Numerical Methods

Rafał Zdunek

Direct methods for solving linear systems (3h.)

(Gaussian elimination and matrix decompositions)

Choose yourself and new technologies

HUMAN CAPITAL
HUMAN – BEST INVESTMENT!

Wrocław University of Technology

EUROPEAN SOCIAL FUND

Project co-financed from the EU European Social Fund

Introduction

- Solutions to linear systems
- Gaussian elimination,
- LU factorization,
- Gauss-Jordan elimination,
- Pivoting

Bibliography

[1] A. Bjorck, Numerical Methods for Least Squares Problems, SIAM, Philadelphia, 1996,
[2] G. Golub, C. F. Van Loan, Matrix Computations, The John Hopkins University Press, (Third Edition), 1996,
[3] J. Stoer R. Bulirsch, Introduction to Numerical Analysis (Second Edition), Springer-Verlag, 1993,
[4] C. D. Meyer, Matrix Analysis and Applied Linear Algebra, SIAM, 2000,
[5] Ch. Zarowski, An Introduction to Numerical Analysis for Electrical and Computer Engineers, Wiley, 2004,
[6] G. Strang, Linear Algebra and Its Applications, Harcourt Brace & Company International Edition, 1998,

Applications

There are two basic classes of methods for solving a system of linear equations: ***direct*** and ***iterative*** methods. Direct methods theoretically give an exact solution in a (predictable) finite number of steps. Unfortunately, this does not have to be true in computational approach due to rounding errors: an error made in one step spreads in all following steps. Classical direct methods usually involve a variety of matrix factorization techniques such as the LU, LDU, LUP, Cholesky, QR, EVD, SVD, and GSVD.

Direct methods are computationally efficient only for a specific class of problems where the system matrix is sparse and reveals some patterns, e.g. it is a banded matrix. Otherwise, iterative methods are usually more efficient.

Solutions to linear systems

A system of linear equations can be expressed in the following matrix form:

$$\mathbf{Ax} = \mathbf{b}\,, \qquad (1)$$

where $\mathbf{A} = [a_{ij}] \in \Im^{M \times N}$ is a coefficient matrix, $\mathbf{b} = [b_i] \in \Im^{M}$ is a data vector, and $\mathbf{x} = [x_j] \in \Im^{N}$ is a solution to be estimated. Let $\mathbf{B} = [\mathbf{A} \quad \mathbf{b}] = \Im^{M \times (N+1)}$ be the augmented matrix to the system (1). The system of linear equations may behave in any one of three possible ways:

A The system has no solution, if $rank(\mathbf{A}) < rank(\mathbf{B})$.

B The system has a single unique solution, if $rank(\mathbf{A}) = rank(\mathbf{B}) = N$.

C The system has infinitely many solutions, if $rank(\mathbf{A}) = rank(\mathbf{B}) < N$.

Solutions to linear systems

If the case B takes place, the system (1) can be basically solved with *direct* or *iterative* methods. When we have the case A (which often occurs in practice), only an approximate solution to (1) can be estimated, usually by formulating a least squares problem. The case C occurs for rank-deficient problems or under-determined problems.

Gaussian elimination (example)

$$\begin{cases} 2x_1 + x_2 + x_3 = 1, \\ 6x_1 + 2x_2 + x_3 = -1, \\ -2x_1 + 2x_2 + x_3 = 7, \end{cases} \begin{matrix} \leftarrow (E_1) \\ \leftarrow (E_2) \\ \leftarrow (E_3) \end{matrix}$$

(Pivot)

$$\begin{cases} 2x_1 + x_2 + x_3 = 1, \\ 6x_1 + 2x_2 + x_3 = -1, \\ -2x_1 + 2x_2 + x_3 = 7, \end{cases} \quad \begin{cases} 2x_1 + x_2 + x_3 = 1, \\ 0x_1 - x_2 - 2x_3 = -4, \\ -2x_1 + 2x_2 + x_3 = 7, \end{cases} \quad (E_2 - 3E_1)$$

$$\begin{cases} 2x_1 + x_2 + x_3 = 1, \\ 0x_1 - x_2 - 2x_3 = -4, \\ 0x_1 + 3x_2 + 2x_3 = 8, \end{cases}$$

$(E_3 - E_1)$

Gaussian elimination (example)

$$\begin{cases} 2x_1 + x_2 + x_3 = 1, \\ 0x_1 - x_2 - 2x_3 = -4, \\ 0x_1 + 3x_2 + 2x_3 = 8, \end{cases}$$

(Pivot)

$$\begin{cases} 2x_1 + x_2 + x_3 = 1, \\ 0x_1 - x_2 - 2x_3 = -4, \\ 0x_1 + 0x_2 - 4x_3 = -4, \end{cases}$$

$(E_3 - 3E_2)$

(Triangularization)

Back-substitution:

$$x_3 = 1,$$

$$x_2 = 4 - 2x_3 = 4 - 2 \cdot 1 = 2,$$

$$x_1 = \frac{1}{2}(1 - x_2 - x_3) = \frac{1}{2}(1 - 2 - 1) = -1,$$

Gaussian elimination (example)

Augmented matrix: $[\mathbf{A}\,|\,\mathbf{b}]$

Let

$$\begin{cases} 2x_1 + x_2 + x_3 = 1, \\ 6x_1 + 2x_2 + x_3 = -1, \\ -2x_1 + 2x_2 + x_3 = 7, \end{cases}$$

$$[\mathbf{A}\,|\,\mathbf{b}] = \left[\begin{array}{ccc|c} 2 & 1 & 1 & 1 \\ 6 & 2 & 1 & -1 \\ -2 & 2 & 1 & 7 \end{array}\right] \underset{\substack{R_2 - 3R_1 \\ R_3 + R_1}}{\rightarrow} \left[\begin{array}{ccc|c} 2 & 1 & 1 & 1 \\ 0 & -1 & -2 & -4 \\ 0 & 3 & 2 & 8 \end{array}\right] \underset{R_3 + 3R_2}{\rightarrow} \left[\begin{array}{ccc|c} 2 & 1 & 1 & 1 \\ 0 & -1 & -2 & -4 \\ 0 & 0 & -4 & -4 \end{array}\right]$$

Gaussian elimination (example)

Recursive technique:

Step 1: $k=1,$ $\mathbf{A}^{(1)}=\mathbf{A},$ $\mathbf{b}^{(1)}=\mathbf{b},$ $\mathbf{A}^{(1)}\mathbf{x}=\mathbf{b}^{(1)},$

$$\left[\mathbf{A}^{(1)}\,|\,\mathbf{b}^{(1)}\right]=\left[\begin{array}{ccc|c} a_{11}^{(1)} & a_{12}^{(1)} & a_{13}^{(1)} & b_1^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & a_{23}^{(1)} & b_2^{(1)} \\ a_{31}^{(1)} & a_{32}^{(1)} & a_{33}^{(1)} & b_3^{(1)} \end{array}\right] \begin{array}{l} \longleftarrow (R_1^{(1)}) \\ \longleftarrow (R_2^{(1)}) \\ \longleftarrow (R_3^{(1)}) \end{array}$$

$$\left[\mathbf{A}^{(2)}\,|\,\mathbf{b}^{(2)}\right]=\left[\begin{array}{ccc|c} a_{11}^{(1)} & a_{12}^{(1)} & a_{13}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & b_2^{(2)} \\ 0 & a_{32}^{(2)} & a_{33}^{(2)} & b_3^{(2)} \end{array}\right]$$

$$R_2^{(2)} \leftarrow R_2^{(1)} - \left(\frac{a_{21}^{(1)}}{a_{11}^{(1)}}\right) R_1^{(1)},$$

$$R_3^{(2)} \leftarrow R_3^{(1)} - \left(\frac{a_{31}^{(1)}}{a_{11}^{(1)}}\right) R_1^{(1)},$$

Gaussian elimination (example)

Step 2: $k = 2,$

$$\left[\mathbf{A}^{(2)} \,|\, \mathbf{b}^{(2)}\right] = \left[\begin{array}{ccc|c} a_{11}^{(1)} & a_{12}^{(1)} & a_{13}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & b_2^{(2)} \\ 0 & 0 & a_{33}^{(3)} & b_3^{(3)} \end{array}\right]$$

$$R_3^{(3)} \leftarrow R_3^{(2)} - \left(\frac{a_{32}^{(2)}}{a_{22}^{(2)}}\right) R_2^{(2)},$$

In general:

$$a_{ij}^{(k+1)} \leftarrow a_{ij}^{(k)} - \left(\frac{a_{ik}^{(k)} a_{kj}^{(k)}}{a_{kk}^{(k)}}\right)$$

Determinant: $$\det\left(\mathbf{A}^{(2)}\right) = \pm \det\left(\mathbf{A}^{(1)}\right)$$

Computational cost

Gaussian Elimination Operation Counts

Gaussian elimination with back substitution applied to an $n \times n$ system requires

$$\frac{n^3}{3} + n^2 - \frac{n}{3} \quad \text{multiplications/divisions}$$

and

$$\frac{n^3}{3} + \frac{n^2}{2} - \frac{5n}{6} \quad \text{additions/subtractions.}$$

As n grows, the $n^3/3$ term dominates each of these expressions. Therefore, the important thing to remember is that Gaussian elimination with back substitution on an $n \times n$ system requires about $n^3/3$ multiplications/divisions and about the same number of additions/subtractions.

Gaussian elimination (example)

$$\begin{cases} x_2 - x_3 = 3, \\ -2x_1 + 4x_2 - x_3 = 1, \\ -2x_1 + 5x_2 - 4x_3 = -2, \end{cases}$$

Augmented matrix:

$$[\mathbf{A}\,|\,\mathbf{b}] = \left[\begin{array}{ccc|c} 0 & 1 & -1 & 3 \\ -2 & 4 & -1 & 1 \\ -2 & 5 & -4 & -2 \end{array}\right] \xrightarrow{\text{Interchange } R_1 \text{ and } R_2} \left[\begin{array}{ccc|c} -2 & 4 & -1 & 1 \\ 0 & 1 & -1 & 3 \\ -2 & 5 & -4 & -2 \end{array}\right]$$

$$\xrightarrow[R_3 - R_1]{} \left[\begin{array}{ccc|c} -2 & 4 & -1 & 1 \\ 0 & 1 & -1 & 3 \\ 0 & 1 & -3 & -3 \end{array}\right] \xrightarrow{R_3 - R_2} \left[\begin{array}{ccc|c} -2 & 4 & -1 & 1 \\ 0 & 1 & -1 & 3 \\ 0 & 0 & -2 & -6 \end{array}\right]$$

Gauss-Jordan elimination

The difference between the Gaussian and Gauss-Jordan elimination:

1. At each step, the pivot element is forced to be 1,

2. At each step, all terms ***above*** the pivot as well as all terms below the pivot are eliminated.

$$\left(\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n \end{array}\right) \longrightarrow \left(\begin{array}{cccc|c} 1 & 0 & \cdots & 0 & s_1 \\ 0 & 1 & \cdots & 0 & s_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & s_n \end{array}\right)$$

Solution

Gauss-Jordan elimination (example)

$$\begin{cases} 2x_1 + 2x_2 + 6x_3 = 4, \\ 2x_1 + x_2 + 7x_3 = 6, \\ -2x_1 - 6x_2 - 7x_3 = -1, \end{cases}$$

$$[\mathbf{A}\,|\,\mathbf{b}] = \left[\begin{array}{ccc|c} 2 & 2 & 6 & 4 \\ 2 & 1 & 7 & 6 \\ -2 & -6 & -7 & -1 \end{array}\right] \underset{R_1/2}{\rightarrow} \left[\begin{array}{ccc|c} 1 & 1 & 3 & 2 \\ 2 & 1 & 7 & 6 \\ -2 & -6 & -7 & -1 \end{array}\right]$$

$$\underset{\substack{R_2-2R_1 \\ R_3+2R_1}}{\rightarrow} \left[\begin{array}{ccc|c} 1 & 1 & 3 & 2 \\ 0 & -1 & 1 & 2 \\ 0 & -4 & -1 & 3 \end{array}\right] \underset{-R_2}{\rightarrow} \left[\begin{array}{ccc|c} 1 & 1 & 3 & 2 \\ 0 & 1 & -1 & -2 \\ 0 & -4 & -1 & 3 \end{array}\right] \underset{\substack{R_1-R_2 \\ R_3+4R_2}}{\rightarrow} \left[\begin{array}{ccc|c} 1 & 0 & 4 & 4 \\ 0 & 1 & -1 & -2 \\ 0 & 0 & -5 & -5 \end{array}\right]$$

$$\underset{-R_3/5}{\rightarrow} \left[\begin{array}{ccc|c} 1 & 0 & 4 & 4 \\ 0 & 1 & -1 & -2 \\ 0 & 0 & 1 & 1 \end{array}\right] \underset{\substack{R_1-4R_3 \\ R_2+R_3}}{\rightarrow} \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 1 \end{array}\right]$$

Solution: $\mathbf{x} = \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix}$,

Computational cost

Gauss–Jordan Operation Counts

For an $n \times n$ system, the Gauss–Jordan procedure requires

$$\frac{n^3}{2} + \frac{n^2}{2} \quad \text{multiplications/divisions}$$

and

$$\frac{n^3}{2} - \frac{n}{2} \quad \text{additions/subtractions.}$$

In other words, the Gauss–Jordan method requires about $n^3/2$ multiplications/divisions and about the same number of additions/subtractions.

Partial Pivoting

Partial Pivoting

At each step, search the positions on and below the pivotal position for the coefficient of *maximum magnitude*. If necessary perform the appropriate row interchange to bring this maximal coefficient into the pivotal position. Illustrated below is the third step in a typical case:

$$\left(\begin{array}{ccccc|c} * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & \textcircled{S} & * & * & * \\ 0 & 0 & S & * & * & * \\ 0 & 0 & S & * & * & * \end{array}\right).$$

Strategy: Maximize the magnitude of the pivot at each step only by using row interchanges.

Partial Pivoting (example)

$$\begin{cases} -10^{-4}x_1 + x_2 = 1, \\ x_1 + x_2 = 2, \end{cases}$$

If real numbers are rounded to 3 floating digits, *without partial pivoting*, we have:

$$[\mathbf{A}\,|\,\mathbf{b}] = \left[\begin{array}{cc|c} -10^{-4} & 1 & 1 \\ 1 & 1 & 2 \end{array}\right] \underset{R_2+10^4 R_1}{\rightarrow} \left[\begin{array}{cc|c} -10^{-4} & 1 & 1 \\ 0 & 10^4 & 10^4 \end{array}\right]$$

Round-off:

$$float\left(1+10^4\right) = float\left(.10001\times 10^5\right) = 10^4, \quad float\left(2+10^4\right) = float\left(.10002\times 10^5\right) = 10^4,$$

Back-substitution gives:

$$\mathbf{x} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

Partial Pivoting (example)

$$\begin{cases} -10^{-4}x_1 + x_2 = 1, \\ x_1 + x_2 = 2, \end{cases}$$

If real numbers are rounded to 3 floating digits, ***with*** *partial pivoting*, we have:

$$[\mathbf{A}\,|\,\mathbf{b}] = \left[\begin{array}{cc|c} -10^{-4} & 1 & 1 \\ 1 & 1 & 2 \end{array}\right] \rightarrow \left[\begin{array}{cc|c} 1 & 1 & 2 \\ -10^{-4} & 1 & 1 \end{array}\right] \underset{R_2+10^{-4}R_1}{\rightarrow} \left[\begin{array}{cc|c} 1 & 1 & 2 \\ 0 & 1 & 1 \end{array}\right]$$

Round-off:

$$float\left(1+10^{-4}\right) = float\left(.10001\times 10^{1}\right) = 1, \qquad float\left(1+2\times 10^{-4}\right) = float\left(.10002\times 10^{1}\right) = 1,$$

Back-substitution gives:

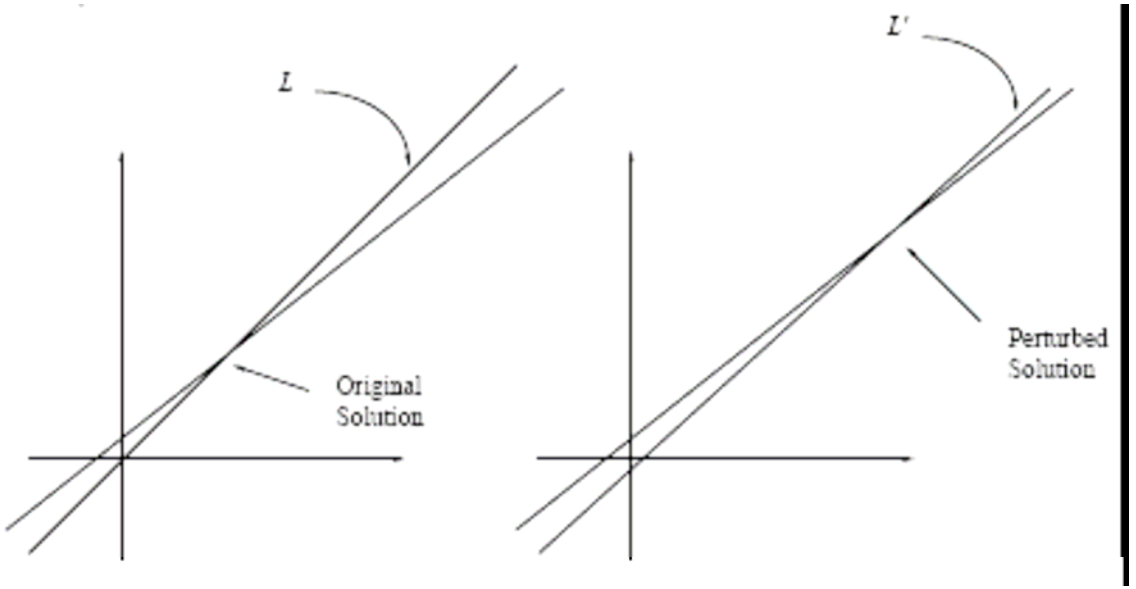
$$\mathbf{x} = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

Ill-conditioned systems

$$\mathbf{Ax} = \mathbf{b}$$

Condition number:

$$\kappa(\mathbf{A}) = \|\mathbf{A}\|_2 \cdot \|\mathbf{A}^{-1}\|_2$$



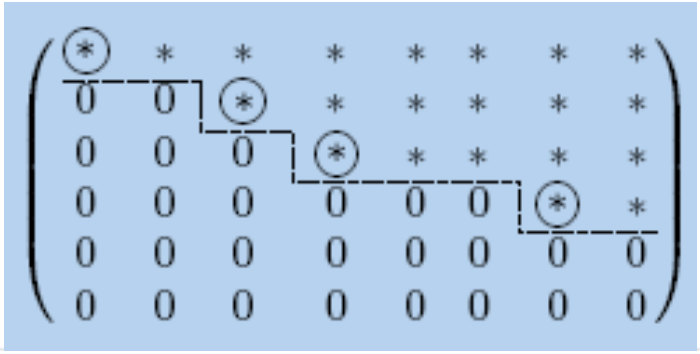
A system of linear equations is said to be ***ill-conditioned*** when some small perturbation in the system can produce relatively large changes in the exact solution. Otherwise, the system is said to be ***well-conditioned.***

Row Echelon Form

Let $\mathbf{A} \in \mathfrak{I}^{m \times n}$, $(\mathbf{A})_{*j}$ - *j*-th column of **A** $(\mathbf{A})_{i*}$ - *i*-th row of **A**

Row Echelon Form:

1. If $(\mathbf{A})_{i*}$ consists entirely of zeros, then all rows below $(\mathbf{A})_{i*}$ are also entirely zero rows;
2. If the first non-zero entry in $(\mathbf{A})_{i*}$ lies in the *j*-th position, then all entries below the *i*-th position in columns $(\mathbf{A})_{*1}$, $(\mathbf{A})_{*2}$, ..., $(\mathbf{A})_{*j}$ are zero.



Rank

Suppose $\mathbf{A} \in \mathfrak{I}^{m \times n}$ is reduced by row operations to the Row Echelon Form (REF). The **rank** of **A** is defined to be the number:

$$\begin{aligned} \textit{rank}(\mathbf{A}) &= \text{number of pivots,} \\ &= \text{number of non-zero rows in REF,} \\ &= \text{number of basic columns in A,} \end{aligned}$$

where the *basic columns* of **A** are these columns that contain pivots.

Reduced Row Echelon Form

Reduced Row Echelon Form (RREF):

1. REF is in row echelon form,
2. The first non-zero entry in each row (i.e. each pivot) is one,
3. All entries above each pivot are zeros.

$$\left(\begin{array}{cccccccc} \textcircled{1} & * & 0 & 0 & * & * & 0 & * \\ 0 & 0 & \textcircled{1} & 0 & * & * & 0 & * \\ 0 & 0 & 0 & \textcircled{1} & * & * & 0 & * \\ 0 & 0 & 0 & 0 & 0 & 0 & \textcircled{1} & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right|$$

RREF (example)

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 2 & 3 & 1 \\ 2 & 4 & 4 & 6 & 2 \\ 3 & 6 & 6 & 9 & 6 \\ 1 & 2 & 4 & 5 & 3 \end{bmatrix} \qquad RREF = \begin{bmatrix} 1 & 2 & 2 & 3 & 1 \\ 2 & 4 & 4 & 6 & 2 \\ 3 & 6 & 6 & 9 & 6 \\ 1 & 2 & 4 & 5 & 3 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 2 & 3 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 2 & 2 & 2 \end{bmatrix}$$

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

We have $rank(\mathbf{A}) = 3,$

Basic columns: $\{\mathbf{A}_{*1}, \mathbf{A}_{*3}, \mathbf{A}_{*5}\}$

Matrix Inversion

Existence of an Inverse

For an $n \times n$ matrix $\mathbf{A}$, the following statements are equivalent

- $\mathbf{A}^{-1}$ exists ($\mathbf{A}$ is nonsingular).
- $rank\,(\mathbf{A}) = n$.
- $\mathbf{A} \xrightarrow{\text{Gauss–Jordan}} \mathbf{I}$.
- $\mathbf{Ax} = \mathbf{0}$ implies that $\mathbf{x} = \mathbf{0}$.

If **A** is nonsingular:

$$[\mathbf{A} \mid \mathbf{I}] \xrightarrow{Gauss-Jordan} [\mathbf{I} \mid \mathbf{A}^{-1}],$$

Matrix Inversion (example)

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

$$[\mathbf{A} \mid \mathbf{I}] = \left[\begin{array}{ccc|ccc} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 2 & 2 & 0 & 1 & 0 \\ 1 & 2 & 3 & 0 & 0 & 1 \end{array}\right] \rightarrow \left[\begin{array}{ccc|ccc} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & -1 & 1 & 0 \\ 0 & 1 & 2 & -1 & 0 & 1 \end{array}\right] \rightarrow \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 2 & -1 & 0 \\ 0 & 1 & 1 & -1 & 1 & 0 \\ 0 & 0 & 1 & 0 & -1 & 1 \end{array}\right]$$

$$\rightarrow \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 2 & -1 & 0 \\ 0 & 1 & 0 & -1 & 2 & -1 \\ 0 & 0 & 1 & 0 & -1 & 1 \end{array}\right]$$

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

Matrix Factorizations

A matrix factorization (or decomposition) decomposes a matrix **A** into a product of other objects (or factors) $\mathbf{F}_1, \ldots, \mathbf{F}_k$:

$$\mathbf{A} = \mathbf{F}_1 \mathbf{F}_2 \cdots \mathbf{F}_k ,$$

where the number of factor matrices *k* depends on the factorization. Most often, $k = 2$ or $k = 3$. There are many different matrix decompositions; each one finds use among a particular class of problems. The most known matrix decompositions or factorizations include: LU, LDU, LUP, Block LU, Cholesky, QR, EVD, SVD, GSVD, Jordan, Jordan-Chevalley, Schur, QZ, Takagi, Polar, Rank factorization, and several forms of NMF.

Gaussian elimination algorithm

for $k = 1$ **to** $n - 1$ ← Sweep over the columns
  for $i = k + 1$ **to** n
    $\ell_{ik} = a_{ik}/a_{kk}$ ← Multipliers for the *k*-th column; They form a lower triangular matrix. **(LU factorization)**
  end
  for $j = k + 1$ **to** n
    for $i = k + 1$ **to** n
      $a_{ij} = a_{ij} - \ell_{ik} a_{kj}$
    end
  end
end

LU factorization

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1(n-1)} & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2(n-1)} & a_{2n} \\ a_{31} & a_{32} & \cdots & a_{3(n-1)} & a_{3n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{n(n-1)} & a_{nn} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ l_{21} & 1 & \cdots & 0 & 0 \\ l_{31} & l_{33} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{n(n-1)} & 1 \end{bmatrix} \cdot \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1(n-1)} & u_{1n} \\ 0 & u_{22} & \cdots & u_{2(n-1)} & u_{2n} \\ 0 & 0 & \cdots & u_{3(n-1)} & u_{3n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & u_{nn} \end{bmatrix}$$

$\mathbf{A} \in \Re^{nxn}$ $\mathbf{L} \in \Re^{nxn}$ $\mathbf{U} \in \Re^{nxn}$

Determinant: $$\det(\mathbf{A}) = \det(\mathbf{L})\det(\mathbf{U}) = 1 \cdot \prod_{i=1}^{n} u_{ii} = \prod_{i=1}^{n} u_{ii}$$

LU factorization

For each *k*, we have $\mathbf{L}^{(k)} = \begin{bmatrix} 1 & 0 & & & 0 \\ 0 & \ddots & \ddots & & \\ & \ddots & 1 & \ddots & \\ & 0 & -l_{(k+1)k} & \ddots & 0 \\ 0 & \cdots & -l_{nk} & 0 & 1 \end{bmatrix}$, where $l_{ik} = \dfrac{a_{ik}^{(k)}}{a_{kk}^{(k)}}$,

Recursive technique: $\mathbf{A}^{(k+1)} = \mathbf{L}^{(k)}\mathbf{A}, \Rightarrow \mathbf{U} = \mathbf{A}^{(n)} = \mathbf{L}^{(n-1)} \ldots \mathbf{L}^{(2)}\mathbf{L}^{(1)}\mathbf{A} = \mathbf{L}^{-1}\mathbf{A},$

Since $\left(\mathbf{L}^{(k)}\right)^{-1} = -\mathbf{L}^{(k)},$ we have

$$\left(\mathbf{L}^{(n-1)} \ldots \mathbf{L}^{(2)}\mathbf{L}^{(1)}\right)^{-1} = \left(\mathbf{L}^{(1)}\right)^{-1}\left(\mathbf{L}^{(2)}\right)^{-1} \ldots \left(\mathbf{L}^{(n-1)}\right)^{-1} = \left(-\mathbf{L}^{(1)}\right)\left(-\mathbf{L}^{(2)}\right)\ldots\left(-\mathbf{L}^{(n-1)}\right)$$

Finally $\boxed{\mathbf{A} = \mathbf{LU}}$

LDU factorization

$$\mathbf{A} = \mathbf{L}\mathbf{D}\tilde{\mathbf{U}}$$

where $\mathbf{D} = diag\left\{\frac{1}{a_{kk}^{(k)}}\right\}$, and $\mathbf{D}\tilde{\mathbf{U}} = \mathbf{U}$,

$$\tilde{\mathbf{U}} = \begin{bmatrix} 1 & u_{12} & \cdots & u_{1(n-1)} & u_{1n} \\ 0 & 1 & \cdots & u_{2(n-1)} & u_{2n} \\ 0 & 0 & \cdots & u_{3(n-1)} & u_{3n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

If **A** is symmetric, $\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^T$

Determinant: $$\det(\mathbf{A}) = \det(\mathbf{L})\det(\mathbf{D})\det(\tilde{\mathbf{U}}) = \prod_{i=1}^{n}\left(d_{kk}^{(k)}\right)^{-1}$$

Crout's algorithm

Set $u_{kk} = 1$ for all k

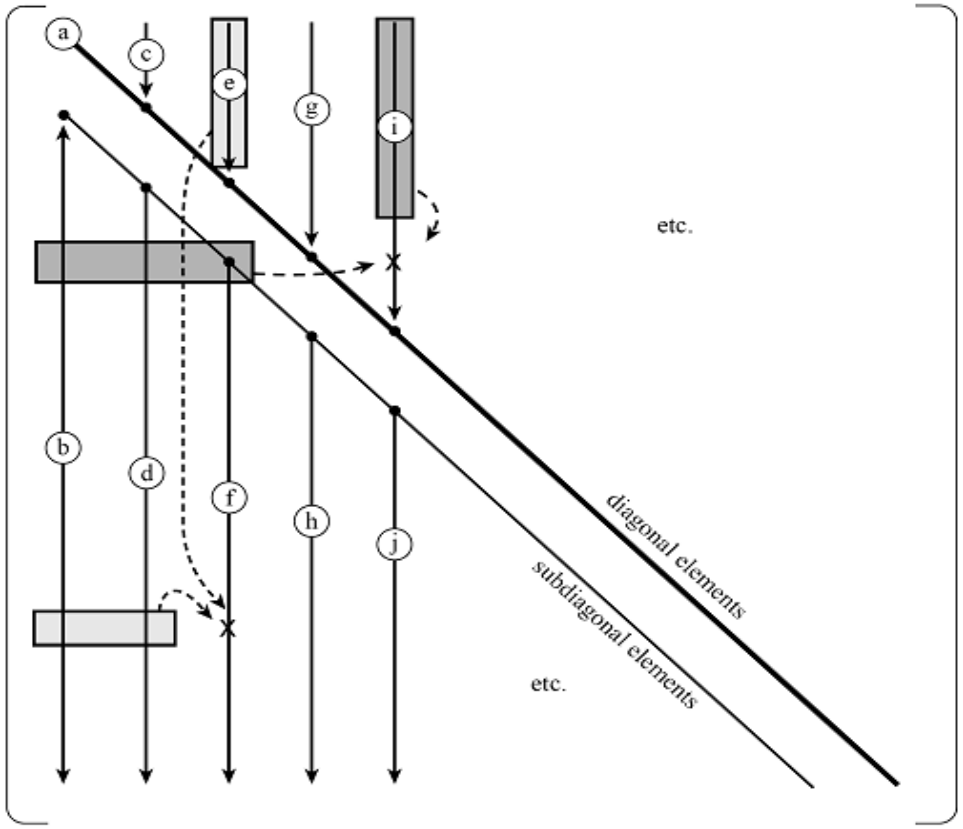
For $k = 1,\ldots,n$

For $i = k, k+1, \ldots, n$

$$l_{ik} = a_{ik} - \sum_{p=1}^{k-1} l_{ip} u_{pk}$$

For $j = k+1,\ldots, n$

$$u_{kj} = \frac{1}{l_{kk}} \left(a_{kj} - \sum_{p=1}^{k-1} l_{kp} u_{pj} \right)$$



LUP factorization

If a **zero pivot** occurs, the basic LU factorization cannot be used.

If so, the partial pivoting should be applied, i.e. by interchanging the rows in **A** to maximize the pivotal element.

$$\mathbf{PA} = \mathbf{LU}$$

P – permutation matrix

If no nonzero pivot is available, **A** is singular.

LUP factorization

The **LUP decomposition** algorithm (by Cormen et al.) generalizes the Crout's method. It can be described as follows:

1. If **A** has a nonzero entry in its first row, then permute the columns in **A** to have a nonzero entry in its upper left corner. Let $\mathbf{A}_1 = \mathbf{A}\mathbf{P}_1$, where $\mathbf{P}_1$ is a permutation matrix.

2. Let $\mathbf{A}_2$ be the matrix obtained from $\mathbf{A}_1$ by deleting both the first row and the first column. Decompose $\mathbf{A}_2 = \mathbf{L}_2\mathbf{U}_2\mathbf{P}_2$ recursively. Make **L** from $\mathbf{L}_2$ by adding a zero row above and then the first column of $\mathbf{A}_1$ at the left.

3. Create $\mathbf{U}_3$ from $\mathbf{U}_2$ by first adding a zero row above and a zero column at the left and then replacing the upper left entry (which is 0 at this point) by 1. Create $\mathbf{P}_3$ from $\mathbf{P}_2$ in a similar manner and define $\mathbf{A}_3 = \mathbf{A}_1\,\mathbf{P}_3^{-1} = \mathbf{A}\mathbf{P}_1\mathbf{P}_3^{-1}$. Then $\mathbf{P} = \mathbf{P}_3\,\mathbf{P}_1^{-1}$.

4. At this point, $\mathbf{A}_3$ is the same as $\mathbf{L}\mathbf{U}_3$, except (possibly) at the first row. If the first row of **A** is zero, then $\mathbf{A}_3 = \mathbf{L}\mathbf{U}_3$, since both have first row zero, and $\mathbf{A} = \mathbf{L}\mathbf{U}_3\mathbf{P}$ follows, as desired. Otherwise, $\mathbf{A}_3$ and $\mathbf{L}\mathbf{U}_3$ have the same nonzero entry in the upper left corner, and $\mathbf{A}_3 = \mathbf{L}\mathbf{U}_3\mathbf{U}_1$ for some upper triangular square matrix $\mathbf{U}_1$ with ones on the diagonal ($\boldsymbol{U}_1$ clears entries of $\mathbf{L}\mathbf{U}_3$ and adds entries of $\mathbf{A}_3$ by changing the upper left corner). Now $\mathbf{A} = \mathbf{L}\mathbf{U}_3\mathbf{U}_1\mathbf{P}$ is a decomposition of the desired form.

Tiangular systems

Let $$\begin{bmatrix} l_{11} & 0 \\ l_{21} & l_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

If $l_{11} l_{22} \neq 0$: $$x_1 = \frac{b_1}{l_{11}}, \qquad x_2 = \frac{(b_2 - l_{21} x_1)}{l_{22}},$$

Forward Substitution: $$x_i = \frac{1}{l_{ii}} \left(b_i - \sum_{j=1}^{i-1} l_{ij} x_j \right),$$ (Lower triangular) for $\mathbf{Lx} = \mathbf{b}$

Back Substitution: $$x_i = \frac{1}{u_{ii}} \left(b_i - \sum_{j=i+1}^{n} u_{ij} x_j \right),$$ (Upper triangular) for $\mathbf{Ux} = \mathbf{b}$

Solving linear systems with LUP

Let $\boxed{\mathbf{Ax} = \mathbf{b}}$, where $\mathbf{A} \in \Re^{m \times n}$ has a full column rank, and $m \geq n$,

then: $\mathbf{PAx} = \mathbf{Pb}$

$\mathbf{PA} = \mathbf{LU}$ (LUP factorization)

1. Solve $\mathbf{Uy} = \mathbf{Pb}$ Back Substitution
2. Solve $\mathbf{Lx} = \mathbf{y}$ Forward Substitution

Cholesky (Banachiewicz) factorization

If **A** is a symmetric and positive-definite matrix, all $d_{ii} > 0$ for $\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^T$.

The Crout's algorithm leads to: $\mathbf{A} = \mathbf{L}\mathbf{L}^T$

For $k = 1,\ldots,n$

$$l_{kk} = \sqrt{a_{kk} - \sum_{p=1}^{k-1} l_{kp}^2}$$

For $j = k+1,\ldots, n$

$$l_{jk} = \frac{1}{l_{kk}}\left(a_{jk} - \sum_{p=1}^{k-1} l_{jp} l_{kp} \right)$$

Another version:

$$\mathbf{A} = \mathbf{R}^T\mathbf{R}$$

R – upper triangular matrix

QR factorization

Let $\mathbf{A} \in \Re^{m \times n}$ have a full column rank, and let $m \geq n$,

QR factorization: $\mathbf{A} = \mathbf{Q} \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}$,

where $\mathbf{Q} \in \Re^{m \times m}$ is an orthogonal matrix, i.e. $\mathbf{Q}^T \mathbf{Q} = \mathbf{I}_m$,

and $\mathbf{R} \in \Re^{n \times n}$ is an upper triangular matrix.

If $\mathbf{A} \in \Im^{m \times n}$ has a full column rank, and $m \geq n$,

then $\mathbf{Q} \in \Im^{m \times m}$ is a unitary matrix, i.e. $\mathbf{Q}^H \mathbf{Q} = \mathbf{I}_m$,

and $\mathbf{R} \in \Im^{n \times n}$ is an upper triangular complex matrix.

QR factorization

If $\mathbf{A} \in \Re^{m \times n}$, $m \geq n$, $r = rank(\mathbf{A})$, and $r < n$, (rank-deficient matrix)

then $\mathbf{Q} \in \Re^{m \times m}$ is an orthogonal matrix, and $\mathbf{R} \in \Re^{n \times n}$ is upper trapezoidal,

that is, the last *n-r* rows have zero-values.

Determinant: $\det(\mathbf{A}) = \det(\mathbf{Q}) \det(\mathbf{R}) = \prod_{i=1}^{n} r_{ii}$

If **A** is a rank-deficient matrix: $r_{ii} = 0$ for $i = r+1, \ldots, n$, then det(A) = 0.

Gram-Schmidt ortogonalization

Given the set $\{v_1, v_2, ..., v_k\}$ of linearly independent vectors, the GS orthogonalization generates the set $\{u_1, u_2, ..., u_k\}$ of orthogonal vectors ($k \leq n$).

The GS orthogonalization algorithm:

$$\mathbf{u}_1 = \mathbf{v}_1,$$

$$\mathbf{u}_2 = \mathbf{v}_2 - P_{\mathbf{u}_1}(\mathbf{v}_2),$$

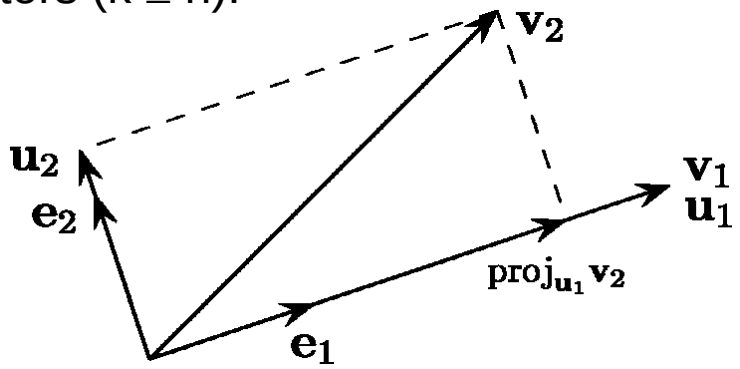
...

$$\mathbf{u}_k = \mathbf{v}_k - \sum_{j=1}^{k-1} P_{\mathbf{u}_j}(\mathbf{v}_k),$$

$$\mathbf{e}_1 = \frac{\mathbf{u}_1}{\|\mathbf{u}_1\|_2}$$

$$\mathbf{e}_2 = \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|_2}$$

$$\mathbf{e}_k = \frac{\mathbf{u}_k}{\|\mathbf{u}_k\|_2}$$



$$P_{\mathbf{u}_j}(\mathbf{v}_k) = \frac{\langle \mathbf{v}_k, \mathbf{u}_j \rangle}{\langle \mathbf{u}_j, \mathbf{u}_j \rangle} \mathbf{u}_j$$

$\{e_1, e_2, ..., e_k\}$ - orthonormal vectors

QR with Gram-Schmidt ortogonalization

Let $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n] \in \Re^{n \times n}$

Applying the GS orthogonalization to the column vectors in **A**, we have:

$$\mathbf{u}_1 = \mathbf{a}_1, \qquad \mathbf{q}_1 = \frac{\mathbf{u}_1}{\|\mathbf{u}_1\|_2}$$

$$\mathbf{u}_2 = \mathbf{a}_2 - P_{\mathbf{u}_1}(\mathbf{a}_2), \qquad \mathbf{q}_2 = \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|_2}$$

...

$$\mathbf{u}_n = \mathbf{a}_n - \sum_{j=1}^{n-1} P_{\mathbf{u}_j}(\mathbf{a}_n), \qquad \mathbf{q}_n = \frac{\mathbf{u}_n}{\|\mathbf{u}_n\|_2}$$

$$P_{\mathbf{u}_j}(\mathbf{a}_k) = \frac{\langle \mathbf{a}_k, \mathbf{u}_j \rangle}{\langle \mathbf{u}_j, \mathbf{u}_j \rangle} \mathbf{u}_j$$

Thus $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n] \in \Re^{n \times n}$

QR with Gram-Schmidt ortogonalization

Then $\mathbf{a}_1 = \langle \mathbf{q}_1, \mathbf{a}_1 \rangle \mathbf{q}_1,$

$\mathbf{a}_2 = \langle \mathbf{q}_1, \mathbf{a}_2 \rangle \mathbf{q}_1 + \langle \mathbf{q}_2, \mathbf{a}_2 \rangle \mathbf{q}_2,$

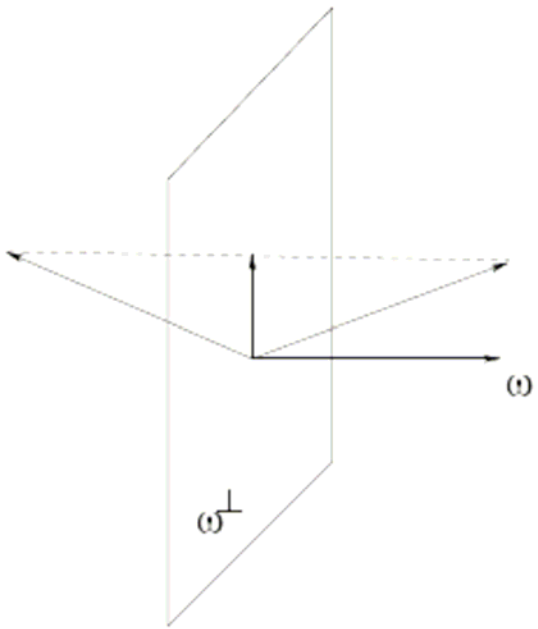
...

$$\mathbf{a}_n = \sum_{j=1}^{n} \langle \mathbf{q}_j, \mathbf{a}_n \rangle \mathbf{q}_j, \qquad \text{where} \qquad \|\mathbf{u}_j\| = \langle \mathbf{q}_j, \mathbf{a}_j \rangle$$

Thus

$$\mathbf{R} = \begin{bmatrix} \langle \mathbf{q}_1, \mathbf{a}_1 \rangle & \langle \mathbf{q}_1, \mathbf{a}_2 \rangle & \cdots & \langle \mathbf{q}_1, \mathbf{a}_n \rangle \\ 0 & \langle \mathbf{q}_2, \mathbf{a}_2 \rangle & \cdots & \langle \mathbf{q}_2, \mathbf{a}_n \rangle \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \langle \mathbf{q}_n, \mathbf{a}_n \rangle \end{bmatrix}$$

Householder reflections



Let $\mathbf{w} \in \Re^n, \quad \|\mathbf{w}\|_2 = 1,$ Householder transformation:

$$\mathbf{H} = \mathbf{I}_n - 2\mathbf{w}\mathbf{w}^T$$

Properties:

$\mathbf{H} = \mathbf{H}^T$ (symmetric)

$\mathbf{H}^T\mathbf{H} = \mathbf{I}_n$ (orthogonal)

For $\mathbf{x} \in \Re^n$: $\mathbf{H}\mathbf{x} = \mathbf{x} - 2\left(\mathbf{w}^T\mathbf{x}\right)\mathbf{w}$ (reflection)

Assuming $\mathbf{w} = \dfrac{\mathbf{x} - \alpha\mathbf{e}_1}{\|\mathbf{x} - \alpha\mathbf{e}_1\|_2},$ $\mathbf{e}_1 = [1, 0, \ldots, 0]^T \in \Re^n$

$\alpha = \pm\|\mathbf{x}\|_2,$ we have $\mathbf{H}\mathbf{x} = \alpha\mathbf{e}_1$

QR with Householder reflections

Let $\mathbf{A} = \left[\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n\right] \in \Re^{m \times n}$, and $\mathbf{A}^{(1)} = \mathbf{A}$,

Recursive algorithm: **Step 1:** $\mathbf{x} := \mathbf{a}_1^{(1)}$, $\mathbf{w}^{(1)} = \dfrac{\mathbf{a}_1^{(1)} - \alpha_1 \mathbf{e}_1}{\left\| \mathbf{a}_1^{(1)} - \alpha_1 \mathbf{e}_1 \right\|_2}$, $\alpha_1 = \pm \left\| \mathbf{a}_1^{(1)} \right\|_2$,

$$\mathbf{H}^{(1)} = \mathbf{I}_m - 2\mathbf{w}^{(1)} \left(\mathbf{w}^{(1)} \right)^T$$

Thus $$\mathbf{H}^{(1)} \mathbf{A}^{(1)} = \begin{bmatrix} \alpha_1 & * & \cdots & * \\ 0 & & & \\ \vdots & & \mathbf{A}^{(2)} & \\ 0 & & & \end{bmatrix} \in \Re^{m \times n}$$

QR with Householder reflections

Recursive algorithm: $\mathbf{A}^{(2)} = \left[\mathbf{a}_1^{(2)}, \mathbf{a}_2^{(2)}, \ldots, \mathbf{a}_{n-1}^{(2)}\right]$, (obtained from $H^{(1)}A$ by deleting the first row and the first column)

Step 2: $\mathbf{x} := \mathbf{a}_1^{(2)}, \quad \mathbf{w}^{(2)} = \dfrac{\mathbf{a}_1^{(2)} - \alpha_2 \mathbf{e}_1}{\left\| \mathbf{a}_1^{(2)} - \alpha_2 \mathbf{e}_1 \right\|_2}, \qquad \alpha_2 = \pm \left\| \mathbf{a}_1^{(2)} \right\|_2,$

$$\mathbf{H}^{(2)} = \mathbf{I}_{m-1} - 2\mathbf{w}^{(2)} \left(\mathbf{w}^{(2)} \right)^T$$

$$\mathbf{H}^{(2)}\mathbf{A}^{(2)} = \begin{bmatrix} \alpha_2 & * & \cdots & * \\ 0 & & & \\ \vdots & & \mathbf{A}^{(3)} & \\ 0 & & & \end{bmatrix} \in \Re^{(m-1)\times(n-1)}$$

QR with Householder reflections

Recursive algorithm: For $k = 1, \ldots, m,$

$$\mathbf{Q}^{(k)} = \begin{bmatrix} \mathbf{I}_{k-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}^{(k)} \end{bmatrix} \in \Re^{m \times m}$$

After t iterations of this recurence, where $t = \min(m\text{-}1, n)$:

$$\mathbf{R} = \mathbf{Q}^{(t)} \cdots \mathbf{Q}^{(2)} \mathbf{Q}^{(1)} \mathbf{A},$$

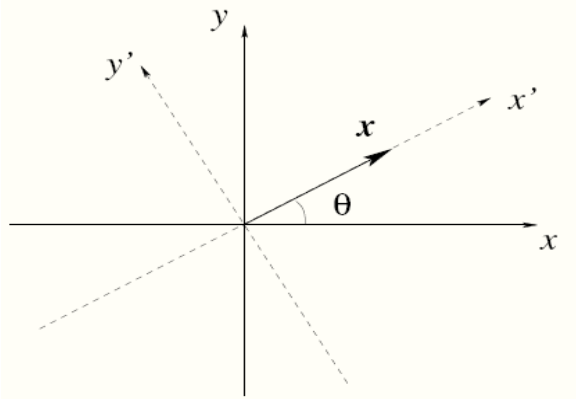
$$\mathbf{Q} = \left(\mathbf{Q}^{(1)}\right)^T \left(\mathbf{Q}^{(2)}\right)^T \cdots \left(\mathbf{Q}^{(t)}\right)^T,$$

This method has much better numerical stability than the QR with GR orthogonalization.

Givens rotations

The **Givens rotation**, represented by an orthogonal matrix **G**, rotates a given vector **x** in the plane spanned by two coordinate axes by an angle θ clockwise.

$$\mathbf{G} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}, \qquad c = \cos\theta, \qquad s = \sin\theta,$$



Let $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, then $\mathbf{y} = \mathbf{G}\mathbf{x} = \begin{bmatrix} cx_1 + sx_2 \\ -sx_1 + cx_2 \end{bmatrix}$.

If $c = \dfrac{x_1}{\sqrt{x_1^2 + x_2^2}}$, $s = \dfrac{x_2}{\sqrt{x_1^2 + x_2^2}}$, then $\mathbf{G}\mathbf{x} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \sqrt{x_1^2 + x_2^2} \\ 0 \end{bmatrix}$

Givens rotations

For $\mathbf{A} \in \Re^{m \times n}$,

$$\mathbf{G}(i,j,\theta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & -s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \in \Re^{m \times m},$$

i ←

j ←

$$\mathbf{G}(i,j,\theta)\mathbf{A} = \begin{cases} a_{i'k} \leftarrow ca_{ik} + sa_{jk} & \text{if } i' = i, j' = j \\ a_{j'k} \leftarrow -sa_{ik} + ca_{jk} & \text{if } i' = i, j' = j \\ a_{i'k} = a_{ik}, \quad a_{j'k} = a_{jk} & \text{otherwise} \end{cases}$$

Givens rotations

Let $\mathbf{A} = \left[\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n\right] \in \Re^{m \times n}$, and $\mathbf{A}^{(1)} = \mathbf{A}$,

thus $\mathbf{G}^{(1)} = \mathbf{G}\left(1,2,\theta\right)\ldots\mathbf{G}\left(1,m-1,\theta\right)\mathbf{G}\left(1,m,\theta\right)$ brings the first column of $\mathbf{A}^{(1)}$ to the vector $\left[\alpha_1, 0, \ldots, 0\right]^T \in \Re^m$.

$$\mathbf{G}^{(1)}\mathbf{A}^{(1)} = \begin{bmatrix} \alpha_1 & * & \cdots & * \\ 0 & & & \\ \vdots & & \mathbf{A}^{(2)} & \\ 0 & & & \end{bmatrix} \in \Re^{m \times n}$$ where $\mathbf{A}^{(2)} \in \Re^{(m-1)\times(n-1)}$

Givens rotations

Then $\mathbf{G}^{(2)} = \mathbf{G}(2,3,\theta)\ldots\mathbf{G}(2,m-1,\theta)\mathbf{G}(2,m,\theta)$ brings the first column of $\mathbf{A}^{(2)}$ to the vector $[*,\alpha_2,0,\ldots,0]^T \in \Re^m$.

$$\mathbf{G}^{(2)}\mathbf{G}^{(1)}\mathbf{A}^{(1)} = \begin{bmatrix} \alpha_1 & * & \cdots & * \\ 0 & \alpha_2 & & \\ \vdots & 0 & \mathbf{A}^{(3)} & \\ 0 & 0 & & \end{bmatrix} \in \Re^{m\times n}, \quad \text{where} \quad \mathbf{A}^{(3)} \in \Re^{(m-2)\times(n-2)}$$

Finally $\mathbf{R} = \mathbf{G}^{(n-1)}\ldots\mathbf{G}^{(2)}\mathbf{G}^{(1)}\mathbf{A}$ and $\mathbf{Q} = \left(\mathbf{G}^{(1)}\right)^T\left(\mathbf{G}^{(2)}\right)^T\ldots\left(\mathbf{G}^{(n-1)}\right)^T$