System of Linear Equations

\[
\begin{align*}
    a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\
    a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\
    \cdots \quad \cdots \quad \cdots \\
    a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n
\end{align*}
\]

where \( a_{ij} \) are coefficients, \( x_i \) are unknowns, and \( b_i \) are right-hand sides.

Written in a compact form is

\[
\sum_{j=1}^{n} a_{ij}x_j = b_i, \quad i = 1, \ldots, n
\]

The system can also be written in a matrix form

\[
 Ax = b
\]

where the matrix is

\[
 A = \begin{bmatrix}
    a_{11} & a_{12} & \cdots & a_{1n} \\
    a_{21} & a_{22} & \cdots & a_{2n} \\
    \vdots & \vdots & \ddots & \vdots \\
    a_{n1} & a_{n2} & \cdots & a_{nn}
\end{bmatrix}
\]

and \( x = [x_1, x_2, \ldots, x_n]^T, b = [b_1, b_2, \ldots, b_n]^T \)
An upper triangular system

\[
\begin{align*}
    a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1 \\
    a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n &= b_2 \\
    a_{33}x_3 + \cdots + a_{3n}x_n &= b_3 \\
    & \vdots \\
    a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_n &= b_{n-1} \\
    a_{nn}x_n &= b_n
\end{align*}
\]

is much easier to find the solution:

\[
x_n = \frac{b_n}{a_{nn}}
\]

from the last equation and substitute its value in other equations and repeat the process

\[
x_i = \frac{1}{a_{ii}} \left( b_i - \sum_{j=i+1}^{n} a_{ij}x_j \right)
\]

for \( i = n - 1, n - 2, \ldots, 1 \)
Karl Friedrich Gauss (April 30, 1777 – February 23, 1855)
German Mathematician and Scientist
Linear systems are solved by Gaussian elimination, which involves repeated procedure of multiplying a row by a number and adding it to another row to eliminate a certain variable.

For a particular step, this amounts to

\[ a_{ij} \leftarrow a_{ij} - \frac{a_{ik}}{a_{kk}} a_{kj} \quad (k \leq j \leq n) \]

\[ b_i \leftarrow b_i - \frac{a_{ik}}{a_{kk}} b_k \]

After this step, the variable \( x_k \) is eliminated in the \((k + 1)\)th and in the later equations.

The Gaussian elimination modifies a matrix into an upper triangular form such that \( a_{ij} = 0 \) for all \( i > j \). The solution of an upper triangular system is then easily obtained by a back substitution procedure.
Illustration of Gaussian Elimination
Given a system of linear algebraic equations

\[
\begin{bmatrix}
\alpha_{11} & \alpha_{12} & \ldots & \alpha_{1n} \\
\alpha_{21} & \alpha_{22} & \ldots & \alpha_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
\alpha_{m1} & \alpha_{m2} & \ldots & \alpha_{mn}
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_m
\end{bmatrix}
= 
\begin{bmatrix}
\delta_1 \\
\delta_2 \\
\vdots \\
\delta_m
\end{bmatrix}
\]

Step 1: Each row times \(\alpha_{11}/\alpha_{k1}\),

then use row one to subtract other rows.

\[
\Rightarrow
\begin{bmatrix}
\alpha_{11} & \alpha_{12} & \ldots & \alpha_{1n} \\
0 & \check{\alpha}_{22} & \ldots & \check{\alpha}_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
0 & \check{\alpha}_{m2} & \ldots & \check{\alpha}_{mn}
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n
\end{bmatrix}
= 
\begin{bmatrix}
\delta_1 \\
\delta_2 \\
\vdots \\
\delta_m
\end{bmatrix}
\]

Step 2: The second row and down multiply by \(\check{\alpha}_{22}/\check{\alpha}_{k2}\),

then use row two to subtract every row below.

\[
\Rightarrow
\begin{bmatrix}
\alpha_{11} & \alpha_{12} & \ldots & \alpha_{1n} \\
0 & \check{\alpha}_{22} & \ldots & \check{\alpha}_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
0 & \check{\alpha}_{m2} & \ldots & \check{\alpha}_{mn}
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n
\end{bmatrix}
= 
\begin{bmatrix}
\delta_1 \\
\delta_2 \\
\vdots \\
\delta_m
\end{bmatrix}
\]

Step 3: Similar to the previous two steps, repeat until all elements in the lower triangle of the matrix \(A\) become zeros.

\[
\Rightarrow
\begin{bmatrix}
\alpha_{11} & \alpha_{12} & \ldots & \alpha_{1n} \\
0 & \check{\alpha}_{22} & \ldots & \check{\alpha}_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & \check{\alpha}_{nn}
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
\vdots \\
x_n
\end{bmatrix}
= 
\begin{bmatrix}
\delta_1 \\
\delta_2 \\
\vdots \\
\delta_m
\end{bmatrix}
\]
The obtained upper triangular system is
\[ a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n = b_1 \]
\[ a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n = b_2 \]
\[ a_{33}x_3 + \cdots + a_{3n}x_n = b_3 \]
\[ \vdots \]
\[ a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_n = b_{n-1} \]
\[ a_{nn}x_n = b_n \]

We can compute
\[ x_n = \frac{b_n}{a_{nn}} \]

from the last equation and substitute its value in other equations and repeat the process
\[ x_i = \frac{1}{a_{ii}} \left( b_i - \sum_{j=i+1}^{n} a_{ij}x_j \right) \]
for \( i = n - 1, n - 2, \ldots, 1 \)
Example: The system of equations \[
\begin{cases}
x + y + z = 3 \\
2x + 3y + 7z = 0 \\
x + 3y - 2z = 17
\end{cases}
\] has augmented matrix
\[
\begin{bmatrix}
1 & 1 & 1 & | & 3 \\
2 & 3 & 7 & | & 0 \\
1 & 3 & -2 & | & 17
\end{bmatrix}
\]
Row operations can be used to express the matrix in row-echelon form.
\[
\begin{bmatrix}
1 & 1 & 1 & | & 3 \\
2 & 3 & 7 & | & 0 \\
1 & 3 & -2 & | & 17
\end{bmatrix}
\rightarrow
\begin{bmatrix}
1 & 1 & 1 & | & 3 \\
0 & 1 & 5 & | & -6 \\
0 & 2 & -3 & | & 14
\end{bmatrix}
\rightarrow
\begin{bmatrix}
1 & 1 & 1 & | & 3 \\
0 & 1 & 5 & | & -6 \\
0 & 0 & -13 & | & 26
\end{bmatrix}
\rightarrow
\begin{bmatrix}
1 & 1 & 1 & | & 3 \\
0 & 1 & 5 & | & -6 \\
0 & 0 & 1 & | & -2
\end{bmatrix}
\]
The system has become \[
\begin{cases}
x + y + z = 3 \\
y + 5z = -6
\end{cases}
\]
By back-substitution we find that \( x = 1, \ y = 4, \) and \( z = -2 \).
A quantity used to measure the quality of a matrix is called condition number, defined as

$$\rho(A) = \|A\| \|A^{-1}\|$$

The condition number measures the transfer of error from the matrix $A$ to the right-hand side vector $b$. If $A$ has a large condition number, small error in $b$ may yield large error in the solution $x = A^{-1}b$. Such a matrix is called ill-conditioned.

The error $e$ is defined as the difference between a computed solution and the exact solution

$$e = x - \tilde{x}$$

Since the exact solution is generally unknown, we measure the residual

$$r = b - A \tilde{x}$$

as an indicator of the size of the error.
Is this BMW ill-conditioned?
Small Pivot

\[ \varepsilon x_1 + x_2 = 1 \]
\[ x_1 + x_2 = 2 \]

for some small \( \varepsilon \). After the step of Gaussian elimination

\[ \varepsilon x_1 + x_2 = 1 \]
\[ (1 - \frac{1}{\varepsilon})x_2 = 2 - \frac{1}{\varepsilon} \]

We have

\[ x_2 = \frac{2 - 1/\varepsilon}{1 - 1/\varepsilon} \]
\[ x_1 = \frac{1 - x_2}{\varepsilon} \]

For very small \( \varepsilon \), the computer result will be \( x_2 = 1 \) and \( x_1 = 0 \). The correct results are

\[ x_1 = \frac{1}{1 - \varepsilon} \approx 1 \]
\[ x_2 = \frac{1 - 2\varepsilon}{1 - \varepsilon} \approx 1 \]
We need to choose an element which is large relative to other elements of the same row as the pivot.

Let $L = (l_1, l_2, \ldots, l_n)$ be an index array of integers. We first compute an array of scale factor as $S = (s_1, s_2, \ldots, s_n)$ where

$$s_i = \max_{1 \leq j \leq n} \left| a_{ij} \right| \quad (1 \leq i \leq n)$$

The first row $i$ is chosen such that the ratio $|a_{i1}|/s_i$ is the greatest. Suppose this index is $l_1$, then appropriate multipliers of equation $l_1$ are subtracted from the other equations to eliminate $x_1$ from the other equations.

Suppose initially $L = (l_1, l_2, \ldots, l_n) = (1, 2, \ldots, n)$, if our first choice is $l_j$, we will interchange $l_j$ and $l_1$ in the index set, not actually interchange the first and the $l_j$ rows, to avoid moving data around the memory. The remaining subsystem uses the same scale factors.
Example

Straightforward Gaussian elimination does not work well (not robust)

\[ \varepsilon x_1 + x_2 = 1 \]
\[ x_1 + x_2 = 2 \]

The scale factor will be computed as \( S = \{1, 1\} \). In the first step, the scale factor ratio array \( \{\varepsilon, 1\} \). So the 2\(^{nd}\) row is the pivoting row

After eliminating \( x_1 \) from the 1\(^{st}\) equation, we have

\[ (1 - \varepsilon)x_2 = 1 - 2\varepsilon \]
\[ x_1 + x_2 = 2 \]

It follows that

\[ x_2 = \frac{1 - 2\varepsilon}{1 - \varepsilon} \approx 1 \]
\[ x_1 = 2 - x_2 \approx 1 \]

We computed correct results by using scaled partial pivoting strategy
Gaussian Elimination with Scaled Partial Pivoting

\[
\begin{bmatrix}
1 & 2 & 3 & 1 \\
2 & 3 & 5 & 0 \\
3 & 4 & 5 & 0 \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
3 & 4 & 5 & 0 \\
2 & 3 & 5 & 0 \\
1 & 2 & 3 & 1 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
3 & 4 & 5 & 0 \\
0 & 1/3 & 5/3 & 0 \\
0 & 2/3 & 4/3 & 1 \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
3 & 4 & 5 & 0 \\
0 & 2/3 & 4/3 & 1 \\
0 & 1/3 & 5/3 & 0 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
3 & 4 & 5 & 0 \\
0 & 2/3 & 4/3 & 1 \\
0 & 0 & 1 & -\frac{1}{2} \\
\end{bmatrix}
\rightarrow
\begin{bmatrix}
3 & 4 & 5 & 0 \\
0 & 2/3 & 4/3 & 1 \\
0 & 0 & 1 & -\frac{1}{2} \\
\end{bmatrix}
\]
We count the number of multiplications and divisions, ignore summations and subtractions in scaled Gaussian elimination

The 1\(^{st}\) step, finding a pivoting costs \(n\) divisions

Additional \(n\) operations are needed to multiply a factor to the pivoting row for each of the \(n - 1\) eliminations. The cost is \(n(n - 1)\) operations. The total cost of this step is \(n^2\) operations

The computation is repeated on the remaining \((n - 1)\) equations. The total cost of Gaussian elimination with scaled partial pivoting is

\[
n^2 + (n-1)^2 + \cdots + 4^2 + 3^2 + 2^2 = \\
\frac{n(n+1)(2n+1)}{6} - 1 \approx \frac{n^3}{3}
\]

Back substitution costs \(n(n - 1)/2\) operations
Banded system has a coefficient matrix such that $a_{ij} = 0$ if $|i - j| \geq w$. For a tridiagonal system, $w = 2$

$$
\begin{bmatrix}
  d_1 & c_1 & & & \\
  a_1 & d_2 & c_2 & & \\
  & a_2 & d_3 & c_3 & \\
  & & \vdots & \ddots & \ddots \\
  & & & a_{n-2} & d_{n-1} & c_{n-1} \\
  & & & a_{n-1} & d_n &
\end{bmatrix}
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  \vdots \\
  x_{n-1} \\
  x_n
\end{bmatrix}
= 
\begin{bmatrix}
  b_1 \\
  b_2 \\
  b_3 \\
  \vdots \\
  b_{n-1} \\
  b_n
\end{bmatrix}
$$

General elimination procedure

$$
d_i \leftarrow d_i - \left( \frac{a_{i-1}}{d_{i-1}} \right) c_{i-1}
$$

$$
b_i \leftarrow b_i - \left( \frac{a_{i-1}}{d_{i-1}} \right) b_{i-1}
$$

The array $c_i$ is not modified. No additional nonzero is created.

Matrix can be stored in three vector arrays.
The back substitution is straightforward

\[ x_n = \frac{b_n}{d_n} \]

\[ x_i = \frac{b_i - c_i x_{i+1}}{d_i} \quad (i = n-1, \ldots, 1) \]

No pivoting is performed, otherwise the procedure will be quite different due to the fill-in (the array \( c \) will be modified)

**Diagonal dominance:** A matrix \( A = (a_{ij})_{n \times n} \) is diagonally dominant if

\[ |a_{ii}| > \sum_{j=1,j\neq i}^{n} |a_{ij}| \quad (1 \leq i \leq n) \]

For a diagonally dominant tridiagonal system, no pivoting is needed, i.e., no division by zero will happen

We want to show Gaussian elimination preserves diagonal dominance, i.e.,

\[ |d_i| > |a_{i-1}| + |c_i| \]
The new coefficient matrix has 0 elements at the $a_i$'s places. The new diagonal elements are determined recursively as

$$\hat{d}_1 = d_1$$

$$\hat{d}_i = d_i - c_{i-1} \left( \frac{a_{i-1}}{d_{i-1}} \right) \quad (2 \leq i \leq n)$$

We assume that

$$|d_i| > |a_{i-1}| + |c_i|$$

We want to show that

$$|\hat{d}_i| > |c_i|$$

We use induction to prove the inequality

It is obviously true for $i = 1$, as

$$\hat{d}_1 = d_1$$
If we assume that

$$|\hat{d}_{i-1}| \geq |c_{i-1}|$$

We prove for index $i$, as

$$|d_i| = |d_i - c_{i-1} \left( \frac{a_{i-1}}{\hat{d}_{i-1}} \right)|$$

$$\geq |d_i| - |a_{i-1}| \left| \frac{c_{i-1}}{\hat{d}_{i-1}} \right|$$

$$> |a_{i-1}| + |c_i| - |a_{i-1}| = |c_i|$$

It follows that the new diagonal entries will not be zero, the Gaussian elimination procedure can be carried out without any problem.
Example of a pentadiagonal matrix. It is nearly tridiagonal. The matrix with only nonzero entries on the main diagonal, and the first two bi-diagonals above and below it:
Matrix for Discrete Poisson Problem

\[
\begin{bmatrix}
4 & -1 & & & & & \\
-1 & 4 & -1 & & & & \\
& -1 & 4 & -1 & & & \\
& & -1 & 4 & -1 & & \\
& & & -1 & 4 & -1 & \\
& & & & -1 & 4 & \\
& & & & & -1 & 4 \\
& & & & & & -1
\end{bmatrix}
\]
LU Factorization 1

As we showed before, an $n \times n$ system of linear equations can be written in a matrix form as

$$Ax = b$$

where the coefficient matrix $A$ has the form

$$A = \begin{bmatrix}
    a_{11} & a_{12} & \cdots & a_{1n} \\
    a_{21} & a_{22} & \cdots & a_{2n} \\
    \vdots & \vdots & \ddots & \vdots \\
    a_{n1} & a_{n2} & \cdots & a_{nn}
\end{bmatrix}$$

$x$ is the unknown vector and $b$ is the right-hand side known vector.

We also assume $A$ is of full rank, and most entries of $A$ are not zero.
LU Factorization 2

There are two special forms of matrices. One is called (unit) lower triangular

\[
L = \begin{bmatrix}
1 & 0 & \cdots & 0 \\
l_{21} & 1 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
l_{n1} & l_{n2} & \cdots & 1
\end{bmatrix}
\]

The other is upper triangular

\[
U = \begin{bmatrix}
u_{11} & u_{12} & \cdots & u_{1n} \\
0 & u_{22} & \cdots & u_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & u_{nn}
\end{bmatrix}
\]

We want to find a pair of \(L\) and \(U\) matrices, such that

\[
A = LU
\]
Example

Take a system of linear equations

\[
\begin{bmatrix}
  6 & -2 & 2 & 4 \\
  12 & -8 & 6 & 10 \\
  3 & -13 & 9 & 3 \\
  -6 & 4 & 1 & -18
\end{bmatrix}
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  x_4
\end{bmatrix}
= 
\begin{bmatrix}
  16 \\
  26 \\
  -19 \\
  -34
\end{bmatrix}
\]

The Gaussian elimination process finally yields an upper triangular system

\[
\begin{bmatrix}
  6 & -2 & 2 & 4 \\
  0 & -4 & 2 & 2 \\
  0 & 0 & 2 & -5 \\
  0 & 0 & 0 & -3
\end{bmatrix}
\begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  x_4
\end{bmatrix}
= 
\begin{bmatrix}
  16 \\
  -6 \\
  -9 \\
  -3
\end{bmatrix}
\]

This could be achieved by multiplying the original system with a matrix \( M \), such that

\[
M\mathbf{x} = \mathbf{Mb}
\]
Example

We want the matrix $M$ to be special, so that $MA$ is upper triangular

$$MA = \begin{bmatrix} 6 & -2 & 2 & 4 \\ 0 & -4 & 2 & 2 \\ 0 & 0 & 2 & -5 \\ 0 & 0 & 0 & -3 \end{bmatrix} = U$$

The question is: can we find such a matrix $M$? Look at the first step of the Gaussian elimination

$$\begin{bmatrix} 6 & -2 & 2 & 4 \\ 0 & -4 & 2 & 2 \\ 0 & -12 & 8 & 1 \\ 0 & 2 & 3 & -14 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 16 \\ -6 \\ -27 \\ -18 \end{bmatrix}$$

This step can be achieved by multiplying the original system with a lower triangular matrix

$$M_1Ax = M_1b$$
Here the lower triangular matrix $M_1$ is

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \end{bmatrix}$$

This matrix is nonsingular, because it is lower triangular with a main diagonal containing all 1’s.

The nonzero elements in the first columns are the negatives of the multipliers in the positions where 0’s were created in the original matrix

$$M_1Ax = M_1b$$
Example

If we continue to the second step of the Gaussian elimination, we have

\[
\begin{bmatrix}
6 & -2 & 2 & 4 \\
0 & -4 & 2 & 2 \\
0 & 0 & 2 & -5 \\
0 & 0 & 4 & -13
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{bmatrix}
= 
\begin{bmatrix}
16 \\
-6 \\
-9 \\
-21
\end{bmatrix}
\]

This can be achieved with the multiplication of another lower triangular matrix

\[
M_2 = 
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & -3 & 1 & 0 \\
0 & \frac{1}{2} & 0 & 1
\end{bmatrix}
\]

Thus, we have

\[
M_2M_1Ax = M_2M_1b
\]
The last step is

\[
\begin{bmatrix}
6 & -2 & 2 & 4 \\
0 & -4 & 2 & 2 \\
0 & 0 & 2 & -5 \\
0 & 0 & 0 & -3 \\
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4 \\
\end{bmatrix}
= 
\begin{bmatrix}
16 \\
-6 \\
-9 \\
-3 \\
\end{bmatrix}
\]

with another lower triangular matrix

\[
M_3 = 
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & -2 & 1 \\
\end{bmatrix}
\]

And, we have

\[M_3 M_2 M_1 A x = M_3 M_2 M_1 b\]
If we define

\[ M = M_3 M_2 M_1 \]

Then \( M \) is lower triangular and \( MA \) is upper triangular

Since

\[ M_3 M_2 M_1 A = U \]

the inverse of a lower triangular is again a lower triangular matrix. The product of two lower triangular matrices is a lower triangular matrix

This shows that we have transformed a matrix \( A \) into the product of a lower triangular matrix and an upper triangular matrix

This process is called \( LU \) factorization or \( LU \) decomposition
Example

The lower triangular matrix is

\[ L = M_1^{-1} M_2^{-1} M_3^{-1} \]

which in the current example is

\[
L = \begin{bmatrix}
1 & 0 & 0 & 0 \\
2 & 1 & 0 & 0 \\
1 & 3 & 1 & 0 \\
-1 & -\frac{1}{2} & 2 & 1
\end{bmatrix}
\]

And, we have

\[
LU = \begin{bmatrix}
1 & 0 & 0 & 0 \\
2 & 1 & 0 & 0 \\
\frac{1}{2} & 3 & 1 & 0 \\
-1 & -\frac{1}{2} & 2 & 1
\end{bmatrix}\begin{bmatrix}
6 & -2 & 2 & 4 \\
0 & -4 & 2 & 2 \\
0 & 0 & 2 & -5 \\
0 & 0 & 0 & -3
\end{bmatrix} = \begin{bmatrix}
6 & -2 & 2 & 4 \\
12 & -8 & 6 & 10 \\
3 & -13 & 9 & 3 \\
-6 & 4 & 1 & -18
\end{bmatrix} = A
\]
Illustration of LU Factorization
The original system

\[ Ax = b \]

can be factored into

\[ LUx = b \]

The solution process can be separated into two phases. First, forward elimination with an intermediate variable \( z \), as

\[ Lz = b \]

Second, back substitution step

\[ Ux = z \]

The advantage of \( LU \) factorization or \( LU \) decomposition is that several linear systems with the same coefficient matrix but different right-hand side vectors can be solved more efficiently. We only need do the factorization once. The forward elimination and back substitution steps are \( O(n^2) \) operations.
The system of linear equations

\[ Ax = b \]

can be symbolically solved as

\[ x = A^{-1}b \]

But the inverse of \( A \) is seldom computed explicitly. This is because performing Gaussian elimination is less expensive.

If the explicit inverse of a matrix \( A \) is needed, it can be obtained by using LU factorization of \( A \), note that

\[ AX = I \]

We can compute a series of linear systems as

\[ LU[x^{(1)}, x^{(2)}, \ldots, x^{(n)}] = [I^{(1)}, I^{(2)}, \ldots, I^{(n)}] \]

And

\[ A^{-1} = [x^{(1)}, x^{(2)}, \ldots, x^{(n)}] \]
Singular Value Decomposition (SVD)

The eigenvalues and eigenvectors of a matrix $A$, are

$$Ax = \lambda x$$

I.e., the application of $A$ on the vector of $x$ is equivalent to a scaling. Here $\lambda$ is an eigenvalue and $x$ is the associated eigenvector.

The singular values of a matrix $A$ are the nonnegative square roots of the eigenvalues of $A^T A$

By the Spectral Theorem for Matrices, $A^T A$ can be diagonalized by an orthogonal matrix, $Q$, as

$$A^T A = Q\Sigma Q^{-1}$$

where (orthogonality property)

$$Q^T Q = QQ^T = I, \quad \text{i.e.,} \quad Q^T = Q^{-1}$$

The diagonal matrix $\Sigma$ contains the eigenvalues of $A^T A$ on its diagonal.
Singular Value Decomposition II

We can see that

$$A^T A Q = Q \Sigma$$

So the columns of $Q$ are eigenvectors of $A^T A$

If $\lambda$ is an eigenvalue of $A^T A$ and $x$ is a corresponding eigenvector, then

$$A^T A x = \lambda x$$

And

$$\| A x \|^2 = (Ax)^T (Ax) = x^T A^T A x = x^T \lambda x = \lambda \| x \|^2$$

It follows that the eigenvalue $\lambda$ is real and nonnegative.

Since $Q$ is an orthogonal matrix, its columns form an orthonormal base. They are unit eigenvectors of $A^T A$

If $v_j$ is the $j$th column of $Q$, we have

$$A^T A v_j = \lambda_j v_j$$
Singular Value Decomposition

- $A$ is an $m$-by-$n$ matrix with $\text{rank}(A) = r$. It could be decomposed into three matrices.
  \[ A_{m\times n} = U D V^T \]
- $U$ is an $m$-by-$m$ orthogonal matrix and $V$ is an $n$-by-$n$ orthogonal matrix.

- $D = \begin{bmatrix} D_{r\times r} & O_{r\times(n-r)} \\ O_{(m-r)\times r} & O_{(m-r)\times(n-r)} \end{bmatrix}_{m\times n}$

- is the singular value matrix, where $D$ is a diagonal matrix with the singular values on its diagonal.
Singular Value Decomposition cont.

The singular values in $D$ have the property $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0$. 
SVD-based Latent Factor Model cont.

Singular Value Decomposition (SVD)
**EXAMPLE OF TRANSACTIONS**

<table>
<thead>
<tr>
<th>User</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>u01</td>
<td>Bread, Milk</td>
</tr>
<tr>
<td>u02</td>
<td>Bread, Diaper, Beer, Eggs</td>
</tr>
<tr>
<td>u03</td>
<td>Milk, Diaper, Beer, Coke</td>
</tr>
<tr>
<td>u04</td>
<td>Bread, Milk, Diaper, Beer</td>
</tr>
<tr>
<td>u05</td>
<td>Bread, Milk, Diaper, Coke</td>
</tr>
</tbody>
</table>

- Each user may purchase several items
- Each item could be purchased by a few users
- Several items may be purchased together by one user
## A User-Item Rating Matrix

<table>
<thead>
<tr>
<th></th>
<th>Bread</th>
<th>Milk</th>
<th>Diaper</th>
<th>XBox</th>
<th>iPad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jenny</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Emily</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Diane</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Jeremy</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Stefan</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

*item vector*

*user vector*
**SVD Factorization**

The image illustrates the SVD factorization with a user–category similarity matrix and an item–category similarity matrix. The user–category similarity matrix is shown as a square matrix with the following entries:

```
[9.49  0  
 0   3.16]
```

The item–category similarity matrix is shown as a square matrix with the following entries:

```
[0.58 0.58 0.58 0  0  
 0   0   0   0.71 0.71]
```

The user–category similarity matrix is multiplied by the strength of digital fans, resulting in the item–category similarity matrix.
Gene H. Golub (February 29, 1932 – November 16, 2007)
American Mathematician and Computer Scientist
Factor Face Images to Find the Basis

Nonnegative matrix Factorization

\[ W^*H = V \]

Iterative Solution Methods

Obtain an approximate solution to a linear system iteratively

\[
\begin{align*}
    a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1 \\
    a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n &= b_2 \\
    \quad \quad \cdots \quad \quad \cdots \quad \cdots \\
    a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \cdots + a_{nn}x_{nn} &= b_n
\end{align*}
\]

Create a single equation for each variable

\[
\begin{align*}
    x_1 &= \frac{b_1 - (a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n)}{a_{11}} \\
    x_2 &= \frac{b_2 - (a_{21}x_1 + a_{23}x_3 + \cdots + a_{2n}x_n)}{a_{22}} \\
    \quad \quad \cdots \quad \quad \cdots \\
    x_n &= \frac{b_n - (a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \cdots)}{a_{nn}}
\end{align*}
\]

We assign an arbitrary value to each of the variables in the right-hand side to start iteration
Jacobi Iteration

We repeatedly use each set of computed values as the next iterates, and hope that the process will converge to the true solution. (May stop early if necessary.) This iteration procedure is called Jacobi iteration method. Convergence is not guaranteed for general matrices

\[
x_1^{(k)} = \frac{b_1 - (a_{12}x_2^{(k-1)} + a_{13}x_3^{(k-1)} + \cdots + a_{1n}x_n^{(k-1)})}{a_{11}}
\]

\[
x_2^{(k)} = \frac{b_2 - (a_{21}x_1^{(k-1)} + a_{23}x_3^{(k-1)} + \cdots + a_{2n}x_n^{(k-1)})}{a_{22}}
\]

\[\vdots\quad \vdots\]

\[
x_n^{(k)} = \frac{b_n - (a_{n1}x_1^{(k-1)} + a_{n2}x_2^{(k-1)} + a_{n3}x_3^{(k-1)} + \cdots)}{a_{nn}}
\]
Carl Gustav Jacobi (December 10, 1804 – February 18, 1851)
Prussian Mathematician
Gauss-Seidel Iteration

If the iteration process converges, it may be faster if we use the new values as soon as they are available. This gives the Gauss-Seidel iteration method

\[
\begin{align*}
  x_1^{(k)} &= \frac{b_1 - (a_{12}x_2^{(k-1)} + a_{13}x_3^{(k-1)} + \cdots + a_{1n}x_n^{(k-1)})}{a_{11}} \\
  x_2^{(k)} &= \frac{b_2 - (a_{21}x_1^{(k)} + a_{23}x_3^{(k-1)} + \cdots + a_{2n}x_n^{(k-1)})}{a_{22}} \\
  &\vdots \\
  x_n^{(k)} &= \frac{b_n - (a_{n1}x_1^{(k)} + a_{n2}x_2^{(k)} + a_{n3}x_3^{(k)} + \cdots)}{a_{nn}}
\end{align*}
\]

In general, Gauss-Seidel is faster than Jacobi, but the later is more attractive to run on parallel computers. No convergence is guaranteed.
Philipp Ludwig von Seidel (October 23, 1821 – August 13, 1896)
German Mathematician

Seidel was a student of Jacobi
Around 1950, David Young at Harvard University found that iteration can be accelerated by using a weighted average of the successive approximate solutions. Given some parameter $0 < \omega < 2$, we perform

$$x_i^{(k)} = \omega \tilde{x}_i^{(k)} + (1 - \omega)x_i^{(k-1)} \quad i = 1, 2, \ldots, n$$

This step symbolized the beginning of modern iterative methods.

With an arbitrary initial guess, convergence of these basic iterative methods is guaranteed if

1.) The matrix is strictly diagonally dominant;

2.) The matrix is symmetric positive definite, i.e., $A = A^T$ and $x^TAx > 0$
David M. Young, Jr. (1923-2008)
American Mathematician and Scientist
Splitting Coefficient Matrix

The system of linear equations is

\[ Ax = b \]

We can split the coefficient matrix \( A \) as

\[ A = M - N \]

Such that \( M \) is nonsingular and \( M^{-1} \) is easy to compute. So we have

\[(M - N)x = b\]

and

\[ Mx = Nx + b\]

So

\[ x = M^{-1}(Nx + b) = M^{-1}Nx + M^{-1}b \]

We can use this relationship for an iteration scheme

\[ x^{(k)} = M^{-1}Nx^{(k-1)} + M^{-1}b \]
Iteration Matrix

The matrix $M^{-1}N$ is called the iteration matrix

For the iteration process to be inexpensive to carry out, $M$ must be easy to invert

$$x^{(k)} = M^{-1}Nx^{(k-1)} + M^{-1}b$$

Since

$$N = M - A$$

We have

$$x^{(k)} = M^{-1}(M - A)x^{(k-1)} + M^{-1}b = (I - M^{-1}A)x^{(k-1)} + M^{-1}b$$

If $x^{(k)}$ converges to the solution $x$, then

$$x = (I - M^{-1}A)x + M^{-1}b$$

It follows that

$$Ax = b$$
Convergence

We define the error vector at each iteration step as

\[ e^{(k)} = x^{(k)} - x \]

It follows that

\[ e^{(k)} = x^{(k)} - x = (I - M^{-1}A)x^{(k-1)} - x + M^{-1}b \]

\[ = (I - M^{-1}A)x^{(k-1)} - (I - M^{-1}A)x \]

\[ = (I - M^{-1}A)(x^{(k-1)} - x) \]

\[ = (I - M^{-1}A)e^{(k-1)} \]

If we want the error to become smaller and smaller, we want the iteration matrix

\[ I - M^{-1}A \]

to be small, in some sense
Iteration Matrix

So we want the matrix

\[ M^{-1}A \]

to be close to the identity matrix. It is to say that we want \( M \) is close to \( A \)

If fact, if all eigenvalues of the iteration matrix

\[ I - M^{-1}A \]

is smaller than one, then the iteration

\[ e^{(k)} = (I - M^{-1}A)e^{(k-1)} \]

converges to zero and the original iteration converges. This requires

\[ \rho(I - M^{-1}A) < 1 \]

I.e., that the spectral radius of the iteration matrix is smaller than one guarantees the convergence of the induced iteration method.
Stationary Iterative Methods

For the Jacobi iteration

\[
A = D - (L + U)
\]

For the Gauss-Seidel Method

\[
A = (L + D) - U
\]

For the SOR method, the iteration is

\[
(D - \omega L)x^{(k)} = [\omega U + (1 - \omega)D]x^{(k-1)} + \omega b
\]

These iterative methods are called stationary methods, or classic iterative methods. They converge slowly.

Modern iterative methods, such as multigrid method and Krylov subspace methods converge much more rapidly. Those methods will be studied in CS623.