TOPIC 3: SYSTEMS OF LINEAR EQUATIONS

A system of linear equations has the form $Ax = b$, where $A$ is a square $n \times n$ matrix, $b$ is a given $n \times 1$ column vector, and $x$ is the unknown $n \times 1$ column vector to be found. There may or may not be a solution to this system, and if there is a solution, it may or may not be unique. To summarize the possibilities, for a given matrix $A$ and vector $b$, the system may have:

No solution: $A$ is singular.
One solution: $A$ is non-singular.
Infinitely many solutions: $A$ is singular.

(Remember that a square $n \times n$ matrix $A$ is singular if it has any one
of the following (equivalent) properties:
• A has no inverse
• $\det(A) = 0$
• rank $(A) < n$, the rank of a matrix being the maximum number of linearly independent rows or columns it contains
• $Ay = 0$ for some non-zero vector $y$.

For example, the 2 x 2 system

$2x_1 + 3x_2 = b_1$
$5x_1 + 4x_2 = b_2$

i.e.

$$\begin{bmatrix} 2 & 3 \\ 5 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$
is nonsingular regardless of the values of $b_1$, $b_2$. If $b = \begin{bmatrix} 8 \\ 13 \end{bmatrix}$, for example, then $x = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$.

The $2 \times 2$ system
\[
\begin{bmatrix} 2 & 3 \\ 4 & 6 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}
\]
is singular regardless of the values of $b_1$, $b_2$. If $b = \begin{bmatrix} 4 \\ 7 \end{bmatrix}$, there is no solution. If $b = \begin{bmatrix} 4 \\ 8 \end{bmatrix}$, then
\[
x = \begin{bmatrix} \alpha \\ \frac{4-2\alpha}{3} \end{bmatrix}
\] for any real $\alpha$. 
Let us look at some procedures for obtaining solutions to systems of linear equations. We will consider the system
\[ \begin{align*}
x_1 - 2x_2 + 3x_3 &= -1 \\
2x_1 - x_2 + 2x_3 &= 2 \\
3x_1 + x_2 + 2x_3 &= 3
\end{align*} \]
which in matrix form is
\[
\begin{bmatrix}
1 & -2 & 3 \\
2 & -1 & 2 \\
3 & 1 & 2
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}
= 
\begin{bmatrix}
-1 \\
2 \\
3
\end{bmatrix}
\]
and we consider the augmented matrix
\[
\begin{bmatrix}
1 & -2 & 3 & -1 \\
2 & -1 & 2 & 2 \\
3 & 1 & 2 & 3
\end{bmatrix}
\]
We carry out a sequence of row operations on this augmented matrix so that every non-diagonal number is set to zero and every diagonal number is set to 1 (Gauss-Jordan elimination method).

Column 1:
\[
\begin{bmatrix}
1 & -2 & 3 & -1 \\
0 & 3 & -4 & 4 \\
0 & 7 & -7 & 6 \\
\end{bmatrix}
\]

Unitize the entry at (2,2)
\[
\begin{bmatrix}
1 & -2 & 3 & -1 \\
0 & 1 & -4/3 & 4/3 \\
0 & 7 & -7 & 6 \\
\end{bmatrix}
\]

Column 2:
\[
\begin{bmatrix}
1 & 0 & 1/3 & 5/3 \\
0 & 1 & -4/3 & 4/3 \\
0 & 0 & 7/3 & -10/3 \\
\end{bmatrix}
\]
Unitize the entry at (3,3)

\[
\begin{bmatrix}
1 & 0 & 1/3 & 5/3 \\
0 & 1 & -4/3 & 4/3 \\
0 & 0 & 1 & -10/7
\end{bmatrix}
\]

\[R_3*3/7\]

Column 3 :

\[R_1-\frac{1}{3}R_3\]

\[R_2+\frac{4}{3}R_3\]

Hence the required solutions are

\[x_1 = \frac{15}{7}\]

\[x_2 = -\frac{4}{7}\]

\[x_3 = -\frac{10}{7}\]
The Gaussian elimination method operates in a similar manner, except that only those elements below the diagonal are reduced to zero. At the conclusion of the row operations, back substitution is then used to compute the values of $x_n, x_{n-1}, \ldots, x_1$ in turn.

Consider the same augmented matrix as before:

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

Column 1:

$$
\begin{bmatrix}
1 & -2 & 3 & -1 \\
0 & 3 & -4 & 4 \\
0 & 7 & -7 & 6 \\
\end{bmatrix}
$$
Unitize the entry at (2,2)

\[
\begin{bmatrix}
1 & -2 & 3 & -1 \\
0 & 1 & -4/3 & 4/3 \\
0 & 7 & -7 & 6 \\
\end{bmatrix}
\]

\(R_2 \times \frac{1}{3}\)

Column 2:

\[
\begin{bmatrix}
1 & -2 & 3 & -1 \\
0 & 1 & -4/3 & 4/3 \\
0 & 7/3 & -10/3 \\
\end{bmatrix}
\]

\(R_3 - 7R_2\)

Unitize the entry at (3,3)

\[
\begin{bmatrix}
1 & -2 & 3 & -1 \\
0 & 1 & -4/3 & 4/3 \\
0 & 0 & 1 & -10/7 \\
\end{bmatrix}
\]

from which \(x_1 - 2x_2 + 3x_3 = -1\)

\[
x_2 - \frac{4}{3}x_3 = \frac{4}{3}
\]

\[
x_3 = -\frac{10}{7}
\]
Putting \( x_3 = \frac{-10}{7} \) in the second equation gives
\[
x_2 - \frac{4}{3} \cdot \frac{10}{7} = \frac{4}{3}
\]
\[
x_2 = -\frac{4}{7}
\]
and then \( x_1 \) is given by
\[
x_1 - 2 \cdot \frac{4}{7} + 3 \cdot \frac{10}{7} = -1
\]
\[
x_1 = \frac{15}{7}
\]
(Note that we could have carried out the row operations to obtain 0's above the main diagonal instead of below, and we would then have finished the solution by a process known as forward substitution.)
The work required in the Gauss-Jordan method clearly exceeds that for Gaussian elimination, but by how much? The handout shows how to calculate the number of operations required in Gauss's method; you need to be able to carry out a similar computation for the Gauss-Jordan method.

In these two elimination procedures, one of the steps which is repeated for each row is to divide by the current value of the diagonal element of that row. If this number was zero, this operation would not be admissible. If the element was quite small, the division operation would lead to very large numbers that may invalidate the remainder of the calculations. When this problem arises, it can be resolved by interchanging the row
containing the offending diagonal element with any row below it for which the element in this particular column is neither zero nor very small. In practice, from column 1 onwards, we choose the entry of largest magnitude on or below the diagonal, and the row containing this large element is then interchanged with the row under consideration. Such a technique is called "partial pivoting", the name arising from the fact that only the current column (and, in fact, only that part of the current column on or below the diagonal) is searched for a suitable pivot; in complete pivoting, the entire remaining unreduced submatrix is searched for the entry of largest magnitude, which
is then permuted into the diagonal pivot position by interchanging columns as well as rows.
Let us repeat the previous example, applying partial pivoting:
The initial augmented matrix is:
\[
\begin{bmatrix}
1 & -2 & 3 & -1 \\
2 & -1 & 2 & 2 \\
3 & 1 & 2 & 3
\end{bmatrix}
\]
The element of largest magnitude in the first column is 3, in position (3,1); so interchange rows 1 and 3:
\[
\begin{bmatrix}
3 & 1 & 2 & 3 \\
2 & -1 & 2 & 2 \\
1 & -2 & 3 & -1
\end{bmatrix}
\]
Unitize the element at (1,1):

\[
R_1 \cdot \frac{1}{3} \begin{bmatrix} 1 & 1/3 & 2/3 & 1 \\ 2 & -1 & 2 & 2 \\ 1 & -2 & 3 & -1 \end{bmatrix}
\]

Column 1:

\[
R_2 - 2R_1 \begin{bmatrix} 1 & 1/3 & 2/3 & 1 \\ 0 & -5/3 & 2/3 & 0 \\ 0 & -7/3 & 7/3 & -2 \end{bmatrix}
\]

The element of largest magnitude on or below the diagonal in column 2 is \(-\frac{7}{3}\) in position (3,2); so interchange rows 2 and 3:

\[
\begin{bmatrix} 1 & 1/3 & 2/3 & 1 \\ 0 & -7/3 & 7/3 & -2 \\ 0 & -5/3 & 2/3 & 0 \end{bmatrix}
\]
Unitize the element at (2,2)

\[
\begin{bmatrix}
1 & 1/3 & 2/3 & 1 \\
0 & 1 & -1 & 6/7 \\
0 & -5/3 & 2/3 & 0
\end{bmatrix}
\]

Column 2 :

\[
\begin{bmatrix}
1 & 1/3 & 2/3 & 1 \\
0 & 1 & -1 & 6/7 \\
0 & 0 & -1 & 10/7
\end{bmatrix}
\]

\[\begin{array}{c}
R_3 + \frac{5}{3} R_2 \\
R_3 - 1
\end{array}\]

Unitize the element at (3,3) :

\[
\begin{bmatrix}
1 & 1/3 & 2/3 & 1 \\
0 & 1 & -1 & 6/7 \\
0 & 0 & 1 & -10/7
\end{bmatrix}
\]
Hence \( x_1 + \frac{1}{3}x_2 + \frac{2}{3}x_3 = 1 \)

\[
x_2 - x_3 = \frac{6}{7}
\]

\[
x_3 = -\frac{10}{7}
\]

which leads to \( x_3 = -\frac{10}{7} \),

\[
x_2 = -\frac{4}{7}
\]

\[
x_1 = \frac{15}{7}, \text{ as before.}
\]

The operations of Gaussian elimination in a form which will facilitate the writing of a computer program are as follows:
Gaussian Elimination

To solve a system of $n$ linear equations: $Ax = b$.

For $j = 1$ To $(n - 1)$
  pvt = $\{a[j, j]\}$
  pivot[$j$] = $j$
  ipvt_temp = $j$

  For $i = j + 1$ To $n$ (Find pivot row)
    If $|a[i, j]| > pvt$ Then
      pvt = $|a[i, j]|$
      ipvt_temp = $i$
    End If
  End For

(Switch rows if necessary)
If pivot[$j$] < ipvt_temp
  [switch rows (rows $j$ and ipvt_temp)]
End If

  For $i = j + 1$ to $n$ (Store multipliers)
    $a[i, j] = a[i, j]/a[j, j]$
  End For

(Create zeros below the main diagonal)
For $i = j + 1$ To $n$
  For $k = j + 1$ To $n$
    $a[i, k] = a[i, k] - a[i, j]*a[j, k]$
  End For
  $b[i] = b[i] - a[i, j]*b[j]$
End For

(Back Substitution Part)
$x[n] = b[n]/a[n, n]$
For $j = n - 1$ Down To 1
  $x[j] = b[j]$
  For $k = n$ Down To $j + 1$
    $x[j] = x[j] - x[k] * a[j, k]$
  End For
  $x[j] = x[j]/a[j, j]$
End For
Another situation which may affect the solution of linear equations is that they may involve relationships between quantities measured in different units which may result in equations having coefficients of vastly different sizes.

Consider, for example, the equations

\[
\begin{bmatrix}
3 & 2 & 100 \\
-1 & 3 & 100 \\
1 & 2 & -1
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} =
\begin{bmatrix}
105 \\
102 \\
2
\end{bmatrix}
\]

If we work with 3 digits and partial pivoting, then

\[x_3 = 1.00,\]
\[x_2 = 1.09,\]
\[x_1 = 0.94\]

which substitution shows to be not correct.
If we scale the coefficients before solution by dividing each row by the coefficient of largest magnitude, we obtain
\[
\begin{bmatrix}
.03 & .02 & 1.00 \\
-.01 & .03 & 1.00 \\
.50 & 1.00 & -.50
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}
= 
\begin{bmatrix}
1.05 \\
1.02 \\
1.00
\end{bmatrix}
\]
from which
\[
x_3 = 1.00,
\]
\[
x_2 = 1.00,
\]
\[
x_1 = 1.00
\]
which are correct.
The reason for the improvement is that, in the second case, rows are interchanged after scaling has been done, whereas no interchanges are indicated in the unscaled equations.
Having dealt with the Gaussian and the Gaussian-Jordan elimination procedures for solving systems of linear equations, we turn now to the LU-factorization method. An \( n \times n \) matrix of the form

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

having only 0's below the main diagonal is called an upper triangular matrix. Similarly,

\[
L = \begin{bmatrix}
  v_{11} & 0 & \ldots & 0 \\
  v_{21} & v_{22} & \ldots & 0 \\
  \vdots & \vdots & \ddots & \vdots \\
  v_{n1} & v_{n2} & \ldots & v_{nn}
\end{bmatrix}
\]
having only 0's above the main diagonal is called a lower triangular matrix.
The LU-factorization method factors the matrix $A$ as $A = LU$, where $U$ is upper triangular and $L$ is lower triangular with 1's along the main diagonal. Having found such a factorization, the system of equations $Ax = b$ is then solved by first solving $Lz = b$ by forward substitution and secondly solving $Ux = z$ by back substitution; this gives the desired result since $b = Lz = LUx = Ax$.

Consider again our matrix

$$A = \begin{bmatrix} 1 & -2 & 3 \\ 2 & -1 & 2 \\ 3 & 1 & 2 \end{bmatrix}$$
\[
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\end{bmatrix}
\begin{bmatrix}
1 & -2 & 3 \\
2 & -1 & 2 \\
3 & 1 & 2 \\
\end{bmatrix}
\]
Without unitizing or swapping rows, we begin the process of obtaining 0’s below the main diagonal.
\[
A = \begin{bmatrix}
1 & 0 & 0 \\
2 & 1 & 0 \\
0 & 0 & 1 \\
\end{bmatrix}
\begin{bmatrix}
1 & -2 & 3 \\
0 & 3 & -4 \\
3 & 1 & 2 \\
\end{bmatrix} R_2 - 2R_1
\]
As we are making the entry 0 in position (2,1) by \( R_2 - 2R_1 \) we record this in the first matrix by placing a 2 in position (2,1); note that the sign is opposite to that of the multiplier -2. The product of the resulting matrices is still A.
Next we attack the entry in position (3,1):
\[
A = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & -2 & 3 \\ 0 & 3 & -4 \\ 0 & 7 & -7 \end{bmatrix} R_3 - 3R_1
\]
Finally we attack the entry at position (3,2):
\[
A = LU =
\begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & \frac{7}{3} & 1 \end{bmatrix} \begin{bmatrix} 1 & -2 & 3 \\ 0 & 3 & -4 \\ 0 & \frac{7}{3} & \frac{7}{3} \end{bmatrix} R_3 - \frac{7}{3} R_2
\]
Next we solve \( Lz = b \) by forward substitution:
\[
\begin{align*}
z_1 &= -1 \\
2z_1 + z_2 &= 2 \\
3z_1 + \frac{7}{3}z_2 + z_3 &= 3
\end{align*}
\]
from which
\[ z_1 = -1, \]
\[ z_2 = 2 - 2z_1 = 2 - 2(-1) = 4 \]
\[ z_3 = 3 - 3z_1 - \frac{7}{3}z_2 = 3 - 3(-1) - \frac{7}{3} \cdot 4 = -\frac{10}{3} \]

Finally we solve $Ux = z$ by back substitution:
\[ x_1 - 2x_2 + 3x_3 = -1 \]
\[ 3x_2 - 4x_3 = 4 \]
\[ \frac{7}{3}x_3 = -\frac{10}{3} \]
\[ x_3 = -\frac{10}{7} \]

from which
\[ x_3 = \frac{3}{7} \cdot -\frac{10}{3} = -\frac{10}{7} \]
\[ x_2 = \frac{1}{3} (4 + 4x_3) = \frac{1}{3} (4 + 4 \cdot -\frac{10}{7}) = -\frac{4}{7} \]
\[ x_1 = -1 + 2x_2 - 3x_3 = -1 + 2 \cdot -\frac{4}{7} - 3 \cdot -\frac{10}{7} = \frac{15}{7} \]
which agrees with our previous solutions.
A flowchart for the LU factorization procedure is given in the handout.
Norms

When we discuss multicomponent entities such as matrices and vectors, we need a way to express their magnitude (note that we are not talking about the number of elements they contain, but rather about the "bigness" of these elements).

The magnitude of a matrix or a vector is called the norm; the norm of a matrix $A$ is denoted by $\|A\|$. A suitable measure of the magnitude of a matrix should have four properties:

• $\|A\| \geq 0$ and $\|A\| = 0$ if and only if $A = 0$

• $\|kA\| = |k| \|A\|$ for a scalar $k$.

• $\|A+B\| \leq \|A\| + \|B\|$

• $\|AB\| \leq \|A\| \|B\|$
For vectors in 2- and 3-dimensions, the length satisfies these properties and can be used as the norm; this norm is called the Euclidean norm, and is computed thus:

2-D: \( \sqrt{x_1^2 + x_2^2} \)

3-D: \( \sqrt{x_1^2 + x_2^2 + x_3^2} \)

The Euclidean norm of a vector with more than three components is

\[
\sqrt{x_1^2 + x_2^2 + \ldots + x_n^2} = \left( \sum_{i=1}^{n} x_i^2 \right)^{1/2}
\]

This is not the only way to compute a vector norm; the sum of the absolute values of the \( x_i \) and the maximum value of the magnitudes of the \( x_i \) will also serve. We define
\[ \| x \|_1 = \sum_{i=1}^{n} |x_i| = \text{sum of magnitudes} \]

\[ \| x \|_2 = \left( \sum_{i=1}^{n} x_i^2 \right)^{1/2} = \text{Euclidean norm} \]

\[ \| x \|_\infty = \max_{1 \leq i \leq n} |x_i| = \text{maximum-magnitude norm}. \]

So, if \( x = (1.25, 0.02, -5.15, 0) \)

\[ \| x \|_1 = |1.25| + |0.02| + |-5.15| + |0| = 6.42 \]

\[ \| x \|_2 = \left[ 1.25^2 + 0.02^2 + (-5.15)^2 + 0^2 \right]^{1/2} = 5.2996 \]

\[ \| x \|_\infty = |-5.15| = 5.15 \]

For any vector \( x \) in \( \mathbb{R}^n \),

\[ \| x \|_1 \geq \| x \|_2 \geq \| x \|_\infty \]
and \( \|x\|_2 \leq \|x\|_1 \leq \sqrt{n} \|x\|_2 \)
\( \|x\|_\infty \leq \|x\|_2 \leq \sqrt{n} \|x\|_\infty \)
\( \|x\|_\infty \leq \|x\|_1 \leq n \|x\|_\infty \)
Thus, for a given \( n \), any two of the norms differ by at most a small constant, so they are all equivalent in the sense that if one is small, they must all be proportionally small. Hence, we can choose whichever norm is most convenient in a given context.
Matrix norms that correspond to the above for a square matrix \( A \) are:
\[ \|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{n} |a_{ij}| \]
= maximum column sum
\[ \|A\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{ij}| \]

= maximum row sum
\[ \|A\|_2 = r^{1/2}, \text{ where } r \text{ is the largest eigenvalue of } A^T A. \]

The latter is called the spectral norm of \( A \), and is always less than or equal to \( \|A\|_1 \) and \( \|A\|_\infty \).

The Frobenius norm is defined to be
\[ \|A\|_f = \left( \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}^2 \right)^{1/2} \]

So, if \( A = \begin{bmatrix} 5 & -5 & -7 \\ -4 & 2 & -4 \\ -7 & -4 & 5 \end{bmatrix} \)
\[ \|A\|_f = 15 \]
\[ \|A\|_\infty = 17 \]
\[ \|A\|_1 = 16 \]
\[ \|A\|_2 = 12.03 \]
As there are a number of ways in which the norm of a matrix can be expressed, which way is preferred? Whilst the calculation of the spectral norm usually requires more extensive arithmetic than the others, we often want the norm that puts the smallest upper bound on the magnitude of the matrix, and in this sense the spectral norm is the best.

**Condition Number of a Matrix**

When we solve a set of linear equations \( Ax = b \), our hope is that the calculated solution vector \( \hat{x} \) closely resembles the true solution vector \( x \).
As indicated earlier, pivoting will improve the numerical accuracy of the computed solution. Even so, some linear systems are extremely sensitive to round-off errors and the solution vector is quite inaccurate; a measure of this sensitivity is the condition number.

We can get an appreciation of ill-conditioning by examining the following situation. Consider the system

\[
\begin{bmatrix}
1.01 & 0.99 \\
0.99 & 1.01
\end{bmatrix}
\begin{bmatrix}
x \\
y
\end{bmatrix}
=
\begin{bmatrix}
2.00 \\
2.00
\end{bmatrix}
\]

for which the solution is \( x = 1, y = 1 \).
Suppose we modify the right-hand side just slightly
\[
\begin{bmatrix}
1.01 & 0.99 \\
0.99 & 1.01 \\
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
\end{bmatrix} =
\begin{bmatrix}
2.02 \\
1.98 \\
\end{bmatrix}
\]
for which the solution is \( x = 2, y = 0 \).
Again, modifying the right-hand side slightly
\[
\begin{bmatrix}
1.01 & 0.99 \\
0.99 & 1.01 \\
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
\end{bmatrix} =
\begin{bmatrix}
1.98 \\
2.02 \\
\end{bmatrix}
\]
gives a solution \( x = 0, y = 2 \).
So, even though the three input vectors for \( b \) are close together, we get very different outputs for the \( \hat{x} \) vectors.
An interesting phenomenon of an ill-conditioned system is that we cannot test for the accuracy of the computed solution by substituting it into the
equations to see whether the right-hand sides are reproduced. To give an example, let us use 3 digit arithmetic to solve the system
\[
\begin{bmatrix}
0.641 & 0.242 \\
0.321 & 0.121
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} =
\begin{bmatrix}
0.883 \\
0.442
\end{bmatrix}
\]
Gaussian elimination with partial pivoting yields
\[
\begin{bmatrix}
0.641 & 0.242 \\
0 & 0.000242
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2
\end{bmatrix} =
\begin{bmatrix}
0.883 \\
0.000383
\end{bmatrix}
\]
and back-substitution then yields
\[
\hat{x} =
\begin{bmatrix}
0.782 \\
1.58
\end{bmatrix}.
\]
The residual for this solution is
\[ r = b - A\hat{x} = \begin{bmatrix} -0.000622 \\ -0.000202 \end{bmatrix} \]
and the exact solution for the system is \( x = \begin{bmatrix} 1.00 \\ 1.00 \end{bmatrix} \).
Hence we get a small residual but a poor solution. The reason is that the condition number, defined as \( \|A\| \|A^{-1}\| \), is large.
Generally, if we let \( e = x - \hat{x} \)
and \( r = b - A\hat{x} \)
\[ = Ax - A\hat{x} \]
\[ = A(x - \hat{x}) \]
\[ = Ae \]
then \( \|r\| \leq \|A\| \|e\| \) ... (1)
And since \( e = A^{-1}r \),
\[ \|e\| \leq \|A^{-1}\| \|r\| \) ... (2)
From (1) and (2),
\[
\frac{\|r\|}{\|A\|} \leq \|e\| \leq \|A^{-1}\| \|r\| \quad (3)
\]
Similarly, for \(b = Ax\) and \(x = A^{-1}b\):
\[
\frac{\|b\|}{\|A\|} \leq \|x\| \leq \|A^{-1}\| \|b\| \quad (4)
\]
From (3) and (4),
\[
\frac{1}{\|A\|\|A^{-1}\|} \frac{\|r\|}{\|b\|} \leq \frac{\|e\|}{\|x\|} \leq \|A\|\|A^{-1}\| \frac{\|r\|}{\|b\|}
\]
That is,
\[
\frac{1}{\text{cond.no.} \|b\|} \frac{\|r\|}{\|x\|} \leq \frac{\|e\|}{\|x\|} \leq \text{cond.no.} \frac{\|r\|}{\|b\|}
\]
(Proof: The inequality on the right is 
\[\|e\| \leq \|A\|\|A^{-1}\| \|r\| \|x\|\]
and this is true since
\[\|e\| = \|A^{-1}(r - Ax)\| \leq \|A^{-1}\| \|r\| \|A\| \|x\|\]
The inequality on the left is
\[ \| r \| \| x \| \leq \| A \| \| A^{-1} \| \| b \| \| e \| \]
and this follows from
\[ \| r \| \| x \| = \| A e \| \| A^{-1} b \| \]
\[ \leq \| A \| \| e \| \| A^{-1} \| \| b \| \]
Hence the relative error in the solution vector can be as great as the relative residual multiplied by the condition number or as small as the relative residual divided by the condition number. When the condition number is large, as it is in the above example (more than 4000), the residual gives little information about the accuracy of \( \hat{x} \). When the condition number is near unity, the relative residual is a good measure of the relative error of \( x \).
Notice that solving a linear system by Gaussian elimination with partial pivoting followed by back-substitution nearly always yields a very small relative residual, no matter how ill-conditioned a system may be. So, a small relative residual is not necessarily a good indicator that a computed solution is close to the true solution unless the system is well-conditioned.

When we solve a linear system of equations, we are usually doing so to determine values for a physical system for which the set of equations is a model. We use measured values to evaluate the coefficients of the equations. Assume that the errors in measurement cause errors
in the coefficients of \( A \) so that the actual set of equations being solved is \((A + E) \hat{x} = b\) or \( \hat{A} \hat{x} = b \).

We want to know the error in \( x \).

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

\[
x - \hat{x} = A^{-1}E \hat{x}
\]

\[
\|x - \hat{x}\| \leq \|A^{-1}\| \|E\| \|\hat{x}\|
\]

\[
\frac{\|x - \hat{x}\|}{\|\hat{x}\|} \leq \text{cond.no.} \frac{\|E\|}{\|A\|}
\]
The effect of this is that, if the coefficients of $A$ are known to 4-digit precision, and the condition number is 1000, the computed vector $x$ may have only 1 digit of accuracy. We can similarly show that the condition number of the matrix determines the possible relative change in the solution due to a given relative change in the right hand side vector:

We have $Ax = b$

And $A\hat{x} = \hat{b}$ for a perturbed RHS

So $A(x - \hat{x}) = b - \hat{b}$

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

$\|x - \hat{x}\| \leq \|A^{-1}\| \|b - \hat{b}\|$
\[
\frac{\|x - \hat{x}\|}{\|x\|} \leq \frac{\|A^{-1}\| \|b - \hat{b}\|}{\|x\|} \\
\frac{\|x - \hat{x}\|}{\|x\|} \leq \frac{\|A\| \|A^{-1}\| \|b - \hat{b}\|}{\|A\| \|x\|}
\]

But \(\|b\| \leq \|A\| \|x\|\)

So \(\frac{\|x - \hat{x}\|}{\|x\|} \leq \text{cond.no.} \frac{\|b - \hat{b}\|}{\|b\|}\)
Special Types of Linear Systems
So far we have assumed that a linear system is dense, i.e. most of the matrix elements are non-zero. If, however, the matrix has special properties, then work and storage can be saved in solving the system. We will deal first with tridiagonal matrices, i.e. those that have non-zero elements only on the diagonal and in positions adjacent to the diagonal. Such matrices occur frequently in real-world problems. Consider, for example, the system

\[-4x_1 + 2x_2 = 0\]
\[x_1 - 4x_2 + x_3 = -4\]
\[x_2 - 4x_3 + x_4 = -11\]
\[x_3 - 4x_4 + x_5 = 5\]
\[2x_4 - 4x_5 = 6\]
We can store the augmented matrix of this system in a 5 x 4 matrix as follows:

\[
\begin{bmatrix}
0 & -4 & 2 & 0 \\
1 & -4 & 1 & -4 \\
1 & -4 & 1 & -11 \\
1 & -4 & 1 & 5 \\
2 & -4 & 0 & 6
\end{bmatrix}
\]

Not only can the equations be economically stored, they can also be solved speedily:

- In tridiagonal systems that are encountered in practice, they are normally diagonally dominant, so that pivoting is unnecessary.
- When Gaussian elimination is used, we get the LU decomposition of the
coefficient matrix at a cost in multiplications/divisions that is only of the order of \( n \), which represents substantial savings over the full system, which is of the order of \( n^3 \).

In general, if

\[
A = \begin{bmatrix}
b_1 & c_1 & 0 & \ldots & 0 \\
 a_2 & b_2 & c_2 & \ddots & \\
 0 & \ddots & \ddots & \ddots & 0 \\
 0 & \ddots & a_{n-1} & b_{n-1} & c_{n-1} \\
 0 & \ldots & 0 & a_n & b_n
\end{bmatrix}
\]
\[
L = \begin{bmatrix}
1 & 0 & \cdots & \cdots & 0 \\
m_2 & 1 & \ddots & \ddots & \\
0 & \ddots & \ddots & \ddots & \\
\vdots & \ddots & m_{n-1} & 1 & 0 \\
0 & \cdots & 0 & m_n & 1 \\
\end{bmatrix}
\]

\[
U = \begin{bmatrix}
d_1 & c_1 & 0 & \cdots & 0 \\
0 & d_2 & c_2 & \ddots & \\
\vdots & \ddots & \ddots & \ddots & 0 \\
\vdots & \ddots & \ddots & d_{n-1} & c_{n-1} \\
0 & \cdots & \cdots & 0 & d_n \\
\end{bmatrix}
\]

where

\[d_1 = b_1\]
for \(i = 2 \text{ to } n\)

\[m_i = a_i/d_{i-1}\]

\[d_i = b_i - m_i c_{i-1}\]
end
In our example

\[
A = \begin{bmatrix}
-4 & 2 & 0 & 0 & 0 & 0 \\
1 & -4 & 1 & 0 & 0 & 0 \\
0 & 1 & -4 & 1 & 0 & 0 \\
0 & 0 & 1 & -4 & 1 & 0 \\
0 & 0 & 0 & 2 & -4 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 \\
.2500 & 1 & 0 & 0 & 0 & 0 \\
0 & -.2857 & 1 & 0 & 0 & 0 \\
0 & 0 & -.2692 & 1 & 0 & 0 \\
0 & 0 & 0 & -.5361 & 1 & 0 \\
\end{bmatrix}
\]

\[
L = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
-.2500 & 1 & 0 & 0 & 0 & 0 \\
0 & -.2857 & 1 & 0 & 0 & 0 \\
0 & 0 & -.2692 & 1 & 0 & 0 \\
0 & 0 & 0 & -.5361 & 1 & 0 \\
\end{bmatrix}
\]

\[
U = \begin{bmatrix}
-4 & 2 & 0 & 0 & 0 & 0 \\
0 & -3.5000 & 1 & 0 & 0 & 0 \\
0 & 0 & -3.7143 & 1 & 0 & 0 \\
0 & 0 & 0 & -3.7308 & 1 & 0 \\
0 & 0 & 0 & 0 & -3.4639 & 0 \\
\end{bmatrix}
\]
We have now completed our consideration of the tridiagonal matrix - a special type of matrix which results in lower storage and a lower operation count than a dense matrix. We move on now to consider symmetric (i.e. $A = A^T$) positive definite (i.e. $x^T Ax > 0$ for all $x \neq 0$) systems, which require only about half as much storage and half as much work as are required for the LU factorization of a general matrix. If a matrix $A$ is symmetric and positive definite, then an LU factorization can be arranged so that $U = L^T$ i.e. $A = LL^T$, where $L$ is lower triangular and has positive diagonal entries (but not unit diagonal entries).
This is known as the Cholesky factorization of $A$.
In the $2 \times 2$ case, we have
\[
\begin{bmatrix}
a_{11} & a_{21} \\
a_{21} & a_{22}
\end{bmatrix} =
\begin{bmatrix}
l_{11} & 0 \\
l_{21} & l_{22}
\end{bmatrix}
\begin{bmatrix}
l_{11} & l_{21} \\
0 & l_{22}
\end{bmatrix}
\]
from which
\[
l_{11} = \sqrt{a_{11}}
\]
\[
l_{21} = \frac{a_{21}}{l_{11}}
\]
\[
l_{22} = \sqrt{a_{22} - l_{21}^2}
\]
In the general $n \times n$ case, an algorithm which performs the computations is as follows:
for $j = 1$ to $n$
  for $k = 1$ to $j - 1$
    for $i = j$ to $n$
      $a_{ij} = a_{ij} - a_{ik} \cdot a_{jk}$
    end
  end
end

$A_{jj} = \sqrt{a_{jj}}$

for $k = j + 1$ to $n$
  $a_{kj} = a_{kj} / a_{jj}$
end

{ for each column $j$ }
{ loop over all prior columns $k$ }
{ subtract a multiple of column $k$ from column $j$ }

{ scale column $j$ by square root of diagonal entry }

Note that no pivoting is required for numerical stability.
To illustrate the Cholesky factorization algorithm, we will compute the value of $L$ for the symmetric positive definite matrix
\[
A = \begin{bmatrix}
5.0 & 0 & 2.5 \\
0 & 2.5 & 0 \\
2.5 & 0 & 2.125 \\
\end{bmatrix}
\]

The first column has no prior columns, so it is scaled by the square root of the diagonal entry, \(\sqrt{5}\), to give
\[
\begin{bmatrix}
2.236 \\
0 & 2.5 \\
1.118 & 0 & 2.125 \\
\end{bmatrix}
\]

The second column now requires updating by subtracting a multiple of the first column. But in our example the multiplier in the second row of the first column is zero, so that the second column is unaffected by the first column. So, the second column
is scaled by the square root of its diagonal entry, $\sqrt{2.5}$, to give
\[
\begin{bmatrix}
2.236 \\
0 & 1.581 \\
1.118 & 0 & 2.125
\end{bmatrix}
\]
Finally, the third column must be updated by subtracting multiples of the previous two columns. The multipliers for the first two columns, found in the third row, are 1.118 and zero. Updating the third column gives
\[
\begin{bmatrix}
2.236 \\
0 & 1.581 \\
1.118 & 0 & 0.875
\end{bmatrix}
\]
Taking the square root of the third diagonal entry then yields the result
\[
L = \begin{bmatrix}
2.236 \\
0 & 1.581 \\
1.118 & 0 & 0.935
\end{bmatrix}
\]

Another illustration of the Cholesky factorization for a 3x3 matrix is given in the handout.