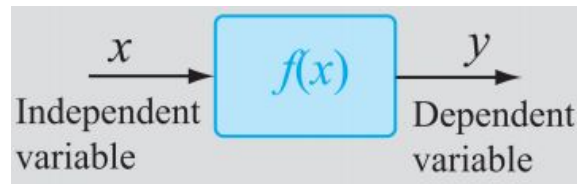


Formulae that are highlighted in green will be given in the exam, unless they are to be derived.

02 Mathematical Background

Functions



The span of x is called the **domain**.

The span of y is called the **range**.

The span can also be called an **interval**:

- Open interval: $]a, b[$ or (a, b) $a > x > b$
- Closed interval: $[a, b]$ $a \geq x \geq b$

Limit of a Function

For continuous functions:

$$\lim_{x \rightarrow a} f(x) = f(a) = L$$

$f(x)$ may be discontinuous or singular at a . In these cases, the limit is said **not to exist**.

Continuity of a Function

A function $f(x)$ at $x = a$ is said to be continuous if:

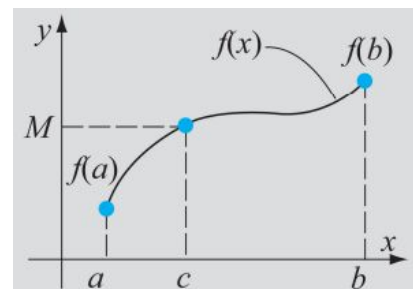
- $f(a)$ exists
- $\lim_{x \rightarrow a} f(x) = f(a)$

The Intermediate Value Theorem

If both:

1. $f(x)$ is continuous in $[a, b]$
2. M is any number between $f(a)$ and $f(b)$

then $\exists c \in [a, b]$ such that $f(c) = M$.



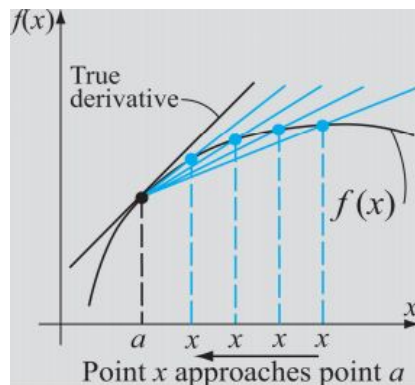
Derivatives of a Function

The derivative of a function $y = f(x)$ is denoted as either:

1. $\frac{dy}{dx}$
2. $\frac{df(x)}{dx}$
3. $f'(x)$

It is defined as:

$$\left. \frac{dy}{dx} \right|_{x=a} = f'(a) = \lim_{x \rightarrow a} \frac{f(x) - f(a)}{x - a}$$



$\frac{f(x) - f(a)}{x - a}$ is the slope of the secant connecting $(a, f(a))$ and $(x, f(x))$.

In the limit as $x \rightarrow a$, we get a slope of tangent at $(a, f(a))$.

This slope is equivalent to the rate of change of $f(x)$ with respect to x at $x = a$.

The Chain Rule

$$\frac{dy}{dx} = \frac{dy}{du} \frac{du}{dx}$$

Example:

$$f(x) = (\sin x)^2$$

$$u = \sin x$$

$$f'(x) = \frac{dy}{dx} = \frac{dy}{du} \frac{du}{dx} = \frac{dy}{d(\sin x)} \frac{d(\sin x)}{dx} = 2(\sin x) * \cos x = 2 \sin x \cos x$$

The Product Rule

$$\frac{du(x)v(x)}{dx} = u(x) \frac{dv(x)}{dx} + v(x) \frac{du(x)}{dx}$$

Example:

$$y = (x^3 + 7x - 1)(5x + 2)$$

$$u(x) = x^3 + 7x - 1$$

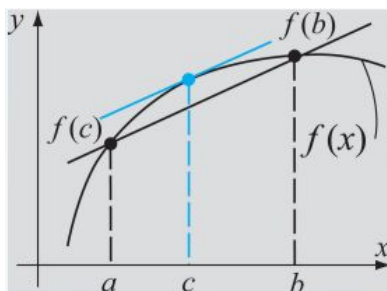
$$\frac{du(x)}{dx} = 3x^2 + 7$$

$$v(x) = 5x + 2$$

$$\frac{dv(x)}{dx} = 5$$

$$\frac{du(x)v(x)}{dx} = u(x) \frac{dv(x)}{dx} + v(x) \frac{du(x)}{dx} = (x^3 + 7x - 1)(5) + (5x + 2)(3x^2 + 7) = \dots$$

The Mean Value Theorem (For Derivatives)



If $f(x)$ is a continuous function in $[a, b]$ and is differentiable in the open interval $]a, b[$, then $\exists c \in]a, b[$ such that:

$$f'(c) = \frac{f(b) - f(a)}{b - a}$$

In other words, at some point in the interval, the slope of the tangent line will equal the slope of the secant line between $f(a)$ and $f(b)$.

Integral / Antiderivative of a Function

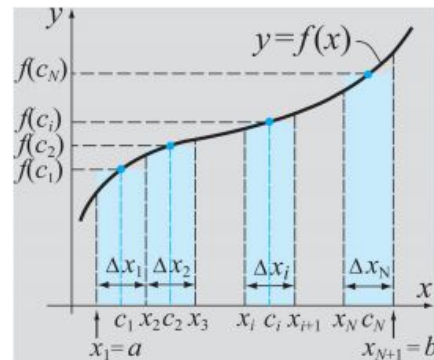
If $f(x) = \frac{dF(x)}{dx}$ then:

- The **indefinite integral** of $f(x)$ is $F(x)$:

$$F(x) = \int f(x) dx$$

- The **definite integral** of $f(x)$ is I :

$$I = \int_a^b f(x) dx$$



Assuming $f(x)$ is defined and continuous over $[a, b]$ then:

$$\int_a^b f(x) dx = \lim_{\Delta x_i \rightarrow 0} \sum_{i=1}^n f(c_i) \Delta x_i$$

Fundamental Theorems of Calculus

$$\int_a^b f(x) dx = F(b) - F(a)$$

and

$$\frac{d}{dx} \left(\int_a^x f(\zeta) d\zeta \right) = f(x)$$

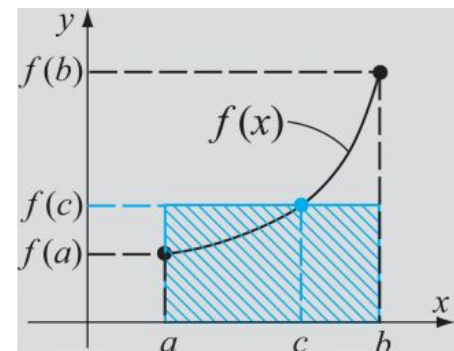
The Mean Value Theorem (For Integrals)

If $f(x)$ is continuous over $[a, b]$ then $\exists c \in [a, b]$ such that:

$$\int_a^b f(x) dx = f(c)(b-a)$$

The **average value** of $f(x)$ over the interval is:

$$\langle f \rangle = \frac{1}{b-a} \int_a^b f(x) dx$$

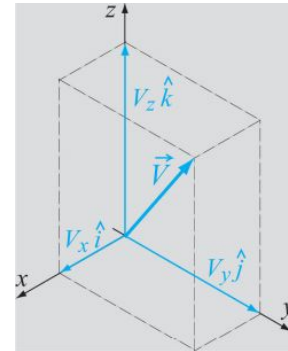


In other words, there must be some value $f(c)$ within the interval that takes on the average value of $f(x)$ over the interval $[a, b]$.

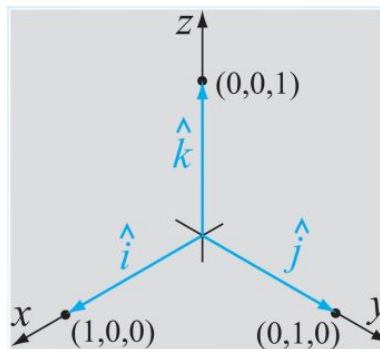
Vectors

Vectors are quantities that have a **magnitude** and **direction**.
Scalars are quantities with **magnitude only**.

To denote the vector V we write \vec{V} (or V^{\rightarrow} in these notes).
The **magnitude** of a vector is denoted V or $|\vec{V}|$.



A vector may be represented graphically as a **directed line segment**.
Projections of the vector onto each of the coordinate axes define the **components** of the vector.



\hat{i} , \hat{j} and \hat{k} are the **unit vectors** in the x , y and z directions respectively.

A unit vector is a vector with a **magnitude of unity** in a particular direction:

$$\hat{V} = \frac{\vec{V}}{|\vec{V}|}$$

We can therefore write:

$$\vec{V} = V_x \hat{i} + V_y \hat{j} + V_z \hat{k}$$

where $V_x \hat{i}$ is the x -component of \vec{V} (the projection of \vec{V} onto the x -axis) and so on.

A vector may be written by listing the **magnitudes of its components** in a row or column:

e.g. $\vec{V} = [V_x \ V_y \ V_z]$

The **magnitude** of a vector in three-dimensional Cartesian space is its **length**:

$$|\vec{V}| = \sqrt{V_x^2 + V_y^2 + V_z^2}$$

The unit vector in the direction of \vec{V} is:

$$\hat{V} = \frac{\vec{V}}{|\vec{V}|} = \frac{V_x \hat{i} + V_y \hat{j} + V_z \hat{k}}{\sqrt{V_x^2 + V_y^2 + V_z^2}} = \hat{l}i + \hat{m}j + \hat{n}k$$

In mathematics, a vector is a set or list of numbers written in a row or column:

e.g. $[V] = [V_1 \ V_2 \ \dots \ V_n]$

Addition and Subtraction of Vectors

$$\begin{aligned} \vec{V} + \vec{U} &= [V_i + U_i] = [V_1 + U_1, V_2 + U_2, \dots, V_n + U_n] \\ \vec{V} - \vec{U} &= [V_i - U_i] = [V_1 - U_1, V_2 - U_2, \dots, V_n - U_n] \end{aligned}$$

Multiplication by a Scalar

$$\alpha \vec{V} = \alpha [V_i] = [\alpha V_1, \alpha V_2, \dots, \alpha V_n]$$

Transpose of a Vector

If $\vec{V} = [V_1, V_2, \dots, V_n]$ then:

$$\vec{V}^T = \begin{bmatrix} V_1 \\ V_2 \\ \dots \\ V_n \end{bmatrix}$$

Multiplication of Two Vectors

Dot Product / Scalar Product:

$$\vec{V} \cdot \vec{U} = [V_i][U_i] = V_1 U_1 + V_2 U_2 + \dots + V_n U_n$$

or

$$\vec{V} \cdot \vec{U} = |\vec{V}| |\vec{U}| \cos \theta \quad (\text{Multiply the magnitudes of both vectors and by } \cos \theta)$$

Cross Product / Vector Product:

$$\vec{W} = \vec{V} \times \vec{U} = \vec{V} \otimes \vec{U} = |\vec{V}| |\vec{U}| \sin \theta \hat{w}$$

$$\begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ V_x & V_y & V_z \\ U_x & U_y & U_z \end{vmatrix} = \begin{vmatrix} V_y & V_z \\ U_y & U_z \end{vmatrix} \hat{i} - \begin{vmatrix} V_x & V_z \\ U_x & U_z \end{vmatrix} \hat{j} + \begin{vmatrix} V_x & V_y \\ U_x & U_y \end{vmatrix} \hat{k}$$

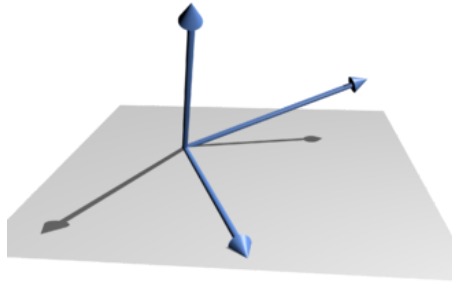
$$= (V_y U_z - V_z U_y) \hat{i} - (V_x U_z - V_z U_x) \hat{j} + (V_x U_y - V_y U_x) \hat{k}$$

Linear Dependence and Independence on a Set of Vectors

A set of vectors $\vec{V}_1, \vec{V}_2, \dots, \vec{V}_n$ is said to be **linearly independent** if:

$\alpha_1 \vec{V}_1 + \alpha_2 \vec{V}_2 + \dots + \alpha_n \vec{V}_n = \vec{0}$ is satisfied *iff* $\alpha_1 = \alpha_2 = \dots = \alpha_n = 0$.

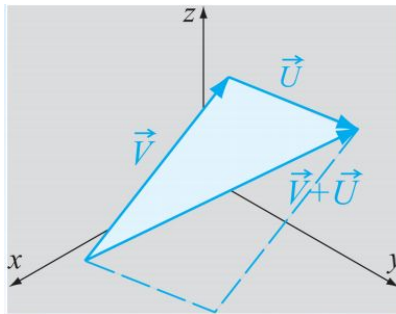
Otherwise, the set of vectors is said to be **linearly dependent**.



i.e. If one of the vectors in the set can be defined as a linear combination of the others, then the set of vectors is said to be linearly dependent.

The Triangle Inequality

$$|\vec{V} + \vec{U}| \leq |\vec{V}| + |\vec{U}|$$



Matrices & Linear Algebra

A matrix is a rectangular array of numbers. The size of the matrix refers to the number of rows and columns that it contains.

$$[a] = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{mn} \end{bmatrix}$$

An element a_{ij} refers to row i and column j .

A **row vector** is a $1 \times n$ matrix.

A **column vector** is a $n \times 1$ matrix.

Multiplication by a Scalar

If $[a] = [a_{ij}]$ is a matrix, and α is a scalar, then $\alpha[a] = [\alpha a_{ij}]$ is obtained by **multiplying every element** of $[a]$ by α .

Addition and Subtraction of Matrices

Matrices can only be added or subtracted if they are both of the **same size**.

This is done by adding / subtracting the **corresponding entries** of both matrices.

The resulting matrix $[c]$ is the same size where:

- $[c_{ij}] = [a_{ij}] + [b_{ij}]$
- $[c_{ij}] = [a_{ij}] - [b_{ij}]$

Transpose of a Matrix

The transpose of a matrix is a matrix with the **rows and columns interchanged**.


$$\begin{bmatrix} 2 & -1 & 0 \\ 5 & 3 & 1 \\ 6 & 1 & -4 \\ 7 & -2 & 9 \end{bmatrix}^T = \begin{bmatrix} 2 & 5 & 6 & 7 \\ -1 & 3 & 1 & -2 \\ 0 & 1 & -4 & 9 \end{bmatrix}$$

$$[a]^T = [a_{ij}^T] = [a_{ji}]$$

Multiplication of Matrices

The multiplication $[c] = [a][b]$ is defined only if the numbers of columns of $[a]$ equals the number of rows of matrix $[b]$.

The resulting matrix $[c]$ has the same number of rows as $[a]$ and the same number of columns as $[b]$.

$$[c]_{mn} = [a]_{mq} [b]_{qn}$$


If $[a]$ is $m \times q$ and $[b]$ is $q \times n$ then $[c]$ is $m \times n$.

$$c_{ij} = \sum_{k=1}^q a_{ik} b_{kj}$$

For example:

$\begin{aligned} c_{11} &= a_{11} b_{11} + a_{12} b_{21} + a_{13} b_{31} + a_{14} b_{41} \\ c_{12} &= a_{11} b_{12} + a_{12} b_{22} + a_{13} b_{32} + a_{14} b_{42} \\ c_{21} &= a_{21} b_{11} + a_{22} b_{21} + a_{23} b_{31} + a_{24} b_{41} \end{aligned}$	$\begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \\ c_{31} & c_{32} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \\ b_{41} & b_{42} \end{bmatrix}$
---	--

Special Matrices

<p>Square Matrix: Has the same number of columns as rows.</p> <p>Entries a_{ii} are known as the diagonal elements, and all others are the off-diagonal elements.</p>	$[A] = \begin{bmatrix} 4 & 5 & -3 \\ 8 & 1 & 2 \\ 4 & 6 & 3 \end{bmatrix}$
<p>Diagonal Matrix: $[D]$ A square matrix with diagonal elements that are non-zero, and off-diagonal elements that are zero.</p>	$[D] = \begin{bmatrix} 6 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 7 \end{bmatrix}$
<p>Upper-Triangular Matrix: $[U]$ A square matrix whose sub-diagonal entries are all zero.</p>	$\begin{bmatrix} 1 & -3 & 1 \\ 0 & 5 & 12 \\ 0 & 0 & -7 \end{bmatrix}$
<p>Lower-Triangular Matrix: $[L]$ A square matrix whose super-diagonal entries are all zero.</p>	$\begin{bmatrix} 1 & 0 & 0 \\ 5 & 5 & 0 \\ 9 & -7 & -7 \end{bmatrix}$
<p>Identity Matrix: $[I]$ A square matrix whose diagonal elements are all ones, and off-diagonal entries are all zero.</p> <p>Any matrix multiplied by the identity matrix remains unchanged. $[a][I] = [a]$</p>	$[I] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
<p>Zero Matrix: A matrix whose entries are all zero.</p> <p>Any matrix multiplied by a zero matrix will yield a zero matrix. $[z][a] = [z]$</p>	$[z] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
<p>Symmetric Matrix: A square matrix whose lower-diagonal entries mirror its upper-diagonal entries. $[a_{ij}] = [a_{ji}]$</p> <p>The transpose of a symmetric matrix is the matrix itself. $[a]^T = [a]$</p>	$\begin{bmatrix} 3 & 4 & 8 \\ 4 & 1 & 2 \\ 8 & 2 & -7 \end{bmatrix}$

Inverse of a Matrix

A matrix is invertible (its multiplicative inverse can be found) if there exists a square matrix $[b]$ of the same size such that $[a][b] = [I]$.

$$[a][a]^{-1} = [a]^{-1}[a] = [I]$$

$$[a]^{-1} = [b]$$

Determinant of a Matrix

This is a useful quantity to determine if a matrix has an inverse.

$$\det(A) = |A| = \sum_j (-1)^k a_{1j_1} a_{2j_2} \dots a_{nj_n}$$

Example:

$$A = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

$$|A| = a(ei - fh) - b(di - fg) + c(dh - eg)$$

Cramer's Rule and Solution to a System of Linear Equations

Given a set of n simultaneous equations with n unknowns:

$$\begin{aligned}a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n &= b_1 \\a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n &= b_2 \\&\dots + \dots + \dots + \dots = \dots \\a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n &= b_n\end{aligned}$$

The system of equations can be written compactly using matrices:

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{bmatrix}$$

i.e. $[a][x] = [b]$

Cramer's Rule states that the solution for $[x]$, if it exists, is given by:

$$x_j = \frac{\det(a'_j)}{\det(a)} \text{ for } j = 1, 2, \dots, n$$

where a'_j is the matrix formed by replacing the j^{th} column of the matrix $[a]$ with the column vector $[b]$.

Note: $[a]^{-1}$ exists iff $\det(a) \neq 0$

$\det(a) = 0$ if one or more columns or rows of $[a]$ are **not** linearly independent.

Example:

Find the solution of the following system of equations using Cramer's rule.

$$\begin{aligned}2x + 3y - z &= 5 \\4x + 4y - 3z &= 3 \\-2x + 3y - z &= 1\end{aligned}\tag{2.45}$$

SOLUTION

Step 1: Write the system of equations in a matrix form $[a][x] = [b]$.

$$\begin{bmatrix} 2 & 3 & -1 \\ 4 & 4 & -3 \\ -2 & 3 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 5 \\ 3 \\ 1 \end{bmatrix}\tag{2.46}$$

Step 2: Calculate the determinant of the matrix of coefficients.

$$\begin{aligned}\det(A) &= 2[(4 \times -1) - (-3 \times 3)] - 3[(4 \times -1) - (-3 \times -2)] - 1[(4 \times 3) - (4 \times -2)] \\ &= 2(5) - 3(-10) - 1(20) = 10 + 30 - 20 = 20\end{aligned}$$

Step 3: Apply Eq. (2.44) to find x , y , and z . To find x , the modified matrix a'_x is created by replacing its first column with $[b]$.

$$x = \frac{\det \begin{pmatrix} \begin{bmatrix} 5 & 3 & -1 \\ 3 & 4 & -3 \\ 1 & 3 & -1 \end{bmatrix} \end{pmatrix}}{20} = \frac{(5 \cdot 5) - (3 \cdot 0) - (1 \cdot 5)}{20} = 1$$

In the same way, to find y , the modified matrix a'_y is created by replacing its second column with $[b]$.

$$y = \frac{\det \begin{pmatrix} \begin{bmatrix} 2 & 5 & -1 \\ 4 & 3 & -3 \\ -2 & 1 & -1 \end{bmatrix} \end{pmatrix}}{20} = \frac{(20 \cdot 0) - (5 \cdot -10) - (1 \cdot 10)}{20} = 2$$

Finally, to determine the value of z , the modified matrix a'_z is created by replacing its third column with $[b]$.

$$z = \frac{\det \begin{pmatrix} \begin{bmatrix} 2 & 3 & 5 \\ 4 & 4 & 3 \\ -2 & 3 & 1 \end{bmatrix} \end{pmatrix}}{20} = \frac{(2 \cdot -5) - (3 \cdot 10) - (5 \cdot 20)}{20} = 3$$

To check the answer, the matrix of coefficients $[a]$ is multiplied by the solution:

$$\begin{bmatrix} 2 & 3 & -1 \\ 4 & 4 & -3 \\ -2 & 3 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 2 + 6 - 3 \\ 4 + 8 - 9 \\ -2 + 6 - 3 \end{bmatrix} = \begin{bmatrix} 5 \\ 3 \\ 1 \end{bmatrix}$$

The right-hand side is equal to $[b]$, which confirms that the solution is correct.

Ordinary Differential Equations (ODEs)

An ordinary differential equation is one that contains:

1. **One dependent variable** (e.g. y).
2. **One independent variable** (e.g. x).
3. **Ordinary derivatives** of the dependent variable (as opposed to partial).

Examples:

$$\frac{dy}{dx} = 10x$$

$$c \frac{dx}{dt} + kx = -m \frac{d^2x}{dt^2}$$

An ordinary differential equation is **nonlinear** if any of the **coefficients** are **functions of the dependent variable**.

Examples:

$$\frac{d^2y}{dt^2} + \sin y = 4$$

$$y \frac{d^2y}{dt^2} + 3y = 8$$

An ODE is said to be **homogenous** if the **coefficient of the independent variables is zero**. Otherwise, it is said to be nonhomogeneous.

Examples:

$$\frac{dy}{dx} = 10x \text{ is a linear, nonhomogeneous ODE.}$$

$$\frac{d^2y}{dt^2} + \sin y = 4 \text{ is a nonlinear, homogeneous ODE.}$$

The **order** of an ODE is the **highest derivative** that appears in the equation.

Example:

$$\frac{d^2y}{dt^2} + \sin y = 4 \text{ is a second order ODE.}$$

Boundary Conditions & Initial Conditions

To eliminate integration constraints when solving a differential equation we apply **constraints**.

These are referred to as **boundary conditions** (solutions to the DE at particular points) and in the case of **time-dependent DEs**, **initial conditions** (or the solution to the DE at $t = 0$).

Examples:

Nonhomogeneous Linear First-Order ODE:

$$\frac{dy}{dx} + P(x)y = Q(x)$$

Multiply both sides of the equation by the following integrating factor:

$$u(x) = \int_e P(x)dx$$

This gives:

$$\frac{d}{dx}(y\mu) = Q(x)\mu(x)$$

Integrating both sides gives:

$$y(x)\mu(x) = \int Q(x)\mu(x) + C_1$$

Dividing through by $\mu(x)$ gives:

$$y(x) = \frac{1}{\mu(x)} \int Q(x)\mu(x) + \frac{C_1}{\mu(x)}$$

The constant of integration is determined from a constraint which is problem-dependent.

Homogeneous Linear Second-Order ODE:

Consider:

$$\frac{d^2y}{dx^2} + b\frac{dy}{dx} + cy = 0 \text{ where } a \text{ and } b \text{ are constants.}$$

The general solution to this equation is found by substituting $y = e^{sx}$.

The resulting equation is called the **characteristic equation**:

$$s^2 + bs + c = 0$$

The solution is obtained from the quadratic formula:

$$s = \frac{-b \pm \sqrt{b^2 - 4c}}{2}$$

The general solution is then:

$$y(x) = e^{-bx/2} \left[C_1 e^{\frac{\sqrt{b^2 - 4c}}{2}x} + C_2 e^{-\frac{\sqrt{b^2 - 4c}}{2}x} \right]$$

If $b^2 < 4ac$ we get:

$$y(x) = e^{-bx/2} \left[C_1 e^{\frac{i\sqrt{4ac - b^2}}{2}x} + C_2 e^{-\frac{i\sqrt{4ac - b^2}}{2}x} \right]$$

Functions of Two or More Independent Variables

Consider the function:

$$z = f(x, y) = \frac{x^2}{4} + \frac{y^2}{9}$$

This is a function of two variables where z is the dependent variable and x and y are the independent variables.

The Partial Derivative

For $z = f(x, y)$, the first partial derivative of f with respect to x is denoted by $\frac{\partial f}{\partial x}$ or f_x and is defined by:

$$\frac{\partial f}{\partial x} = \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x, y) - f(x, y)}{\Delta x}$$

if the limit exists.

Also:

$$\frac{\partial f}{\partial y} = \lim_{\Delta y \rightarrow 0} \frac{f(x, y+\Delta y) - f(x, y)}{\Delta y}$$

The partial derivative with respect to a certain variable (say x) simply by treating (or holding) all other independent variables (say y and z) to constants.

Example:

$$z = f(x, y) = x^2 + xy + y^2$$

$$\frac{\partial z}{\partial x} = 2x + y$$

$$f_y(x) = x^2 + xy + y^2$$

$$f_a(x) = x^2 + xa + a^2$$

The **total differential** (or exact differential) of a function of two variables, say $f(x, y)$ is given by:

$$df = \frac{\partial f}{\partial x} dx + \frac{\partial f}{\partial y} dy$$

Using the formula for the total differential we can state:

$$\frac{df}{dt} = \frac{\partial f}{\partial x} \frac{dx}{dt} + \frac{\partial f}{\partial y} \frac{dy}{dt}$$

where x and y are both functions of the independent variable t .

Consider $f(x, y)$ where $y = g(x)$ then:

$$\frac{df}{dx} = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} \frac{\partial y}{\partial x}$$

Consider $f(x, y, z)$ where $z = h(x, y)$ then:

$$\left. \frac{\partial f}{\partial x} \right|_y = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial z} \frac{\partial z}{\partial x} \Big|_y$$

or

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial z} \frac{\partial z}{\partial x}$$

The Jacobian

The Jacobian is a quantity that arises when solving **systems of nonlinear equations**.

Given $f_1(x,y) = a$ and $f_2(x,y) = b$ (where a and b are constants) then the **Jacobian matrix** is:

$$[J] = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix}$$

The **Jacobian determinant** or simply the **Jacobian** is:

$$\det \begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{pmatrix}$$

$$= \left(\frac{\partial f_1}{\partial x} \right) \left(\frac{\partial f_2}{\partial y} \right) - \left(\frac{\partial f_1}{\partial y} \right) \left(\frac{\partial f_2}{\partial x} \right)$$

In general:

$$J(f_1, f_2, \dots, f_n) = \det \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \dots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}$$

Taylor Series Expansion of Functions

Used to represent a function as an **infinite power series**.

This can be useful when trying to **integrate** certain non-analytic functions since a truncated series may result in a good analytic **approximation** of the function.

It may also be used in approximating functions to make them more easily manipulated for certain purposes such as **increasing computational speed**.

Given a function $f(x)$ that is differentiable $n + 1$ times in an interval containing $x = x_0$, Taylor's theorem states that $\exists \xi \in [x, x_0]$ such that:

$$f(x) = f(x_0) + (x - x_0) \frac{df}{dx} \Big|_{x=x_0} + \frac{(x-x_0)^2}{2!} \frac{d^2f}{dx^2} \Big|_{x=x_0} + \frac{(x-x_0)^3}{3!} \frac{d^3f}{dx^3} \Big|_{x=x_0} + \dots + \frac{(x-x_0)^n}{n!} \frac{d^n f}{dx^n} \Big|_{x=x_0} + R_n(x)$$

where $R_n(x)$ called the **remainder** is given by:

$$R_n = \frac{(x-x_0)^{n+1}}{(n+1)!} \frac{d^{n+1}f}{dx^{n+1}} \Big|_{x=\xi}$$

The value of the remainder R_n cannot actually be calculated since the value of ξ is not known.

For $n = 1$, Taylor's theorem reduces to:

$$f(x) = f(x_0) + (x - x_0) \frac{df}{dx} \Big|_{x=\xi}$$

or

$$\frac{df}{dx} \Big|_{x=\xi} = \frac{f(x) - f(x_0)}{(x - x_0)}$$

which is a statement of the **mean value theorem for derivatives**.

Example:

Approximate the function $y = \sin(x)$ by using Taylor series expansion about $x = 0$, using two, four, and six terms.

(a) In each case, calculate the approximate value of the function at $x = \frac{\pi}{12}$, and at $x = \frac{\pi}{2}$.

(b) Using MATLAB, plot the function and the three approximations for $0 \leq x \leq \pi$.

SOLUTION

The first five derivatives of the function $y = \sin(x)$ are:

$$y' = \cos(x), \quad y'' = -\sin(x), \quad y^{(3)} = -\cos(x), \quad y^{(4)} = \sin(x), \quad \text{and} \quad y^{(5)} = \cos(x)$$

At $x = 0$, the values of these derivatives are:

$$y' = 1, \quad y'' = 0, \quad y^{(3)} = -1, \quad y^{(4)} = 0, \quad \text{and} \quad y^{(5)} = 1$$

Substituting this information and $y(0) = \sin(0) = 0$ in Eq. (2.74) gives:

$$y(x) = 0 + x + 0 - \frac{x^3}{3!} + 0 + \frac{x^5}{5!} \quad (2.77)$$

(a) For $x = \frac{\pi}{12}$, the exact value of the function is $y = \sin\left(\frac{\pi}{12}\right) = \frac{1}{4}(\sqrt{6} + \sqrt{2}) = 0.2588190451$

The approximate values using two, four, and six terms of the Taylor series expansion are:

Using two terms in Eq. (2.77) gives: $y(x) = x = \frac{\pi}{12} = 0.2617993878$

Using four terms in Eq. (2.77) gives: $y(x) = x - \frac{x^3}{3!} = \frac{\pi}{12} - \frac{(\pi/12)^3}{3!} = 0.2588088133$

Using six terms in Eq. (2.77) gives: $y(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} = \frac{\pi}{12} - \frac{(\pi/12)^3}{3!} + \frac{(\pi/12)^5}{5!} = 0.2588190618$

Matlab:

```
% Generate a linearly-spaced vector between 0 and pi with 100 elements
x = linspace(0, pi, 100);
y = sin(x); % Function we are approximating with Taylor series

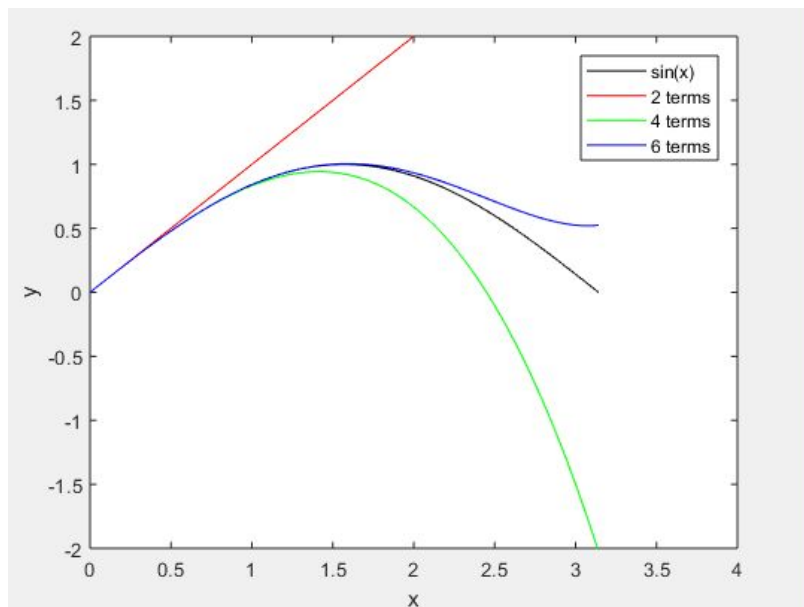
% Terms of Taylor series
y2 = x; % First two terms.
y4 = y2 - (x.^3)/factorial(3); % .^ is element-wise power
y6 = y4 + (x.^5)/factorial(5); % First six terms.

% Plot graph
plot(x, y, 'k', ... % black line
      x, y2, 'r', ... % ... allows multi-line functions
      x, y4, 'g', ...
      x, y6, 'b');

axis([0, 4, -2, 2]); % 0<=x<=4, -2<=y<=2

legend('sin(x)', ... % line 1
       '2 terms', ... % line 2
       '4 terms', ...
       '6 terms');

xlabel('x');
ylabel('y');
```



Taylor Series for a Function of Two Variables

Taylor's expansion for a function of two variables is done the same way as for a function of one independent variable, except that the differentiation involves **partial derivatives**.

$$\begin{aligned} f(x, y) = & f(x_0, y_0) + \frac{1}{1!} \left[(x - x_0) \frac{\partial f}{\partial x} \Big|_{x_0, y_0} + (y - y_0) \frac{\partial f}{\partial y} \Big|_{x_0, y_0} \right] + \\ & \frac{1}{2!} \left[(x - x_0)^2 \frac{\partial^2 f}{\partial x^2} \Big|_{x_0, y_0} + 2(x - x_0)(y - y_0) \frac{\partial^2 f}{\partial x \partial y} \Big|_{x_0, y_0} + (y - y_0)^2 \frac{\partial^2 f}{\partial y^2} \Big|_{x_0, y_0} \right] + \\ & + \dots + \frac{1}{n!} \left[\sum_{k=0}^n \frac{n!}{k!(n-k)!} (x - x_0)^k (y - y_0)^{n-k} \frac{\partial^n f}{\partial x^k \partial y^{n-k}} \Big|_{x_0, y_0} \right] \end{aligned}$$

Inner Product & Orthogonality

The **inner product** (scalar product or dot product) of two vectors V^{\rightarrow} and U^{\rightarrow} is denoted:

$$\langle V^{\rightarrow} | U^{\rightarrow} \rangle \text{ or } \langle V^{\rightarrow}, U^{\rightarrow} \rangle \text{ or } V^{\rightarrow} \cdot U^{\rightarrow}$$

Say $V^{\rightarrow} = V_x \hat{i} + V_y \hat{j} + V_z \hat{k}$ and $U^{\rightarrow} = U_x \hat{i} + U_y \hat{j} + U_z \hat{k}$

Then:

$$\langle V^{\rightarrow} | U^{\rightarrow} \rangle = V^{\rightarrow} \cdot U^{\rightarrow} = |V^{\rightarrow}| |U^{\rightarrow}| \cos \theta = V_x U_x + V_y U_y + V_z U_z$$

The vectors are said to be **orthogonal** to each other if:

$$\langle V^{\rightarrow} | U^{\rightarrow} \rangle = 0$$

and **parallel** if:

$$\langle V^{\rightarrow} | U^{\rightarrow} \rangle = 1$$

The inner product of two functions $f(x)$ and $g(x)$ over an interval $[a, b]$ is given by:

$$\langle f(x) | g(x) \rangle = \int_a^b f(x)g(x)dx$$

Again, they are said to be orthogonal if:

$$\langle f(x) | g(x) \rangle = 0$$

The **sine** and **cosine** functions are orthogonal at all frequencies:

$$\langle \sin(kx) | \cos(mx) \rangle = \int_{-\pi}^{\pi} \sin(kx)\cos(mx)dx = 0 \text{ for both } k = m \text{ and } k \neq m$$

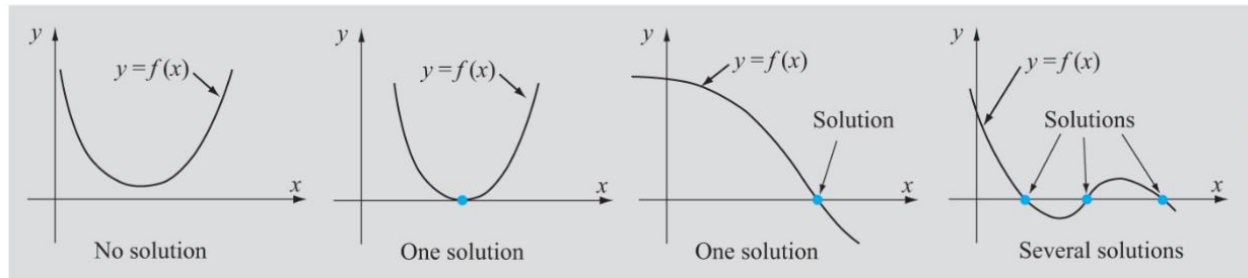
03 Solving Nonlinear Equations

An equation of one variable can be written in the form:

$$f(x) = 0$$

A solution to the equation (a root of the equation) is a numerical value of x that satisfies the equation.

Graphically, this is the point (or points) where the function $f(x)$ intersects the x -axis.



Sometimes the solutions can be obtained analytically (such as using the formula for solving a quadratic equation), but sometimes there is **no analytic solution** and the equation must be solved **numerically**.

A numerical solution of an equation $f(x) = 0$ is a value of x that satisfies the equation **approximately**.

When x is substituted in the equation, the value of $f(x)$ is **close** to zero, but not exactly.

We will always have an error in our approximation. This error **must be quantified** as it tells us the range in which the exact solution lies.

Estimation of Errors in Numerical Solutions

Let:

1. x_{TS} be the true (exact) solution such that $f(x_{TS}) = 0$
2. x_{NS} be the numerical solution such that $f(x_{NS}) = \varepsilon$ where ε is a small number.

True Error

$$TrueError = x_{TS} - x_{NS}$$

In general x_{TS} is **unknown** so we need to use other measures.

Tolerance (in $f(x)$)

$$ToleranceInf = |f(x_{TS}) - f(x_{NS})| = |0 - \varepsilon| = |\varepsilon|$$

If it is known that the solution is in the domain $[a, b]$ then:

$$x_{NS} = \frac{a+b}{2}$$

plus or minus a tolerance of:

$$\left| \frac{b-a}{2} \right|$$

True Relative Error

$$TrueRelativeError = \left| \frac{x_{TS} - x_{NS}}{x_{TS}} \right|$$

but since x_{TS} is unknown we use the estimated relative error.

Estimated Relative Error

When two numerical estimates for the solution are known (from multiple iterations):

$$EstimatedRelativeError = \left| \frac{x_{NS}^{(n)} - x_{NS}^{(n-1)}}{x_{NS}^{(n)}} \right|$$

When the numerical solution is close to the true solution then:

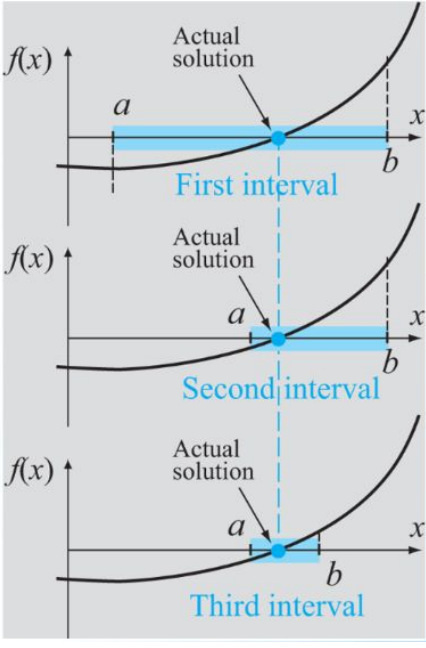
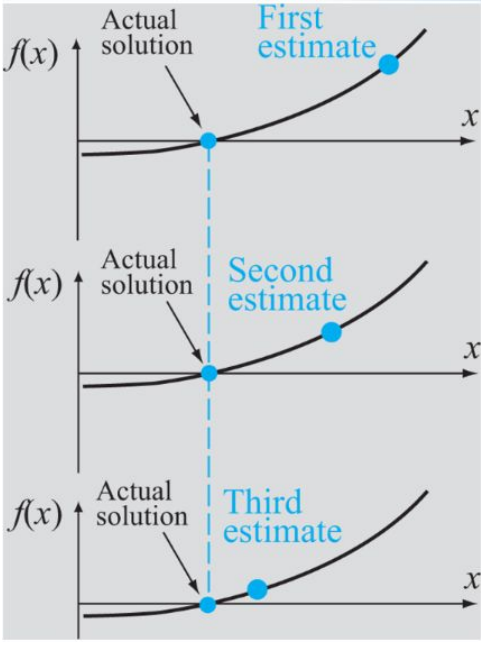
$$x_{NS}^{(n)} - x_{NS}^{(n-1)} \ll x_{NS}^{(n)}$$

i.e. the difference $x_{NS}^{(n)} - x_{NS}^{(n-1)}$ is small compared to the value of $x_{NS}^{(n)}$.

The estimated relative error is approximately the same as the true relative error.

The methods for solving nonlinear equations are divided into:

1. Bracketing methods
2. Open methods

Bracketing Methods	Open Methods
<p>In bracketing methods, the interval containing the solution is identified.</p> <p>The size of the interval is successively reduced until the distance between the endpoints is less than the desired accuracy of the solution.</p>	<p>In open methods, an initial guess is made at the solution before narrowing down the search through further guesswork.</p> <p>Solutions that have a higher error than the current solution are discarded.</p>
Always converge to a solution.	More efficient , but may not yield a solution.
<p>Examples:</p> <ol style="list-style-type: none"> 1. Bisection Method 2. Regula Falsi Method 	<p>Examples:</p> <ol style="list-style-type: none"> 1. Newton's Method 2. Secant Method 3. Fixed-Point Iteration
	

All these methods assume that $f(x)$:

1. Is **continuous**.
2. Is **differentiable** over the appropriate interval.
3. Has a **solution**.

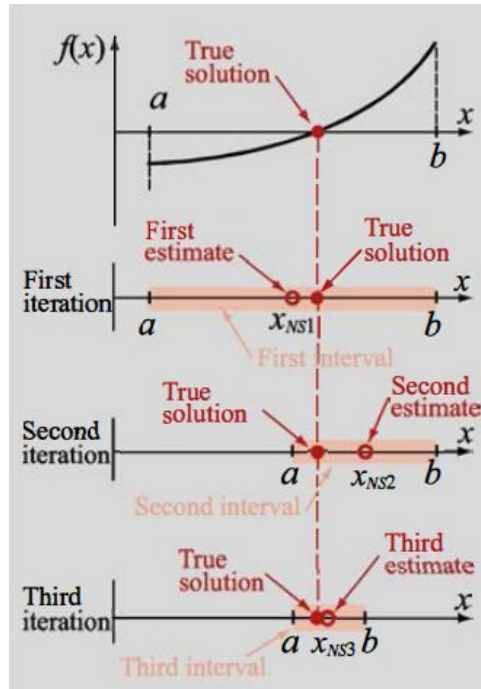
Bracketing Methods

1) The Bisection Method

Used for finding the solution of $f(x) = 0$ when it is known that the solution exists within a given interval $[a, b]$.

Algorithm:

1. Choose the **first interval** by finding points a and b such that a solution exists in this domain.
 $f(a)$ and $f(b)$ have **different signs** so that $f(a)f(b) < 0$.
2. The first estimate of the numerical solution is then:
$$x_{NS1} = \frac{a+b}{2}$$
3. Determine whether the true solution is between (a and x_{NS1}) or (x_{NS1} and b).
This is done by checking the sign of $f(a)f(x_{NS1})$:
 1. If $f(a)f(x_{NS1}) < 0$, then the true solution is between a and x_{NS1} .
 2. If $f(a)f(x_{NS1}) > 0$, then the true solution is between x_{NS1} and b .
4. Select the subinterval that contains the true solution.
Go to step 2.



Matlab:

```
% Define an anonymous function f(x)=8-4.5(x-sin(x))
f = @ (x) 8 - 4.5*(x - sin(x));

% Define initial interval a<=x<=3
a = 2;
b = 3;

maxIterations = 100; % Perform at most 100 iterations.
tolerance = 1e-10; % Terminate after tolerance is reached.

% Make sure that initial interval is valid.
if (f(a)*f(b)>0)
    disp("Error, a and b are invalid bounds.")
    disp(strcat("f(a)=", num2str(f(a))));
    disp(strcat("f(b)=", num2str(f(b))));
end

% Perform Bisection method until either max iterations or tolerance are
% reached.
for i = 0 : maxIterations
    mid = (a+b)/2;

    % Stop iterating if within tolerance.
    if abs(f(mid)) < tolerance
        break;
    end;

    % Calculate new interval.
    sign = f(a)*f(mid);
    if (sign < 0)
        b = mid;
    else
        a = mid;
    end
end

% Print results of x, f(x) and number of iterations.
disp(strcat("x=", num2str(mid)));
disp(strcat("f(x)=", num2str(f(mid))));
disp(strcat("iterations=", num2str(i)));
```

2) The Regula Falsi Method

Approximates the solution by calculating the intercept of the line containing $f(a)$ and $f(b)$ with the interval $[a, b]$ where the solution is known to exist.

The equation of the line that connects $(a, f(a))$ and $(b, f(b))$ is:

$$y = \frac{f(b)-f(a)}{b-a}(x-a) + f(a) \quad (y = mx + c)$$

Its intercept with the x -axis is determined by subbing $y = 0$ into the previous formula and solving the equation for x :

$$x_{NS} = \frac{af(b)-bf(a)}{f(b)-f(a)}$$

Algorithm:

1. Choose the **first interval** by finding points a and b such that a solution exists in this domain.

$f(a)$ and $f(b)$ have **different signs** so that $f(a)f(b) < 0$.

2. The first estimate of the numerical solution is then:

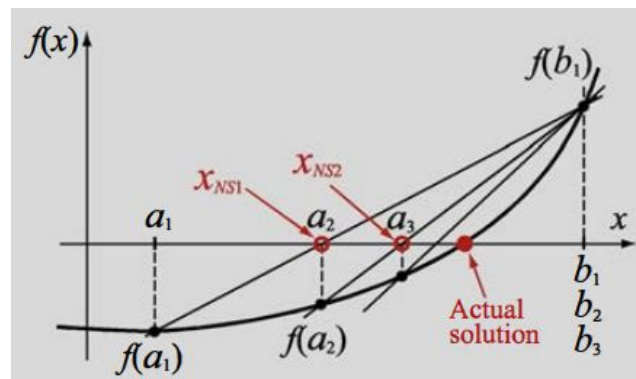
$$x_{NS1} = \frac{af(b)-bf(a)}{f(b)-f(a)}$$

3. Determine whether the true solution is between $(a$ and $x_{NS1})$ or $(x_{NS1}$ and $b)$.

This is done by checking the sign of $f(a)f(x_{NS1})$:

- a. If $f(a)f(x_{NS1}) < 0$, then the true solution is between a and x_{NS1} .
 - b. If $f(a)f(x_{NS1}) > 0$, then the true solution is between x_{NS1} and b .
4. Select the subinterval that contains the true solution.

Go to step 2.



Open Methods

1) Newton's Method (Newton-Raphson Method)

Approximates the solution initially as the intercept of the tangent to the function, at an initial guess-point, with the x -axis.

Subsequent iterations approximate the solution as the intercept of the tangent to the function, at the point defined by the previous estimate, with the x -axis.

This algorithm does **not** necessarily converge.

Algorithm:

1. Choose $(x_1, f(x_1))$ as a starting point near the solution.

2. For the first iteration, the slope of the tangent at $(x_1, f(x_1))$ is given by:

$$f'(x_1) = \frac{f(x_1) - 0}{x_1 - x_2} \quad (\text{rise over run, slope between } (x_1, f(x_1)) \text{ and } (x_2, 0))$$

Solving for x_2 gives:

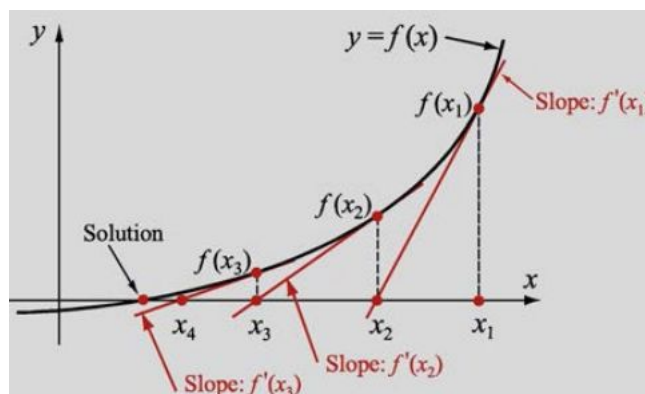
$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

3. Generalising the previous formula, we can **repeat step 3**:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

Newton's method may not converge, in particular if $f(x) \approx 0$ in the vicinity of the solution.

If $f(x)$ and $f'(x)$ are continuous, $f(x) \neq 0$ at the solution and if the starting value $(x_1, f(x_1))$ is near the solution, then the method converges.



Matlab:

```
% Define an anonymous function f(x)=8-4.5(x-sin(x))
f = @ (x) 8 - 4.5*(x - sin(x)); % f(x)
dfdx = @ (x) -4.5 + 4.5*cos(x); % f'(x)

% Define starting point (x, f(x))
x = 5;

maxIterations = 100; % Perform at most 100 iterations.
tolerance = 1e-10; % Terminate after tolerance is reached.

% Perform Newton's method until either max iterations or tolerance are
% reached.
for i = 0 : maxIterations
    % Calculate numerical solution.
    x = x - f(x) / dfdx(x);

    % Stop iterating if within tolerance.
    if abs(f(x)) < tolerance
        break;
    end;
end

% Print results of x, f(x) and number of iterations.
disp(strcat("x=", num2str(x)));
disp(strcat("f(x)", num2str(f(x))));
disp(strcat("iterations=", num2str(i)));
```


2) The Secant Method

Similar to Newton's Method, however we use a **secant** to the function instead of a tangent. This means we initially need **two points** near the solution.

Algorithm:

1. Choose $(x_1, f(x_1))$, $(x_2, f(x_2))$ as starting points near the solution.
2. For the first iteration, the slope of the secant line is given by:

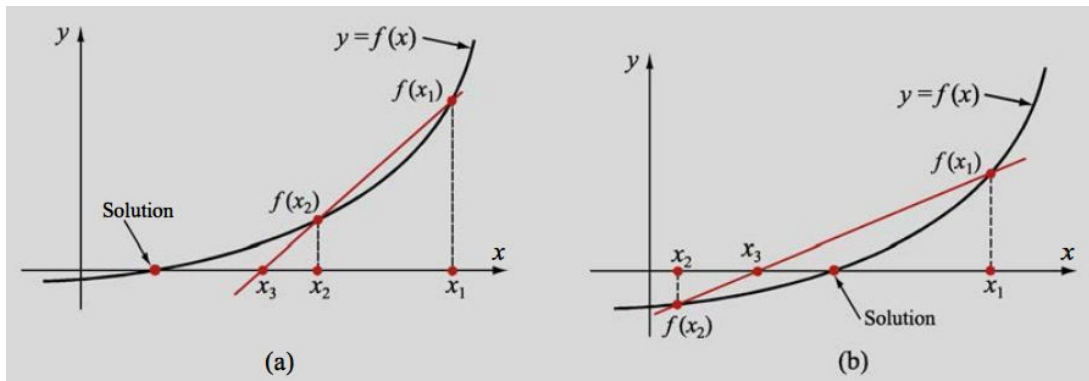
$$\frac{f(x_1) - f(x_2)}{x_1 - x_2} = \frac{f(x_2) - 0}{x_2 - x_3}$$

Solving for x_3 gives:

$$x_3 = x_2 - \frac{f(x_2)(x_1 - x_2)}{f(x_1) - f(x_2)}$$

3. Generalising the previous formula, we can **repeat step 3**:

$$x_{i+1} = x_i - \frac{f(x_i)(x_{i-1} - x_i)}{f(x_{i-1}) - f(x_i)}$$



Matlab:

```
% Define an anonymous function f(x)=8-4.5(x-sin(x))
f = @(x) 8 - 4.5*(x - sin(x)); % f(x)

% Define starting points (x1, f(x1)), (x2, f(x2))
x1 = 6;
x2 = 5;

maxIterations = 100; % Perform at most 100 iterations.
tolerance = 1e-10; % Terminate after tolerance is reached.

% Perform Secant method until either max iterations or tolerance are
% reached.
for i = 0 : maxIterations
    % Calculate numerical solution.
    new = x2 - (f(x2)*(x1-x2)) / (f(x1)-f(x2));
    x1 = x2;
    x2 = new;

    % Stop iterating if within tolerance.
    if abs(f(x2)) < tolerance
        break;
    end;
end

% Print results of x, f(x) and number of iterations.
disp(strcat("x2=", num2str(x2)));
disp(strcat("f(x2)", num2str(f(x2))));
disp(strcat("iterations=", num2str(i)));
```

3) The Fixed-Point Iteration Method

Used for solving an equation of the form:

$$f(x) = 0$$

This function is written in the form:

$$x = g(x).$$

This equation holds true at the **intersection** of the line $y = x$ and the function $y = g(x)$.

This intersection is called the fixed point.

The problem boils down to obtaining the x -coordinate of this point of intersection.

Algorithm:

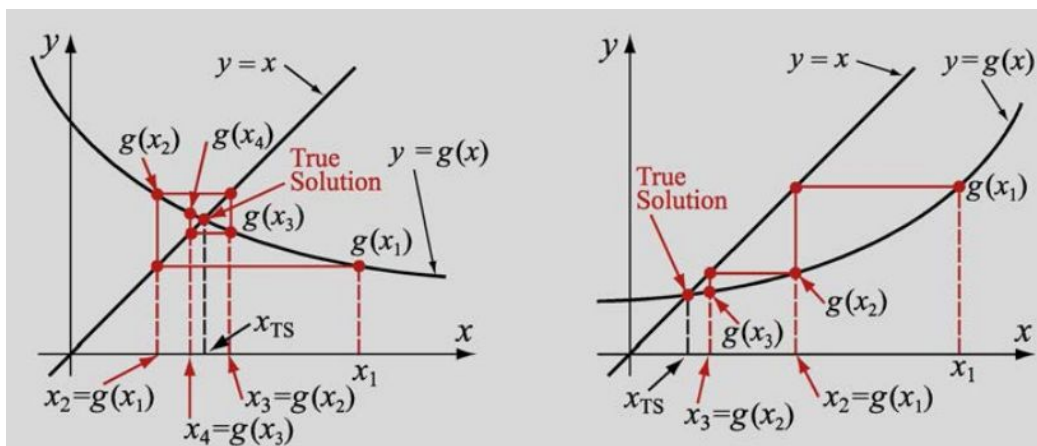
1. Take a value of x near the solution (the solution is the point of intersection between the line $y = x$ and the function $y = g(x)$).

This is our first estimate of the solution.

2. Find $g(x)$.

3. Use this y -ordinate to obtain an x -ordinate using the line $y = x$.

Go to step 2.



This method is prone to diverge depending on the manner in which $x = g(x)$ is expressed.

If the function is slowly varying such that in the neighbourhood of the solution:

$$|g'(x)| < 1$$

then the process will converge to the correct solution.

Example:

Considering $xe^{0.5x} + 1.2x - 5 = 0$

We can write $x = g(x)$ a number of different ways:

1. $x = \frac{5 - xe^{0.5x}}{1.2}$
2. $x = \frac{5}{xe^{0.5x} + 1.2}$
3. $x = \frac{5 - 1.2x}{e^{0.5x}}$

Only when expressed as the second option do we get:

$$|g'(x)| < 1; \quad x = 1, 2$$

Matlab:

```
% f(x) = 8-4.5(x-sin(x)) = 0
% rewrite into form x = g(x)

% Define an anonymous function g(x)
g = @(x) sin(x) + 8/4.5;

% Define starting x value
x = 5;

maxIterations = 100; % Perform at most 100 iterations.
tolerance = 1e-10; % Terminate after tolerance is reached.

% Perform Secant method until either max iterations or tolerance are
% reached.
for i = 0 : maxIterations
    % Calculate numerical solution.
    y = g(x);
    x = y;

    % Stop iterating if within tolerance.
    if abs(g(x) - x) < tolerance
        break;
    end;
end

% Print results of x, g(x) and number of iterations.
disp(strcat("x=", num2str(x)));
disp(strcat("g(x)", num2str(g(x))));
disp(strcat("iterations=", num2str(i)));
```

Systems of Nonlinear Equations

A system of nonlinear equations consists of **two or more nonlinear equations** that have to be solved simultaneously.

1) Newton's Method (For Systems of Nonlinear Equations)

A system of two equations with two unknowns x and y can be written as:

$$f_1(x, y) = 0$$

$$f_2(x, y) = 0$$

The solution process begins by choosing an **estimated solution** x_1 and y_1 .

If x_2 and y_2 are the true (unknown) solutions to the system and are sufficiently close to x_1 and y_1 , then the values of f_1 and f_2 at x_2 and y_2 can be expressed using a **Taylor series expansion** of the functions $f_1(x, y) = 0$ and $f_2(x, y) = 0$ about (x_1, y_1) .

$$f_1(x_2, y_2) = f_1(x_1, y_1) + (x_2 - x_1) \frac{\partial f_1}{\partial x} \Big|_{x_1, y_1} + (y_2 - y_1) \frac{\partial f_1}{\partial y} \Big|_{x_1, y_1} + \dots$$

$$f_2(x_2, y_2) = f_2(x_1, y_1) + (x_2 - x_1) \frac{\partial f_2}{\partial x} \Big|_{x_1, y_1} + (y_2 - y_1) \frac{\partial f_2}{\partial y} \Big|_{x_1, y_1} + \dots$$

Since x_2 and y_2 are close to x_1 and y_1 , approximate values for $f_1(x_2, y_2)$ and $f_2(x_2, y_2)$ can be calculated by **neglecting the higher-order terms**.

Since $f_1(x_2, y_2) = 0$ and $f_2(x_2, y_2) = 0$, our Taylor-expanded equations can be rewritten as:

$$\frac{\partial f_1}{\partial x} \Big|_{x_1, y_1} \Delta x + \frac{\partial f_1}{\partial y} \Big|_{x_1, y_1} \Delta y = -f_1(x_1, y_1)$$

$$\frac{\partial f_2}{\partial x} \Big|_{x_1, y_1} \Delta x + \frac{\partial f_2}{\partial y} \Big|_{x_1, y_1} \Delta y = -f_2(x_1, y_1)$$

where $\Delta x = x_2 - x_1$ and $\Delta y = y_2 - y_1$.

The system can be solved using Cramer's rule:

$$\Delta x = \frac{-f_1(x_1, y_1) \frac{\partial f_2}{\partial y} \Big|_{x_1, y_1} + f_2(x_1, y_1) \frac{\partial f_1}{\partial y} \Big|_{x_1, y_1}}{J(f_1(x_1, y_1), f_2(x_1, y_1))}$$

$$\Delta y = \frac{-f_2(x_1, y_1) \frac{\partial f_2}{\partial x} \Big|_{x_1, y_1} + f_1(x_1, y_1) \frac{\partial f_1}{\partial x} \Big|_{x_1, y_1}}{J(f_1(x_1, y_1), f_2(x_1, y_1))}$$

where:

$$J(f_1, f_2) = \det \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix}$$

$J(f_1, f_2)$ is the Jacobian. Once Δx and Δy are known, the values of x_2 and y_2 are calculated by:

$$x_2 = x_1 + \Delta x$$

$$y_2 = y_1 + \Delta y$$

Newton's method can be generalised for a system of n nonlinear equations which have the form:

$$f_1(x_1, x_2, \dots, x_n) = 0$$

$$f_2(x_1, x_2, \dots, x_n) = 0$$

...

$$f_n(x_1, x_2, \dots, x_n) = 0$$

which gives:

$$\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \dots \\ \Delta x_n \end{bmatrix} = \begin{bmatrix} -f_1 \\ -f_2 \\ \dots \\ -f_n \end{bmatrix} \quad (3.55)$$

Algorithm:

1. Estimate an initial solution:

$$x_{1,i}, x_{2,i}, x_{3,i}, \dots$$

2. Solve equation 3.55 for $\Delta x_1, \Delta x_2, \Delta x_3, \dots$
3. Calculate a new estimate of the solution using:

$$x_{1,i+1} = x_{1,i} + \Delta x_1$$

$$x_{2,i+1} = x_{2,i} + \Delta x_2$$

...

$$x_{n,i+1} = x_{n,i} + \Delta x_n$$

4. Repeat until the desired accuracy is reached.

Newton's method does not necessarily converge, but when it does it does so quickly. The Jacobian must be non-zero and the initial estimates must be close to the solution.

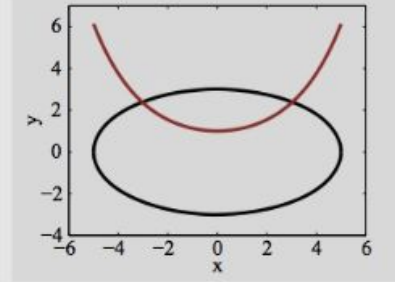
Example 3-5: Solution of a system of nonlinear equations using Newton's method.

The equations of the catenary curve and the ellipse, which are shown in the figure, are given by:

$$f_1(x, y) = y - \frac{1}{2}(e^{x/2} + e^{(-x)/2}) = 0 \quad (3.49)$$

$$f_2(x, y) = 9x^2 + 25y^2 - 225 = 0 \quad (3.50)$$

Use Newton's method to determine the point of intersection of the curves that resides in the first quadrant of the coordinate system.

**SOLUTION**

Equations (3.49) and (3.50) are a system of two nonlinear equations. The points of intersection are given by the solution of the system. The solution with Newton's method is obtained by using Eqs. (3.43) and (3.44). In the present problem, the partial derivatives in the equations are given by:

$$\frac{\partial f_1}{\partial x} = -\frac{1}{4}(e^{x/2} - e^{(-x)/2}) \quad \text{and} \quad \frac{\partial f_1}{\partial y} = 1 \quad (3.51)$$

$$\frac{\partial f_2}{\partial x} = 18x \quad \text{and} \quad \frac{\partial f_2}{\partial y} = 50y \quad (3.52)$$

The Jacobian is given by:

$$J(f_1, f_2) = \det \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix} = \det \begin{bmatrix} -\frac{1}{4}(e^{x/2} - e^{(-x)/2}) & 1 \\ 18x & 50y \end{bmatrix} = -\frac{1}{4}(e^{x/2} - e^{(-x)/2})50y - 18x \quad (3.53)$$

Substituting Eqs. (3.51)–(3.53) in Eqs. (3.45) and (3.46) gives the solution for Δx and Δy .

The problem is solved in the MATLAB program that is listed below. The order of operations in the program is:

- The solution is initiated by the initial guess, $x_i = 2.5$, $y_i = 2.0$.
- The iterations start. Δy and Δx are determined by substituting x_i and y_i in Eqs. (3.45) and (3.46).
- $x_{i+1} = x_i + \Delta x$, and $y_{i+1} = y_i + \Delta y$ are determined.
- If the estimated relative error (Eq. (3.9)) for both variables is smaller than 0.001, the iterations stop. Otherwise, the values of x_{i+1} and y_{i+1} are assigned to x_i and y_i , respectively, and the next iteration starts.

The program also displays the solution and the error at each iteration.


```

% Solution of Chapter 3 Example 5
F1 = @ (x,y) y - 0.5*(exp(x/2) + exp(-x/2));
F2 = @ (x,y) 9*x^2 + 25*y^2 - 225;
F1x = @ (x) -(exp(x/2) - exp(-x/2))/4;
F2x = @ (x) 18*x;
F2y = @ (y) 50*y;
Jacob = @ (x,y) -(exp(x/2) - exp(-x/2))/4*50*y - 18*x;
xi = 2.5; yi = 2; Err = 0.001;
for i = 1:5
    Jac = Jacob(xi,yi);
    Delx = (-F1(xi,yi)*F2y(yi) + F2(xi,yi))/Jac;
    Dely = (-F2(xi,yi)*F1x(xi) + F1(xi,yi)*F2x(xi))/Jac;
    xipl = xi + Delx;
    yipl = yi + Dely;
    Errx = abs((xipl - xi)/xi);
    Erry = abs((yipl - yi)/yi);
    fprintf('i=%2.0f x=%7.4f y=%7.4f Error in x=%7.4f Error in y=%7.4f\n',i,xipl,yipl,Errx,Erry)
    if Errx < Err & Erry < Err
        break
    else
        xi = xipl; yi = yipl;
    end
end

```

Assign the initial estimate of the solution.

Start the iterations.

Calculate Δx and Δy with Eqs. (3.45) and (3.46).

Calculate x_{i+1} and y_{i+1} .

If the error is not small enough, assign x_{i+1} to x_i , and y_{i+1} to y_i .

2) Fixed-Point Method (For Systems of Nonlinear Equations)

Algorithm:

1. We begin with:

$$f_1(x_1, x_2, \dots, x_n) = 0$$

$$f_2(x_1, x_2, \dots, x_n) = 0$$

...

$$f_n(x_1, x_2, \dots, x_n) = 0$$

2. This system can be rewritten in the form:

$$x_1 = g_1(x_1, x_2, \dots, x_n)$$

$$x_2 = g_2(x_1, x_2, \dots, x_n)$$

...

$$x_n = g_n(x_1, x_2, \dots, x_n)$$

3. We then choose an initial estimate for the solution $x_{1,1}, x_{2,1}, x_{3,1} \dots$ which is substituted into the RHS of our rewritten system.
The LHS is our new second estimate of the solution.
4. We continue this process until a desired accuracy is reached.

The fixed-point iteration method will converge under the following (but not necessary) conditions:

1. If the functions g and their derivatives are continuous in the neighbourhood of the solution.
2. If $\sum_i \left| \frac{\partial g_i}{\partial x_i} \right| \leq 1$
3. The initial estimate is close to the solution.

04 Solving a System of Linear Equations

The general form of a system of n linear algebraic equations is:

$$\begin{aligned}a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n &= b_1 \\a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n &= b_2 \\&\dots + \dots + \dots + \dots = \dots \\a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n &= b_n\end{aligned}$$

Which can be written as:

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{bmatrix}$$

Methods for Solving Systems of Linear Equations

These equations can be solved with **analytic**, **direct** or **iterative** methods. However, analytic methods such as Kramer's rule become impossible to compute when the number of equations is large.

Direct methods	Iterative methods
The solution is calculated using an efficient numerical algorithm which performs arithmetic operations on equations.	An initial approximate solution is assumed and is then used in an iterative process to obtain successively more-accurate results.

Direct Methods

In direct methods, the initial system of equations is manipulated into an **equivalent** system of equations that can be easily solved.

Three systems of linear equations that can be **easily solved** are:

1. Upper triangular form
2. Lower triangular form
3. Diagonal form

Upper-Triangular Form

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

This equation can be easily solved by **back-substitution**:

1. We start with the last equation and solve for x_n .
2. Knowing x_n , we can then solve for x_{n-1} and so on.

The general form for the solution using back-substitution is:

$$x_n = \frac{b_n}{a_{nn}}$$
$$x_i = \frac{b_i - \sum_{j=i+1}^n a_{ij}x_j}{a_{ii}} \quad i = n-1, n-2, \dots, 1$$

Lower-Triangular Form

Likewise, a lower-triangular form can be solved using forward substitution.

$$\begin{bmatrix} a_{11} & 0 & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ a_{31} & a_{32} & a_{33} & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

Gaussian Elimination

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

Here, the full set of linear equations is manipulated to be in **upper-triangular form**, which can then be solved using back-substitution.

Algorithm:

1. We take the **first equation**, called the **pivot equation**.

a_{11} is called the pivot coefficient or **pivot element**.

The pivot element is always a diagonal element.

To eliminate the term $a_{21}x_1$, we multiply the pivot equation by $m_{21} = \frac{a_{21}}{a_{11}}$.

The resulting equation is subtracted from the second equation thus eliminating x_1 .

$$\begin{array}{r} a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 = b_2 \\ - \\ m_{21}(a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4) = m_{21}b_1 \\ \hline 0 + (a_{22} - m_{21}a_{12})x_2 + (a_{23} - m_{21}a_{13})x_3 + (a_{24} - m_{21}a_{14})x_4 = b_2 - m_{21}b_1 \\ \underbrace{\hspace{1.5cm}}_{a'_{22}} \quad \underbrace{\hspace{1.5cm}}_{a'_{23}} \quad \underbrace{\hspace{1.5cm}}_{a'_{24}} \quad \underbrace{\hspace{1.5cm}}_{b'_2} \end{array}$$

Note that the **pivot equation remains unchanged**.

The second equation now has new coefficients.

The same process is used to eliminate x_1 from all subsequent equations using $m_{i1} = \frac{a_{i1}}{a_{11}}$ for the i^{th} equation giving:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a'_{22} & a'_{23} & a'_{24} \\ 0 & a'_{32} & a'_{33} & a'_{34} \\ 0 & a'_{42} & a'_{43} & a'_{44} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b'_2 \\ b'_3 \\ b'_4 \end{bmatrix}$$

2. The process is now repeated operating on the third and subsequent equations using the second as the pivot equation and a_{22} as the pivot element.
This eliminates x_2 from these equations.

Continuing in this fashion yields an upper-triangular system which can be solved using back-substitution.

Problems with zero or near-zero pivoting elements can be remedied by **changing the order** of the system of equations such that **no zero or near-zero elements appear on the diagonal**. This process is known as **pivoting**.

Gauss-Jordan Elimination

Gauss-Jordan elimination is similar to Gaussian elimination except that:

1. The pivot equation is normalised by **dividing all the terms** in the pivot equation by the **pivot coefficient**. This makes the pivot coefficient equal to 1.
2. The pivot equation is used to eliminate the off-diagonal terms in **all equations**.

The result is a **diagonal matrix** that can be easily solved.
Pivoting applies as with Gaussian elimination.

LU Decomposition

In this method, we begin with the equation to be solved:

$$[a][x] = [b] \text{ where } [x] \text{ and } [b] \text{ are vectors.}$$

$$\text{Then } [a] = [L][U]$$

where $[L]$ and $[U]$ are lower and upper triangular matrices respectively.

We then write:

$$[L][U][x] = [b]$$

We then say:

$$[U][x] = [y]$$

and

$$[L][y] = [b]$$

$[L][y] = [b]$ is then first solved for $[y]$ thus enabling $[U][x] = [y]$ to be solved for $[x]$.

The advantage of this method is that $[L][U][x] = [b]$ can be easily solved by back and forward substitution.

LU Decomposition Using The Gaussian Elimination Procedure

When the Gaussian Elimination procedure is applied to a matrix $[a]$, the elements of the matrices $[L]$ and $[U]$ are actually calculated.

The upper-triangular matrix $[U]$ is obtained at the **end** of the Gaussian Elimination procedure.

The lower-triangular matrix $[L]$ is a matrix with 1s as its diagonal elements and the multipliers m_{ij} that multiply the pivot equation as its off-diagonal elements.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ m_{21} & 1 & 0 & 0 \\ m_{31} & m_{32} & 1 & 0 \\ m_{41} & m_{42} & m_{43} & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a'_{22} & a'_{23} & a'_{24} \\ 0 & 0 & a''_{33} & a''_{34} \\ 0 & 0 & 0 & a'''_{44} \end{bmatrix}$$

LU Decomposition Using Crout's Method

Crout's method is particularly advantageous since we do not have to use vector $[b]$ at all, which means that we can solve for **multiple** $[b]$ s once $[L]$ and $[U]$ have been determined.

In Crout's method, the LU decomposition matrix has the form:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} = \begin{bmatrix} L_{11} & 0 & 0 & 0 \\ L_{21} & L_{22} & 0 & 0 \\ L_{31} & L_{32} & L_{33} & 0 \\ L_{41} & L_{42} & L_{43} & L_{44} \end{bmatrix} \begin{bmatrix} 1 & U_{12} & U_{13} & U_{14} \\ 0 & 1 & U_{23} & U_{24} \\ 0 & 0 & 1 & U_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Executing the matrix multiplication on the RHS gives:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} = \begin{bmatrix} L_{11} & (L_{11}U_{12}) & (L_{11}U_{13}) & (L_{11}U_{14}) \\ L_{21} & (L_{21}U_{12}+L_{22}) & (L_{21}U_{13}+L_{22}U_{23}) & (L_{21}U_{14}+L_{22}U_{24}) \\ L_{31} & (L_{31}U_{12}+L_{32}) & (L_{31}U_{13}+L_{32}U_{23}+L_{33}) & (L_{31}U_{14}+L_{32}U_{24}+L_{33}U_{34}) \\ L_{41} & (L_{41}U_{12}+L_{42}) & (L_{41}U_{13}+L_{42}U_{23}+L_{43}) & (L_{41}U_{14}+L_{42}U_{24}+L_{43}U_{34}+L_{44}) \end{bmatrix}$$

This equation is then solved by **equating** the elements of the product with that of matrix $[a]$.

In general:

1. Calculate the **first column** of $[L]$ using:

$$\text{for } i = 1, 2, \dots, n \quad L_{i1} = a_{i1}$$

2. Substitute 1s in the **diagonal** of $[U]$:

$$\text{for } i = 1, 2, \dots, n \quad U_{ii} = 1$$

3. Calculate the elements of the **first row** of $[U]$ **except** for U_{11} :

$$\text{for } j = 2, \dots, n \quad U_{1j} = \frac{a_{1j}}{L_{11}}$$

4. Then, calculate:

$$\text{for } j = 2, \dots, i \quad L_{ij} = a_{ij} - \sum_{k=1}^{j-1} L_{ik}U_{kj}$$

$$\text{for } j = (i+1), (i+2), \dots, n \quad U_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} L_{ik}U_{kj}}{L_{ii}}$$

Inverse of a Matrix

This is the most advantageous since it can easily be reused.

Consider:

$$[a][a]^{-1} = [a]^{-1}[a] = [I]$$

For $[x] = [a]^{-1}$ we have:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

We then have:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{21} \\ x_{31} \\ x_{41} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_{12} \\ x_{22} \\ x_{32} \\ x_{42} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_{13} \\ x_{23} \\ x_{33} \\ x_{43} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} x_{14} \\ x_{24} \\ x_{34} \\ x_{44} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Solving these four systems of equations gives the four columns of $[a]^{-1}$.

Among other methods, this can be done by **Gauss-Jordan elimination** and by **LU decomposition**.


Inverse of a Matrix by LU Decomposition & Gauss-Jordan Elimination

We obtain each column of the inverse matrix by solving the system of equations.

This can be done by **Gauss-Jordan elimination** or by **LU decomposition** on each of the four equations.

Iterative Methods

For the Jacobi and Gauss-Seidel iterative methods, the system of equations is written in **explicit form**:

$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 &= b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 &= b_3 \\ a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 &= b_4 \end{aligned}$ <p style="text-align: center;">(a)</p>	<p>Writing the equations in an explicit form.</p> 	$\begin{aligned} x_1 &= [b_1 - (a_{12}x_2 + a_{13}x_3 + a_{14}x_4)]/a_{11} \\ x_2 &= [b_2 - (a_{21}x_1 + a_{23}x_3 + a_{24}x_4)]/a_{22} \\ x_3 &= [b_3 - (a_{31}x_1 + a_{32}x_2 + a_{34}x_4)]/a_{33} \\ x_4 &= [b_4 - (a_{41}x_1 + a_{42}x_2 + a_{43}x_3)]/a_{44} \end{aligned}$ <p style="text-align: center;">(b)</p>
---	---	---

The idea is to **assume an initial solution** for $[x]$, and then **recursively substitute** these values in to the RHS of the above explicit form and so obtain new estimates for $[x]$.

$$x_i = \frac{1}{a_{ii}} \left[b_i - \sum_{j=1, j \neq i}^{j=n} a_{ij}x_j \right] \quad i = 1, 2, \dots, n \quad (4.51)$$

A sufficient (but not necessary) **condition for convergence** is:

$$|a_{ii}| > \sum_{j=1, j \neq i}^{j=n} |a_{ij}|$$

The Jacobi Iterative Method

An **initial solution** for $[x]$ is chosen. If no information is available then $[x] = 0$.

The values for $[x]$ are substituted into the RHS of the explicit representation of the system of equations and a new estimate for $[x]$ is found. This process is repeated recursively.


That is:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left[b_i - \sum_{j=1, j \neq i}^{j=n} a_{ij}x_j^{(k)} \right] \quad i = 1, 2, \dots, n$$

The process is continued until the absolute value of the estimated relative error of all unknowns is less than some predetermined value:

$$\left| \frac{x_i^{(k+1)} - x_i^{(k)}}{x_i^{(k)}} \right| < \varepsilon \quad i = 1, 2, \dots, n \quad (4.55)$$

The Gauss-Seidel Iterative Method

$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 &= b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 &= b_3 \\ a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 &= b_4 \end{aligned}$ <p style="text-align: center;">(a)</p>	<p>Writing the equations in an explicit form.</p> 	$\begin{aligned} x_1 &= [b_1 - (a_{12}x_2 + a_{13}x_3 + a_{14}x_4)]/a_{11} \\ x_2 &= [b_2 - (a_{21}x_1 + a_{23}x_3 + a_{24}x_4)]/a_{22} \\ x_3 &= [b_3 - (a_{31}x_1 + a_{32}x_2 + a_{34}x_4)]/a_{33} \\ x_4 &= [b_4 - (a_{41}x_1 + a_{42}x_2 + a_{43}x_3)]/a_{44} \end{aligned}$ <p style="text-align: center;">(b)</p>
---	---	---

Algorithm:

1. First iteration:

To begin with, all unknowns x_j except for x_1 are initialised with some approximate values. These can be zero if approximate values are unknown.

$$x_i = \frac{1}{a_{ii}} \left[b_i - \sum_{j=1, j \neq i}^{j=n} a_{ij}x_j \right] \quad i = 1, 2, \dots, n \quad \text{(4.51 again)}$$

Using these values, with $i = 1$ in equation (4.51), we calculate x_1 .

With $i = 2$ in equation (4.51), we calculate a new value for x_2 and so on.

2. Second iteration:

The next iteration uses $i = 1$ in equation (4.51), where we calculate a new value for x_1 .

With $i = 2$ in equation (4.51), we calculate a new value for x_2 and so on.

These iterations are repeated until:

$$\left| \frac{x_i^{(k+1)} - x_i^{(k)}}{x_i^{(k)}} \right| < \varepsilon \quad i = 1, 2, \dots, n \quad \text{(4.55 again)}$$

Mathematically, the Gauss-Seidel algorithm can be written:

$$\begin{aligned} x_1^{(k+1)} &= \frac{1}{a_{11}} \left[b_1 - \sum_{j=2}^{j=n} a_{1j}x_j^{(k)} \right] \\ x_i^{(k+1)} &= \frac{1}{a_{ii}} \left[b_i - \left(\sum_{j=1}^{j=i-1} a_{ij}x_j^{(k+1)} + \sum_{j=i+1}^{j=n} a_{ij}x_j^{(k)} \right) \right] \quad i = 2, 3, \dots, n-1 \quad (4.56) \\ x_n^{(k+1)} &= \frac{1}{a_{nn}} \left[b_n - \sum_{j=1}^{j=n-1} a_{nj}x_j^{(k+1)} \right] \end{aligned}$$

The Gauss-Seidel algorithm **converges faster** than the Jacobi method.

Like the Jacobi method, the sufficient but not necessary condition for convergence is that $[a]$ is diagonally dominant.

Tridiagonal Systems of Equations

$$\begin{bmatrix} A_{11} & A_{12} & 0 & 0 & 0 & 0 & 0 & 0 \\ A_{21} & A_{22} & A_{23} & 0 & 0 & 0 & 0 & 0 \\ 0 & A_{32} & A_{33} & A_{34} & 0 & 0 & 0 & 0 \\ & & \dots & \dots & \dots & & & \\ & & & \dots & \dots & \dots & & \\ 0 & 0 & 0 & 0 & A_{n-2, n-3} & A_{n-2, n-2} & A_{n-2, n-1} & 0 \\ 0 & 0 & 0 & 0 & 0 & A_{n-1, n-2} & A_{n-1, n-1} & A_{n-1, n} \\ 0 & 0 & 0 & 0 & 0 & 0 & A_{n, n-1} & A_{n, n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ \dots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \\ B_3 \\ \dots \\ \dots \\ B_{n-2} \\ B_{n-1} \\ B_n \end{bmatrix}$$

These can be readily solved using previous methods but many elements of $[a]$ are zero. This means more storage space and many redundant zero multiplications and additions.

The **Thomas algorithm** is a “common sense” approach to the problem. Instead of performing Gaussian elimination on $[a]$ as a whole, we store the **diagonal** and **off-diagonal** entries as **separate vectors**. We then apply **Gaussian elimination** as before.

Error & Residual

Consider the equation $[a][x] = [b]$

The **true error** is the vector $[e]$:

$$[e] = [x_{TS}] - [x_{NS}] \quad (TS = \text{True Solution and } NS = \text{Numerical solution})$$

But because $[x_{TS}]$ cannot be calculated in general, we instead use the **residual error** $[r]$:

$$\begin{aligned} [r] &= [a][x_{TS}] - [a][x_{NS}] = [b] - [a][x_{NS}] \\ \Rightarrow [e] &= [a]^{-1}[r] \end{aligned}$$

The smaller the elements in $[r]$ the better $[x_{NS}]$ satisfies the equation. However, this does not necessarily tell us how close $[x_{NS}]$ is to $[x_{TS}]$.

Norms

The norm of a vector or matrix is a number that in some way indicates collectively the **size** of the vector or matrix **elements**.

There are various definitions for the norm of a vector $[v]$ or matrix $[a]$.

We will concern ourselves with the **Euclidean 2-norm** on this course.

Vector norms:

1. The **infinity norm** (l_∞ norm):

$$\|v\|_\infty = \max_{1 \leq i \leq n} |v_i|$$

2. The **1-norm** (l_1 norm):

$$\|v\|_1 = \sum_{i=1}^n |v_i|$$

3. The **Euclidean 2-norm** (l_2 norm):

$$\|v\|_2 = \left(\sum_{i=1}^n v_i^2 \right)^{1/2}$$

The Euclidean 2-norm is also referred to as the **magnitude of the vector** $[v]$.

Matrix norms:

1. The **infinity norm** (l_∞ norm):

$$\|[a]\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^n |a_{ij}|$$

2. The **1-norm** (l_1 norm):

$$\|[a]\|_1 = \sum_{j=1}^n \max_{1 \leq i \leq n} |a_{ij}|$$

3. The **2-norm** (l_2 norm):

$$\|[a]\|_2 = \max \left(\frac{\|[a][v]\|}{\|[v]\|} \right) \text{ where } [v] \text{ is an eigenvector of } [a].$$

4. The **Euclidean norm**:

$$\|[a]\|_{\text{Euclidean}} = \left(\sum_{i=1}^n \sum_{j=1}^n a_{ij}^2 \right)^{1/2} \quad (\text{i.e. The root of the sum of squares of the elements.})$$

Properties of Norms

1. The norm of a vector or matrix $[a]$, denoted $\|[a]\|$, is a **positive quantity** and is equal to zero iff $[a] = 0$.
2. $\|[a]\| = |a| \|[a]\| \quad \forall a$
This means that $\|[-a]\| = \|[a]\| \geq 0$
3. $\|[a][x]\| = \|[a]\| \|x\|$
4. **The triangle inequality:**
 $\|[a+b]\| \leq \|[a]\| + \|b\|$

Using Norms to Determine Error Bounds

It can be shown that:

$$\frac{1}{\|[a]\| \|[a]^{-1}\|} \leq \frac{\|[r]\|}{\|[b]\|} \leq \frac{\|[e]\|}{\|[x_{TS}]\|} \leq \|[a]^{-1}\| \|[a]\| \frac{\|[r]\|}{\|[b]\|}$$

This useful result gives an upper and lower bound to error relative to the true solution.

The Condition Number of a Matrix

This tells us **how stable the solution** to a system of equations is.

This number shows how a small perturbation in $[a]$ will affect the solution. If the condition number is much **greater** than 1, then a small perturbation in $[a]$ will **greatly** affect the solution. In this case, the matrix $[a]$ is said to be **ill-conditioned**.

By contrast, if a small perturbation in $[a]$ does **not** affect the solution greatly, then $[a]$ is said to be **well-conditioned**.

The condition number of a matrix $[a]$ is given by:

$$\text{Cond}[a] = \|[a]\| \|[a]^{-1}\|$$

For example, take the following system of equations:

$\begin{aligned} 6x_1 - 2x_2 &= 10 \\ 11.5x_1 - 3.85x_2 &= 17 \end{aligned}$
--

The solution of this system is:

$$x_1 = 45, x_2 = 130$$

If we now change a_{22} slightly to 3.84, then:

$$x_1 = 110, x_2 = 325$$

It is clear that $[a]$ is **ill-conditioned**. In fact it has a condition number greater than 1500 regardless of which norm is used.

05 Eigenvalues and Eigenvectors

For a given square matrix $[a]$, the number λ is an **eigenvalue** of the matrix if:

$$[a][u] = \lambda[u] \quad (5.1)$$

The vector $[u]$ is a common vector called the **eigenvector associated with the eigenvalue λ** .

There are usually **more than one** eigenvalue and eigenvector. For an $n \times n$ matrix, there are n eigenvalues and an infinite number of eigenvectors.

In general:

$$Lu = \lambda u \quad (5.2)$$

where L is some mathematical operator.

Equation (5.2) is a general statement of an eigenvalue problem where λ is the eigenvalue associated with the operator L .

For example, consider:

$$\frac{d^2 y}{dx^2} = k^2 y$$

where:

- L is the second derivative with respect to x
- y is the **eigenfunction**
- k^2 is the **eigenvalue** associated with L

We will concern ourselves with **matrix operations** only.

Finding Eigenvalues and Eigenvectors

$$[a][u] = \lambda[u] \quad \textbf{(5.1 again)}$$

Equation (5.1) can be rewritten as:

$$[a - \lambda I][u] = 0$$

If $[a - \lambda I]$ has an inverse, then $[u] = 0$.

If on the other hand $[a - \lambda I]$ has no inverse, then a non-trivial solution for $[u]$ is possible.

Another way of stating this problem is to use **Cramer's rule**.

$[a - \lambda I]$ has no inverse (it is singular) if:

$$\det[a - \lambda I] = 0$$

This equation is known as the **characteristic equation** for $[a]$.

The characteristic equation for $[a]$ yields a polynomial equation in λ whose roots are the eigenvalues.

Once the eigenvalues are known then the eigenvectors $[u]$ can be determined by substituting the eigenvalues one at a time into equation (5.1) and solving for $[u]$.

This process is relatively straightforward for small matrices, but for large matrices we need to use **numerical methods** such as the **power method** or the **QR factorisation method**.

The Basic Power Method

This method is an iterative procedure for determining the **largest real eigenvalue** and **corresponding eigenvector** of a matrix.

Proof:

Let A be an $n \times n$ matrix with eigenvalues $\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_n$ which:

1. Are not necessarily distinct.
2. Satisfy the relations $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_n|$

The eigenvalue λ_1 , which is the **largest in magnitude**, is known as the **dominant eigenvalue** of the matrix A .

Assume that the associated eigenvectors $v_1, v_2, v_3, \dots, v_n$ are **linearly independent**, and therefore form a basis for R^n .

It should be noted that **not all matrices** have eigenvalues and eigenvectors which satisfy the conditions we have assumed here.

Let $x^{(0)}$ be a **nonzero** element of R^n . Since the eigenvectors of A form a basis for R^n , it follows that $x^{(0)}$ can be written as a **linear combination** of $v_1, v_2, v_3, \dots, v_n$.

That is, there exists **constants** $\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_n$ such that:

$$x^{(0)} = \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3 + \dots + \alpha_n v_n$$

Next, construct the sequence of vectors $\{x^{(m)}\}$ according to the rule $x^{(m)} = Ax^{(m-1)}$ for $m \geq 1$.

By **direct calculation** we find that:

$$\begin{aligned} x^{(1)} &= Ax^{(0)} = \alpha_1(Av_1) + \alpha_2(Av_2) + \alpha_3(Av_3) + \dots + \alpha_n(Av_n) \\ &= \alpha_1(\lambda_1 v_1) + \alpha_2(\lambda_2 v_2) + \alpha_3(\lambda_3 v_3) + \dots + \alpha_n(\lambda_n v_n) \end{aligned}$$

$$\begin{aligned} x^{(2)} &= Ax^{(1)} = A^2 x^{(0)} = \alpha_1(A^2 v_1) + \alpha_2(A^2 v_2) + \alpha_3(A^2 v_3) + \dots + \alpha_n(A^2 v_n) \\ &= \alpha_1(\lambda_1^2 v_1) + \alpha_2(\lambda_2^2 v_2) + \alpha_3(\lambda_3^2 v_3) + \dots + \alpha_n(\lambda_n^2 v_n) \end{aligned}$$

and in general:

$$\begin{aligned} x^{(m)} &= Ax^{(m-1)} = \dots = A^m x^{(0)} = \alpha_1(A^m v_1) + \alpha_2(A^m v_2) + \alpha_3(A^m v_3) + \dots + \alpha_n(A^m v_n) \\ &= \alpha_1(\lambda_1^m v_1) + \alpha_2(\lambda_2^m v_2) + \alpha_3(\lambda_3^m v_3) + \dots + \alpha_n(\lambda_n^m v_n) \end{aligned}$$

In deriving these expressions, we have made repeated use of the relation $Av_j = \lambda_j v_j$, which follows from the fact that v_j is an **eigenvector** associated with the **eigenvalue** λ_j .

Factoring λ_1^m from the RHS of the equation for $x^{(m)}$ gives:

$$x^{(m)} = \lambda_1^m \left[\alpha_1 v_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^m v_2 + \alpha_3 \left(\frac{\lambda_3}{\lambda_1}\right)^m v_3 + \dots + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right)^m v_n \right] \quad (1)$$

By assumption, $|\lambda_j/\lambda_1| < 1$ for each j , so $|\lambda_j/\lambda_1|^m \rightarrow 0$ as $m \rightarrow \infty$.

It therefore follows that:

$$\lim_{m \rightarrow \infty} \frac{x^{(m)}}{\lambda_1^m} = \alpha_1 v_1$$

Since any nonzero constant times an eigenvector is still an eigenvector associated with the same eigenvalue, we see that the scaled sequence $\{x^{(m)}/\lambda_1^m\}$ converges to an eigenvector associated with the **dominant eigenvalue** provided that $\alpha_1 \neq 0$.

Furthermore, convergence toward the eigenvector is **linear** with asymptotic error constant $|\lambda_j/\lambda_1|$.

An approximation for the dominant eigenvalue of A can be obtained from the sequence $\{x^{(m)}\}$ as follows:

Let i be an index for which $x_i^{(m-1)} \neq 0$, and consider the ratio of the i^{th} element from the vector $x^{(m)}$ to the i^{th} element from $x^{(m-1)}$.

By equation (1):

$$\frac{x_i^{(m)}}{x_i^{(m-1)}} = \frac{\lambda_1^m \alpha_1 v_{1,i} [1 + O((\lambda_2/\lambda_1)^m)]}{\lambda_1^{m-1} \alpha_1 v_{1,i} [1 + O((\lambda_2/\lambda_1)^{m-1})]} = \lambda_1 [1 + O((\lambda_2/\lambda_1)^{m-1})],$$

provided that $v_{1,i} \neq 0$ where $v_{1,i}$ denotes the i^{th} element from the vector v_1 .

Hence the ratio $x^{(m)}/x^{(m-1)}$ converges towards the **dominant eigenvalue**, and the convergence is **linear** with asymptotic rate constant $|\lambda_j/\lambda_1|$.

The Basic Power Method Algorithm

Algorithm:

1. Choose a column vector $[x]_i$ of length n . The vector can be any non-zero vector.
2. Multiply this vector $[x]_i$ by $[A]$. This gives a column vector $[x]_{i+1}$
3. Normalise this column vector.
4. Go to step 2 with $[x]_{i+1} \rightarrow [x]_i$
5. Continue with steps 2 - 4 until a desired accuracy is reached for both the **eigenvalue** and **eigenvector**.

Example 5-2: Using the power method to determine the largest eigenvalue of a matrix.

Determine the largest eigenvalue of the following matrix:

$$\begin{bmatrix} 4 & 2 & -2 \\ -2 & 8 & 1 \\ 2 & 4 & -4 \end{bmatrix} \quad (5.21)$$

Use the power method and start with the vector $x = [1, 1, 1]^T$.

SOLUTION

Starting with $i = 1$, $x_1 = [1, 1, 1]^T$. With the power method, the vector $[x]_2$ is first calculated by $[x]_2 = [a][x]_1$ (Step 2) and is then normalized (Step 3):

$$[x]_2 = [a][x]_1 = \begin{bmatrix} 4 & 2 & -2 \\ -2 & 8 & 1 \\ 2 & 4 & -4 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 7 \\ 2 \end{bmatrix} = 7 \begin{bmatrix} 0.5714 \\ 1 \\ 0.2857 \end{bmatrix} \quad (5.22)$$

For $i = 2$, the normalized vector $[x]_2$ (without the multiplicative factor) is multiplied by $[a]$. This results in $[x]_3$, which is then normalized:

$$[x]_3 = [a][x]_2 = \begin{bmatrix} 4 & 2 & -2 \\ -2 & 8 & 1 \\ 2 & 4 & -4 \end{bmatrix} \begin{bmatrix} 0.5714 \\ 1 \\ 0.2857 \end{bmatrix} = \begin{bmatrix} 3.7143 \\ 7.1429 \\ 4 \end{bmatrix} = 7.1429 \begin{bmatrix} 0.52 \\ 1 \\ 0.56 \end{bmatrix} \quad (5.23)$$

The next three iterations are:

$$i = 3: \quad [x]_4 = [a][x]_3 = \begin{bmatrix} 4 & 2 & -2 \\ -2 & 8 & 1 \\ 2 & 4 & -4 \end{bmatrix} \begin{bmatrix} 0.52 \\ 1 \\ 0.56 \end{bmatrix} = \begin{bmatrix} 2.96 \\ 7.52 \\ 2.8 \end{bmatrix} = 7.52 \begin{bmatrix} 0.3936 \\ 1 \\ 0.3723 \end{bmatrix} \quad (5.24)$$

$$i = 4: \quad [x]_5 = [a][x]_4 = \begin{bmatrix} 4 & 2 & -2 \\ -2 & 8 & 1 \\ 2 & 4 & -4 \end{bmatrix} \begin{bmatrix} 0.3936 \\ 1 \\ 0.3723 \end{bmatrix} = \begin{bmatrix} 2.8298 \\ 7.5851 \\ 3.2979 \end{bmatrix} = 7.5851 \begin{bmatrix} 0.3731 \\ 1 \\ 0.4348 \end{bmatrix} \quad (5.25)$$

$$i = 5: \quad [x]_6 = [a][x]_5 = \begin{bmatrix} 4 & 2 & -2 \\ -2 & 8 & 1 \\ 2 & 4 & -4 \end{bmatrix} \begin{bmatrix} 0.3731 \\ 1 \\ 0.4348 \end{bmatrix} = \begin{bmatrix} 2.6227 \\ 7.6886 \\ 3.0070 \end{bmatrix} = 7.6886 \begin{bmatrix} 0.3411 \\ 1 \\ 0.3911 \end{bmatrix} \quad (5.26)$$

After three more iterations, the results are:

$$i = 8 \quad [x]_9 = [a][x]_8 = \begin{bmatrix} 4 & 2 & -2 \\ -2 & 8 & 1 \\ 2 & 4 & -4 \end{bmatrix} \begin{bmatrix} 0.3272 \\ 1 \\ 0.3946 \end{bmatrix} = \begin{bmatrix} 2.5197 \\ 7.7401 \\ 3.0760 \end{bmatrix} = 7.7401 \begin{bmatrix} 0.3255 \\ 1 \\ 0.3974 \end{bmatrix} \quad (5.27)$$

The results show that the differences between the vector $[x]_i$ and the normalized vector $[x]_{i+1}$ are getting smaller. The value of the multiplicative factor (7.7401) is an estimate of the largest eigenvalue. As shown in Section 5.5, a value of 7.7504 is obtained for the eigenvalue by MATLAB's built-in function `eig`.

The Inverse Power Method

This method is used for finding the **smallest** eigenvalue by applying the power method to $[a]^{-1}$.

The eigenvalues of $[a]^{-1}$ are the reciprocals of the eigenvalues of $[a]$.

This can be seen as follows:

$$[a][x] = \lambda[x]$$

where λ is the eigenvalue associated with the eigenvector $[x]$

Then:

$$[a]^{-1}[a][x] = [a]^{-1}\lambda[x]$$

Hence:

$$[x] = \lambda[a]^{-1}[x]$$

or

$$[a]^{-1}[x] = \frac{1}{\lambda}[x]$$

This shows that $\frac{1}{\lambda}$ is an eigenvalue of $[a]^{-1}$.

Thus the Inverse Power Method can be used to find the largest eigenvalue of $[a]^{-1}$, which is the smallest eigenvalue of $[a]$.

Algorithm:

1. Choose a column vector $[x]_i$ of length n . The vector can be any non-zero vector.
2. Multiply this vector $[x]_i$ by $[a]^{-1}$. This gives a column vector $[x]_{i+1}$
3. Normalise this column vector and multiply it by $[a]^{-1}$ again:

$$[x]_{i+1} = [a]^{-1}[x]_i$$

To avoid calculating $[a]^{-1}$, we use:

$$[a][x]_{i+1} = [x]_i$$

This can be solved for $[x]_{i+1}$ using, for example, LU Decomposition.

4. Go to step 2 with $[x]_{i+1} \rightarrow [x]_i$
5. Continue with steps 2 - 4 until a desired accuracy is reached for both the **eigenvalue** and **eigenvector**.

The Shifted Power Method

Given $[a][x] = \lambda[x]$, then if λ_1 is the largest (or smallest) eigenvalue obtained using the Power Method (or Inverse Power Method), then the eigenvalues of a new shifted matrix formed by $[a - \lambda I]$ are:

$$\alpha = 0, \lambda_2 - \lambda_1, \lambda_3 - \lambda_1, \dots, \lambda_n - \lambda_1$$

The reason for this is as follows:

$$[a - \lambda I][x] = \alpha[x]$$

where the α s are the eigenvalues of $[a - \lambda I]$.

But also $[a][x] = \lambda[x]$, so:

$$(\lambda - \lambda_1)[x] = \alpha[x]$$

where $\lambda = \lambda_1, \lambda_2, \lambda_3, \dots, \lambda_n$

Algorithm:

1. Determine the **largest eigenvalue** λ_1 of $[a]$ using the Power Method.
2. Determine the **largest eigenvalue** α_2 of the shifted matrix $[a - \lambda I]$ using the power method.
Then $\alpha_2 = \lambda_2 - \lambda_1$ from which λ_2 can be determined.
3. Continue in this fashion until **all the eigenvalues** of $[a]$ are determined.
This is done in a total of n steps where $[a]$ is an $n \times n$ matrix.

The Shifted Power Method is an **inefficient process**. A preferred method for obtaining all eigenvalues of a matrix is the QR Factorisation Method.

The QR Factorisation Method

Some definitions:

Two square matrices $[a]$ and $[b]$ are **similar** if:

$$[a] = [c]^{-1} [b] [c]$$

where $[c]$ is an **invertible matrix**.

$[a]$ and $[b]$ have the **same eigenvalues**.

A matrix is **orthogonal** if:

$$[a]^{-1} = [a]^T$$

A matrix is **symmetric** if:

$$[a] = [a]^T$$

The **eigenvalues** of an **upper-triangular matrix** are its **diagonal elements**.

QR Factorisation is used to find **all the eigenvalues** of a matrix.

We begin with the matrix $[a]_1$, whose eigenvalues are to be determined.

Iteration 1:

Let $[a]_1 = [Q]_1 [R]_1$

where:

- $[Q]_1$ is an **orthogonal** matrix
- $[R]_1$ is an **upper-triangular** matrix

We now define $[a]_2 = [R]_1 [Q]_1$

But $[R]_1 = [Q]_1^{-1} [a]_1 = [Q]_1^T [a]_1$

So we can write:

$$[a]_2 = [Q]_1^T [a]_1 [Q]_1$$

This means that $[a]_1$ and $[a]_2$ are similar, thus having the **same eigenvalues**.

Iteration 2:

We now write $[a]_2 = [Q]_2[R]_2$

Then define $[a]_3 = [R]_2[Q]_2$ where $[R]_2 = [Q]_2^{-1}[a]_2 = [Q]_2^T[a]_2$ so that $[a]_3 = [Q]_2^T[a]_2[Q]_2$ as in the first iteration.

These iterations continue until the sequence of matrices $[a]_{n-1}$ generates results in an **upper-triangular matrix** whose eigenvalues are its **diagonal elements** which are the same as the eigenvalues of $[a]_1$.

We now obtain $[Q]_k$ and $[R]_k$ ($[Q]$ and $[R]$ for the k^{th} iteration).

To do this, we use the **Householder matrix** that has the form:

$$[H] = [I] - \frac{2}{[v]^T[v]} [v][v]^T$$

where $[I]$ is the $(n \times n)$ identity matrix and $[v]$ is an n -element column vector given by:

$$[v] = [c] + \|c\|_2 [e]$$

where:

$$\|c\|_2 = \sqrt{c_1^2 + c_2^2 + \dots + c_n^2}$$

Note that $[v]^T[v]$ is a **scalar**, but $[v][v]^T$ is an $(n \times n)$ matrix.

The **Householder matrix** is both **symmetric** and **orthogonal**. This means that:

$$[H] = [H]^T = [H]^{-1}$$

Hence $[a] = [H][b][H]$ yields a matrix $[b]$ that is similar to $[a]$.

QR Factorisation

Factoring the $(n \times n)$ matrix $[a]$ into an orthogonal matrix $[Q]$ and an upper-triangular matrix $[R]$ such that $[a] = [Q][R]$ is done in $n - 1$ steps.

Step 1:

We choose $[c]$ to be the first column of $[a]$:

$$[c] = \begin{bmatrix} a_{11} \\ a_{21} \\ \dots \\ a_{n1} \end{bmatrix}$$

The vector $[e]$ is defined as:

$$[e] = \begin{bmatrix} \pm 1 \\ 0 \\ 0 \\ \dots \\ 0 \end{bmatrix}$$

The first element of $[e]$ is $+1$ if the first element of $[c]$ is positive. Otherwise, it is negative.

Now we have enough information to calculate $[H]^{-1}$.

The matrix $[a]$ is factored into $[Q]^{(1)}[R]^{(1)}$ where:

$$[Q]^{(1)} = [H]^{(1)} \text{ and } [R]^{(1)} = [H]^{(1)} [a]$$

The matrix $[Q]^{(1)}$ is orthogonal because $[H]^{(1)}$ is orthogonal.

$[R]^{(1)}$ is a matrix with zeroes as the elements in the first column below the diagonal.

$$\begin{bmatrix} R_{11}^{(1)} & R_{12}^{(1)} & R_{13}^{(1)} & R_{14}^{(1)} & R_{15}^{(1)} \\ 0 & R_{22}^{(1)} & R_{23}^{(1)} & R_{24}^{(1)} & R_{25}^{(1)} \\ 0 & R_{32}^{(1)} & R_{33}^{(1)} & R_{34}^{(1)} & R_{35}^{(1)} \\ 0 & R_{42}^{(1)} & R_{43}^{(1)} & R_{44}^{(1)} & R_{45}^{(1)} \\ 0 & R_{52}^{(1)} & R_{53}^{(1)} & R_{54}^{(1)} & R_{55}^{(1)} \end{bmatrix}$$

Step 2:

The vector $[c]$ is defined as the second column of the $[R]^{(1)}$ matrix with its first element set to zero:

$$[c] = \begin{bmatrix} 0 \\ R_{22}^{(1)} \\ R_{32}^{(1)} \\ \dots \\ R_{n2}^{(1)} \end{bmatrix}$$

The vector $[e]$ is:

$$[e] = \begin{bmatrix} 0 \\ \pm 1 \\ 0 \\ \dots \\ 0 \end{bmatrix}$$

The second element of $[e]$ is $+1$ if the second element of $[c]$ is positive. Otherwise, it is negative.

Like before in *step 1*, a new Householder matrix $[H]^{(2)}$ is constructed.

We factor the matrix $[a]$ into $[Q]^{(2)}[R]^{(2)}$ where:

$$[Q]^{(2)} = [Q]^{(1)}[H]^{(2)} \text{ and } [R]^{(2)} = [H]^{(2)}[R]^{(1)}$$

The matrix $[Q]^{(2)}$ is orthogonal because $[H]^{(2)}$ is orthogonal.

$[R]^{(2)}$ is a matrix of zeros as the elements in the second column below the diagonal.

$$\begin{bmatrix} R_{11}^{(2)} & R_{12}^{(2)} & R_{13}^{(2)} & R_{14}^{(2)} & R_{15}^{(2)} \\ 0 & R_{22}^{(2)} & R_{23}^{(2)} & R_{24}^{(2)} & R_{25}^{(2)} \\ 0 & 0 & R_{33}^{(2)} & R_{34}^{(2)} & R_{35}^{(2)} \\ 0 & 0 & R_{43}^{(2)} & R_{44}^{(2)} & R_{45}^{(2)} \\ 0 & 0 & R_{53}^{(2)} & R_{54}^{(2)} & R_{55}^{(2)} \end{bmatrix}$$

Step 3:

Moving to the third column of $[a]$, the vectors $[c]$ and $[e]$ are defined as:

$$[c] = \begin{bmatrix} 0 \\ 0 \\ R_{33}^{(2)} \\ R_{34}^{(2)} \\ \dots \\ R_{n3}^{(2)} \end{bmatrix}$$

$$[e] = \begin{bmatrix} 0 \\ 0 \\ \pm 1 \\ 0 \\ \dots \\ 0 \end{bmatrix}$$

where the sign rule applies as before.

$[R]^{(3)}$ is then:

$$\begin{bmatrix} R_{11}^{(3)} & R_{12}^{(3)} & R_{13}^{(3)} & R_{14}^{(3)} & R_{15}^{(3)} \\ 0 & R_{22}^{(3)} & R_{23}^{(3)} & R_{24}^{(3)} & R_{25}^{(3)} \\ 0 & 0 & R_{33}^{(3)} & R_{34}^{(3)} & R_{35}^{(3)} \\ 0 & 0 & 0 & R_{44}^{(3)} & R_{45}^{(3)} \\ 0 & 0 & 0 & R_{54}^{(3)} & R_{55}^{(3)} \end{bmatrix}$$

We continue in this latter fashion until $[R]^{(n-1)}$ is **upper-triangular**.

Then we have:

$$[a] = [Q]^{n-1} [R]^{n-1}$$

This completes the factorisation of $[a]$.

Algorithm:

1. Factor $[a]_1$ into an orthogonal matrix $[Q]_1$ and an upper triangular matrix $[R]_1$.

This is done in $n - 1$ steps using a Householder matrix such that:

$$[a]_1 = [Q]_1 [R]_1$$

2. Calculate:

$$[a]_2 = [R]_1 [Q]_1$$

3. Repeat steps 1 and 2 until $[a]_{n-1}$ is reached.

At this point, $[a]_{n-1}$ is upper-triangular and so its eigenvalues (which are the same as those of $[a]$) appear on its diagonal.

Example 5-3: QR factorization of a matrix.

Factor the following matrix $[a]$ into an orthogonal matrix $[Q]$ and an upper triangular matrix $[R]$:

$$[a] = \begin{bmatrix} 6 & -7 & 2 \\ 4 & -5 & 2 \\ 1 & -1 & 1 \end{bmatrix} \quad (5.56)$$

SOLUTION

The solution follows the steps listed in pages 176–179. Since the matrix $[a]$ is (3×3) , the factorization requires only two steps.

Step 1: The vector $[c]$ is defined as the first column of the matrix $[a]$:

$$[c] = \begin{bmatrix} 6 \\ 4 \\ 1 \end{bmatrix}$$

The vector $[e]$ is defined as the following three-element column vector:

$$[e] = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Using Eq. (5.40), the Euclidean norm, $\|c\|_2$, of $[c]$ is:

$$\|c\|_2 = \sqrt{c_1^2 + c_2^2 + c_3^2} = \sqrt{6^2 + 4^2 + 1^2} = 7.2801$$

Using Eq. (5.39), the vector $[v]$ is:

$$[v] = [c] + \|c\|_2[e] = \begin{bmatrix} 6 \\ 4 \\ 1 \end{bmatrix} + 7.2801 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 13.2801 \\ 4 \\ 1 \end{bmatrix}$$

Next, the products $[v]^T[v]$ and $[v][v]^T$ are calculated:

$$[v]^T[v] = \begin{bmatrix} 13.2801 & 4 & 1 \end{bmatrix} \begin{bmatrix} 13.2801 \\ 4 \\ 1 \end{bmatrix} = 193.3611$$

$$[v][v]^T = \begin{bmatrix} 13.2801 \\ 4 \\ 1 \end{bmatrix} \begin{bmatrix} 13.2801 & 4 & 1 \end{bmatrix} = \begin{bmatrix} 176.3611 & 53.1204 & 13.2801 \\ 53.1204 & 16 & 4 \\ 13.2801 & 4 & 1 \end{bmatrix}$$

The Householder matrix $[H]^{(1)}$ is then:

$$[H]^{(1)} = [I] - \frac{2}{[v]^T[v]}[v][v]^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{2}{193.3611} \begin{bmatrix} 176.3611 & 53.1204 & 13.2801 \\ 53.1204 & 16 & 4 \\ 13.2801 & 4 & 1 \end{bmatrix} = \begin{bmatrix} -0.8242 & -0.5494 & -0.1374 \\ -0.5494 & 0.8345 & -0.0414 \\ -0.1374 & -0.0414 & 0.9897 \end{bmatrix}$$

Once the Householder matrix $[H]^{(1)}$ is constructed, $[a]$ can be factored into $[Q]^{(1)}[R]^{(1)}$, where:

$$[Q]^{(1)} = [H]^{(1)} = \begin{bmatrix} -0.8242 & -0.5494 & -0.1374 \\ -0.5494 & 0.8345 & -0.0414 \\ -0.1374 & -0.0414 & 0.9897 \end{bmatrix}$$

and

$$[R]^{(1)} = [H]^{(1)}[a] = \begin{bmatrix} -0.8242 & -0.5494 & -0.1374 \\ -0.5494 & 0.8345 & -0.0414 \\ -0.1374 & -0.0414 & 0.9897 \end{bmatrix} \begin{bmatrix} 6 & -7 & 2 \\ 4 & -5 & 2 \\ 1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} -7.2801 & 8.6537 & -2.8846 \\ 0 & -0.2851 & 0.5288 \\ 0 & 0.1787 & 0.6322 \end{bmatrix}$$

This completes the first step.

Step 2: The vector $[c]$, which has three elements, is now defined as:

$$[c] = \begin{bmatrix} 0 \\ R_{22}^{(1)} \\ R_{32}^{(1)} \end{bmatrix} = \begin{bmatrix} 0 \\ -0.2851 \\ 0.1787 \end{bmatrix}$$

The vector $[e]$ is defined as the following three-element column vector:

$$[e] = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix}$$

Using Eq. (5.40), the Euclidean norm, $\|c\|_2$, of $[c]$ is:

$$\|c\|_2 = \sqrt{c_1^2 + c_2^2 + c_3^2} = \sqrt{0^2 + (-0.2851)^2 + 0.1787^2} = 0.3365$$

Using Eq. (5.39), the vector $[v]$ is:

$$[v] = [c] + \|c\|_2 [e] = \begin{bmatrix} 0 \\ -0.2851 \\ 0.1787 \end{bmatrix} + 0.3365 \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ -0.6215 \\ 0.1787 \end{bmatrix}$$

Next, the products $[v]^T[v]$ and $[v][v]^T$ are calculated:

$$[v]^T[v] = \begin{bmatrix} 0 & -0.6215 & 0.1787 \end{bmatrix} \begin{bmatrix} 0 \\ -0.6215 \\ 0.1787 \end{bmatrix} = 0.4183$$

$$[v][v]^T = \begin{bmatrix} 0 \\ -0.6215 \\ 0.1787 \end{bmatrix} \begin{bmatrix} 0 & -0.6215 & 0.1787 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.3864 & -0.1111 \\ 0 & 0.1111 & 0.0319 \end{bmatrix}$$

The Householder matrix $[H]^{(2)}$ is then:

$$[H]^{(2)} = [I] - \frac{2}{[v]^T[v]} [v][v]^T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{2}{0.4183} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.3864 & -0.1111 \\ 0 & 0.1111 & 0.0319 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -0.8474 & 0.5311 \\ 0 & 0.5311 & 0.8473 \end{bmatrix}$$

Once the Householder matrix $[H]^{(2)}$ is constructed, $[a]$ can be factored into $[Q]^{(2)}[R]^{(2)}$, where:

$$[Q]^{(2)} = [Q]^{(1)}[H]^{(2)} = \begin{bmatrix} -0.8242 & -0.5494 & -0.1374 \\ -0.5494 & 0.8345 & -0.0414 \\ -0.1374 & -0.0414 & 0.9897 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -0.8474 & 0.5311 \\ 0 & 0.5311 & 0.8473 \end{bmatrix} = \begin{bmatrix} -0.8242 & 0.3927 & -0.4082 \\ -0.5494 & -0.7291 & 0.4082 \\ -0.1374 & 0.5607 & 0.8166 \end{bmatrix}$$

and

$$[R]^{(2)} = [H]^{(2)}[R]^{(1)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -0.8474 & 0.5311 \\ 0 & 0.5311 & 0.8473 \end{bmatrix} \begin{bmatrix} -7.2801 & 8.6537 & -2.8846 \\ 0 & -0.2851 & 0.5288 \\ 0 & 0.1787 & 0.6322 \end{bmatrix} = \begin{bmatrix} -7.2801 & 8.6537 & -2.8846 \\ 0 & 0.3365 & -0.1123 \\ 0 & 0 & 0.8165 \end{bmatrix}$$

This completes the factorization, which means that:

$$[a] = [Q]^{(2)}[R]^{(2)} \quad \text{or} \quad \begin{bmatrix} 6 & -7 & 2 \\ 4 & -5 & 2 \\ 1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} -0.8242 & 0.3927 & -0.4082 \\ -0.5494 & -0.7291 & 0.4082 \\ -0.1374 & 0.5607 & 0.8166 \end{bmatrix} \begin{bmatrix} -7.2801 & 8.6537 & -2.8846 \\ 0 & 0.3365 & -0.1123 \\ 0 & 0 & 0.8165 \end{bmatrix}$$

The results can be verified by using MATLAB. First, it is verified that the matrix $[Q]^{(2)}$ is orthogonal. This is done by calculating the inverse of $[Q]^{(2)}$ with MATLAB's built-in function, `inv`, and verifying that it is equal to the transpose of $[Q]^{(2)}$. Then the multiplication $[Q]^{(2)}[R]^{(2)}$ is done with MATLAB, and the result is compared with $[a]$.

```
>> Q2=[-0.8242 0.3927 -0.4082; -0.5494 -0.7291 0.4082; -0.1374 0.5607 0.8166];
>> R2=[-7.2801 8.6537 -2.8846; 0 0.3365 -0.1123; 0 0 0.8165];
>> invQ2=inv(Q2)
```

Google Docs is getting sluggish AF, so the rest of the notes will be in "All Notes 2/2".