A linear equation is one in which all the variables are raised, at most, to the first power and no two variables are multiplied together.

E.g., \(3x + 2y + z = 3\) is linear.

\(3x^2 + 2y - 3xy = 0\) is not linear. (Why not?)

A system of linear equations is any set of linear equations.

A system can be OVER-DETERMINED \(\rightarrow\) more equations than variable, or UNDER-DETERMINED \(\rightarrow\) fewer equations than variables.
A system is **LINEARLY INDEPENDENT** if no single equation is a linear combination of the others. Otherwise, it is **LINEARLY DEPENDENT**.

E.g., \[
\begin{align*}
x + y &= 1 \\
2x + y &= 1 \\
4x + 3y &= 3
\end{align*}
\]

**NOT** linearly independent since (3) = 2 \times (1) + (2)

BF-9 §8.2, p 512, Def 8.1
We speak about an $n \times m$ ("$n$ by $m$") linear system or "$n$ equations in $m$ unknowns."

**First** number refers to the number of **EQUATIONS**.

**Second** number refers to the number of **VARIABLES**.

In many applications, $n = m$.

We can employ a notational abstraction for the equation formal of linear systems to simplify our analysis, and make use of matrices and vectors.
[4-3]

A **matrix** is a rectangular arrangement (array) of numbers.

[NOTATION: A matrix is usually delimited by (large) parentheses or brackets.]

We speak about an \( n \times m \) matrix, i.e., \( n \) rows and \( m \) columns, corresponding to \( n \) equations in \( m \) unknowns.

E.g., \[
\begin{pmatrix}
3 & 2 & 1 \\
1 & 2 & 5
\end{pmatrix}
\] is a \( 2 \times 3 \) matrix.
A **vector** is a matrix in which one dimension is 1. We usually speak of row vectors or column vectors. “Default” vector is a column vector.

E.g., \[
\begin{pmatrix}
0 \\
1 \\
1
\end{pmatrix}
\] is a (column) vector.

\[
\begin{pmatrix}
0 \\
1 \\
1
\end{pmatrix}
\] \text{3×1 matrix}

**NOTATION:** Frequently, for the sake of typography, a column vector is written as the transpose of a row vector, e.g., we write \((1, 2, 3)^T\) instead of \[
\begin{pmatrix}
1 \\
2 \\
3
\end{pmatrix}
\], where “transpose” indicates that the values of the \(i^{th}\) row/column become the values of the \(i^{th}\) column/row of the transposed vector/matrix.
NOTATION: Matrix $A$ has elements $a_{ij}$ with $i$ indicating the row and $j$ indicating the column.

Some books use small Greek letters for matrix elements, e.g., $\alpha_{ij}$.

We write

$$A = (a_{ij})_{n \times m} = (\alpha_{ij})_{n \times m}$$

If $n = m$, $A$ is a SQUARE matrix.

In a square matrix $A$, the elements $\{a_{ii} | i = 1, \ldots, n\}$ form the MAIN DIAGONAL. We also speak about the SUPERDIAGONAL (elements to the immediate right of the main diagonal) and the SUBDIAGONAL (elements to the immediate left of the main diagonal).
MATRIX ARITHMETIC [4-4]

Pre-note: Arithmetic operations can be defined on matrices, but we have to be concerned about rules regarding appropriate dimensions.

I EQUALITY: $A = B$ iff both are of the same dimensions (e.g., $n \times m$), and iff $a_{ij} = b_{ij}$ for every $i, j$.

III SCALAR MULTIPLICATION is defined as follows: If $A = (a_{ij})$ and $k$ is a scalar, then $kA = (ka_{ij})$.

II ADDITION/SUBTRACTION of $A$ and $B$ is defined element-wise (for corresponding elements) iff $A$ and $B$ are both of the same dimensions. Normal arithmetic rules hold for matrix addition (assoc., commut.). The additive identity is the zero matrix, $0_{n \times m}$, in which all elements are zero.
IV VECTOR INNER PRODUCT (Scalar Product of vectors [so-called because it gives a scalar result], “Dot” product) is defined between two vectors of the same length as follows:

If \( \vec{a} = (a_i)_n \), \( \vec{b} = (b_i)_n \)

then \( \vec{a}^T \cdot \vec{b} = \vec{a}^T \vec{b} = (\vec{a}, \vec{b}) = \sum_{i=1}^{n} a_i b_i. \)

E.g., Given \( \vec{a} = (2, 3, 1)^T \), \( \vec{b} = (1, 4, 2)^T \)
we have \( (\vec{a}, \vec{b}) = \vec{a}^T \cdot \vec{b} = 2 \cdot 1 + 3 \cdot 4 + 1 \cdot 2 \)
\[ = 2 + 12 + 2 = 16. \]

**NOTE:** Technically, \( \vec{a} \cdot \vec{b} \) is incorrect, since by the normal rules for matrix multiplication, one multiplies a row by a column and \( \vec{a} \) is a column vector.
V MATRIX MULTIPLICATION is defined between “compatible” matrices rather than equi-sized matrices.

If $A$ is $n \times p$ and $B$ is $p \times m$, then multiplication is defined and if $C = A \cdot B$, then $C$ is $n \times m$ and $c_{ij} =$ vector inner product of row $i$ of $A$ and column $j$ of $B$.

E.g.,

$$
\begin{pmatrix}
3 & 2 \\
1 & 2 \\
5 & 0
\end{pmatrix}
\begin{pmatrix}
2 & 1 \\
2 & 0
\end{pmatrix}
= 
\begin{pmatrix}
3 \cdot 2 + 2 \cdot 2 & 3 \cdot 1 + 2 \cdot 0 \\
1 \cdot 2 + 2 \cdot 2 & 1 \cdot 1 + 2 \cdot 0 \\
5 \cdot 2 + 0 \cdot 2 & 5 \cdot 1 + 0 \cdot 0
\end{pmatrix}
= 
\begin{pmatrix}
10 & 3 \\
6 & 1 \\
10 & 5
\end{pmatrix}
$$
NOTE: Matrix multiplication is **NOT** commutative in general, i.e., $AB \neq BA$. In fact, $BA$ may not even be defined!

The multiplicative identity is the **Identity Matrix**, a square matrix with 1’s on the main diagonal and zeroes elsewhere. An identity matrix of $m$ rows and columns is denoted by $I_m$.

VI Standard multiplication rules hold (i.e., assoc., distrib.), except for commutativity.

**NOTATION:** If $B \cdot B$ is defined, i.e., when $B$ is square, we can write $B^2$, etc.
STRASSEN’S ALGORITHM [4-6]

Since more arithmetic operations frequently lead to more inaccuracy, we can and should ask the question whether we can speed up the matrix multiplication of $A \cdot B = C$.

Suppose

$$A \cdot B = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \cdot \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}$$

By definition of matrix multiplication,

$$c_{11} = a_{11}b_{11} + a_{12}b_{21} \quad c_{12} = a_{11}b_{12} + a_{12}b_{22}$$
$$c_{21} = a_{21}b_{11} + a_{22}b_{21} \quad c_{22} = a_{21}b_{12} + a_{22}b_{22}$$

Thus, to get $C$, the operation count is 8 multiplications and 4 additions.
Instead, try to following, which uses intermediate variables.

\[ m_1 = (a_{12} - a_{22})(b_{21} + b_{22}) \]
\[ m_2 = (a_{11} + a_{22})(b_{11} + b_{22}) \]
\[ m_3 = (a_{11} - a_{21})(b_{11} + b_{12}) \]
\[ m_4 = (a_{11} + a_{12})b_{22} \]
\[ m_5 = a_{11}(b_{12} - b_{22}) \]
\[ m_6 = a_{22}(b_{21} - b_{11}) \]
\[ m_7 = (a_{21} + a_{22})b_{11} \]
\[ c_{11} = m_1 + m_2 - m_4 + m_6 \]
\[ c_{12} = m_4 + m_5 \]
\[ c_{21} = m_6 + m_7 \]
\[ c_{22} = m_2 - m_3 + m_5 - m_7 \]

Here the operation count is 7 multiplications and 18 additions/subtractions.

The 14 additional additions/subtractions may be worth it to reduce the multiplications by 1, especially if matrix multiplication is done often or is part of a larger scheme to multiply larger matrices together.
BLOCK MULTIPLICATION OF MATRICES [4-7]

Strassen’s algorithm is only defined on $2 \times 2$ matrices, but it can be extended if we take the block matrix multiplication into account.

Suppose

\[
A_{2n \times 2n} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \quad B_{2n \times 2n} = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}
\]

where $A_{ij}, B_{ij}$ are all $n \times n$. 
If $C = A \cdot B$, then $C$ can be computed by using the normal rules of matrix multiplication applied to the “blocks” (i.e., four submatrices) of $A$ and $B$. I.e.,

$$C = \begin{pmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{pmatrix}$$

Thus, in theory, one could program Strassen’s algorithm recursively (since the formulas make no use of commutativity), until reaching the $2 \times 2$ matrix cases.
SPECIAL MATRICES [4-8]

Certain types of matrices (linear systems) occur frequently enough to warrant special titles.

UPPER TRIANGULAR

everything below the main diagonal is 0.

\[
\begin{pmatrix}
X & X & X & \cdots & X \\
0 & X & X & \cdots & X \\
0 & 0 & X & \cdots & X \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & X
\end{pmatrix}
\]

LOWER TRIANGULAR

everything above the main diagonal is 0.

\[
\begin{pmatrix}
X & X & 0 & \cdots & 0 \\
X & X & 0 & \cdots & 0 \\
X & X & X & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
X & X & X & \cdots & X
\end{pmatrix}
\]
Banded everything above and below certain minor diagonals is 0. Here the “bandwidth” is five diagonals.

\[
\begin{pmatrix}
X & X & X & 0 & \cdots & 0 \\
X & X & X & X & \cdots & 0 \\
X & X & X & X & \cdots & 0 \\
0 & X & X & X & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \ddots \\
0 & 0 & 0 & 0 & \cdots & X
\end{pmatrix}
\]

Tri-Diagonal (Special case of banded.) everything other than the main diagonal, and 2 diagonals on either side, is 0.

\[
\begin{pmatrix}
X & X & 0 & \cdots & 0 \\
X & X & X & \cdots & 0 \\
0 & X & X & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \ddots \\
0 & 0 & 0 & \cdots & X
\end{pmatrix}
\]
These are important because:

1) similarity transformations may make these matrices equivalent to “full” matrices.

2) storage restrictions may require the use of matrices with many zeroes (e.g., a tridiagonal matrix only needs $3n - 2$ words as compared to $n^2$ words for a full matrix).

3) certain procedures are drastically simplified with “special” matrices.
Sparse matrices are those with very many zeroes. Frequently, sparse matrices are banded or can easily be transformed to banded matrices. (Matlab has special functions for sparse matrices and a special storage format – see Matlab HELP for more information.)

NOTE: Sparse, banded matrices are frequently stored as 3 vectors (if tridiagonal) – 1 vector for each diagonal.

— This leads to a drastic saving of storage. As noted earlier, a tridiagonal matrix requires $3n - 2$ words as opposed to $n^2$. If $n = 100$, tridiagonal storage would require 298 words as opposed to 10,000.

— Special rules (i.e., subroutines, functions) must be developed for matrix-matrix and matrix-vector products to take into account sparsity and storage method.
BRIEF ANALYSIS OF MATRIX OPERATIONS [4-10]

MATRIX MULT: \( C = AB \). Assume all are \( n \times n \).

To help analysis, think of \( A \) as a column vector of row vectors \( \vec{a}_i^T \) and \( B \) as a row vector of column vectors \( \vec{b}_i \).

Thus \( A = \begin{pmatrix} \vec{a}_1^T \\ \vec{a}_2^T \\ \vdots \\ \vec{a}_n^T \end{pmatrix} \), \( B = (\vec{b}_1 \vec{b}_2 \cdots \vec{b}_n) \) and \( C = (c_{i,j}) = (\vec{a}_i^T \cdot \vec{b}_j) \).

Thus to compute \( C \), we need \( n^2 \) elements, each of which is a vector product, and EACH vector (dot) product needs \( n \) multiplications and \( n - 1 \) additions.

Thus, the total mults for matrix mult. is \( n^2 \cdot n = n^3 \). (Cf. p 4-6)
E.g.,

\[
\begin{pmatrix}
1 & 2 & 3 \\
2 & 3 & 4 \\
1 & 1 & 1
\end{pmatrix}
\begin{pmatrix}
1 & 2 & 1 \\
2 & 3 & 1 \\
3 & 4 & 1
\end{pmatrix}
= 
\begin{pmatrix}
(1 & 2 & 3) \\
(2 & 3 & 4) \\
(1 & 1 & 1)
\end{pmatrix}
\begin{pmatrix}
1 \\
2 \\
3
\end{pmatrix}
\begin{pmatrix}
2 \\
3 \\
4
\end{pmatrix}
\begin{pmatrix}
1
\end{pmatrix}
= 
\begin{pmatrix}
\vec{a}_1^T \\
\vec{a}_2^T \\
\vec{a}_3^T
\end{pmatrix}
(\vec{b}_1 \vec{b}_2 \vec{b}_3)
= 
\begin{pmatrix}
\vec{a}_1^T \cdot \vec{b}_1 & \vec{a}_1^T \cdot \vec{b}_2 & \vec{a}_1^T \cdot \vec{b}_3 \\
\vec{a}_2^T \cdot \vec{b}_1 & \vec{a}_2^T \cdot \vec{b}_2 & \vec{a}_2^T \cdot \vec{b}_3 \\
\vec{a}_3^T \cdot \vec{b}_1 & \vec{a}_3^T \cdot \vec{b}_2 & \vec{a}_3^T \cdot \vec{b}_3
\end{pmatrix}
\]

There are \(9 = 3^2\) elements in the product matrix.

Each element is a vector (dot) product.

Each vector product requires 3 mults and 2 adds.

Thus, total mults is \(27 = 3 \cdot 3^2 = 3^3\).
MATRIX-VECTOR PRODUCT $\vec{c} = A\vec{b}$  \[4-11\]

Similar analysis and set-up for a “MATVEG” (matrix-vector product).

Let $A = \begin{pmatrix} \vec{a}_1^T \\ \vec{a}_2^T \\ \vdots \\ \vec{a}_n^T \end{pmatrix}$. Then $A\vec{b} = \begin{pmatrix} \vec{a}_1^T \\ \vec{a}_2^T \\ \vdots \\ \vec{a}_n^T \end{pmatrix} \vec{b} = \vec{c} = (c_i) = (\vec{a}_i^T \vec{b})$.

Thus, to compute $\vec{c}$, we need $n$ elements, each of which is a vector product. Therefore, using similar analysis as before, the total mults for a matrix-vector product is $n \cdot n = n^2$. 


SPECIAL MATRICES: TRIDIAGONAL

Suppose in the matrix-vector product, $A$ were tridiagonal and we have a special routine which does only the necessary multiplications (i.e., it ignores 0 entries). Therefore, for $i = 1, \ldots, n$

$$c_i = a_{i1} b_{1i}^* + a_{i2} b_{2i}^* + a_{i3} b_{3i}^*$$

(where the subscripts are appropriately defined).

Therefore, to compute all of $\vec{c}$, we need to compute $n c_i$'s, each of which requires 3 mults (actually $c_1$ and $c_n$ only require 2 mults each), for a total of $3n(-2)$ mults.
Def. (BF-9 Def 6.11) If $A$ is square, $n \times n$, and if there exists a matrix $B$ such that

$$AB = BA = I_n$$

then $A$ is called **non-singular** or **invertible**.

Def. If $A$ is invertible, and $AB = I_n$, then $B$ is called the **inverse** of $A$.

Def. If $A$ is **not** invertible, $A$ is said to be **singular**.

Notation: The inverse of $A$ is written $A^{-1}$.
Examples: (1) For \( n = 2 \) and \( A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \) then \( A^{-1} = \frac{1}{\text{det}(A)} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \)

where \( \text{det}(A) = \text{determinant of } A = |A| = ad - bc \).

(2) \( A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \) then \( A^{-1} = \begin{pmatrix} -2 & 1 \\ 3/2 & -1/2 \end{pmatrix} \)

(3) \( A_1 = \begin{pmatrix} 1 & 2 & 1 \\ 1 & 0 & 1 \\ 2 & 2 & 1 \end{pmatrix} \) then \( A_1^{-1} = \begin{pmatrix} -1 & 0 & 1 \\ 1/2 & -1/2 & 0 \\ 1 & 1 & -1 \end{pmatrix} \)

\( A_2 = \begin{pmatrix} 1 & 2 & 1 \\ 1 & 0 & 1 \\ 2 & 2 & 2 \end{pmatrix} \) then \( A \) is singular (note that the third row is the sum of the first two rows and the last element is the only difference between \( A_2 \) and \( A_1 \)).
Def. (BF-9 §9.2, Def 9.11, p 571) If $A$ and $B$ are square and equi-sized matrices, and if there exists a non-singular matrix $P$ such that

$$B = P^{-1}AP,$$

then $A$ and $B$ are said to be SIMILAR.

COMMENT: (cf. notes [4-18]) Given $Ax = b$, suppose we also know $A^{-1}$. Then,

$$A^{-1}Ax = A^{-1}b$$

$$\implies Ix = A^{-1}b$$

or $$x = A^{-1}b$$

BUT, it is difficult to find $A^{-1}$ QUICKLY and ACCURATELY, especially if $A$ is large. There are too many operations. Thus, this seemingly straightforward approach (similar to solving a single linear equation, $ax = b \implies x = a^{-1}b = b/a$), is NOT practical.