Gauss elimination

$$\boldsymbol{A}\vec{x}=\vec{b}$$

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1N} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2N} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3N} \\ \dots & \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & a_{N3} & \dots & a_{NN} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_N \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \dots \\ b_N \end{pmatrix}$$

Can permute rows of **A** and *b* simultaneously; columns of **A** and corresponding rows of x; form linear combinations of rows of **A** and *b*.

(second row) – (first row) x (a_{21}/a_{11}):

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1N} \\ a_{21}-\frac{a_{21}}{a_{11}}a_{11} & a_{22}-\frac{a_{21}}{a_{11}}a_{12} & a_{23}-\frac{a_{21}}{a_{11}}a_{13} & \dots & a_{2N}-\frac{a_{21}}{a_{11}}a_{1N} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3N} \\ \dots & \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & a_{N3} & \dots & a_{NN} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_N \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2-\frac{a_{21}}{a_{11}}b_1 \\ b_3 \\ \dots \\ b_N \end{pmatrix}$$

(*i*th row) – (first row) x (a_{i1}/a_{11}):

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1N} \\ 0 & a_{22}-\frac{a_{21}}{a_{11}}a_{12} & a_{23}-\frac{a_{21}}{a_{11}}a_{13} & \dots & a_{2N}-\frac{a_{21}}{a_{11}}a_{1N} \\ 0 & a_{32}-\frac{a_{31}}{a_{11}}a_{12} & a_{33}-\frac{a_{31}}{a_{11}}a_{13} & \dots & a_{3N}-\frac{a_{31}}{a_{11}}a_{1N} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & a_{N2}-\frac{a_{N1}}{a_{11}}a_{12} & a_{N3}-\frac{a_{N1}}{a_{11}}a_{13} & \dots & a_{NN}-\frac{a_{N1}}{a_{11}}a_{1N} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_N \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2-\frac{a_{21}}{a_{11}}b_1 \\ b_3-\frac{a_{31}}{a_{11}}b_1 \\ \dots \\ b_N-\frac{a_{N1}}{a_{11}}b_1 \end{pmatrix}$$

a_{11} is called a pivot in this context.

Problem if $a_{11}=0$. Even if just small, numerical instability (severe round-off errors)

Solution: find the largest element by absolute value in the first column; exchange rows so this element ends up as a_{11}. This is called partial pivoting (partial since only rows are exchanged). If there is no nonzero element, the matrix is singular (either no unique solution or an infinite number of them).

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1N} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2N} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3N} \\ \dots & \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & a_{N3} & \dots & a_{NN} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_N \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \dots \\ b_N \end{pmatrix}$$

Suppose a_{31} is the largest element by absolute value in the first column.

$$\begin{pmatrix} a_{31} & a_{32} & a_{33} & \dots & a_{3N} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2N} \\ a_{11} & a_{12} & a_{13} & \dots & a_{1N} \\ \dots & \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & a_{N3} & \dots & a_{NN} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_N \end{pmatrix} = \begin{pmatrix} b_3 \\ b_2 \\ b_1 \\ \dots \\ b_N \end{pmatrix}$$

Then do a linear combination of rows as before.

$$\begin{pmatrix} a_{11}^{(2)} & a_{12}^{(2)} & a_{13}^{(2)} & \dots & a_{1N}^{(2)} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & \dots & a_{2N}^{(2)} \\ 0 & a_{32}^{(2)} & a_{33}^{(2)} & \dots & a_{3N}^{(2)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & a_{N2}^{(2)} & a_{N3}^{(2)} & \dots & a_{NN}^{(2)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_N \end{pmatrix} = \begin{pmatrix} b_1^{(2)} \\ b_2^{(2)} \\ b_3^{(2)} \\ \dots \\ b_N^{(2)} \end{pmatrix}$$

Assume that pivoting is already done for $a^{(2)}_{22}$, so this element is the largest in the column.

(ith row) – (first row) x (a_{i1}/a_{11}) for $i>2$:

$$\begin{pmatrix} a_{11}^{(2)} & a_{12}^{(2)} & a_{13}^{(2)} & \dots & a_{1N}^{(2)} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & \dots & a_{2N}^{(2)} \\ 0 & 0 & a_{33}^{(2)} - \frac{a_{32}^{(2)}}{a_{22}^{(2)}} a_{23}^{(2)} & \dots & a_{3N}^{(2)} - \frac{a_{32}^{(2)}}{a_{22}^{(2)}} a_{2N}^{(2)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & a_{N3}^{(2)} - \frac{a_{N2}^{(2)}}{a_{22}^{(2)}} a_{23}^{(2)} & \dots & a_{NN}^{(2)} - \frac{a_{N2}^{(2)}}{a_{22}^{(2)}} a_{2N}^{(2)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_N \end{pmatrix} = \begin{pmatrix} b_1^{(2)} \\ b_2^{(2)} \\ b_3^{(2)} - \frac{a_{32}^{(2)}}{a_{22}^{(2)}} b_2^{(2)} \\ \dots \\ b_N^{(2)} - \frac{a_{N2}^{(2)}}{a_{22}^{(2)}} b_2^{(2)} \end{pmatrix}$$

After N-1 such transformations, get an upper triangular matrix:

$$\begin{pmatrix} a_{11}^{(N)} & a_{12}^{(N)} & a_{13}^{(N)} & \dots & a_{1,N-2}^{(N)} & a_{1,N-1}^{(N)} & a_{1N}^{(N)} \\ 0 & a_{22}^{(N)} & a_{23}^{(N)} & \dots & a_{2,N-2}^{(N)} & a_{2,N-1}^{(N)} & a_{2N}^{(N)} \\ 0 & 0 & a_{33}^{(N)} & \dots & a_{3,N-2}^{(N)} & a_{3,N-1}^{(N)} & a_{3N}^{(N)} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & a_{N-2,N-2}^{(N)} & a_{N-2,N-1}^{(N)} & a_{N-2,N}^{(N)} \\ 0 & 0 & 0 & \dots & 0 & a_{N-1,N-1}^{(N)} & a_{N-1,N}^{(N)} \\ 0 & 0 & 0 & \dots & 0 & 0 & a_{NN}^{(N)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_{N-2} \\ x_{N-1} \\ x_N \end{pmatrix} = \begin{pmatrix} b_1^{(N)} \\ b_2^{(N)} \\ b_3^{(N)} \\ \dots \\ b_{N-2}^{(N)} \\ b_{N-1}^{(N)} \\ b_N^{(N)} \end{pmatrix}$$

Now, solve easily by backsubstitution:

$$a_{NN}^{(N)} x_N = b_N^{(N)} \quad \Rightarrow \quad x_N = \frac{b_N^{(N)}}{a_{NN}^{(N)}}$$

$$a_{N-1,N-1}^{(N)} x_{N-1} + a_{N-1,N}^{(N)} x_N = a_{N-1,N-1}^{(N)} x_{N-1} + a_{N-1,N}^{(N)} \frac{b_N^{(2)}}{a_{NN}^{(N)}} = b_{N-1}^{(N)} \quad \Rightarrow \quad x_{N-1}$$

Can solve for several right-hand sides simultaneously.

$$\boldsymbol{A}\vec{x}=\vec{b} \qquad \boldsymbol{A}\vec{x}=\vec{c}$$

$$\left(\begin{array}{cccccc|cc}
a_{11} & a_{12} & a_{13} & \dots & a_{1,N-1} & a_{1N} & b_1 & c_1 \\
a_{21} & a_{22} & a_{23} & \dots & a_{2,N-1} & a_{2N} & b_2 & c_2 \\
a_{31} & a_{32} & a_{33} & \dots & a_{3,N-1} & a_{3N} & b_3 & c_3 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
a_{N-1,1} & a_{N-1,2} & a_{N-1,3} & \dots & a_{N-1,N-1} & a_{N-1,N} & b_{N-1} & c_{N-1} \\
a_{N1} & a_{N2} & a_{N3} & \dots & a_{N,N-1} & a_{NN} & b_N & c_N
\end{array}\right)$$

After transformation:

$$\left(\begin{array}{cccccc|cc}
a_{11}^{(N)} & a_{12}^{(N)} & a_{13}^{(N)} & \dots & a_{1,N-1}^{(N)} & a_{1N}^{(N)} & b_1^{(N)} & c_1^{(N)} \\
0 & a_{22}^{(N)} & a_{23}^{(N)} & \dots & a_{2,N-1}^{(N)} & a_{2N}^{(N)} & b_2^{(N)} & c_2^{(N)} \\
0 & 0 & a_{33}^{(N)} & \dots & a_{3,N-1}^{(N)} & a_{3N}^{(N)} & b_3^{(N)} & c_3^{(N)} \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & 0 & 0 & \dots & a_{N-1,N-1}^{(N)} & a_{N-1,N}^{(N)} & b_{N-1}^{(N)} & c_{N-1}^{(N)} \\
0 & 0 & 0 & \dots & 0 & a_{NN}^{(N)} & b_N^{(N)} & c_N^{(N)}
\end{array}\right)$$

A variant: Gauss-Jordan elimination. Make elements above diagonal 0 as well.

$$\begin{pmatrix} a_{11}^{(2)} & a_{12}^{(2)} & a_{13}^{(2)} & \dots & a_{1N}^{(2)} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & \dots & a_{2N}^{(2)} \\ 0 & a_{32}^{(2)} & a_{33}^{(2)} & \dots & a_{3N}^{(2)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & a_{N2}^{(2)} & a_{N3}^{(2)} & \dots & a_{NN}^{(2)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_N \end{pmatrix} = \begin{pmatrix} b_1^{(2)} \\ b_2^{(2)} \\ b_3^{(2)} \\ \dots \\ b_N^{(2)} \end{pmatrix}$$

(ith row) – (first row) x (a_{i1}/a_{11}) for $i \neq 2$:

$$\begin{pmatrix} a_{11}^{(2)} & 0 & a_{13}^{(2)} - \frac{a_{12}^{(2)}}{a_{22}^{(2)}} a_{23}^{(2)} & \dots & a_{1N}^{(2)} - \frac{a_{12}^{(2)}}{a_{22}^{(2)}} a_{2N}^{(2)} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & \dots & a_{2N}^{(2)} \\ 0 & 0 & a_{33}^{(2)} - \frac{a_{32}^{(2)}}{a_{22}^{(2)}} a_{23}^{(2)} & \dots & a_{3N}^{(2)} - \frac{a_{32}^{(2)}}{a_{22}^{(2)}} a_{2N}^{(2)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & a_{N3}^{(2)} - \frac{a_{N2}^{(2)}}{a_{22}^{(2)}} a_{23}^{(2)} & \dots & a_{NN}^{(2)} - \frac{a_{N2}^{(2)}}{a_{22}^{(2)}} a_{2N}^{(2)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_N \end{pmatrix} = \begin{pmatrix} b_1^{(2)} - \frac{a_{12}^{(2)}}{a_{22}^{(2)}} b_2^{(2)} \\ b_2^{(2)} \\ b_3^{(2)} - \frac{a_{32}^{(2)}}{a_{22}^{(2)}} b_2^{(2)} \\ \dots \\ b_N^{(2)} - \frac{a_{N2}^{(2)}}{a_{22}^{(2)}} b_2^{(2)} \end{pmatrix}$$

End up with a diagonal matrix:

$$\begin{pmatrix} a_{11}^{(N)} & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & a_{22}^{(N)} & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & a_{33}^{(N)} & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & a_{N-2,N-2}^{(N)} & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & a_{N-1,N-1}^{(N)} & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & a_{NN}^{(N)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_{N-2} \\ x_{N-1} \\ x_N \end{pmatrix} = \begin{pmatrix} b_1^{(N)} \\ b_2^{(N)} \\ b_3^{(N)} \\ \dots \\ b_{N-2}^{(N)} \\ b_{N-1}^{(N)} \\ b_N^{(N)} \end{pmatrix}$$

No need for backsubstitution. However, Gaussian elimination requires ~$2N^3/3$ operations and Gauss-Jordan requires ~$1\times N^3$ operations, whereas backsubstitution is only $O(N^2)$. So only starts to make sense if one needs to solve for O(N) right-hand sides simultaneously. One application of that is when one needs to calculate a matrix inverse, rather than just solve a linear system.

$\boldsymbol{A}\vec{x}=\vec{b}$ $\boldsymbol{A}\boldsymbol{X}=\boldsymbol{B}$ If ***B*** = ***I***, then ***X*** = $\boldsymbol{A}^{-1}$.

$$\left(\begin{array}{ccccc|ccccc}
a_{11} & a_{12} & a_{13} & \dots & a_{1N} & 1 & 0 & 0 & \dots & 0 \\
a_{21} & a_{22} & a_{23} & \dots & a_{2N} & 0 & 1 & 0 & \dots & 0 \\
a_{31} & a_{32} & a_{33} & \dots & a_{3N} & 0 & 0 & 1 & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
a_{N1} & a_{N2} & a_{N3} & \dots & a_{NN} & 0 & 0 & 0 & \dots & 1
\end{array}\right)$$

After Gauss-Jordan elimination and dividing each row by diagonal element,

$$\begin{pmatrix}
a_{11} & a_{12} & a_{13} & \dots & a_{1N} & 1 & 0 & 0 & \dots & 0 \\
a_{21} & a_{22} & a_{23} & \dots & a_{2N} & 0 & 1 & 0 & \dots & 0 \\
a_{31} & a_{32} & a_{33} & \dots & a_{3N} & 0 & 0 & 1 & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
a_{N1} & a_{N2} & a_{N3} & \dots & a_{NN} & 0 & 0 & 0 & \dots & 1
\end{pmatrix}$$

However, the inverse matrix itself is rarely needed in practice. E.g., Newton-Raphson: $\vec{x}^{(k+1)}=\vec{x}^{(k)}-\boldsymbol{J}^{-1}\vec{f}\left(\vec{x}^{(k)}\right)$ $\vec{y}=\boldsymbol{J}^{-1}\vec{f}\left(\vec{x}^{(k)}\right)$ can be viewed as the solution of $\boldsymbol{J}\vec{y}=\vec{f}\left(\vec{x}^{(k)}\right)$.

Gauss elimination and pivoting transformations can be viewed as multiplying by a matrix.

$$
\begin{pmatrix}
a_{11} & a_{12} & \dots & a_{1N} \\
\dots & \dots & \dots & \dots \\
a_{i1} & a_{i2} & \dots & a_{iN} \\
a_{i+1,1}-\alpha_{i+1}a_{i1} & a_{i+1,2}-\alpha_{i+1}a_{i2} & \dots & a_{i+1,N}-\alpha_{i+1}a_{iN} \\
a_{i+2,1}-\alpha_{i+2}a_{i1} & a_{i+2,2}-\alpha_{i+2}a_{i2} & \dots & a_{i+2,N}-\alpha_{i+2}a_{iN} \\
\dots & \dots & \dots & \dots \\
a_{N,1}-\alpha_{N}a_{i1} & a_{N,2}-\alpha_{N}a_{i2} & \dots & a_{N,N}-\alpha_{N}a_{iN}
\end{pmatrix} =
$$

$$
\begin{array}{cc}
 & \begin{array}{cccccc} & & & i & & \end{array} \\
\begin{array}{c} \\ \\ \\ i \\ \\ \\ \\ \\ \end{array} &
\begin{pmatrix}
1 & 0 & \dots & 0 & \dots & 0 \\
0 & 1 & \dots & 0 & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots \\
0 & 0 & \dots & 1 & \dots & 0 \\
0 & 0 & \dots & -\alpha_{i+1} & \dots & 0 \\
0 & 0 & \dots & -\alpha_{i+2} & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots \\
0 & 0 & \dots & -\alpha_{N} & \dots & 1
\end{pmatrix}
\end{array}
\begin{pmatrix}
a_{11} & a_{12} & \dots & a_{1N} \\
\dots & \dots & \dots & \dots \\
a_{i1} & a_{i2} & \dots & a_{iN} \\
a_{i+1,1} & a_{i+1,2} & \dots & a_{i+1,N} \\
a_{i+2,1} & a_{i+2,2} & \dots & a_{i+2,N} \\
\dots & \dots & \dots & \dots \\
a_{N,1} & a_{N,2} & \dots & a_{N,N}
\end{pmatrix}
$$

A lower triangular matrix

$$
\begin{pmatrix}
1 & 0 & \dots & 0 & \dots & 0 \\
0 & 1 & \dots & 0 & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots \\
0 & 0 & \dots & 1 & \dots & 0 \\
0 & 0 & \dots & -\alpha_{i+1} & \dots & 0 \\
0 & 0 & \dots & -\alpha_{i+2} & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots \\
0 & 0 & \dots & -\alpha_{N} & \dots & 1
\end{pmatrix}^{-1}
=
\begin{pmatrix}
1 & 0 & \dots & 0 & \dots & 0 \\
0 & 1 & \dots & 0 & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots \\
0 & 0 & \dots & 1 & \dots & 0 \\
0 & 0 & \dots & \alpha_{i+1} & \dots & 0 \\
0 & 0 & \dots & \alpha_{i+2} & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots \\
0 & 0 & \dots & \alpha_{N} & \dots & 1
\end{pmatrix}
$$

For pivoting:

$$
\begin{pmatrix}
a_{11} & a_{12} & \dots & a_{1N} \\
\dots & \dots & \dots & \dots \\
a_{j1} & a_{j2} & \dots & a_{jN} \\
\dots & \dots & \dots & \dots \\
a_{i1} & a_{i2} & \dots & a_{iN} \\
\dots & \dots & \dots & \dots \\
a_{N,1} & a_{N,2} & \dots & a_{N,N}
\end{pmatrix}
=
\begin{pmatrix}
1 & \dots & 0 & \dots & 0 & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & \dots & 0 & \dots & 1 & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & \dots & 1 & \dots & 0 & \dots & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & \dots & 0 & \dots & 0 & \dots & 1
\end{pmatrix}
\begin{pmatrix}
a_{11} & a_{12} & \dots & a_{1N} \\
\dots & \dots & \dots & \dots \\
a_{i1} & a_{i2} & \dots & a_{iN} \\
\dots & \dots & \dots & \dots \\
a_{j1} & a_{j2} & \dots & a_{jN} \\
\dots & \dots & \dots & \dots \\
a_{N,1} & a_{N,2} & \dots & a_{N,N}
\end{pmatrix}
$$

If we consider eliminations without pivoting, $\boldsymbol{L}_{N-1}\boldsymbol{L}_{N-2}\ldots\boldsymbol{L}_2\boldsymbol{L}_1\boldsymbol{A}=U$

$$\boldsymbol{A}=\boldsymbol{L}_1^{-1}\boldsymbol{L}_2^{-1}\ldots\boldsymbol{L}_{N-2}^{-1}\boldsymbol{L}_{N-1}^{-1}\boldsymbol{U}=\boldsymbol{L}\boldsymbol{U}$$

With pivoting, $\boldsymbol{L}_{N-1}\boldsymbol{P}_{N-1}\boldsymbol{L}_{N-2}\boldsymbol{P}_{N-2}\ldots\boldsymbol{L}_2\boldsymbol{P}_2\boldsymbol{L}_1\boldsymbol{P}_1\boldsymbol{A}=U$

It can be shown that this is equivalent to

$$\tilde{\boldsymbol{L}}_{N-1}\tilde{\boldsymbol{L}}_{N-2}\ldots\tilde{\boldsymbol{L}}_2\tilde{\boldsymbol{L}}_1\boldsymbol{P}_{N-1}\boldsymbol{P}_{N-2}\ldots\boldsymbol{P}_2\boldsymbol{P}_1\boldsymbol{A}=U$$

$$\boldsymbol{P}_{N-1}\boldsymbol{P}_{N-2}\ldots\boldsymbol{P}_2\boldsymbol{P}_1\boldsymbol{A}=\tilde{\boldsymbol{L}}_1\tilde{\boldsymbol{L}}_2\ldots\tilde{\boldsymbol{L}}_{N-2}\tilde{\boldsymbol{L}}_{N-1}U \quad\Rightarrow\quad \boldsymbol{P}\boldsymbol{A}=\boldsymbol{L}\boldsymbol{U}$$

LU decomposition of a matrix. Such decompositions are not unique, but become unique if the diagonal elements of either ***L*** or ***U*** are fixed at 1:

$$\boldsymbol{P}\begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1N} \\ a_{21} & a_{22} & \ldots & a_{2N} \\ \ldots & \ldots & \ldots & \ldots \\ a_{N1} & a_{N2} & \ldots & a_{NN} \end{pmatrix}=\begin{pmatrix} 1 & 0 & \ldots & 0 \\ l_{21} & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ l_{N1} & l_{N2} & \ldots & 1 \end{pmatrix}\cdot\begin{pmatrix} u_{11} & u_{12} & \ldots & u_{1N} \\ 0 & u_{22} & \ldots & u_{2N} \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & u_{NN} \end{pmatrix}$$

If the LU decomposition is known explicitly, then solving a system with an arbitrary right-hand side takes only O(N²) operations:

$$\boldsymbol{L}\,\boldsymbol{U}\,\vec{x}=\boldsymbol{P}\,\boldsymbol{A}\,\vec{x}=\boldsymbol{P}\,\vec{b}$$

First solve $\boldsymbol{L}\,\vec{y}=\boldsymbol{P}\,\vec{b}$ Forward substitution

$$\begin{pmatrix} 1 & 0 & \dots & 0 \\ l_{21} & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ l_{N1} & l_{N2} & \dots & 1 \end{pmatrix}\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_N \end{pmatrix}=\begin{pmatrix} b_{i_1} \\ b_{i_2} \\ \dots \\ b_{i_N} \end{pmatrix}$$

$$y_1=b_{i_1}$$

$$l_{21}\,y_1+y_2=l_{21}\,b_{i_1}+y_2=b_{i_2} \quad \Rightarrow \quad y_2$$

$$\dots\dots\dots\dots\dots$$

Then solve $\boldsymbol{U}\,\vec{x}=\vec{y}$ Backsubstitution

A big advantage if we don't know all the right-hand sides in advance.

Crout's method (sometimes called Doolittle's method, and then Crout is when ***U*** has 1's on the diagonal). Ignore pivoting for now.

$$\begin{pmatrix} 1 & 0 & \ldots & 0 \\ l_{21} & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ l_{N1} & l_{N2} & \ldots & 1 \end{pmatrix} \cdot \begin{pmatrix} u_{11} & u_{12} & \ldots & u_{1N} \\ 0 & u_{22} & \ldots & u_{2N} \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & u_{NN} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1N} \\ a_{21} & a_{22} & \ldots & a_{2N} \\ \ldots & \ldots & \ldots & \ldots \\ a_{N1} & a_{N2} & \ldots & a_{NN} \end{pmatrix}$$

$$u_{11} = a_{11} \qquad l_{i1} u_{11} = l_{i1} a_{11} = a_{i1} \;\Rightarrow\; l_{i1} = a_{i1}/a_{11} \qquad u_{12} = a_{12}$$

$$l_{21} u_{12} + u_{22} = a_{22} \;\Rightarrow\; u_{22} \qquad l_{i1} u_{12} + l_{i2} u_{22} = a_{i2} \;\Rightarrow\; l_{i2}$$

In general:

For j=1,...,N:

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj}, \quad i = 1, \ldots, j$$

$$l_{ij} = \frac{1}{u_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj} \right), \quad i = j+1, \ldots, N$$

About the same computational complexity as straightforward Gaussian elimination.

Special case: symmetric positive definite matrices

For any $\vec{x}$, $\vec{x}^T \boldsymbol{A} \vec{x} > 0$ Physical example: a system of interacting particles in equilibrium. Potential near equilibrium for small displacements $\vec{x}$:

$U = \frac{1}{2} \vec{x}^T \boldsymbol{A} \vec{x}$ If there is an external force $\vec{F}$, the displacement $\vec{x} = \boldsymbol{A}^{-1} \vec{F}$

Cholesky decomposition: $\boldsymbol{A} = \boldsymbol{L} \boldsymbol{L}^T$

$$\begin{pmatrix} l_{11} & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ l_{N1} & l_{N2} & \dots & l_{NN} \end{pmatrix} \cdot \begin{pmatrix} l_{11} & l_{21} & \dots & l_{N1} \\ 0 & l_{22} & \dots & l_{N2} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & l_{NN} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{pmatrix}$$

$l_{11}^2 = a_{11} \Rightarrow l_{11} = \sqrt{a_{11}}$ $\qquad l_{i1} l_{11} = l_{i1} \sqrt{a_{11}} = a_{i1} \Rightarrow l_{i1} = a_{i1} / \sqrt{a_{11}}$

$l_{11} l_{21} = a_{12} \Rightarrow l_{21}$ $\qquad l_{21}^2 + l_{22}^2 = a_{22} \Rightarrow l_{22}$ $\qquad l_{i1} l_{21} + l_{i2} l_{22} = a_{i2} \Rightarrow l_{i2}$

In general: for i=1,..., N

$$l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2} \qquad l_{ji} = \frac{1}{l_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} l_{ik} l_{jk} \right), \quad j = i+1, \dots, N$$

No need for pivoting. Computational cost ~1/2 of that for Gaussian elimination.

Tridiagonal systems

$$\begin{pmatrix} b_1 & c_1 & 0 & 0 & \dots & 0 & 0 & 0 \\ a_2 & b_2 & c_2 & 0 & \dots & 0 & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & a_{N-1} & b_{N-1} & c_{N-1} \\ 0 & 0 & 0 & 0 & \dots & 0 & a_N & b_N \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_{N-1} \\ x_N \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ \dots \\ r_{N-1} \\ r_N \end{pmatrix}$$

Apply Gaussian elimination (no pivoting):

$$\begin{pmatrix} b_1 & c_1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & b_2 - \frac{a_2}{b_1} c_1 & c_2 & 0 & \dots & 0 & 0 & 0 \\ 0 & a_3 & b_3 & c_3 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & a_{N-1} & b_{N-1} & c_{N-1} \\ 0 & 0 & 0 & 0 & \dots & 0 & a_N & b_N \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_{N-1} \\ x_N \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 - \frac{a_2}{b_1} r_1 \\ r_3 \\ \dots \\ r_{N-1} \\ r_N \end{pmatrix}$$

At every step, only two subtractions.

$$
\begin{pmatrix}
b'_1 & c_1 & 0 & 0 & \dots & 0 & 0 & 0 \\
0 & b'_2 & c_2 & 0 & \dots & 0 & 0 & 0 \\
0 & 0 & b'_3 & c_3 & \dots & 0 & 0 & 0 \\
\dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
0 & 0 & 0 & 0 & \dots & 0 & b'_{N-1} & c_{N-1} \\
0 & 0 & 0 & 0 & \dots & 0 & 0 & b'_N
\end{pmatrix}
\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_{N-1} \\ x_N \end{pmatrix}
=
\begin{pmatrix} r'_1 \\ r'_2 \\ r'_3 \\ \dots \\ r'_{N-1} \\ r_N \end{pmatrix}
$$

$$b'_i = b_i - \frac{c_{i-1} a_i}{b'_{i-1}} \qquad r'_i = r_i - \frac{r'_{i-1} a_i}{b'_{i-1}}$$

Then backsubstitution: $x_n = \dfrac{r'_n}{b'_n}$ $\qquad x_{i-1} = \dfrac{r'_{i-1} - c_{i-1} x_i}{b'_{i-1}}$

Another formulation of this in terms of the LU decomposition.

No pivoting is justified when the matrix is diagonally dominant $\left(|b_i| > |a_i| + |c_i|\right)$

Block-tridiagonal matrices: each “element” is a block. Use the same equations formally; will need to do small matrix inverses (or solve small linear systems).

Simplifications for other sparse patterns are possible.

If there are only a small number of pattern violations:

$$\left(\boldsymbol{A}+\vec{u}\otimes\vec{v}\right)^{-1}=\left(\boldsymbol{I}+\boldsymbol{A}^{-1}\cdot\vec{u}\otimes\vec{v}\right)^{-1}\cdot\boldsymbol{A}^{-1}$$

$$=\left(\boldsymbol{I}-\boldsymbol{A}^{-1}\cdot\vec{u}\otimes\vec{v}+\boldsymbol{A}^{-1}\cdot\vec{u}\otimes\vec{v}\cdot\boldsymbol{A}^{-1}\cdot\vec{u}\otimes\vec{v}-\ldots\right)\cdot\boldsymbol{A}^{-1}$$

$$=\boldsymbol{A}^{-1}-\boldsymbol{A}^{-1}\cdot\vec{u}\otimes\vec{v}\cdot\boldsymbol{A}^{-1}\left(1-\lambda+\lambda^{2}-\ldots\right)=\boldsymbol{A}^{-1}-\frac{\left(\boldsymbol{A}^{-1}\cdot\vec{u}\right)\otimes\left(\vec{v}\cdot\boldsymbol{A}^{-1}\right)}{1+\lambda},$$

where $\left(\vec{u}\otimes\vec{v}\right)_{ij}=u_i v_j,\qquad \lambda=\vec{v}\cdot\boldsymbol{A}^{-1}\cdot\vec{u}$

Special matrices for which there are O(N^2) methods:

Vandermonde

$$\begin{pmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^{N-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{N-1} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & x_N & x_N^2 & \ldots & x_N^{N-1} \end{pmatrix}$$

Toeplitz

$$\begin{pmatrix} R_0 & R_{-1} & R_{-2} & \ldots & R_{-(N-2)} & R_{-(N-1)} \\ R_1 & R_0 & R_{-1} & \ldots & R_{-(N-3)} & R_{-(N-2)} \\ R_2 & R_1 & R_0 & \ldots & R_{-(N-4)} & R_{-(N-3)} \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ R_{N-2} & R_{N-3} & R_{N-4} & \ldots & R_0 & R_{-1} \\ R_{N-1} & R_{N-2} & R_{N-3} & \ldots & R_1 & R_0 \end{pmatrix}$$

Accuracy of the solution

$$A\vec{x}=\vec{b}$$

Suppose A and $\vec{b}$ have errors δA and $\delta\vec{b}$ associated with them (e.g., round-off errors). How sensitive is $\vec{x}$ to these errors?

For simplicity consider the case $\delta A=0$ and assume that the matrix is symmetric and positive definite (so its eigenvalues are real and positive).

$$A(\vec{x}+\delta\vec{x})=(\vec{b}+\delta\vec{b}) \qquad A\delta\vec{x}=\delta\vec{b} \;\Rightarrow\; \delta\vec{x}=A^{-1}\delta\vec{b}$$

$$\|\vec{x}\|=\|A^{-1}\vec{b}\| \qquad \|\delta\vec{x}\|=\|A^{-1}\delta\vec{b}\|$$

Eigenvectors $\vec{v}_i$: $\quad A\vec{v}_i=\lambda_i\vec{v}_i; \qquad A^{-1}\vec{v}_i=\lambda_i^{-1}\vec{v}_i$

$$\frac{\|\delta\vec{x}\|}{\|\vec{x}\|}\le\frac{\max\,\|\delta\vec{x}\|}{\min\,\|\vec{x}\|}=\frac{\max\,\|A^{-1}\delta\vec{b}\|}{\min\,\|A^{-1}\vec{b}\|}=\frac{\lambda_{min}^{-1}\|\delta\vec{b}\|}{\lambda_{max}^{-1}\|\vec{b}\|}=\frac{\lambda_{max}}{\lambda_{min}}\frac{\|\delta\vec{b}\|}{\|\vec{b}\|}=K(A)\frac{\|\delta\vec{b}\|}{\|\vec{b}\|}$$

K(***A***) is (spectral) condition number of matrix ***A***. If it is large, the matrix is called ill-conditioned.

Accuracy of the solution

More generally, when $\delta \boldsymbol{A} \neq 0$,

$$\frac{\|\delta \vec{x}\|}{\|\vec{x}\|} \leq \frac{K(\boldsymbol{A})}{1-\lambda_{max}(\delta \boldsymbol{A})/\lambda_{min}(\boldsymbol{A})}\left(\frac{\lambda_{max}(\delta \boldsymbol{A})}{\lambda_{max}(\boldsymbol{A})}+\frac{\|\delta \vec{b}\|}{\|\vec{b}\|}\right)$$

Even more generally, for arbitrary matrices,

$$\frac{\|\delta \vec{x}\|}{\|\vec{x}\|} \leq \frac{K_2(\boldsymbol{A})}{1-K_2(\boldsymbol{A})\|\delta \boldsymbol{A}\|_2/\|\boldsymbol{A}\|_2}\left(\frac{\|\delta \boldsymbol{A}\|_2}{\|\boldsymbol{A}\|_2}+\frac{\|\delta \vec{b}\|}{\|\vec{b}\|}\right)$$

Wilkinson showed that $\dfrac{\|\delta \boldsymbol{A}\|_2}{\|\boldsymbol{A}\|_2} \leq \rho\, n\, \epsilon$

ρ is the growth factor. In theory, can be 2^{N-1} for partial pivoting, but in practice this rarely happens. Better for full pivoting.

Iterative methods

$$\boldsymbol{A}\vec{x}=\vec{b} \qquad \boldsymbol{A}=\boldsymbol{A}_1+\boldsymbol{A}_2 \qquad \boldsymbol{A}_1\vec{x}=\vec{b}-\boldsymbol{A}_2\vec{x}$$

$$\vec{x}=\boldsymbol{A}_1^{-1}\left(\vec{b}-\boldsymbol{A}_2\vec{x}\right)$$

Iterate: $\vec{x}^{(n+1)}=\boldsymbol{A}_1^{-1}\left(\vec{b}-\boldsymbol{A}_2\vec{x}^{(n)}\right)$

$$\vec{x}^{(n+1)}=\boldsymbol{A}_1^{-1}\left(\vec{b}-\left(\boldsymbol{A}-\boldsymbol{A}_1\right)\vec{x}^{(n)}\right)=\vec{x}^{(n)}-\boldsymbol{A}_1^{-1}\left(\boldsymbol{A}\vec{x}^{(n)}-\vec{b}\right)$$

Jacobi method

$\boldsymbol{A}=\boldsymbol{D}+\boldsymbol{L}+\boldsymbol{U}$ $\qquad$ $\boldsymbol{A}_1=\boldsymbol{D}$ $\qquad$ Diagonal+everything else

Of course, the diagonal part is easy to invert.

$$x_i^{(n+1)}=-\frac{1}{a_{ii}}\sum_{j\neq i}x_j^{(n)}+\frac{1}{a_{ii}}b_i$$

Converges slowly. Reduction of the error by factor 10^{-p} requires about pN/2 iterations. Each iteration has cost $O(N^2)$.

Jacobi – $x_i^{(n+1)} = -\frac{1}{a_{ii}} \sum_{j \neq i} x_j^{(n)} + \frac{1}{a_{ii}} b_i$

Gauss-Seidel method

$$x_i^{(n+1)} = \frac{1}{a_{ii}} \left(-\sum_{j<i} x_j^{(n+1)} - \sum_{j>i} x_j^{(n)} + \frac{1}{a_{ii}} b_i \right)$$

I.e., almost the same as Jacobi, but use the already updated values.

Converges faster by roughly a factor of 2.

Actually corresponds to $\boldsymbol{A}_1 = \boldsymbol{D} + \boldsymbol{L}$

Successive over-relaxation (SOR)

$$x_i^{(n+1)} = (1-\omega) x^{(n)} + \frac{\omega}{a_{ii}} \left(-\sum_{j<i} x_j^{(n+1)} - \sum_{j>i} x_j^{(n)} + \frac{1}{a_{ii}} b_i \right)$$

Over-relaxation for $1 < \omega < 2$

For optimal ω for error reduction by factor 10^{-p} about $\frac{1}{3} p \sqrt{N}$ iterations needed.