hence $A = U \Sigma V^T$ with

$$U = \begin{bmatrix} 0.32 & -0.95 \\ 0.95 & 0.32 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} 6.71 & \\ & 2.24 \end{bmatrix}, \quad V = \begin{bmatrix} 0.71 & -0.71 \\ 0.71 & 0.71 \end{bmatrix}$$

Navigation icons

The geometry of the SVD

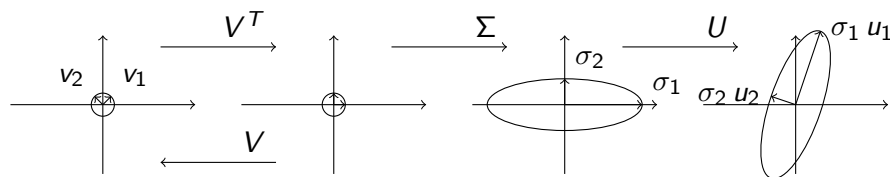
The SVD decomposes the matrix as

orthogonal \times diagonal \times orthogonal .

The unit sphere \mathbb{S}_n of \mathbb{R}^n is sent to the ellipsoid $A \mathbb{S}_n$ of \mathbb{R}^n centered at the origin and with axes

$$\sigma_i u_i, \quad i = 1, \dots, r.$$

In dimension 2 we can draw the process: for $A = \begin{bmatrix} 3 & 0 \\ 4 & 5 \end{bmatrix}$ it gives



Navigation icons

Some particular cases

- Let $S = Q \Lambda Q^T$ be the diagonalization of a SPD matrix. Then

$$U = V = Q \quad \text{and} \quad \Sigma = \Lambda$$

- The singular values of an orthonormal $n \times n$ matrix Q are all equal to 1.
- Let $A = x y^T$ be an $m \times n$ matrix of rank 1, with $x \in \mathbb{R}^m$ and $y \in \mathbb{R}^n$. Its reduced SVD is $A = U_1 \Sigma_1 V_1^T$

$$U_1 = \frac{x}{\|x\|_2}, \quad \Sigma_1 = [\|x\|_2 \|y\|_2], \quad V_1 = \frac{y}{\|y\|_2}$$

Navigation icons

Computing the 2-norm and the Frobenius norm

The 2-norm of A can be computed in terms of its SVD: this norm is invariant with respect to multiplication by orthogonal matrices and so

$$\|A\|_2 = \|U^T A V\|_2 = \|\Sigma\|_2 = \sigma_1$$

The Frobenius norm is also invariant with respect to multiplication by orthogonal matrices, and so

$$\|A\|_F = \|U^T A V\|_F = \|\Sigma\|_F = \left(\sum_{i=1}^r \sigma_i^2 \right)^{1/2}$$

Navigation icons

The best low rank approximation

Eckart-Young theorem: for $k = 1, \dots, r$, the matrix

$$A_k = \sum_{i=1}^k \sigma_i u_i v_i^T = U_k \Sigma_k V_k^T$$

is the best rank k approximation of A with respect to both the 2-norm and the Frobenius norm

Hence for $\|\cdot\| = \|\cdot\|_2$ or $\|\cdot\| = \|\cdot\|_F$ we have that

$$\|A - B\| \geq \|A - A_k\|$$

for any other $m \times n$ matrix of rank $\leq k$



Image compression

A B/W image of $m \times n$ pixels can be coded by an $m \times n$ matrix A with entries $a_{i,j} \in [0,1]$, indicating the brightness of the (i,j) -pixel:

0 (black) \cdots gray \cdots 1 (white)

Instead of transmitting/storing A , we can replace it by its k -th rank approximation

$$A_k = \sum_{i=1}^k \sigma_i u_i v_i^T,$$

which has size $k(m+n+1)$ (or $k(m+n)$ if we store $\sigma_i u_i$)

The *relative error of the approximation* is

$$\frac{\|A - A_k\|_2}{\|A\|_2} = \frac{\sigma_{k+1}}{\sigma_1}$$

and its *compression ratio* is

$$\frac{k(m+n)}{mn}.$$



The best low rank approximation (cont.)

For both norms, by the orthogonal invariance we have that

$$\|A - A_k\| = \|\Sigma - \Sigma_k\| = \begin{bmatrix} 0 & & & & \\ & \ddots & & & \\ & & 0 & & \\ & & & \sigma_{k+1} & \\ & & & & \ddots \\ & & & & & \sigma_r \end{bmatrix}$$

Hence

$$\|A - A_k\|_2 = \sigma_{k+1} \quad \text{and} \quad \|A - A_k\|_F = \left(\sum_{i=k+1}^r \sigma_i \right)^{1/2}$$



Image compression (cont.)

Here is a 320×200 -picture of a clown and its approximations:

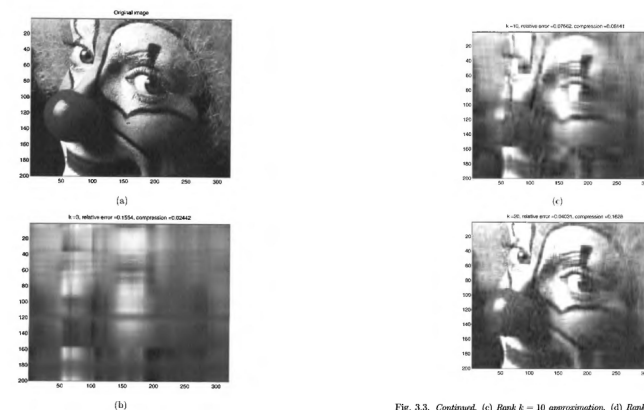


Fig. 3.3. Image compression using the SVD. (a) Original image. (b) Rank $k = 3$ approximation.

Fig. 3.3. Continued. (c) Rank $k = 10$ approximation. (d) Rank $k = 20$ approximation.



Principal component analysis

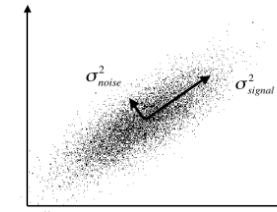
Let M be a data $m \times n$ matrix, for instance:

$$M = \begin{bmatrix} & \text{age} & \text{height} \\ & & \\ & & \\ & & \end{bmatrix} \text{ kids}$$

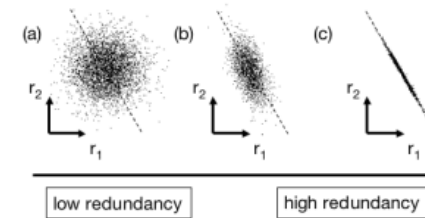
How can we find a simpler description?

Correlated vs uncorrelated data

Plotting the samples in \mathbb{R}^n and centering them, we might find that they are *correlated*:



This correlation might be higher (*more significant*) or lower (*less significant*).



Centering the data

The key parameters in probability and statistics are

mean and variance

The *mean* of the variables is the row n -vector

$$\mu = \frac{1}{n} \sum_{i=1}^m \text{row}_i(M)$$

The *centered data* is the $m \times n$ matrix

$$A = \begin{bmatrix} \text{row}_1(M) - \mu \\ \vdots \\ \text{row}_m(M) - \mu \end{bmatrix}$$

\rightsquigarrow the mean of each variable (=column) in A is zero

The covariance matrix

The *covariance matrix* of M is the SPD $n \times n$ matrix

$$S = \frac{1}{m-1} A^T A$$

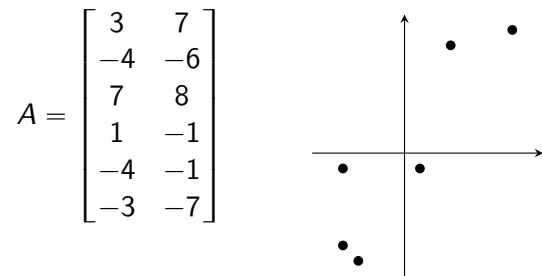
Its diagonal and off-diagonal entries are the *variances* and *covariances* of the variables in A :

setting $a_j = \text{col}_j(A)$, $j = 1, \dots, n$, we have that

$$s_{k,k} = \frac{\langle a_k, a_k \rangle}{m-1} = \text{Var}(a_k) \quad \text{and} \quad s_{k,l} = \frac{\langle a_k, a_l \rangle}{m-1} = \text{Cov}(a_k, a_l)$$

Example

Consider the centered data matrix of ages and weights



Its covariance matrix is

$$S = \frac{1}{6-1} A^T A = \begin{bmatrix} 20 & 25 \\ 25 & 40 \end{bmatrix}$$

Navigation icons

Orthonormal changes of variables

Let q_i , $i = 1, \dots, n$, be an orthonormal basis of \mathbb{R}^n

For an n -vector x , its representation with respect to this basis is

$$x = \sum_{j=1}^n \langle q_j, x \rangle q_j = \sum_{j=1}^n (x^T q_j) q_j$$

Consider the orthogonal $n \times n$ matrix $Q = [q_1 \cdots q_n]$ and set

$$B = A Q = [b_1 \cdots b_n]$$

so that $b_{i,j} = \text{row}_i(A) q_j$

We have that

- the j -variable in B is the linear combination $\langle q_j, x \rangle$ of the variables in A
- for each k , the $m \times k$ matrix $B_k = [b_1 \cdots b_k]$ gives the projection of the centered data matrix A into the k -th linear subspace

$$\text{span}(b_1, \dots, b_k)$$

Navigation icons

The total variance

The *total variance* of the variables in A is

$$\text{Var}(A) = \sum_{j=1}^n \text{var}(a_j) = \frac{1}{m-1} \sum_{i,j} a_{i,j}^2 = \frac{1}{m-1} \|A\|_F^2$$

Consider the full SVD

$$A = U \Sigma V^T$$

By the orthogonal invariance of the Frobenius norm:

$$\text{Var}(A) = \frac{1}{m-1} \|U \Sigma V^T\|_F^2 = \frac{1}{m-1} \|\Sigma\|_F^2 = \frac{1}{m-1} \sum_{j=1}^n \sigma_j^2$$

Navigation icons

Orthonormal changes of variables (cont.)

By the orthogonal invariance of the Frobenius norm

$$\text{Var}(B) = \frac{1}{m-1} \|B\|_F^2 = \frac{1}{m-1} \|A\|_F^2 = \text{Var}(A)$$

the total variance is invariant by orthogonal changes of variables

Consider again the full SVD

$$A = U \Sigma V^T$$

For $Q = V$ we have that

$$\text{Var}(B_k) = \text{Var}(A V_k) = \frac{1}{m-1} \|U \Sigma_k\|_F^2 = \frac{1}{m-1} \sum_{j=1}^k \sigma_j^2$$

This is the *maximal* total variance among all possible orthogonal projections of the data into a k -th linear subspace of \mathbb{R}^n

Navigation icons

The smallest distance

Also, the sum of the squares of the distances between the samples and its projections is *minimal* for this k -th linear subspace:

$$\begin{aligned} \sum_{i=1}^m \left\| \text{row}_i(A) - \sum_{j=1}^k b_{i,j} v_j \right\|_2^2 &= \sum_{i=1}^m \left\| \sum_{j=k+1}^n b_{i,j} v_j \right\|_2^2 \\ &= \|A \cdot [0 \cdots 0 v_{k+1} \cdots v_n]\|_{\mathbb{F}}^2 \\ &= \sum_{j=k+1}^n \sigma_j^2 \end{aligned}$$

This is a consequence of the Eckart-Young theorem



Principal components and principal directions (cont.)

The principal components $b_i = \sigma_i u_i$, $i = 1, \dots, k$, account for

$$\frac{\sum_{i=1}^k \sigma_i^2}{\sum_{i=1}^n \sigma_i^2}$$

of the total variance of the data

A good choice of k keeps the true *signal* and discards the *noise*



Principal components and principal directions

The j -th principal direction and the j -th principal component of the data matrix M are

$$q_j \in \mathbb{R}^n \quad \text{and} \quad b_j = \begin{bmatrix} \text{row}_1(A) q_j \\ \vdots \\ \text{row}_m(A) q_j \end{bmatrix} \in \mathbb{R}^m$$

These variables are not correlated, and are ordered according to their variances:

$$\langle b_i, b_j \rangle = 0 \text{ for all } i \neq j \quad \text{and} \quad \langle b_i, b_i \rangle = \sigma_i^2 \text{ for each } i$$



Example

The full SVD of A is given by

$$\begin{bmatrix} 3 & 7 \\ -4 & -6 \\ 7 & 8 \\ 1 & -1 \\ -4 & -1 \\ -3 & -7 \end{bmatrix} = \begin{bmatrix} -0.44 & -0.36 \\ 0.43 & 0.01 \\ -0.63 & 0.33 \\ 0.02 & 0.35 \\ 0.18 & -0.70 \\ 0.44 & 0.37 \end{bmatrix} \begin{bmatrix} 16.87 & 0 \\ 0 & 3.92 \end{bmatrix} \begin{bmatrix} -0.56 & -0.83 \\ 0.83 & -0.56 \end{bmatrix}$$

- The columns of $U\Sigma$ are the first and second principal components, and the columns of V give the first and second principal directions
- The variances and covariances of the principal components are given by

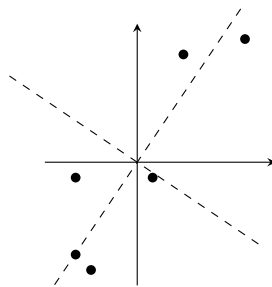
$$\frac{1}{6-1}(AV)^T(AV) = \begin{bmatrix} 56.92 & 0 \\ 0 & 3.07 \end{bmatrix},$$



Example (cont.)

Graphically

$$\begin{bmatrix} 3 & 7 \\ -4 & -6 \\ 7 & 8 \\ 1 & -1 \\ -4 & -1 \\ -3 & -7 \end{bmatrix} = \begin{bmatrix} -0.44 & -0.36 \\ 0.43 & 0.01 \\ -0.63 & 0.33 \\ 0.02 & 0.35 \\ 0.18 & -0.70 \\ 0.44 & 0.37 \end{bmatrix} \begin{bmatrix} 16.87 & 0 \\ 0 & 3.92 \end{bmatrix} \begin{bmatrix} -0.56 & -0.83 \\ 0.83 & -0.56 \end{bmatrix}$$



Navigation icons

The rank deficient LSP revisited

The SVD also applies to the LSP, and it is particularly appropriate in the rank deficient case

Let A be an $m \times n$ matrix of rank r . If $r < n$ then the solution x_{\min} of the LSP

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2$$

is not unique since

$$Ax_{\min} = A(x_{\min} + y)$$

for any $y \in \text{Ker}(A) \simeq \mathbb{R}^{n-r}$

A reasonable choice is the minimizer having the *smallest norm*

Navigation icons

The full and the reduced SVD's

$${}_m \begin{bmatrix} A \end{bmatrix}^n = {}_m \begin{bmatrix} U_r \end{bmatrix}^m [\Sigma] [\begin{bmatrix} V \end{bmatrix}]^T = {}_m \begin{bmatrix} U_r \end{bmatrix}^m [\Sigma_r] [\begin{bmatrix} V_r \end{bmatrix}]^T,$$

that is, if $U = [u_1 \cdots u_m]$ and $V = [v_1 \cdots v_n]$ then $U_r = [u_1 \cdots u_r]$ and $V_r = [v_1 \cdots v_r]$, and if

$$\Sigma = \begin{bmatrix} \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_r & \\ & & & 0 \end{bmatrix}$$

then $\Sigma_r = \text{diag}(\sigma_1, \dots, \sigma_r)$

Navigation icons

The Moore-Penrose inverse and the LSP

The *Moore-Penrose inverse* (or *pseudo-inverse*) of A is the $n \times m$ matrix

$$A^+ = V_r \Sigma_r^{-1} U_r^T$$

\rightsquigarrow the solution of the LSP can be written as

$$x_{\min} = A^+ b$$

When A is rank deficient, it is the solution with the *smallest norm*

Navigation icons

The Moore-Penrose inverse and the LSP (cont.)

The solution given by the pseudo-inverse is well-conditioned when the smallest nonzero singular value of A is not too small:

Changing b to $b + \delta b$ changes x to $x + \delta x$ with

$$\|\delta x\|_2 \leq \frac{\|\delta b\|_2}{\sigma_r}$$

because

$$\delta x = V_r \Sigma_r^{-1} U_r^T \delta b$$

and so

$$\|\delta x\|_2 \leq \|\Sigma_r^{-1}\|_2 \|\delta b\|_2 = \frac{\|\delta b\|_2}{\sigma_r}$$

A practical strategy

The rank is not continuous, and so it might be affected by small perturbations: in the example, round off will make perturbations of size

$$O(\varepsilon) \|A\|_2$$

that might increase the condition number from $\frac{1}{\sigma_r}$ to $\frac{1}{\varepsilon}$

The SVD is *backward stable*: round off gives

$$(U + \delta U)(\Sigma + \delta \Sigma)(V + \delta V)^T = A + \delta A$$

with $\|\delta A\|_2 \leq O(\varepsilon) \|A\|_2$

\rightsquigarrow the computed singular values $\sigma_i + \delta \sigma_i$ verify

$$|\delta \sigma_i| \leq O(\varepsilon) \|A\|_2$$

Example

Let

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Then

$$A^+ = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

and so

$$\hat{x} = A^+ b = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

with condition number $\frac{1}{\sigma_1} = 1$

Setting $A_\varepsilon = \begin{bmatrix} 1 & 0 \\ 0 & \varepsilon \end{bmatrix}$ for $\varepsilon > 0$ gives

$$\hat{x}_\varepsilon = \begin{bmatrix} 1 \\ \frac{1}{\varepsilon} \end{bmatrix}$$

A practical strategy

Let $\text{tol} > 0$ be a user supplied measure of uncertainty in A , e.g.

$$\text{tol} = C \varepsilon \|A\|_2$$

for some small constant $C > 0$

Given the computed factors in the SVD of A

$$\tilde{U}, \quad \tilde{\Sigma}, \quad \tilde{V}$$

set

$$\hat{\sigma}_i = \begin{cases} \tilde{\sigma}_i & \text{if } \tilde{\sigma}_i \geq \text{tol} \\ 0 & \text{else} \end{cases}$$

A practical strategy

Replace $\tilde{\Sigma}$ by the *truncated SVD*

$$\hat{\Sigma} = \begin{bmatrix} \tilde{\sigma}_1 & & & & & \\ & \ddots & & & & \\ & & \tilde{\sigma}_r & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{bmatrix}$$

Then setting

$$\hat{x} = \tilde{V}_r \hat{\Sigma}_r^{-1} \tilde{U}_r^T$$

the error is bounded by $O(\text{tol})$ and the condition number by $\frac{1}{\sigma_r}$



The score vector

Suppose that our web of interest contains n pages, each indexed by an integer $1 \leq k \leq n$

We denote by x_k the *score* of the page k : nonnegative real numbers such $x_j > x_k$ indicates that the page j is more important than the page k

The *score vector*

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}_{\geq 0}^n$$

is normalized so that $\sum_{k=1}^n x_k = 1$



The PageRank algorithm and the eigenvalue problem

PageRank is the basic algorithm of the Google search machine. It had a huge influence on the development and structure of the internet, since it determines which kind of information and services are accessed more often

There are three main steps for ranking web pages:

- 1 Crawl the web and locate all public pages
- 2 Index the data from (1) to allow to search for keywords and phrases within it
- 3 Rate the importance of these pages

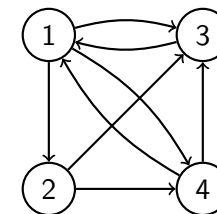
We assume that (1) and (2) are given, and focus on (3):

How can we define and quantify the “importance” of webpages?



A directed graph

The web can be interpreted as a *directed graph*, where the pages correspond to the nodes and the links to the arrows, e.g.

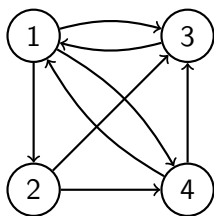


The basic idea

The web is understood as a “democracy” where pages “vote” for the importance of the other pages by linking to them

The simplest approach would consist in *counting links*: in the example this would give

$$x_1 = \frac{2}{8}, \quad x_2 = \frac{1}{8}, \quad x_3 = \frac{3}{8}, \quad x_4 = \frac{2}{8}$$

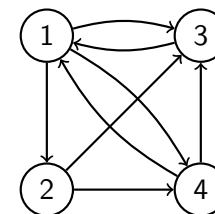


Navigation icons

The basic idea (cont.)

But this forgets an important aspect: receiving a link from an important page should count more!

For instance, pages 1 and 4 have the same number of backlinks (links pointing to them), but page 1 is linked by the important page 3, whereas 4 is linked by the less important page 1



Navigation icons

The basic idea (cont.)

Another aspect to be taken into account is that a page should not increase its influence by increasing its number of outgoing links

If the page j has n_j links, then each should contribute with the score x_j/n_j to the page they link

↪ the overall influence of the page j is its score x_j

The basic idea (cont.)

The score vector should satisfy the equations

$$x_k = \sum_{j \in L_k} \frac{x_j}{n_j}, \quad k = 1, \dots, n,$$

with $L_k \subset \{1, \dots, n\}$ the subset of the indices of the pages linking to the page k

The corresponding *link matrix* $A \in \mathbb{R}^{n \times n}$ is

$$A = \begin{cases} \frac{1}{n_j} & \text{if the page } j \text{ links to the page } k \\ 0 & \text{else} \end{cases}$$

The system of linear equations above is equivalent to

$$Ax = x$$

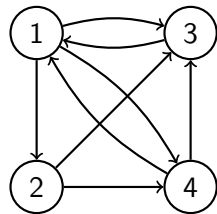
The score vector is an eigenvector of A with eigenvalue 1

Navigation icons

Navigation icons

Example (cont.)

In the example



we have

$$A = \begin{bmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{bmatrix}$$

Navigation icons

Column stochastic matrices

For simplicity: assume that the web has no *dangling nodes*, that is, pages without outgoing links

↪ the link matrix A is *column stochastic*: its entries are nonnegative real numbers and each column sums up to 1:

$$\sum_{i=1}^n a_{i,j} = 1, \quad j = 1, \dots, n$$

Navigation icons

Example (cont.)

The normalized eigenvector of A is

$$x = \begin{bmatrix} \frac{12}{31} \\ \frac{4}{31} \\ \frac{9}{31} \\ \frac{6}{31} \end{bmatrix} = \begin{bmatrix} 0.387 \\ 0.129 \\ 0.290 \\ 0.194 \end{bmatrix}$$

The page 3 (linked by all the others) is less important than the page 1:

Indeed, the page 3 gives all its “vote” to the page 1 and together with the link from the page 2, gives the page 1 the highest score

Navigation icons

Column stochastic matrices (cont.)

Column stochastic matrices have $\lambda = 1$ as one of its eigenvalues: indeed, A and its transposed A^T have the same *characteristic polynomial*:

$$\chi_{A^T} = \det(A^T - t \mathbb{1}_n) = \det(A - t \mathbb{1}_n)^T = \det(A - t \mathbb{1}_n) = \chi_A,$$

↪ the eigenvalues of A and of A^T coincide

Since A is column stochastic then A^T is row stochastic, and so

$$A^T \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

Hence $\lambda = 1$ is an eigenvalue of A^T , and so also an eigenvalue of A

Navigation icons

The eigenspace

Denote by

$$V_1(A) = \{x \in \mathbb{R}^n \mid Ax = x\}.$$

the eigenspace of the eigenvalue $\lambda = 1$

Since this is an eigenvalue of A , we have that $V_1(A) \neq 0$

We would like to have

$$\dim(V_1(A)) = 1$$

so that it defines at most one normalized eigenvector

Not always true...



Example (cont.)

Both

$$x = \begin{bmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \text{and} \quad y = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{2} \\ \frac{1}{2} \\ 0 \end{bmatrix}$$

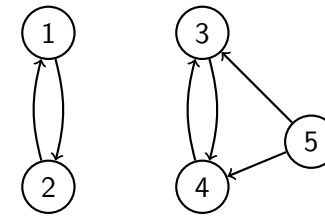
are eigenvectors of A for the eigenvalue 1, and so $V_1(A)$ has dimension at least 2

Which of the vectors in this subspace should be used as a score?



Example

The web



gives the link matrix

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \frac{1}{2} \\ 0 & 0 & 1 & 0 & \frac{1}{2} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$



Disconnected webs

In general, this situation arises when the web is disconnected:

Indeed, if the web consists of t subwebs, then the corresponding link matrix A splits into t blocks, and we have that

$$\dim V_1(A) \geq t$$



A uniform perturbation

To avoid this phenomenon, we modify the link matrix by adding to it a multiple of the *uniform matrix*

$$S = \left[\frac{1}{n} \right]_{i,j} = \begin{bmatrix} \frac{1}{n} & \dots & \frac{1}{n} \\ \vdots & & \vdots \\ \frac{1}{n} & \dots & \frac{1}{n} \end{bmatrix}.$$

Then for $0 \leq \alpha \leq 1$ we set

$$M = (1 - \alpha) A + \alpha S$$

If $\alpha > 0$ then

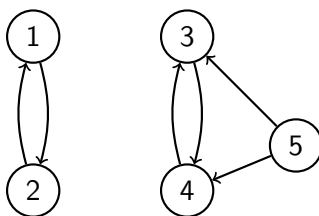
$$\dim V_1(M) = 1$$

Google's value: $\alpha = 0.15$

Navigation icons

Examples (cont.)

For the graph



it gives

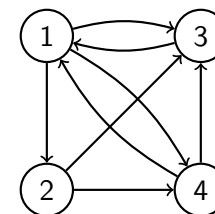
$$x_1 = 0.2, \quad x_2 = 0.2, \quad x_3 = 0.285, \quad x_4 = 0.285, \quad x_5 = 0.003,$$

allowing to compare the different pages

Navigation icons

Examples (cont.)

For the graph



the modified link matrix M gives the scores

$$x_1 = 0.368, \quad x_2 = 0.142, \quad x_3 = 0.288, \quad x_4 = 0.202.$$

Slightly different to those for the link matrix A but giving the same order of importance to the pages

Navigation icons

The eigenvalues of the modified link matrix

The fact that $\dim(V_1(M)) = 1$ follows from the following result:

Theorem: Let

$$\lambda_1 = 1, \lambda_2, \dots, \lambda_n$$

be the eigenvalues of A , repeated according to their multiplicities

Then

$$|\lambda_i| \leq 1 \quad \text{for all } i$$

and the eigenvalues of M are

$$\lambda_1 = 1, (1 - \alpha) \lambda_2, \dots, (1 - \alpha) \lambda_n$$

Navigation icons

Proof

Then

and

because A is column stochastic

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ 🔍 ↺

Proof (cont.)

We have that

and so

Hence the eigenvalues of M are $1, (1 - \alpha)\lambda_2, \dots, (1 - \alpha)\lambda_n$, as stated.

Proof (cont.)

This is a unit n -vector such that $S = e e^T$

where the second equality follows from the fact that A is column stochastic, and the fourth from the fact that e is orthogonal to U_1

Computing the score vector

A set of navigation icons typically found in Beamer presentations, including symbols for back, forward, search, and other slide controls.

Computing the score vector (cont.)

The theorem implies that

$$|\lambda_2| \leq 1 - \alpha = 0.85$$

For the base $b = 10$ we have that

$$\log_b \left(\frac{1}{|\lambda_2|} \right) \geq 0.07$$

and so

$$-\log_b \|x - x_k\|_1 \geq 0.07 k + \text{constant}$$



Eigenvalues

Let A be an $n \times n$ matrix with complex coefficients

An element $\lambda \in \mathbb{C}$ is an *eigenvalue* of A if there is $x \in \mathbb{C}^n \setminus \{0\}$ such that

$$Ax = \lambda x$$

that is, if A is a homothety in the direction of this n -vector

We set

$$\lambda(A) = \{\text{eigenvalues of } A\} \quad \text{the spectrum of } A$$



Computing the score vector (cont.)

Currently there are $\approx 4 \cdot 10^8$ active public webpages

Each iteration of the power method consists of a matrix-vector multiplication. In a practical implementation, it is computed for $v \in \mathbb{R}_{\geq 0}^n$ with $\|v\|_1 = 1$, as

$$M \mathbf{v} = (1 - \alpha) \mathbf{A} \mathbf{v} + \alpha \begin{bmatrix} \frac{1}{n} \\ \vdots \\ \frac{1}{n} \end{bmatrix}$$

because the multiplication of v by the uniform matrix S gives

$$S_v = \begin{bmatrix} \frac{1}{n} \\ \vdots \\ \frac{1}{n} \end{bmatrix}$$



The roots of the characteristic polynomial

The *characteristic polynomial* of the $n \times n$ -matrix A is

$$\chi_A = \det(A - z \mathbb{1}_n) \in \mathbb{C}[z]_n,$$

where $\mathbb{C}[z]_n$ denotes the space of degree n polynomials with complex coefficients

Let $\lambda \in \mathbb{C}$. TFAE:

- $\lambda \in \lambda(A)$
- $A - \lambda \mathbb{1}_n$ is singular
- $\det(A - \lambda \mathbb{1}_n) = 0$
- λ is a root of χ_A

Hence

$$\lambda(A) = \{\lambda \in \mathbb{C} \mid \chi_A(\lambda) = 0\}.$$



Eigenvalues can be complex even if A is real!

For instance, if

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

then $\lambda(A) = \{\pm i\}$

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ↺ 🔍 ↻

Similarities

An $n \times n$ matrix B is *similar* to A if there is a nonsingular $n \times n$ matrix S such that

$$A = S B S^{-1}$$

The matrices A and B have the same eigenvalues, and the eigenvectors of A can be read from those of B :

$$\lambda(A) = \lambda(B),$$

and y is an eigenvector of B for λ if and only if $S y$ is an eigenvector of A for λ

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ 🔍 ↺ ↻

Eigenspaces and multiplicities

For $\lambda \in \lambda(A)$, the associate *eigenspace* is

$$V_\lambda(A) = \{x \in \mathbb{C}^n \mid Ax = \lambda x\}$$

It is an *invariant subspace*:

$$A V_\lambda(A) \subset V_\lambda(A)$$

- The *geometric multiplicity* of λ is the dimension $\dim(V_\lambda(A))$
- The *algebraic multiplicity* of λ , denoted by $e_\lambda(A)$, is the exponent of the factor $\lambda - z$ in the irreducible factorization of χ_A

These quantities are related by

$$1 \leq \dim(V_\lambda(A)) \leq e_\lambda(A)$$

A set of navigation icons typically found in Beamer presentations, including symbols for back, forward, search, and other slide controls.

Similarities (cont.)

If $B = \text{diag}(\lambda_1, \dots, \lambda_n)$ is a diagonal matrix, then

$$e_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}_i$$

is an eigenvector of B for λ_i and so the i -th column in the matrix S

$$s_j = S e_j$$

is an eigenvector of A for λ_i

A set of small navigation icons typically found in Beamer presentations, including symbols for back, forward, search, and other slide controls.

Decoupling

Many eigenvalue computations involve breaking down the problem into smaller ones (*decoupling*): if

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ \mathbb{0} & A_{2,2} \end{bmatrix}$$

then $\lambda(A) = \lambda(A_{1,1}) \cup \lambda(A_{2,2})$

In particular, if A is upper triangular then its eigenvalues coincide with its diagonal entries.



The Schur decomposition (cont.)

The existence of this factorization can be verified by induction:

$n = 1$:

$$Q = [1] \quad \text{and} \quad T = A$$

$n > 1$:

Take an eigenvalue $\lambda \in \lambda(A)$ and a unit eigenvector $u \in \mathbb{C}^n$ of it, and choose an $n \times (n-1)$ matrix \tilde{U} such that $U = [u \tilde{U}]$ is unitary. Then

$$U^* A U = \begin{bmatrix} u^* \\ \tilde{U}^* \end{bmatrix} A \begin{bmatrix} u & \tilde{U} \end{bmatrix} = \begin{bmatrix} u^* A u & u^* A \tilde{U} \\ \tilde{U}^* A u & \tilde{U}^* A \tilde{U} \end{bmatrix}$$

We have that $u^* A u = u^* \lambda u = \lambda$ and $\tilde{U}^* A u = \tilde{U}^* \lambda u = \mathbb{0}$, and so

$$U^* A U = \begin{bmatrix} \lambda & \tilde{a}_{1,2} \\ \mathbb{0} & \tilde{A}_{2,2} \end{bmatrix}$$



The Schur decomposition

An $n \times n$ matrix Q is *unitary* if

$$Q^* = Q^{-1}$$

For reasons of numerical stability, we prefer to consider similarities given by unitary matrices

\leadsto the *Schur decomposition*:

$$A = Q T Q^*$$

with Q unitary and T upper triangular



The Schur decomposition (cont.)

By induction, there is an $(n-1) \times (n-1)$ -unitary matrix P and an $(n-1) \times (n-1)$ -upper triangular matrix \tilde{T} such that

$$\tilde{A}_{2,2} = P \tilde{T} P^*$$

Hence

$$A = U \begin{bmatrix} \lambda & \tilde{a}_{1,2} \\ \mathbb{0} & \tilde{A}_{2,2} \end{bmatrix} U^* = Q T Q^*$$

with

$$Q = U \begin{bmatrix} 1 & \mathbb{0} \\ \mathbb{0} & P \end{bmatrix} \quad \text{and} \quad T = \begin{bmatrix} \lambda & \tilde{a}_{1,2} P \\ \mathbb{0} & \tilde{T} \end{bmatrix}$$



Some remarks

- The Schur decomposition is *not unique*, e.g. the eigenvalues can appear in the diagonal of T in any possible order
- The columns

$$Q = [q_1 \cdots q_n]$$

are called the *Schur vectors* of A . Writing $T = [t_{i,j}]_{i,j}$, for $k = 1, \dots, n$ we have that

$$A q_k = \sum_{i=1}^k t_{i,k} q_i$$

and so the linear subspace $\text{span}(q_1, \dots, q_k)$ is invariant

- When A is real symmetric, it admits a Schur decomposition $A = Q T Q^*$ with Q orthogonal and T diagonal



The Jordan decomposition

It is the factorization

$$A = S J S^{-1}$$

with J a block diagonal matrix $J = [J_1 \cdots J_q]$ where each block is of the form

$$J_i = \begin{bmatrix} \lambda_i & 1 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 \\ 0 & \cdots & 0 & \lambda_i \end{bmatrix} \in \mathbb{C}^{n_i \times n_i}$$

and $n_1 + \cdots + n_q = n$



Diagonalization

A is *diagonalizable* if there exists a nonsingular matrix S and a diagonal matrix $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$ such that

$$A = S \Lambda S^{-1}.$$

This is equivalent to the fact that the geometric and the algebraic multiplicities of the eigenvalues of A coincide, that is,

$$\dim V_\lambda(A) = e_\lambda(A) \quad \text{for all } \lambda \in \lambda(A)$$



The Jordan decomposition (cont.)

This decomposition gives full information about the eigenvalues and the eigenvectors of A :

- for each i , the eigenvalue of J_i is λ_i with eigenvector e_i
- the eigenvalues of A are the λ_i 's
- for each λ_i the basis of the eigenspace $V_\lambda(A)$ is given by the columns of S corresponding to the first columns of the Jordan blocks with eigenvalue λ_i



Drawbacks

- The Jordan decomposition is *not* continuous
- It is difficult to compute numerically for a non diagonalizable matrix
- It cannot be computed stably: after computing S and J , we cannot guarantee that

$$A + \delta A = S J S^{-1}$$

with δA small, because S might have a large condition number



Example 1

The Jordan form of $A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$ is

$$J = A$$

For $\varepsilon_1 \neq \varepsilon_2$ small, the perturbed matrix

$$\tilde{A} = \begin{bmatrix} \varepsilon_1 & 1 \\ 0 & \varepsilon_2 \end{bmatrix}$$

has two different eigenvalues ε_1 and ε_2 , and so its Jordan form is

$$\tilde{J} = \begin{bmatrix} \varepsilon_1 & 0 \\ 0 & \varepsilon_2 \end{bmatrix}$$



Example 2

Let

$$A = \begin{bmatrix} 1 + \varepsilon & 1 \\ 0 & 1 - \varepsilon \end{bmatrix}$$

with ε small. Up to a scalar, the eigenvectors of A are

$$s_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad s_2 = \begin{bmatrix} 1 \\ -2\varepsilon \end{bmatrix}$$

The Jordan decomposition of A is

$$A = \begin{bmatrix} 1 & 1 \\ 0 & -2\varepsilon \end{bmatrix} \begin{bmatrix} 1 + \varepsilon & 0 \\ 0 & 1 - \varepsilon \end{bmatrix} \begin{bmatrix} 1 & \frac{1}{2\varepsilon} \\ 0 & \frac{-1}{2\varepsilon} \end{bmatrix}$$

and so $\kappa_\infty(S) \approx \varepsilon^{-1}$



Computing the eigenvectors from the Schur decomposition

Consider the Schur decomposition

$$A = Q T Q^*$$

and y an eigenvector of T for an scalar λ . Then

$$A Q y = Q T y = \lambda Q y$$

and so $Q y$ is an eigenvector of A with eigenvalue λ

\rightsquigarrow to find the eigenvalues of A it is enough to find those of T



Computing the eigenvectors (cont.)

Suppose that $\lambda = t_{i,j}$ is *simple*, that is $e_\lambda(A) = 1$

Write the equation $(T - \lambda \mathbb{1}_n)y = 0$ as

$$\begin{aligned} 0 &= \begin{matrix} & i-1 & 1 & n-i \\ \begin{matrix} i-1 \\ 1 \\ n-i \end{matrix} & \begin{bmatrix} T_{1,1} - \lambda \mathbb{1}_{i-1} & T_{1,2} & T_{1,3} \\ \mathbb{0} & 0 & T_{2,3} \\ \mathbb{0} & \mathbb{0} & T_{3,3} - \lambda \mathbb{1}_{n-i} \end{bmatrix} \end{matrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \\ &= \begin{bmatrix} (T_{1,1} - \lambda \mathbb{1}_{i-1}) y_1 + T_{1,2} y_2 + T_{1,3} y_3 \\ T_{2,3} y_3 \\ (T_{3,3} - \lambda \mathbb{1}_{n-i}) y_3 \end{bmatrix} \end{aligned}$$



Example

For $T = \begin{bmatrix} 3 & -3 \\ 0 & 2 \end{bmatrix}$ the eigenvector for $\lambda = 2$ is computed as

$$(T - \lambda 1_2) \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & -3 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} y_1 - 3y_2 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

which gives

$$y = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

Hence for a Schur decomposition $A = Q T Q^*$, the eigenvector for $\lambda = 2$ is

$$Q \begin{bmatrix} 3 \\ 1 \end{bmatrix} = 3q_1 + q_2$$



Computing the eigenvectors (cont.)

Since λ is simple, both $T_{1,1} - \lambda \mathbb{1}_{i-1}$ and $T_{3,3} - \lambda \mathbb{1}_{n-i}$ are nonsingular, and so

$$y_3 = 0 \in \mathbb{R}^{i-1}$$

Set $y_2 = 1$ and compute

$$y_1 = -(T_{1,1} - \lambda \mathbb{1}_{i-1})^{-1} T_{1,2} \in \mathbb{R}^{n-i}$$

solving an upper triangular system

The resulting eigenvector is

$$y = \begin{bmatrix} -(T_{1,1} - \lambda \mathbb{1}_{i-1})^{-1} T_{1,2} \\ 1 \\ 0 \end{bmatrix}$$



Perturbation theory

Eigenvalues are continuous with respect to perturbations of the matrix, but they might have an *infinite* condition number, because in general their variation is not bounded by a linear function



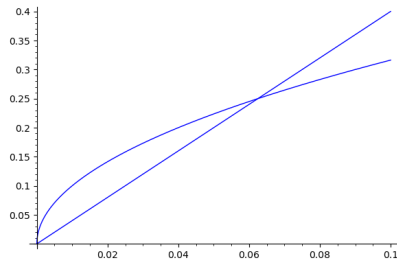
Example

Let

$$A_\varepsilon = \begin{bmatrix} 0 & 1 \\ \varepsilon & 0 \end{bmatrix}$$

with ε small

$\rightsquigarrow \lambda(A_\varepsilon) = \{\pm\varepsilon^{1/2}\}$, and $\varepsilon^{1/2}$ grows faster than $|\varepsilon|$ for $\varepsilon \rightarrow 0$



For $\varepsilon = 0$, the condition number of the eigenvalue of $\lambda = 0$ is $+\infty$

Navigation icons

The condition number

Let A be an $n \times n$ matrix and λ a simple eigenvalue of it. The *condition number* of λ is defined as

$$\kappa(A, \lambda) = \frac{1}{y^* x}$$

where x and y are unit eigenvectors of A for λ and of A^* for $\bar{\lambda}$, respectively:

$$Ax = \lambda x \quad \text{and} \quad y^* A = \bar{\lambda} y^*$$

Navigation icons

Example (cont.)

More formally, set

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad A + \delta A = \begin{bmatrix} 0 & 0 \\ \varepsilon & 0 \end{bmatrix}$$

and let $\lambda = 0$ and $\lambda + \delta\lambda$ be the respective eigenvalues. Hence

$$|\delta\lambda| > c \frac{\|\delta A\|}{\|A\|} \approx \varepsilon$$

for any fixed $c > 0$ and ε sufficiently small

Navigation icons

The condition number (cont.)

For a small perturbation δA we have that

$$\begin{aligned} (A + \delta A) \cdot (x + \delta x) &= (\lambda + \delta\lambda) \cdot (x + \delta x) \\ \frac{A \cdot x + \delta A \cdot x + \delta A \cdot \delta x}{A \cdot x} &= \frac{\lambda x + \delta\lambda x + \delta\lambda \cdot \delta x}{\lambda x} \end{aligned}$$

If we ignore the second order terms and multiply by y^* we obtain

$$y^* \cdot A \cdot \delta x + y^* \cdot \delta A \cdot x = y^* \cdot \lambda \cdot \delta x + y^* \cdot \delta\lambda \cdot x$$

Hence up to second order

$$\delta\lambda = \frac{y^* \cdot \delta A \cdot x}{y^* \cdot x}$$

and so $|\delta\lambda| = \kappa(A, \lambda) \|\delta A\|$

Navigation icons

Example (cont.)

Let

$$A_\varepsilon = \begin{bmatrix} 0 & 1 \\ \varepsilon & 0 \end{bmatrix}$$

with ε small

The eigenvalues of A and A^* for $\lambda_\varepsilon = \varepsilon^{1/2}$ are

$$x_\varepsilon = \frac{1}{(1+\varepsilon)^{1/2}} \begin{bmatrix} 1 \\ \varepsilon^{1/2} \end{bmatrix} \quad \text{and} \quad y_\varepsilon = \frac{1}{(1+\varepsilon)^{1/2}} \begin{bmatrix} \varepsilon^{1/2} \\ 1 \end{bmatrix}$$

Hence

$$\kappa(A_\varepsilon, \lambda_\varepsilon) = \frac{1}{y_\varepsilon^* \cdot x_\varepsilon} = \frac{1 + \varepsilon}{2\varepsilon^{1/2}} \approx \frac{1}{2\varepsilon^{1/2}} \xrightarrow{\varepsilon \rightarrow 0} +\infty$$



Computing eigenvalues and eigenvectors

Let A be an $n \times n$ matrix and consider the *QR iteration*:

$$H_0 \leftarrow Q_0^* A Q_0$$

for $k = 1, 2, \dots$

$$H_{k-1} = Q_k R_k \text{ (QR factorization)}$$

$$H_k \leftarrow R_k Q_k$$

Since $H_k = R_k Q_k = Q_k^* Q_k R_k Q_k = Q_k^* H_{k-1} Q_k$ we have that

$$A = (Q_0 \cdots Q_k) H_k (Q_0 \cdots Q_k)^*$$

and so H_k unitarily similar to A

In most situations

$$H_k \xrightarrow{k \rightarrow \infty} H \quad \text{upper triangular (Schur form)}$$

but this is less obvious!



The symmetric case

When A is symmetric the condition number of a simple eigenvalue is 1: in this case $x = y$ and so

$$\kappa(A, \lambda) = \frac{1}{y^* \cdot x} = \frac{1}{\|x\|_2^2} = 1$$



The power method

Fix a norm $\| \cdot \|$. The *power method* is the iteration

Choose $x_0 \in \mathbb{C}^n$ with $\|x_0\| = 1$

for $k = 0, 1, 2, \dots$

$$y_{k+1} \leftarrow Ax_k$$

$$x_{k+1} \leftarrow \frac{y_{k+1}}{\|y_{k+1}\|}$$

When stopping at an integer l we set

$$\tilde{x} \leftarrow x_{l+1} \quad \text{approximate eigenvector}$$

$$\tilde{\lambda} \leftarrow \tilde{x}^* A \tilde{x} \quad \text{approximate eigenvalue}$$



Error analysis

Suppose A diagonalizable, that is

$$A = S \Lambda S^{-1}$$

with

$$S = [s_1 \cdots s_n] \quad \text{and} \quad \Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$$

and suppose that

$$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$$



Pros and cons

- It is based on matrix-vector multiplications
- Its speed of convergence depends on the ratio $|\lambda_1|/|\lambda_2|$
- It assumes $a_1 \neq 0$

In PageRank:

- Matrix-vector multiplications reduce to multiplications by the link matrix of the web, which is sparse
- The ratio $|\lambda_1|/|\lambda_2|$ is uniformly bounded below:

$$\frac{|\lambda_1|}{|\lambda_2|} \geq 0.85^{-1} \approx 1.18$$

- x_0 is chosen as the score vector of the previous web



Error analysis (cont.)

Write

$$x_0 = a_1 s_1 + \cdots + a_n s_n$$

and suppose furthermore that $a_1 \neq 0$. Then

$$A^k x_0 = a_1 \lambda_1^k s_1 + \cdots + a_n \lambda_n^k s_n = a_1 \lambda_1^k \left(s_1 + \sum_{j=2}^n \frac{a_j}{a_1} \left(\frac{\lambda_j}{\lambda_1} \right)^k s_j \right)$$

and so for $x_k = \frac{A^k x_0}{\|A^k x_0\|}$ and $\tilde{\lambda}_k = x_k^* A x_k$ we have that

$$\|x_k - s_1\|, \|\tilde{\lambda}_k - \lambda_1\| \leq O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)$$

The number of correct digits in base b of these approximations is

$$-\log_b \|x_k - s_1\|_2, -\log_b \|\tilde{\lambda}_k - \lambda_1\|_2 \geq k \log_b \left(\left| \frac{\lambda_1}{\lambda_2} \right| \right) + \text{constant}$$



Pros and cons (cont.)

- It only converges to a pair eigenvalue/eigenvector only if there is a dominant one
- the convergence is only *linear*

The power method does not converge in many situations, including:

- orthogonal matrices (all eigenvalues have absolute value 1)
- real matrices with complex eigenvalues (the eigenvalues come in pairs of conjugate complex numbers)



The QR iteration

Set

$$T_k = Q_k^* A Q_k \quad \text{for } k = 0, 1, 2, \dots$$

The QR iteration arises when considering how to compute T_k directly from T_{k-1}

From the definition of Q_k and R_k and the relation $Q_{k-1} A = R_k Q_k$:

$$T_{k-1} = Q_{k-1}^* A Q_{k-1} = (Q_{k-1}^* Q_k) R_k \quad (6)$$

$$T_k = Q_k^* A Q_k = (Q_k^* A Q_{k-1}) Q_{k-1}^* Q_k = R_k (Q_{k-1}^* Q_k) \quad (7)$$

We have that $Q_{k-1}^* Q_k$ is unitary and (8) is the QR factorization of T_{k-1}



The real QR iteration

Matrices arising from applications have real entries! From now

A real $n \times n$ matrix

The *real QR iteration* is defined by choosing an orthogonal $n \times n$ matrix Q_0 and setting

$$H_0 \leftarrow Q_0^T A Q_0$$

for $k = 1, 2, \dots$

$$H_{k-1} = Q_k R_k \text{ (QR factorization)}$$

$$H_k \leftarrow R_k Q_k$$



Drawbacks

- a single iteration costs $O(n^3)$ flops
- convergence (when it exists) is only linear



The real Schur form

If the eigenvalues of A are not real, then T_k cannot converge to an upper triangular matrix

↪ we content ourselves with convergence to the *real Schur form*:

$$A = Q T Q^T$$

with Q orthogonal and T block upper triangular with 1×1 and 2×2 diagonal blocks, that is

$$T = \begin{bmatrix} T_{1,1} & T_{1,2} & \cdots & T_{1,n} \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & T_{n-1,n} \\ 0 & \cdots & 0 & T_{n,n} \end{bmatrix} \in \mathbb{R}^{n \times n}$$

where each $T_{i,i}$ is a 1×1 block or a 2×2 block with complex conjugate eigenvalues



Hessenberg reduction

To implement the QR iteration, we have to choose Q_0 carefully such that

$$T_0 = Q_0^T A Q_0$$

is an upper Hessenberg matrix, which would lower the complexity to $O(n^2)$ flops



Hessenberg reduction (cont.)

Choose $P_2 = \begin{bmatrix} 1_2 & 0 \\ 0 & \tilde{P}_2 \end{bmatrix}$ with \tilde{P}_2 a 3×3 Householder such that

$$P_2 A_1 = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \end{bmatrix}, \quad A_2 = P_2 A_1 P_2^T = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \end{bmatrix}$$

- P_1 leaves the first and second rows of $P_2 A_1$ unchanged
- P_2^T leaves the first and second columns of $(P_2 A_2) P_2^T$ unchanged



Hessenberg reduction (cont.)

It is a variation of the QR factorization, and can be done with a sequence of Householder reflections

Let $n = 5$ and choose $P_1 = \begin{bmatrix} 1 & 0 \\ 0 & \tilde{P}_1 \end{bmatrix}$ with \tilde{P}_1 a Householder 4×4 matrix such that

$$P_1 A = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \end{bmatrix}, \quad A_1 = P_1 A P_1^T = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \end{bmatrix}$$

- P_1 leaves the first row of $P_1 A$ unchanged
- P_1^T leaves the first column of $(P_1 A) P_1^T$ unchanged, including the zeros



Hessenberg reduction (cont.)

Finally choose $P_3 = \begin{bmatrix} 1_3 & 0 \\ 0 & \tilde{P}_3 \end{bmatrix}$ with \tilde{P}_3 a 2×2 Householder s.t.

$$P_3 A_2 = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}, \quad A_3 = P_3 A_2 P_3^T = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}$$

This constructs an orthogonal $n \times n$ matrix

$$Q_0 = P_{n-2} \cdots P_1$$

such that $H_0 = Q_0^T A Q_0$ is Hessenberg

The complexity of this procedure is $5n^3 + O(n^2)$ flops



The Hessenberg form through the QR iteration

The Hessenberg form is preserved during the QR iteration: let

$$H = Q R \quad \text{and} \quad H_+ = R Q$$

with H Hessenberg

Since R is upper triangular, the j -th column of Q is a linear combination of the first columns of H : write

$$H = [h_1 \cdots h_n] \quad \text{and} \quad Q = [q_1 \cdots q_n]$$

Setting $S = R^{-1} = [s_{i,j}]_{i,j}$ we have that

$$q_j = \sum_{k=1}^j s_{k,j} h_k, \quad j = 1, \dots, n$$

and so Q is Hessenberg

Similarly the j -th row of H_+ is a linear combination of the last j rows of Q , and so H_+ is Hessenberg

Drawbacks

- a single iteration costs $O(n^3)$ flops
- convergence (when it exists) is only linear

The QR iteration

Set

$$T_k = Q_k^* A Q_k \quad \text{for } k = 0, 1, 2, \dots$$

The QR iteration arises when considering how to compute T_k directly from T_{k-1}

From the definition of Q_k and R_k and the relation $Q_{k-1} A = R_k Q_k$:

$$T_{k-1} = Q_{k-1}^* A Q_{k-1} = (Q_{k-1}^* Q_k) R_k \quad (8)$$

$$T_k = Q_k^* A Q_k = (Q_k^* A Q_{k-1}) Q_{k-1}^* Q_k = R_k (Q_{k-1}^* Q_k) \quad (9)$$

We have that $Q_{k-1}^* Q_k$ is unitary and (8) is the QR factorization of T_{k-1}

The real QR iteration

Matrices arising from applications have real entries! From now

A real $n \times n$ matrix

The *real QR iteration* is defined by choosing an orthogonal $n \times n$ matrix Q_0 and setting

$$H_0 \leftarrow Q_0^T A Q_0$$

for $k = 1, 2, \dots$

$$H_{k-1} = Q_k R_k \quad (\text{QR factorization})$$

$$H_k \leftarrow R_k Q_k$$

The real Schur form

If the eigenvalues of A are not real, then T_k cannot converge to an upper triangular matrix

→ we content ourselves with convergence to the *real Schur form*:

$$A = Q T Q^T$$

with Q orthogonal and T block upper triangular with 1×1 and 2×2 diagonal blocks, that is

$$T = \begin{bmatrix} T_{1,1} & T_{1,2} & \dots & T_{1,n} \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & T_{n-1,n} \\ 0 & \dots & 0 & T_{n,n} \end{bmatrix} \in \mathbb{R}^{n \times n}$$

where each $T_{i,i}$ is a 1×1 block or a 2×2 block with complex conjugate eigenvalues



Hessenberg reduction (cont.)

It is a variation of the QR factorization, and can be done with a sequence of Householder reflections

Let $n = 5$ and choose $P_1 = \begin{bmatrix} 1 & 0 \\ 0 & \tilde{P}_1 \end{bmatrix}$ with \tilde{P}_1 a Householder 4×4 matrix such that

$$P_1 A = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \mathbf{0} & \times & \times & \times & \times \\ \mathbf{0} & \times & \times & \times & \times \\ \mathbf{0} & \times & \times & \times & \times \end{bmatrix}, \quad A_1 = P_1 A P_1^T = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \mathbf{0} & \times & \times & \times & \times \\ \mathbf{0} & \times & \times & \times & \times \\ \mathbf{0} & \times & \times & \times & \times \end{bmatrix}$$

- P_1 leaves the first row of $P_1 A$ unchanged
- P_1^T leaves the first column of $(P_1 A) P_1^T$ unchanged, including the zeros



Hessenberg reduction

To implement the QR iteration, we have to choose Q_0 carefully such that

$$T_0 = Q_0^T A Q_0$$

is an upper Hessenberg matrix, which would lower the complexity to $O(n^2)$ flops



Hessenberg reduction (cont.)

Choose $P_2 = \begin{bmatrix} 1_2 & 0 \\ 0 & \tilde{P}_2 \end{bmatrix}$ with \tilde{P}_2 a 3×3 Householder such that

$$P_2 A_1 = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \end{bmatrix}, \quad A_2 = P_2 A_1 P_2^T = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & \times & \times & \times \end{bmatrix}$$

- P_1 leaves the first and second rows of $P_2 A_1$ unchanged
- P_2^T leaves the first and second columns of $(P_2 A_2) P_2^T$ unchanged



Hessenberg reduction (cont.)

Finally choose $P_3 = \begin{bmatrix} 1_3 & 0 \\ 0 & \tilde{P}_3 \end{bmatrix}$ with \tilde{P}_2 a 2×2 Householder s.t.

$$P_3 A_2 = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}, \quad A_3 = P_3 A_2 P_3^T = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \end{bmatrix}$$

This constructs an orthogonal $n \times n$ matrix

$$Q_0 = P_{n-2} \cdots P_1$$

such that $H_0 = Q_0^T A Q_0$ is Hessenberg

The complexity of this procedure is $5n^3 + O(n^2)$ flops



The Hessenberg form through the QR iteration (cont.)

The QR factorization of H is computed with $n - 1$ Givens rotations:

$$Q^T H = R$$

where R is upper triangular and $Q = G_1 \cdots G_{n-1}$ with

$$G_i = \text{Givens}(i, i + 1, \theta_i)$$

Then

$$H_+ = RQ = R(G_1 \cdots G_{n-1})$$

This requires $6n^2 + O(n)$ flops



The Hessenberg form through the QR iteration

The Hessenberg form is preserved during the QR iteration: let

$$H = QR \quad \text{and} \quad H_+ = RQ$$

with H Hessenberg

Since R is upper triangular, the j -th column of Q is a linear combination of the first columns of H : write

$$H = [h_1 \cdots h_n] \quad \text{and} \quad Q = [q_1 \cdots q_n]$$

Setting $S = R^{-1} = [s_{i,j}]_{i,j}$ we have that

$$q_j = \sum_{k=1}^j s_{k,j} h_k, \quad j = 1, \dots, n$$

and so Q is Hessenberg

Similarly the j -th row of H_+ is a linear combination of the last j rows of Q , and so H_+ is Hessenberg



Deflation

A Hessenberg matrix $H \in \mathbb{R}^{n \times n}$ is *reduced* if it has a zero subdiagonal entry

In this case there is $0 < p < n$ such that

$$H = \begin{array}{cc} & \begin{array}{cc} p & n-p \end{array} \\ \begin{array}{cc} p & n-p \end{array} & \begin{bmatrix} H_{1,1} & H_{2,2} \\ \mathbb{O} & H_{2,2} \end{bmatrix} \end{array}$$

and the eigenproblem problem for H decouples into the two smaller problems for $H_{1,1}$ and $H_{2,2}$ (*deflation*)

In practice, this happens when a subdiagonal entry is sufficiently small, for instance when

$$|h_{p+1,p}| \leq c\varepsilon (|h_{p,p}| + |h_{p+1,p+1}|)$$

for ε the machine epsilon and c a small constant

Typically for $p = n - 1$ and $p = n - 2$



The implicit double shift

The *implicit double shift* (or *Francis QR step*) computes this double shift in $O(n^2)$ flops:

- compute $M e_1 =$ multiple of e_1 ,
- determine Householder matrices P_0, P_1, \dots, P_{n-2} such that if $Z = P_0 P_1 \dots P_{n-2}$ then $Z^T H Z$ is Hessenberg and the first columns of Q and of Z are equal.



Implementing the implicit double shift

Set $s = \operatorname{Re}(\lambda)$ and $t = |\lambda|^2$, write

$$M e_1 = \begin{bmatrix} x \\ y \\ z \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

with

$$x = h_{1,1}^2 + h_{1,2} h_{2,1} - s h_{1,1} + t$$

$$y = h_{2,1} (h_{1,1} + h_{2,2} - s)$$

$$z = h_{2,1} h_{3,2}$$



Why does it work?

The implicit QR theorem

If both

$$Q^T A Q = H \quad \text{and} \quad Z^T A Z = Q$$

are Hessenberg, H is unreduced, and Q and Z have the same first column, then

$$G = D^{-1} H D$$

with $D = \text{diag}(\pm 1, \dots, \pm 1)$

- If both $Q^T H Q$ and $Z^T H Z$ are unreduced, then they are essentially equal
- Else the problem decouples into smaller unreduced subproblems



Implementing the implicit double shift

The computation of $M e_1$ and P_0 takes $O(1)$ flops

A similarity with P_0 only changes rows and columns 1, 2 and 3:

$$P_0^T H P_0 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$



Implementing the implicit double shift (cont.)

The mission of P_1, \dots, P_{n-2} is to restore this matrix to Hessenberg form:

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \end{bmatrix} \rightarrow \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \end{bmatrix} \rightarrow \dots$$

Hence

$$P_0 \quad \text{and} \quad Z = P_0 P_1 \cdots P_{n-2}$$

have the same first column

$$\leadsto Z e_1 = Q e_1 \quad \text{and so } Z \text{ and } Q \text{ coincide, up to signs}$$

Basic iterative methods

Let A be a nonsingular $n \times n$ matrix and b an n -vector

Given an *initial* n -vector x_0 , these methods generate a sequence of other n -vectors

$$(x_l)_{l \geq 0}$$

hopefully converging to the solution $x = A^{-1} b$ and where for each $l \geq 0$, the vector x_{l+1} is easy to compute from x_l

Implementing the implicit double shift (cont.)

To implement this strategy, we check when $h_{n-1,n-2}$ is small, in which case the eigenvalues of

$$\begin{bmatrix} h_{n-1,n-1} & h_{n-1,n} \\ h_{n,n-1} & h_{n,n} \end{bmatrix}$$

approximate the eigenvalues of A

In this case

$$s = h_{n-1,n-1} + h_{n,n} \quad (\text{trace})$$

$$t = h_{n-1,n-1} h_{n,n} - h_{n-1,n} h_{n,n-1} \quad (\text{determinant})$$

gives quadratic convergence

On the average, two QR iterations are needed per eigenvalue, and the overall process costs

$$O(n^3) \text{ flops}$$

Splittings

A *splitting* of A is a decomposition

$$A = M - K$$

with M nonsingular

\leadsto iterative method:

$$Ax = b \iff Mx = Kx + b \iff x = M^{-1}Kx + M^{-1}b$$

We then set

$$x_{l+1} = R x_l + c \quad \text{for } l \geq 0 \quad (10)$$

A convergence criterion

The *spectral radius*

$$\rho(R)$$

is the maximum absolute value of the eigenvalues of R

The iteration in (10) converges for every choice of initial vector x_0 if and only if

$$\rho(R) < 1$$



A convergence criterion (cont.)

Conversely, if $\rho(R) \geq 1$ then there is $x_0 \neq x$ such that $x - x_0$ is an eigenvector for an eigenvalue λ of absolute value ≥ 1

Hence

$$x - x_{l+1} = R^{l+1}(x - x_0) = \lambda^{l+1}(x - x_0)$$

which does not approach 0 when $k \rightarrow +\infty$



A convergence criterion (cont.)

Indeed, for every $\varepsilon > 0$ there is a vector norm $\|\cdot\|$ such that the associated operator norm verifies that

$$\|R\| \leq \rho(R) + \varepsilon$$

Hence if $\rho(R) < 1$ then we can choose ε sufficiently small so that the associated vector norm $\|\cdot\|$ verifies that $\|R\| < 1$

Then

$$\begin{aligned} x &= Rx + c \\ \Rightarrow x_{l+1} &= Rx_l + c \\ \Rightarrow x - x_{l+1} &= R(x - x_l) \end{aligned}$$

and so in this case

$$\|x - x_{l+1}\| = \|R(x - x_l)\| \leq \|R\| \|x - x_l\| \leq \|R\|^{l+1} \|x - x_0\| \rightarrow 0$$

when $l \rightarrow +\infty$



The speed of convergence

When $\rho(R) < 1$ the approximation of the l -th iteration is

$$-\log_b \|x - x_l\| \geq -l \log_b \rho(R) - \log_b \|x - x_0\|$$

\rightsquigarrow the increase in precision at each step is *linear* with rate

$$-\log_b \rho(R)$$

Our goal is to find a splitting $A = M - K$ verifying the conditions

- $R = M^{-1}K$ and $c = M^{-1}b$ are easy to compute
- $\rho(R)$ is small



The classical iterative methods

Suppose that no diagonal entry of A is zero and set

$$A = D - \tilde{L} - \tilde{U} = D(1_n - L - U)$$

with

- D is diagonal
- \tilde{L} and L strictly lower triangular
- \tilde{U} and U are strictly upper triangular

Gauss-Seidel method

The *Gauss-Seidel method* takes advantage of the previously computed coordinates $x_{l+1,k}$, $k = 1, \dots, j-1$:

$$\text{for } j = 1, \dots, n$$

$$x_{l+1,j} \leftarrow \frac{1}{a_{j,j}} \left(b_j - \underbrace{\sum_{k < j} a_{j,k} x_{l+1,k}}_{\text{updated x's}} - \underbrace{\sum_{k > j} a_{j,k} x_{l,k}}_{\text{older x's}} \right)$$

Hence for all j

$$a_{j,1}x_{l+1,1} + \cdots + a_{j,j-1}x_{l+1,j-1} + a_{j,j}x_{l+1,j} + a_{l+1,j+1}x_{l,j+1} + \cdots + a_{l,n}x_{l,n} = b_j$$

In matrix notation, $x_{l+1} = R_{GS} x_l + c_{GS}$ with

$$\begin{aligned} R_{\text{GS}} &= (D - \tilde{L})^{-1} \tilde{U} = (\mathbb{1}_n - L)^{-1} U \\ c_{\text{GS}} &= (D - \tilde{L})^{-1} b = (\mathbb{1}_n - L)^{-1} D^{-1} b \end{aligned}$$

Jacobi's method

Jacobi's method can be interpreted as going successively through the equations, changing at the j -th step of the $(l + 1)$ th iteration the variable x_j so that the j -th equation is satisfied:

$$\text{for } j = 1, \dots, n$$

$$x_{l+1,j} \leftarrow \frac{1}{a_{j,j}} (b_j - \sum_{k \neq j} a_{j,k} x_{l,k})$$

Hence for all j we have that

$$a_{j,1}x_{l,1} + \cdots + a_{j,j-1}x_{l,j-1} + a_{j,j}x_{l+1,j} + a_{l,j+1}x_{l,j+1} + \cdots + a_{l,n}x_{l,n} = b_j$$

In matrix notation, $x_{l+1} = R_J x_l + c_J$ with

$$R_J = D^{-1}(\tilde{L} + \tilde{U}) = L + U \quad \text{and} \quad c_J = D^{-1}b$$

SOR

Successive overrelaxation for a parameter $\omega \in \mathbb{R}$ is a weighted average of the vectors x_{l+1} and x_l from the Gauss-Seidel method:

$$x_{l+1}^{\text{SOR}(\omega)} = (1 - \omega) x_l^{\text{GS}} + \omega x_{l+1}^{\text{GS}}$$

for $j = 1, \dots, n$

$$x_{l+1,j} \leftarrow (1 - \omega)x_{l,j} + \frac{\omega}{a_{j,i}} (b_j - \sum_{k < j} a_{j,k} x_{l+1,k} - \sum_{k > j} a_{j,k} x_{l,k})$$

In matrix notation, $x_{l+1} = R_{\text{SOR}(\omega)} x_l + c_{\text{SOR}(\omega)} b$ with

$$\begin{aligned} R_{\text{SOR}(\omega)} &= (D - \omega \tilde{L})^{-1} ((1 - \omega) D + \omega \tilde{U}) \\ &= (\mathbb{1}_n - \omega L)^{-1} ((1 - \omega) \mathbb{1}_n + \omega U) \\ c_{\text{SOR}(\omega)} &= \omega (D - \omega \tilde{L}) = \omega (\mathbb{1}_n - \omega L) D^{-1} \end{aligned}$$

- $\omega = 1$: Gauss-Seidel method
- $\omega > 1$: *overrelaxation*
- $\omega < 1$: *underrelaxation*

Example

Consider the 2×2 matrix and the 2-vector

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

The solution of $Ax = b$ is the 2-vector

$$x = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

The decompositions $A = D - \tilde{L} - \tilde{U} = D(1_2 - L - U)$ are

$$A = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ -1 & 0 \end{bmatrix} - \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix} \\ = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ \frac{-1}{2} & 0 \end{bmatrix} - \begin{bmatrix} 0 & \frac{-1}{2} \\ 0 & 0 \end{bmatrix} \right).$$

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ 🔍 ↺

Example (cont.)

We also have that

$$R_{\text{GS}} = (\mathbb{1}_2 - L)^{-1} U = \begin{bmatrix} 1 & 0 \\ \frac{1}{2} & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0 & -\frac{1}{2} \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{2} \\ 0 & \frac{1}{4} \end{bmatrix}$$

$$c_{\text{GS}} = (D - \tilde{L})^{-1} b = \begin{bmatrix} \frac{1}{2} & 0 \\ -\frac{1}{4} & \frac{1}{2} \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ -\frac{3}{4} \end{bmatrix}$$

and so the Gauss-Seidel iteration writes as

$$\begin{bmatrix} x_{j+1,1} \\ x_{j+1,2} \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{2} \\ 0 & \frac{1}{4} \end{bmatrix} \begin{bmatrix} x_{j,1} \\ x_{j,2} \end{bmatrix} + \begin{bmatrix} \frac{1}{2} \\ -\frac{3}{4} \end{bmatrix} = \begin{bmatrix} \frac{-x_{j,2}}{2} + \frac{1}{2} \\ \frac{x_{j,2}}{4} - \frac{3}{4} \end{bmatrix}$$

◀ ◻ ▶ ◀ 📄 ▶ ◀ ☰ ▶ ◀ ☰ ▶ ☰ ↺ 🔍 ↻

Example (cont.)

Hence

$$R_J = L + U = \begin{bmatrix} 0 & \frac{-1}{2} \\ \frac{-1}{2} & 0 \end{bmatrix} \quad \text{and} \quad c_J = D^{-1}b = \begin{bmatrix} \frac{1}{2} \\ \frac{-1}{2} \end{bmatrix}$$

and so the corresponding Jacobi iteration writes as

$$\begin{bmatrix} x_{l+1,1} \\ x_{l+1,2} \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{2} \\ -\frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} x_{l,1} \\ x_{l,2} \end{bmatrix} + \begin{bmatrix} \frac{1}{2} \\ -\frac{1}{2} \end{bmatrix} = \begin{bmatrix} -\frac{x_{l,2}}{2} + \frac{1}{2} \\ \frac{x_{l,1}}{2} - \frac{1}{2} \end{bmatrix}$$

The characteristic polynomial of the Jacobi matrix is

$$\chi_{R_J} = \det(R_J - t \mathbb{1}_2) = \det \begin{bmatrix} -t & \frac{-1}{2} \\ \frac{-1}{2} & -t \end{bmatrix} = t^2 - \frac{1}{4}$$

Hence $\lambda(R_J) = \{t \mid \chi_{R_J} = 0\} = \{\pm \frac{1}{2}\}$ and so the spectral radius is

$$\rho(R_J) = \frac{1}{2}$$

→ the method converges to x for every choice of initial vector x_0

Example (cont.)

We have that

$$\chi_{R_{\text{GS}}} = \det(R_{\text{GS}} - t \mathbb{1}_2) = \det \begin{bmatrix} -t & \frac{1}{2} \\ 0 & -t + \frac{1}{4} \end{bmatrix} = t^2 - \frac{1}{4}t$$

Hence $\lambda(R_{\text{GS}}) = \{0, \frac{1}{4}\}$ and so $\rho(R_{\text{GS}}) = \frac{1}{4}$

In this example

$$\rho(R_{\text{GS}}) = \rho(R_{\text{J}})^2$$

and so the Gauss-Seidel method converges with the *double of the speed* of the Jacobi method

◀ ◻ ▶ ◀ ◻ ▶ ◀ ≡ ▶ ◀ ≡ ▶ ≡ ◻ ↺ 🔍 ↻

Example (cont.)

For $\omega \in \mathbb{R}$

$$\begin{aligned} R_{\text{SOR}(\omega)} &= (\mathbb{I}_2 - \omega L)^{-1}((1 - \omega) \mathbb{I}_2 + \omega U) \\ &= \begin{bmatrix} 1 & 0 \\ \omega/2 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 - \omega & -\omega/2 \\ 0 & 1 - \omega \end{bmatrix} = \begin{bmatrix} 1 - \omega & -\frac{\omega}{2} \\ \frac{\omega^2}{2} - \frac{\omega}{2} & \frac{\omega^2}{4} - \omega + 1 \end{bmatrix} \end{aligned}$$

and

$$c_{\text{SOR}(\omega)} = \omega (D - \omega \tilde{L})^{-1} b = \omega \begin{bmatrix} 2 & 0 \\ \frac{\omega}{2} & 2 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} \frac{\omega}{4} \\ -\frac{\omega^2}{4} - \frac{\omega}{2} \end{bmatrix}$$

Hence

$$\begin{aligned} \chi_{R_{\text{SOR}(\omega)}} &= \det(R_{\text{SOR}(\omega)} - t \mathbb{I}_2) = \det \begin{bmatrix} 1 - \omega - t & -\frac{\omega}{2} \\ \frac{\omega^2}{2} - \frac{\omega}{2} & \frac{\omega^2}{4} - \omega + 1 - t \end{bmatrix} \\ &= t^2 + \left(\frac{-\omega^2}{4} + 2\omega - 2 \right) t + (\omega^2 - 2\omega + 1). \end{aligned}$$

Navigation icons

Convergence of the basic iterative schemes

Consider a splitting $A = M - K$ with M nonsingular and the associated iterative scheme

$$x_{I+1} \leftarrow R x_I \quad \text{with } R = M^{-1}K \text{ and } c = M^{-1}b$$

If it converges, its limit $x_\infty = \lim_{I \rightarrow \infty} x_I$ satisfies

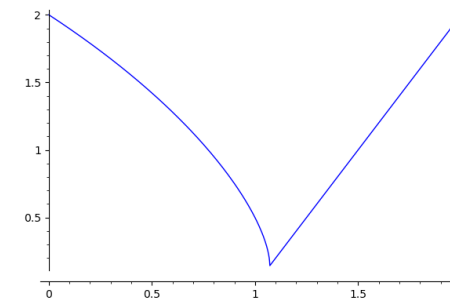
$$x_\infty = R x_\infty + c = M^{-1}K x_\infty + M^{-1}b$$

which implies that x_∞ is the (unique) solution of the equation $Ax = b$

Navigation icons

Example (cont.)

The spectral radius $R_{\text{SOR}(\omega)}$ for $\omega \in [0, 2]$:



The optimal value is $\omega_{\text{opt}} = 1.0717$ and we have that

$$\rho(R_{\text{SOR}(1.0717)}) = 0.1535$$

Navigation icons

Convergence of the basic iterative schemes (cont.)

The convergence of the iteration for an arbitrary initial vector x_0 is equivalent to the condition that

$$\rho(R) < 1$$

In this case, the rate of convergence of the method is linear with coefficient

$$-\log_b \rho(R)$$

In practice, the spectral radius is difficult to compute: need sufficient conditions that are easy to apply!

Navigation icons

Example

The *Richardson iteration* for a parameter $\omega \in \mathbb{R}$ is

$$x_{l+1} \leftarrow R_\omega x_l + \omega b$$

with $R_\omega = \mathbb{1}_n - \omega A$. It corresponds to the splitting

$$A = \omega^{-1} \mathbb{1}_n - (\omega^{-1} \mathbb{1}_n - A)$$

Suppose that the eigenvalues of A are real and ordered as

$$\lambda_1 \geq \dots \geq \lambda_n$$

Then the eigenvalues of R_ω are also real and satisfy that

$$\mu_1 = 1 - \omega \lambda_1 \leq \dots \leq \mu_n = 1 - \omega \lambda_n$$

If $\lambda_n \leq 0$ then $\mu_n \geq 1$ and so $\rho(R_\omega) \geq 1$ and the iteration does not converge



Example (cont.)

The optimal value is given by $-1 + \lambda_1 \omega_{\text{opt}} = 1 - \lambda_n \omega_{\text{opt}}$ and so

$$\omega_{\text{opt}} = \frac{2}{\lambda_1 + \lambda_n}$$

The corresponding spectral radius is

$$\rho(\omega_{\text{opt}}) = \frac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n}$$

- If A has both very large and very small eigenvalues, then the iteration will be slow
- The determination of the optimal parameter ω_{opt} requires knowledge of these eigenvalues



Example (cont.)

Else suppose that $\lambda_n \geq 0$. Then the iteration converges if and only if

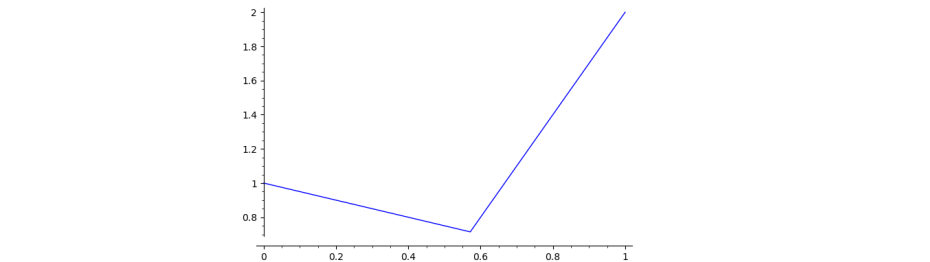
$$-1 \leq 1 - \omega \lambda_1 \quad \text{and} \quad 1 - \omega \lambda_n \leq 1$$

that is, if and only if

$$0 < \omega < \frac{2}{\lambda_1}$$

The spectral radius is the piecewise affine function

$$\rho(R_\omega) = \max(|1 - \omega \lambda_n|, |1 - \omega \lambda_1|)$$



Diagonally dominant matrices

The matrix A is *diagonally dominant* if for all j

$$|a_{j,j}| > \sum_{i \neq j} |a_{i,j}|$$

If A is diagonally dominant then both the Jacobi and the Gauss-Seidel iterations will converge for every choice of initial vector x_0



Diagonally dominant matrices (cont.)

For instance, let

$$R_J = D^{-1}(\tilde{L} + \tilde{U})$$

be the iteration matrix of Jacobi's method

For an eigenvalue $\lambda \in \lambda(R_J)$ let x be a corresponding eigenvector and m the index of its largest component, normalized so that

$$x_m = 1 \quad \text{and} \quad |x_j| \leq 1 \text{ for all } j$$

Then from $R_J x = \lambda x$ we deduce that

$$\lambda x_m = - \sum_{j \neq m} \frac{a_{m,j}}{a_{m,m}} x_j$$

and so

$$|\lambda| \leq \sum_{j \neq m} \frac{|a_{m,j}|}{|a_{m,m}|} |x_j| < 1$$

which proves the result for the Jacobi iteration



Convergence of SOR

Convergence of SOR

For successive overrelaxation, the condition $0 < \omega < 2$ is necessary for the convergence of the method

Indeed

$$R_{\text{SOR}(\omega)} = (\mathbb{I}_n - \omega L)^{-1}((1 - \omega) \mathbb{I}_n + \omega U)$$

and so $\chi_{R_{\text{SOR}(\omega)}}(0) = \det(R_{\text{SOR}(\omega)}) = (1 - \omega)^n$

On the other hand

$$|\chi_{R_{\text{SOR}(\omega)}}(0)| = \prod_{\lambda} |\lambda| \geq \rho(R_{\text{SOR}(\omega)})^n$$

the product being over the eigenvalues of $R_{\text{SOR}(\omega)}$, and so

$$\rho(R_{\text{SOR}(\omega)}) \geq |1 - \omega|$$



When A is symmetric and positive definite,

$$\text{SOR}(\omega)$$

converges for every $0 < \omega < 2$

In particular, in this situation the Gauss-Seidel method ($\omega = 1$) also converges

