Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Introduction to MATLAB

Introduction to MATLAB

The name MATLAB stands for MATrix LABoratory. MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects.

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment: it has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. These factors make MATLAB an excellent tool for teaching and research.

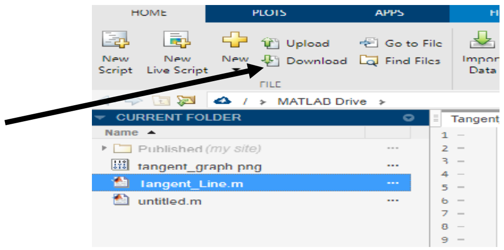
MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. The software package has been commercially available since 1984 and is now considered as a standard tool at most universities and industries worldwide. It has powerful built-in routines that enable a very wide variety of computations. It also has easy-to-use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering.

MATLAB Online

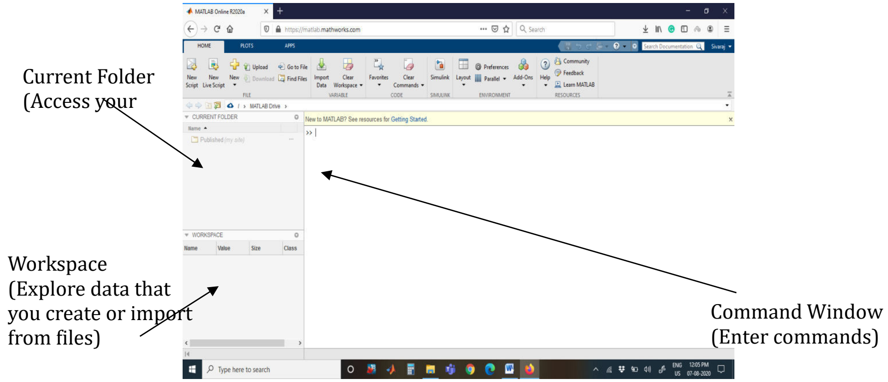
- MATLAB online can be accessed through the link https://matlab.mathworks.com/
- Enter the login credentials provided by VIT. If not sure of the password, you can reset by clicking on 'Forgot Password', after giving your VIT email ID as username.
- After entering the correct username and password, an email will be sent to your VIT email for Profile verification. Open the email and verify your profile. Now you can use the MATLAB online platform.

MATLAB Drive

- MATLAB Drive allows you to securely store and access your MATLAB files from anywhere. MATLAB Drive runs outside of MATLAB and is accessed from the notification area on your computer.
- The programs done on MATLAB online are stored in MATLAB drive. To copy the files to your local drive, you may download the file(s) as shown in below figure.



The graphical interface to MATLAB workspace can be seen in the following figure:



Basic MATLAB Commands

Starting with MATLAB: To start with, MATLAB can be used as a calculator. As an example of a simple interactive calculation, just type the expression you want to evaluate. Let's start at the very beginning. For example, let's suppose you want to calculate the expression, 1 + 2 × 3. You type it at the prompt command (>>) as follows:

```
>> 1+2*3
ans    =
             7
```

You would have noticed that if you do not specify an output variable, MATLAB uses a default variable `ans`, short for answer, to store the results of the current calculation. Note that the variable `ans` is created (or overwritten, if it is already existed). To avoid this, you may assign a value to a variable or output argument name. For example,

```
>> x = 1+2*3
x      =
         7
```

will result in `x` being given the value `1+2×3=7`. This variable name can always be used to refer to the results of the previous computations. Therefore, computing `4x` will result in

```
>> 4*x
ans     =
              28.0000
```

Quitting MATLAB: To end your MATLAB session, type **quit** in the Command Window, or Sign Out from your profile.

Creating MATLAB variables: MATLAB variables are created with an assignment statement. The syntax of variable assignment is

variable name = a value (or an expression)

For example,

```
>> x = expression
```

where expression is a combination of numerical values, mathematical operators, variables, and function calls. On other words, expression can involve:

- manual entry
- built-in functions

• user-defined functions

Overwriting variable: Once a variable has been created, it can be reassigned. In addition, if you do not wish to see the intermediate results, you can suppress the numerical output by putting a semicolon (;) at the end of the line. Then the sequence of commands looks like this:

```
>> t = 5;
>> t = t+1
t =
       6
```

Controlling the appearance of floating point number: MATLAB by default displays only 4 decimals in the result of the calculations. For example `x=7/6` is printed as `1.1667`, however, MATLAB does numerical calculations in double precision, which is 15 digits. The command format controls how the results of computations are displayed. Here are some examples of the different formats together with the resulting outputs.

```
>> format short
>> x=1.6667
```

If we want to see all 15 digits, we use the command format long

```
>> format long
>> x= 1.166666666666667
```

To return to the standard format, enter format short, or simply format.

Managing the workspace: The command clear or clear all removes all variables from the workspace. This frees up system memory.

```
>> clear
```

In order to display a list of the variables currently in the memory, type

```
>> who
```

while, `whos` will give more details which include size, space allocation, and class of the variables.

Getting help: To view the online documentation, select MATLAB Help from Help menu or type help directly in the command window. The preferred method is to use the Help Browser. The Help Browser can be started by selecting the **?** icon from the desktop toolbar. On the other hand, information about any command is available by typing

```
>> help <Command>
```

Vectors: We recall briefly how to enter vectors. Let a = (-1, 2, 4) and b = (1.5, 2, -1). To assign these vectors to variables a and b type either

```
>> a = [-1 2 4]
>> b = [1.5 2 -1]
```

or

```
>> a = [-1,2,4]
>> b = [1.5,2,-1]
```

Thus spaces or commas can be used.

Concatenation is the process of joining arrays to make larger ones. For example, if A=[1 2] and B=[3 4]. To create another matrix C concatenating A and B horizontally, we type

`>> C=[A B]` or `>> C=[A,B]`

Similarly vertical concatenation can be done as `>> C=[A;B]` .

One way of finding the dot product of two vectors, say a.b. Here is another which uses the important idea of element by element multiplication. Typing

```
>> c=a.*b
```


where there is a dot before the multiplication sign *, multiplies the vectors `a` and `b` element-by-element: in this case, `c = (-1.5,4, -4)`. The dot product is then obtained by summing the elements of c:

```
>> sum(c)
```

gives `a.b=-1.5`. Similarly

```
>> sqrt(sum(a.*a))
```

gives the magnitude of `a`. In fact, the MATLAB command, norm, will directly find the magnitude of a vector.

Matrices: A matrix is a two-dimensional array of numbers. To create a matrix in MATLAB, we enter each row as a sequence of comma or space delimited numbers, and then use semicolons to mark the end of each row. For example, consider:

$$A = \begin{bmatrix} -1 & 6 \\ 7 & 11 \end{bmatrix}$$

This matrix is entered in MATLAB using the following syntax:

```
>> A = [-1,6; 7,11];
```

or consider the matrix:

$$B = \begin{bmatrix} 2 & 0 & 1 \\ -1 & 7 & 4 \\ 3 & 0 & 1 \end{bmatrix}$$

We enter it in MATLAB in the following way:

```
>> B = [2,0,1;-1,7,4; 3,0,1]
```

Special Matrix Type: The identity matrix is a square matrix that has ones along the diagonal and zeros elsewhere. To create an $n \times n$ identity matrix, type the following MATLAB command:

`eye(n)`

Let's create a 3 × 3 identity matrix:

```
>> eye(3)
ans =
         1    0    0
         0    1    0
         0    0    1
```

To create an n × n matrix of zeros, we type zeros(n). We can also create a m × n matrix of zeros by typing zeros(m, n). Similarly, we can generate a matrix completely filled with 1's by typing ones(n) or ones(m,n).

Referencing Matrix Elements: Individual elements and columns in a matrix can be referenced using MATLAB.

Consider the matrix:

```
>> A = [1 2 3; 4 5 6; 7 8 9]
A =
       1      2      3
       4      5      6
       7      8      9
```

We can pick out the element at row position m and column position n by typing A(m,n). For example:

```
>> A(2,3)
ans =
        6
```

To reference all the elements in the ith column we write A(:,i). For example, we can pick out the second column of A:

```
>> A(:,2)
ans =
```

```
        2
        5
        8
```

To pick out the elements in the ith through jth columns we type A(:,i:j). Here we return the second and third columns:

```
>> A(:,2:3)
ans =
        2       3
        5       6
        8       9
```

We can pick out pieces or submatrices as well. Continuing with the same matrix, to pick out the elements in the second and third rows that are also in the first and second columns, we write:

```
>> A(2:3,1:2)
ans =
        4       5
        7       8
```

We can change the value of matrix elements using these references as well. Let's change the element in row 1 and column 1 to –8:

```
>> A(1,1) = -8
A =
       -8       2       3
        4       5       6
        7       8       9
```

Symbolic math computations

Symbolic Math Toolbox provides functions for solving, plotting, and manipulating symbolic math equations. The toolbox provides libraries of functions in common mathematical areas such as calculus, linear algebra, algebraic and ordinary differential equations, equation simplification, and equation manipulation. Symbolic Math Toolbox lets you analytically perform differentiation, integration, simplification, transforms, and equation solving. Your computations can be performed either analytically or using variable precision arithmetic, with the results displayed in mathematical typeset.

Key Features

- Symbolic integration, differentiation, transforms, and linear algebra.
- Algebraic and ordinary differential equation (ODE) solvers.
- Simplification and manipulation of symbolic expressions.
- Plotting of analytical functions in 2D and 3D.
- Variable-precision arithmetic.

Create Symbolic Numbers

You can create symbolic numbers by using `sym`. Symbolic numbers are exact representations, unlike floating-point numbers. For example,

```
>> sym(4/5)
ans =
    4/5
```

The symbolic number is represented in exact rational form, while the floating-point number is a decimal approximation. Calculations on symbolic numbers are exact. For example,

```
>> sin(sym(pi))
ans =
0
```

while the numerical approximation for $\sin\pi$ can be obtained as below:

```
>> sin(pi)
ans =
   1.2246e-16
```


Create Symbolic Variables

There are two ways to create symbolic variables, `syms` and `sym`.
The `syms` syntax is a shorthand for `sym`.

To create symbolic variables `x` and `y` using `syms` and `sym` respectively.

```
>> syms x
>> y = sym('y')
```

With `syms`, you can create multiple variables in one command. Create the variables `a, b` and `c`.

```
>> syms a b c
```

To create many numbered variables, namely `a1, ..., a10` try as below:

```
>> A = sym('a', [1,10])
A =
[a1, a2, a3, a4, a5, a6, a7, a8, a9, a10]
```

Create Symbolic Expressions

Suppose we want to study the quadratic function $f = ax^2 + bx + c$, we first create the symbolic variables `a, b, c`, and `x`:

```
>> syms a b c x
```

Then, assign the expression to f:

```
>> f = a*x^2 + b*x + c;
```

Calculus with Symbolic Math

Limits:

The following examples illustrate how to find the limit of a function $\underset{x\to a}{Lim} f(x)$.

Ex. 1. $\underset{x\to\infty}{Lim}\left(1-\frac{2}{x}\right)^x$

Type the following code in a script file and run.

```
syms x
f(x)=(1-2/x)^x;
L = limit(f(x), x, inf)
```

Output:

```
L=
exp(-2)
```

Ex. 2. $\underset{x\to\infty}{Lim}\left(\sqrt{9x^2+x}-3x\right)$

Type the following code in a script file and run.

```
syms x
f = sqrt(9*x^2 + x) - 3*x
L = limit(f, x, inf)
```

Output:

```
L=
1/6
```

Ex. 3. Calculating the left and right limits of $\dfrac{x^2-1}{x^2-6x+5}$ as $x \to 5$.

Type the following code in a script file and run.

```
syms x
f = (x^2 - 1) / (x^2 - 6*x + 5)
LL = limit(f, x, 5, 'left')      % For Left limit x→5⁻
RL = limit(f, x, 5, 'right')     % For right limit x→5⁺
L=limit(f, x, 5)                 % For limit x→5
```

Output:

```
LL=
-Inf

RL=
Inf

L =
NaN
```

Derivative as Rate of Change:

As we know, the derivative or slope of the tangent of $f(x)$ at a point $x=a$ can be determined by $\displaystyle \lim_{x\to a}\left\{\frac{f(x)-f(a)}{x-a}\right\}$ or $\displaystyle \lim_{h\to 0}\left\{\frac{f(a+h)-f(a)}{h}\right\}$. The following example illustrates how to find the derivative of a function $f(x)$ at a point $x=a$.

```
clear all
clc
syms h x y
f(x)=4*x-x^2;
a=1;
Df=limit((f(x+h)-f(x))/h,h,0)  % Derivative of f(x)
m=limit((f(x)-f(a))/(x-1),x,a) % slope of Tangent Line, method 1
m=limit((f(a+h)-f(1))/h,h,0)   % slopt of Tangent Line, method 2
g=f(a)+m*(x-a)
Tangent_Line=y==g
```

Output:

```
Df=

4-2*x

m=
2

m=
2

Tangent_Line =
y == 2*x + 1
```


Derivative of a $f(x)$ can be found directly by the syntax `diff(f(x),x)` and second derivative by `diff(f(x),x,2)`. The following code illustrates how to find derivative of $f(x)=\dfrac{x^2+4x+3}{\sqrt{x}}$.

```
syms x
f(x)=(x^2+4*x+3)/sqrt(x)
df_dx = diff(f(x), x)       %First derivative
d2_fx = diff(f(x), x,2)     %Second derivative
```

Output:

```
df_dx =
(2*x+4)/x^(1/2)-(x^2+4*x+3)/(2*x^(3/2))
d2_fx =
(3*(x^2+4*x+3))/(4*x^(5/2))-(2*x+4)/x^(3/2)+2/x^(1/2)
```

Exercise Questions:

1. Evaluate the following limits with MATLAB

   (a) $\displaystyle \lim_{x\to 0}\left(\frac{\frac{1}{x-1}+\frac{1}{x+1}}{x}\right)$ (b) $\displaystyle \lim_{x\to -1}\left(\frac{\sqrt{x^2+8}-3}{x+1}\right)$ (c) $\displaystyle \lim_{x\to -2}\left(\frac{x+2}{\sqrt{x^2+5}-3}\right)$ (d) $\displaystyle \lim_{x\to -3}\left(\frac{2-\sqrt{x^2-5}}{x+3}\right)$

   (e) $\displaystyle \lim_{x\to 0}\left(\frac{x-x\cos x}{\sin^2 3x}\right)$ (f) $\displaystyle \lim_{x\to 0}\left(\frac{x+x\cos x}{\sin x\cos x}\right)$

2. At time t, the postion of a body moving along the s- axis is $s=t^3-6t^2+9t$ m.
   (a) Find the body's acceleration each time the velocity is zero.
   (b) Find the body's speed each time the acceleration is zero.
   (c) Find the total distance travelled by the body from t=0 to t=2.

3. When a batericide was added to a nutrient broth in which bacteria were growing, the bacterium population continued to grow for a while, but then stopped growing an began to decline. The size of the population at time t (hours) was $b=10^6+10^4t-10^3t^2$. Find the growth rates at
   (a) t=0 hours (b) t=5 hours (c) t=10 hours

4. Find the following derivatives using MATLAB commands

   (a) $\dfrac{d}{dx}(5x^3-x^4)^7$ (b) $\dfrac{d}{dx}\left(\dfrac{1}{3x-2}\right)$ (c) $\dfrac{d}{dx}(\sin^5 x)$ (d) $\dfrac{d}{dx}\left(1-\dfrac{x}{7}\right)^{-7}$

   (e) $\dfrac{d}{dx}\left(\dfrac{x}{2}-1\right)^{-10}$ (f) $\dfrac{d}{dx}\left(\dfrac{x^2}{8}+x-\dfrac{1}{x}\right)^4$ (g) $\dfrac{d}{dx}\sqrt{3x^2-4x+6}$

5. A new product placed in market becomes very popular. Its quantity sold N is given as a function of time t, where t is measured in weeks: $N(t) = \frac{250000\, t^2}{(2t+1)^2}$, $t > 0$. Differentiate this function. Then use the derivative to evaluate $N'(52)$ and $N'(208)$ and interpret these results.
(Hints: $N'(t) = \frac{500000\, t}{(2t+1)^3}$; $N'(52) \approx 22.5$ and $N'(208) \approx 1.4$. The slopes of the tangent lines, representing the change in numbers of units sold per week, are levelling off as t increases. Perhaps the market is becoming saturated with this product: while sales continue to increase, the rate of the sales increase per week is levelling off.)

6. Suppose that an object travels so that its distance s in miles, from its starting point is a function of time t, in hours, as follows: $s(t) = 10\, t^2, t > 0$.
(i) Find the average velocity between the times $t = 2$ hr and $t = 5$ hr,
(ii) Find the (instantaneous) velocity when $t = 4$ hr and
(iii) acceleration.
(Hints: $Avg = \frac{s(5)-s(2)}{3} = 70\,\frac{\text{mi}}{\text{hr}}$; $s'(4) = 80\,\frac{\text{mi}}{\text{hr}}$; $20\frac{\text{mi}}{\text{hr}^2}$)

7. The United States population is becoming more diverse. Based on the U.S. Census population projections for 2000 to 2050, the projected Hispanic population (in millions) can be modelled by the function $H(t) = 37.791(1.021)^t$, where $t = 0$ corresponds to 2000 and $0 \le t \le 50$. Use H to estimate the average rate of change in the Hispanic population from 2000 to 2010.
(Hints: vg $= \frac{\mathrm{H}(10)-\mathrm{H}(0)}{10} = 0.873$. Based on this model, it is estimated that the Hispanic population in the United States increased, on average, at a rate of about 873,000 people per year between 2000 and 2010.)

8. A leaking oil well off the Gulf Coast is spreading a circular film of oil over the water surface. At any time t (in minutes) after the beginning of the leak, the radius of the circular oil slick (in feet) is given by $r(t) = 4t$. Find the rate of change of the area of the oil slick with respect to time.
(Hints: Rate of change in the radius over time is $\frac{dr}{dt} = 4$. The area of the oil slick is $\mathrm{A(r)} = \pi r^2$. The rate of change of the area of the oil slick is $\frac{dA}{dt} = 32\pi t$).

9. The sales as a function of time is given by $\frac{100000}{1+100e^{-0.3\,t}}$. Find the rate of change of sales after 4 years.
(Hints: $s'(t) = \frac{3000000e^{-0.3\,t}}{(1+100e^{-0.3\,t})^2}$; $s'(4) \approx 933$. The rate of change of sales after 4 years is about 933 units per year. The positive number indicates that sales are increasing at this time).

10. Based on projections from the Kelly Blue Book, the resale value of a 2010 Toyota Corolla 4-door sedan can be approximated by the following function $f(t) = 15450 - 13915\log(t+1)$ where t is the number of years since 2010. Find and interpret $f(4)$ and $f'(4)$.
(Hints: $\mathrm{f}(4) \approx 5500.08$. The average resale value of a 2010 Toyota Corolla in 2014 would be approximately \$5500.08. $\mathrm{f}'(t) = -\frac{13915}{t+1}$; $\mathrm{f}'(4) \approx -1208.64$. In 2014, the average resale value of a 2010 Toyota Corolla is decreasing by \$1208.64 per year).

Appendix

Matrices Operations		
Operation	Syntax	Remarks
Matrix Transpose	`>> C=A'`	
Matrix Inverse	`>> C=inv(A)`	In case of Singular matrix warning will be displayed.
Determinant	`>> C=det(A)`	
Addition	`>> C=A+B`	Matrices A and B must be of same order
Subtraction	`>> D=A-B`	Matrices A and B must be of same order
Multiplication	`>> C=A*B`	Matrices A and B must be compatible for multiplication (A is of order mxp and B is pxn, then C is of order mxn.)
Element-wise multiplication	`>> C=A.*B`	Eg. A=[x, y]; B=[a, b], then A.*B=[xa, yb] Matrices A and B must be of same order
Element-wise division	`>> C=A./B`	Eg. A=[x, y]; B=[a, b], then A./B=[x/a, y/b] Matrices A and B must be of same order
Element-wise exponentiation	`>> C=A.^p`	Eg. A=[x, y]; p=2; then A.^p=[x^2, y^2]
Right division	`>> A\b`	Solves the system of linear equations AX=b.
To create an array of n elements	`>>x=linspace(x1,xn,n)`	Creates an array of n elements starting with x1 and ending with xn.
To create an array with h spacing	`>> C=a:h:b`	Creates an array with first element as a and last element as b with step length h.
Determinant	`>> C=det(A)`	
Zero matrix	`>> A=zeros(m,n)`	Creates an mxn matrix of zeros.
Identity matrix	`>> I=eye(n)`	Creates an identity matrix of order n.
Ones	`>> B=ones(m,n)`	Creates a matrix of order mxn of ones.
Length of a vector	`>> n=length(X)`	Displays the number of elements in the vector X
Size of an Array	`>> [m,n]=size(A)`	Displays the number of rows and columns of the matrix.
Diagonal elements	`>> D=diag(A)`	Creates a vector with diagonal entries of a given matrix.
Diagonal matrix	`>>A=diag(x1,x2,…,xn)`	Creates a diagonal matrix with the specified diagonal elements.

Transcendental functions		
Operation	Syntax	Remarks
e^x	`>> exp(x)`	
$\log_e x$	`>> log(x)`	Base e
$\log_{10} x$	`>> log10(x)`	Base 10
$\sin x$	`>> sin(x)`	Here x is in radians. When x is in degrees, we can use sind(x), cosd(x), tand(x), secd(x), cscd(x).
$\cos x$	`>> cos(x)`	
$\tan x$	`>> tan(x)`	
$\sec x$	`>> sec(x)`	
cosec x	`>> csc(x)`	
Inverse trigonometric functions	`>> asin(x)` `>> acos(x)` `>> atan(x)`	

Basic Output Commands		
Operation	Syntax	Remarks
Display	`>> display('text')`	The display command writes text.
Formatting the output	`>> format short`	Scaled fixed point format with 5 digits.
	`>> format short e`	Floating point format with 5 digits.
	`>> format long`	Scaled fixed point format with 15 digits
	`>> format long e`	Floating point format with 15 digits.
Printing the output	`>>sprintf(formatSpec, A1,…,An)`	Formats the data in arrays A1,...,An according to formatSpec in column order
Some format specifiers	`>>sprintf('%f',A)`	Try with %e, %f, and %g specifiers. For example: `>>x=1/exp(1);` `>>sprintf('%0.5f',x)`

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Plotting of 2D curves

Plotting Graphs

MATLAB has many commands that can be used for creating various kinds of 2D plots.

Easy plots: The simplest built-in function for plotting an explicit function is `ezplot` command. For example, we want to plot a parabola on the interval `[-1,1]`

$$f(x)=x^2, -1\le x\le 1.$$

We can simply use the following command:

```
ezplot('x^2',[-1,1])
```

The first input of `ezplot` command is a string describing the function. The second input (which is optional) is the interval of interest in the graph. Note that the function description is not necessarily a function in variable `x`. We can use any variable we like:

```
ezplot('sin(b)',[-2,3])
```

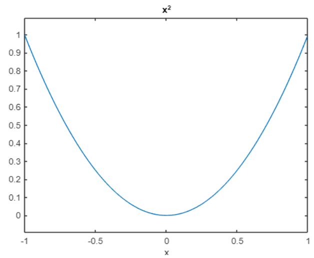
If the function definition involves some constant parameter, write it explicitly:

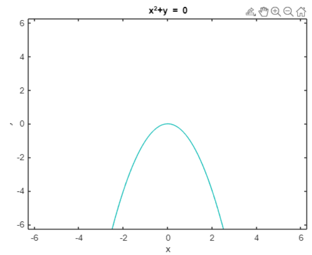
```
ezplot('x^2+2')
```

In this case, if we assign a value 2 to a variable first, says `y=2,` when executing:

```
y = 2;
ezplot('x^2+y')
```

MATLAB will interpret y as a variable (not a value of 2). Hence, it will create a contour plot of the function $x^2+y=0$





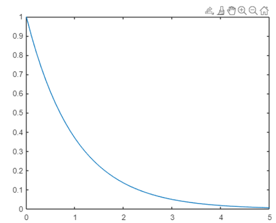
Plot command

The plot command is used to create a two-dimensional plot. The simplest form of the command is:

```
plot(x,y)
```

where x and y are each a vector. Both vector **must** have the same number of elements. For example:

```
x = 0:0.1:5;
y = exp(-x);
plot(x,y)
```



Once the plot command is executed, the figure Window opens and the plot is displayed. The plot appears on the screen in blue which is the default line color. The plot command has additional arguments that can be used to specify the color and style of the line and the color and type of markers, if any are desired. With these options the command has the form:

```
plot(x,y,'linespec','PropertyName',PropertyValue)
```

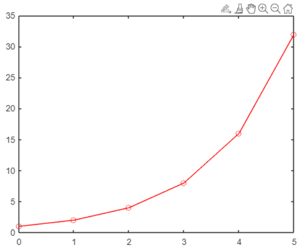
Line Specifiers (linespec): Line specifiers are optional and can be used to define the style and color of the line and the type of markers (if markers are desired).

Line Style	Specifier
solid (default)	-
dashed	--
dotted	:
dash-dot	-.

Line Color	Specifier
red	r
green	g
blue	b
cyan	c
magenta	m
yellow	y
black	k
white	w

Marker Type	Specifier
plus sign	+
circle	o
asterisk	*
point	.
cross	x
triangle (pointed up)	^
triangle (pointed down)	v
square	s
diamond	d
five-pointed star	p
sixed-pointed star	h
triangle (pointed left)	<
triangle (pointed right)	>

The specifiers are typed inside the plot command as strings. Within the string, the specifiers can be typed in any order and the specifiers are optional.



For example:

```
x=0:5;y=2.^x; plot(x,y,'-or')
```

This command creates a plot with solid red line and the marker is a circle.

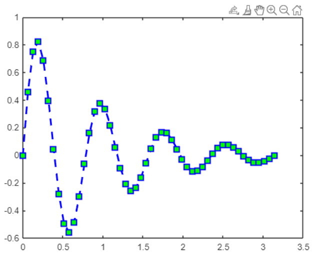
Property Name and PropertyValue: Properties are optional and can be used to specify the thickness of the line, the size of the marker, and the colors of the marker's edge line and fill. The Property Name is typed as a string, followed by a comma and a value for the property, all inside the plot command.

Property Name	Description
linewidth	the width of the line (default 0.5, possible values are 1,2,3,...)
markersize	the size of the marker (e.g., 5,6,...)
markeredgecolor	the color of the edge line for filled marker (e.g., r, b)
markerfacecolor	the color of the filling for filled markers (e.g, r, b)

For example, the command:

```
x=linspace(0,pi,50);
y=exp(-x).*sin(8*x);
plot(x,y,'--sb', 'linewidth', 2,
'markersize',8, 'markerfacecolor', 'g')
```

creates a plot with a blue dashed line and squares as markers. The linewidth is 2 points and the size of the square markers is 8 points. The marker has green filling.



Formatting a plot

Plots can be formatted by using MATLAB command that follow the `plot` or `fplot` commands, or interactively by using the plot editor in the Figure Window. Here we will explain the the first method which is more useful since a formatted plot can be created automatically every time the program is executed.

The xlabel and ylabel command

`xlabel` and `ylabel` create description on the x- and y-axes, respectively after we have plotted a graph. The usages are:

```
xlabel('text as string')
ylabel('text as string')
```

The title command: A title can be added to the plot with the command:

```
title('text as string')
```

The text will appear on the top of the figure as a title.

The text command: A text label can be placed in the plot with the text or gtext commands:

```
text(x,y,'text as string')
gtext('text as string')
```

The text command places the text in the figure such that the first character is positioned at the point with the coordinates x,y(according to the axes of the figure). The gtext command places the text at a position specified by the user (with the mouse).

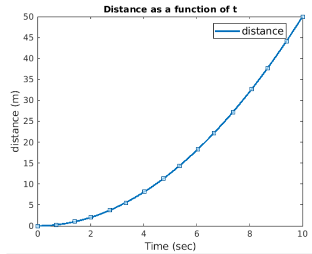
The legend command: The legend command places a legend on the plot. The legend shows a sample of the line type of each graph that is plotted, and places a label, specified by the user, beside the line sample. The usage is:

```
legend('string 1','string 2',... )
```

For example, the command:

```
t=linspace(0,10,100)
s=t.^2/2;
plot(t,s);
xlabel('Time (sec)');
ylabel('distance (m)');
title('Distance as a function of t');
legend('distance')
```

creates a plot as shown here:



Formatting the texts: The texts in the `xlabel, ylabel, title, text` and `legend` commands can be formatted to customize the font, size, style (italic, bold, etc.), and color. Formatting can be done by adding optional `PropertyName` and `PropertyValue` arguments following the string inside the command. For example:

```
text(x,y,'text as string', 'PropertyName', PropertyValue)
```

Some of the PropertyName are:

Property Name	Description
Rotation	the orientation of the text (in degree)
FontSize	the size of the font (in points)
FontWeight	the weight of the characters (light, normal, bold)
Color	the color of the text (e.g., r, b, etc.)

Some formatting can also be done by adding modifiers inside the string. For example, adding \bf, \it, or \rm, will create a text in bold font, italic style, and with normal font, respectively. A single character can be displayed as a subscript or a superscript by typing _ (the underscore character) or ^ in front of the character, respectively. A long superscript or subscript can be typed inside { } following the _or the ^. For example:

```
title('\bf The values of X_{ij}', 'FontSize',18)
```

Greek characters: Greek characters can be included in the text by typing \ (back slash) before the name of the letter.

Character	Letter
\alpha	α
\beta	β
\gamma	γ
\theta	θ
\pi	π
\sigma	σ

The axis command: When the `plot(x,y)` command is executed, MATLAB creates axes with limits that are based on the minimum and maximum values of the elements of x and y. The axis command can be used to change the range of the axes. Here are some possible forms of the axis command:

Commands	Description
axis([xmin, xmax, ymin, ymax])	sets the limits of both x and y
axis equal	sets the same scale for both axes
axis square	sets the axes region to be square
axis tight	sets the axis limits to the range of the data

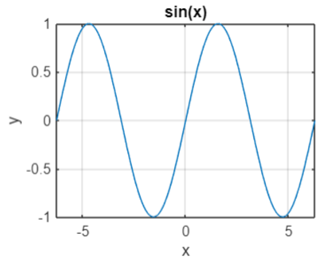
The grid command: 'grid on' adds grid lines to the plot. grid off removes grid lines from the plot.
For example:

```
fplot(@(x) x.^2+4.*sin(2.*x)-1,[-3,3])
grid on
axis([-2 2 -5 5])
```

Plotting with fplot:

`fplot(f)` plots the curve defined by the function $y = f(x)$ over the default interval [-5, 5] for x.
`fplot(f, [xmin, xmax])` plots over the specified interval `[xmin, xmax]`.
For example, to plot $f(x) = \sin x$ curve in the interval $[-2\pi, 2\pi]$.

```
syms x
f(x)=sin(x)
fplot(f(x),[-2*pi, 2*pi])
grid on
title('sin(x)')
xlabel('x');
ylabel('y');
```



Specify Plotting Interval and Plot Piecewise Functions

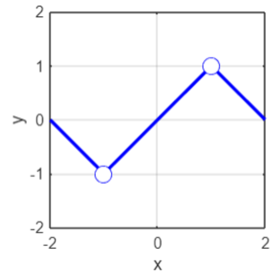
`f(x)=piecewise(I1,f1,I2,f2,...)` can be used to define a piecewise function $f(x)$ whose value is `f1` in the interval `I1` and `f2` in the interval `I2`.

For example to plot a piecewise function $f(x) = \begin{cases} -x-2 & ; -2 \le x \le -1 \\ x & ; -1 < x \le 1 \\ -x+2 & ; 1 < x \le 2 \end{cases}$.

```
clear
clc
syms f(x)
f(x)=piecewise( -2<=x<=-1,-x-2,-1<x<=1,x,1<x<=2,-x+2);
figure
fplot(f(x), [-2,-1], 'b', 'LineWidth', 2);
grid on; hold on
```



```
fplot(f(x), [-0.999, 1], 'b', 'LineWidth', 2);
fplot(f(x), [1.001, 2], 'b', 'LineWidth', 2);
plot(-1, f(-1), 'bo', 'MarkerFaceColor', 'w', 'MarkerSize', 10);
plot(1, f(1), 'bo', 'MarkerFaceColor', 'w', 'MarkerSize', 10);
axis equal; axis([-2 2 -2 2]);
xlabel('x'); ylabel('y');
```



Plotting multiple graphs

In many situations there is a need to make several graphs in the same plot. There are two methods to plot multiple graphs in one figure. One is by using the plot command, the other is by using the hold on, hold off commands. For example:

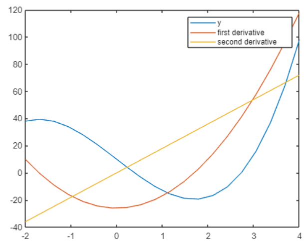
```
plot(x,y,u,v,s,t)
```

creates 3 graphs: y vs x, v vs u, and t vs s, all in the same plot. The vector of each pair must have the same length. MATLAB automatically plots the graphs in different colors so that they can be identified. It is also possible to add line specifiers following each pair. For example:

```
plot(x,y,'-bo',u,v,'--rx',s,t,'g:s')
```

plots y vs x with a solid blue line and circles, v vs u with a dashed red lines with cross signs, t vs s with a dotted green line and square markers.

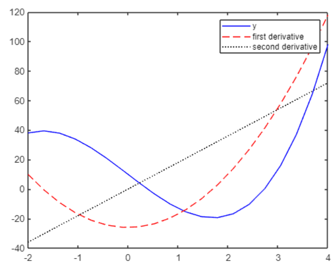
For example, the following script plots the graphs of the functions $y = 3x^3 - 26x + 10$ together with its first derivative $y' = 9x^2 - 26$ and second derivative $y'' = 18x$, for $-2 \le x \le 4$.




```
x = linspace(-2,4,20);
y = 3*x.^3-26*x+10;
dy = 9*x.^2-26;
ddy = 18*x;
ddy = 18*x;
plot(x,y,x,dy,x,ddy);
legend('y','first derivative','second derivative');
```

Using the hold on, hold off command

To plot several graphs using the hold on, hold off commands, one graph is plotted first with the plot command. Then the hold on command is typed. This keeps the Figure Window with the first plot open, including the axis properties and the formatting. Additional graphs can be added with plot commands that are typed next. The hold off command stops this process. It returns MATLAB to the default mode in which the plot command erases the previous plot and resets the axis properties.



With the data from the previous example, we can plot y and its derivatives by typing commands shown in the script:

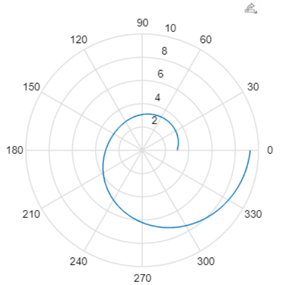
```
plot(x,y,'-b');
hold on
plot(x,dy,'--r');
plot(x,ddy,':k');
hold off
```

Polar plots

The polar command is used to plot functions in polar coordinates. The command has the form:

```
polar(theta,radius,'linespec')
```

where theta and radius are vectors whose elements define the coordinates of the points to be plotted. The line specifiers are the same as in the plot command. To plot a function $r = f(\theta)$ in a certain domain, a vector for values of θ is created first, and then a vector r with the corresponding values of $f(\theta)$ is created using element-wise calculation.



For example, a plot of the function $r = 3\cos^2(\theta/2) + \theta$, for $0 \le \theta \le 2\pi$ is done by:

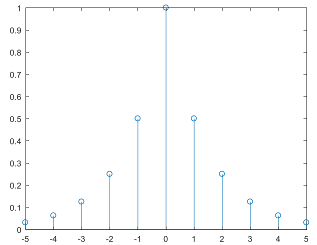
```
theta = linspace(0,2*pi,200);
r = 3*cos(theta/2).^2+theta;
polar(theta,r)
```

Stem plots: A two-dimensional stem plot displays data as lines extending from a baseline along the *x*-axis. A circle (the default) or other marker whose *y*-position represents the data value terminates each stem.

`stem(Y)` plots the data sequence `Y` as stems that extend from equally spaced and automatically generated values along the *x*-axis. When `Y` is a matrix, `stem` plots all elements in a row against the same *x* value. `stem(X,Y)` plots `X` versus the columns of `Y`. `X` and `Y` are vectors or matrices of the same size.

For example, the discrete plot of $y = \left(\frac{1}{2}\right)^{|x|}$, $-5 \le x \le 5$.

```
x=-5:5;
y= 0.5.^abs(x);
stem(x,y);
```



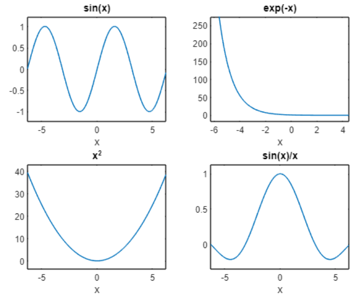
Multiple plots on the same window: Multiple plots on the same page can be created with the subplot command, which has the form:

```
subplot(m,n,p)
```


The command divides the Figure Window into mxn rectangular subplots where plots will be created. The subplots are arranged like elements in a mxn matrix where each element is a subplot. The subplots are numbered from 1 through mn. The number increases from left to right within a row, from the first row to the last. The command subplot(m,n,p) makes the subplot p current. This means that the next plot command will create a plot in this subplot.

For example, we create a plot that has 2 rows and 2 columns:

```
subplot(2,2,1);ezplot('sin(x)');
subplot(2,2,2);ezplot('exp(-x)');
subplot(2,2,3);ezplot('x^2');
subplot(2,2,4);ezplot('sin(x)/x');
```



Exercise:

1. Plot the graph of the function $f(x) = x^3 + 3x^2 + 2$ in the domain $-10 \le x \le 10$. Indicate the x – label, y – label, title of the graph. Set the thickness of the line as 2 pt.

2. In the domain $-3\pi \le x \le 3\pi$, plot $y = \sin x$. On the same graph, superimpose the curve $y = \cos x$ with a different colour. Indicate the x – label, y – label, title of the graph and legend.

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Experiment 1–A
Mean value theorem

Mean value theorem:

Suppose that the function $y = f(x)$ is continuous at every point of the closed interval $[a,b]$ and differentiable at every point in (a,b), then there is at least one number c in (a,b) so that

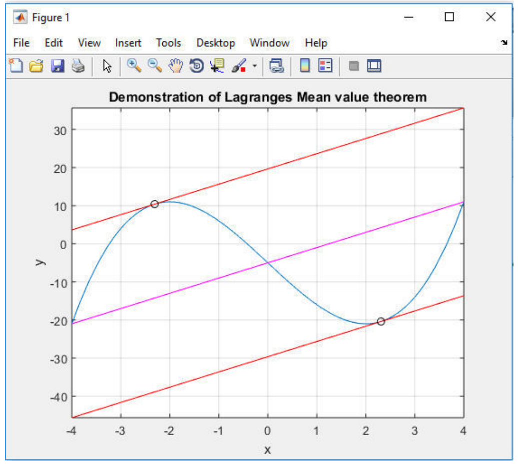
$$f'(c) = \frac{f(b)-f(a)}{b-a}.$$

The code given below illustrates the verification of Lagrange's theorem for the function $f(x) = x^3 - 12x - 5$ on the interval $[-4,4]$.

```
clear
clc
syms x y
f(x)=x^3-12*x-5;I=[-4,4]; % Input the function and interval
a=I(1);b=I(2);
Df=diff(f,x);
m=(f(b)-f(a))/(b-a); %Slope of Secant Line
c=solve(Df==m, x);
c=c(a<c&c<b);
disp('Values of c lying in the interval I are');
disp(double(c));
T=f(c)+m*(x-c); %Tangents at x=c
disp('The Tangent Lines at c are');
disp(vpa(y==T,4));
figure
fplot(f,I); grid on; hold on;
fplot(T, I, 'r');     %Tangent Lines
plot(c, double(f(c)), 'ko');
plot(I, double(f(I)), 'm');   %Secant Line
xlabel('x'); ylabel('y');
title('Demonstration of Lagranges Mean value theorem');
```

Output:

```
Values of c lying in the interval I are
   -2.3094
    2.3094
The Tangent Lines at c are
 y == 4.0*x + 19.63
 y == 4.0*x - 29.63
```

Exercise:

1. Using MATLAB find the tangent to the curves $y = \sqrt{x}$ at $x = 4$ and show graphically.
2. Using MATLAB find the tangent to the curves $y = -\sin(x/2)$ at the origin and and show graphically.
3. Verify Rolle's theorem for the function $(x+2)^3(x-3)^4$ in the interval $[-2,3]$. Plot the curve along with the secant joining the end points and the tangents at points which satisfy Rolle's theorem.
4. Verify Lagrange's mean value theorem for the function $f(x) = x + e^{3x}$ in the interval $[0,1]$. Plot the curve along with the secant joining the end points and the tangents at points which satisfy Lagrange's mean value theorem.

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Experiment 1–B
Maxima and Minima of a function of one variable

Absolute/Global Extrema

- Let $f(x)$ be a function defined in $[a,b]$. Then f has an absolute maximum value on $[a,b]$ at a point $c \in [a,b]$ if $f(x) \le f(c)$ for all $x \in [a,b]$.
- Let $f(x)$ be a function defined in $[a,b]$. Then f has an absolute minimum value on $[a,b]$ at a point $c \in [a,b]$ if $f(x) \ge f(c)$ for all $x \in [a,b]$.

Extreme value theorem

If f is continuous on a closed interval $[a,b]$, then f attains both an absolute maximum value M and an absolute minimum value m in $[a,b]$.

Relative/Local Extreme values

- A function f has a local maximum value at an interior point c of its domain if $f(x) \le f(c)$ for all x in some open interval containing c.
- A function f has a local minimum value at an interior point c of its domain if $f(x) \ge f(c)$ for all x in some open interval containing c.

- A function f has a local maximum or minimum value at an interior point c of its domain, and if f' is defined at c, then $f'(c) = 0$.

MATLAB syntax used in the code:

Here we used the inbuilt function `findpeaks` to find the peak values (maxima) of the function $f(x)$ at the interior points of the given interval.
For finding minimum value of $f(x)$, we have considered $f_1(x) = -f(x)$ and used the `findpeaks` function.

The syntax `[lmax_f,loc]=findpeaks(f)` finds the local maximum value(s) of the discrete function i.e., `lmax_f` and the location (index of x-value) `loc` of the local maxima.

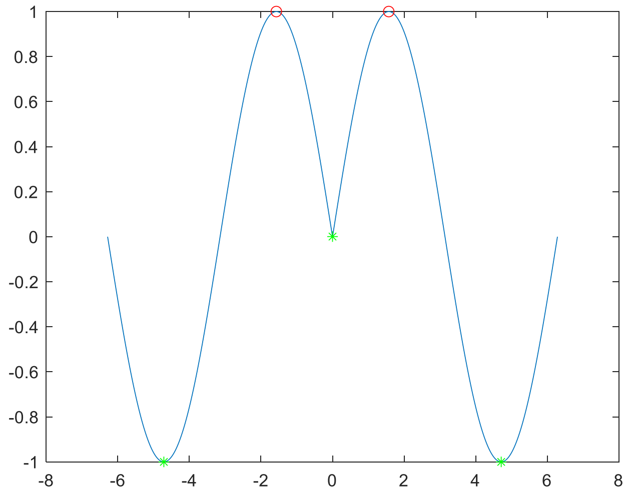
The global maximum and minimum of the function can be clearly visualised from the generated graph.

The following MATLAB code illustrates the evaluation and visualization of local extrema of a function $f(x) = \sin|x|$ in an interval $I = (-2\pi, 2\pi)$.

```
clear
clc
syms x
f(x)=sin(abs(x));
I=[-2*pi,2*pi];
f1(x)=-f(x);
a=I(1);b=I(2);
t=linspace(a,b,10000); %Discretizing the interval I
g=double(f(t)); %Finding the values of f(x) at t values
[lmax_f,loc]=findpeaks(g);
lmax_x=round(t(loc),4);
h=double(f1(t));
[lmin_f,loc]=findpeaks(h);
lmin_x=round(t(loc),4);
disp('Local maximum occur at x=')
disp(lmax_x)
disp('The Local Maximum value(s) of the function are ')
disp(double(f(lmax_x)))
disp('Local minimum occur at x=')
disp(lmin_x)
disp('The Local Minimum value(s) of the function are ')
disp(double(f(lmin_x)))
plot(t,f(t));hold on; %Plotting the function
plot(lmax_x,double(f(lmax_x)),'or');%Pointing the local
maxima on the curve of f(x)
plot(lmin_x,double(f(lmin_x)),'*g');%Pointing the local
minima on the curve of f(x)
hold off
```

Output

```
Local maximum occur at x=
-1.5703    1.5703
The Local Maximum value(s) of the function are
1.0000   1.0000
Local minimum occur at x=
-4.7122   -0.0006    4.7122
The Local Minimum value(s) of the function are
-1.0000    0.0006   -1.0000
```



EXERCISE

1. Find the local and global maxima and minima for the function $x^3-12x-5$ on $x\in(-4,4)$.

2. Find the local and global maxima and minima for the function $x+\sin 2x$ on $x\in(-5,5)$.

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Experiment 2–A
Applications of Integration: finding area, volume of solid of revolution

Area between the curves

If f and g are continuous with $f(x)\ge g(x)$ for $x\in[a,b]$, then the area of the region between the curves $y=f(x)$ and $y=g(x)$ from a to b is the integral

$$A=\int_a^b[f(x)-g(x)]dx.$$

Also, if a region's bounding curves f and g are described by functions of y, where f denotes the right hand curve and g denotes the left hand curve, $f(y)-g(y)$ being non negative, then the area of the region between the curves $x=f(y)$ and $x=g(y)$ from $y=c$ to d is the integral

$$A=\int_c^d[f(y)-g(y)]dy.$$

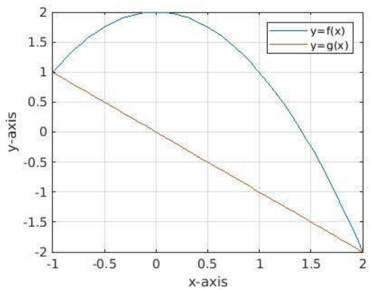
Example 1.

The area bounded by the curves $y=2-x^2$ and the line $y=-x$, from $x=-1$ to 2 is given by the following code:

```
clear
clc
syms x
f(x)=2-x^2; % Upper curve
g(x)=-x;    % Lower curve
I=[-1,2]; % Interval of Integration
a=I(1); b=I(2);
A=int(f(x)-g(x),a,b); % Finding the area by integration
disp('Area bounded by the curves f(x) and g(x) is:' );
disp(A);
fplot(f(x),[a,b]);grid on;hold on; %Plotting the upper curve
fplot(g(x),[a,b]);hold off         %Plotting the lower curve
xlabel('x-axis');ylabel('y-axis');
legend('y=f(x)','y=g(x)');
```

Output

```
Area bounded by the curves f(x) and g(x) is:
9/2
```



Volume of solid of revolution – Disc method

The solid figure formed by revolving a plane curve about an axis is called Solid of revolution.

If the solid is formed by revolving the curve $y = f(x)$ about a line $y = c$ (parallel to the $x-$axis), then the volume of the solid formed is given by $V = \int_a^b \pi[y-c]^2\,dx$.

Note: For $c = 0$, the axis of revolution will be the $x-$axis itself.

Example 2.
The volume of the solid generated by the revolving the curve $y = \sqrt{x}$ about the line $y = 1$ from $x = 1$ to $x = 4$ is given by the following code:

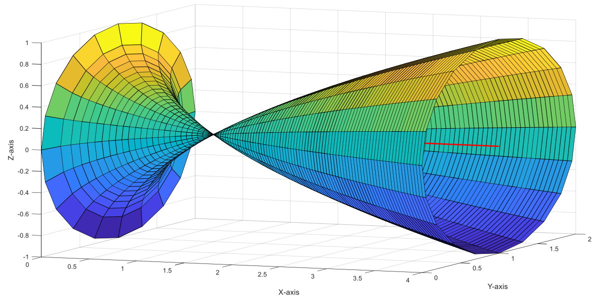
MATLAB Syntax used:
The code below consists of two parts.
The first part evaluates the volume of the solid generated by revolving the curve $y = f(x)$ about the line $y = yr$ (axis of revolution) for $x \in (a,b)$.
In the second part we give the visualization of the solid of revolution in the 3-D space.
Some of the MATLAB commands used in the code are explained here.

`int(f(x),a,b)`	evaluates the integration of $f(x)$ between the limits a and b
`fx=matlabFunction(f)`	This converts Symbolic Function f to Anonymous Function *fx*
`[X,Y,Z]=cylinder(r);`	This returns the *x*-, *y*-, and *z*-coordinates of a cylinder using *r* to define a profile curve. cylinder treats each element in `r` as a radius at equally spaced heights along the unit height of the cylinder. The cylinder has 20 equally spaced points around its circumference.
`surf(X,Y,Z)`	This creates a three-dimensional surface plot. The function plots the values in matrix Z as heights above a grid in the *x*-*y* plane defined by X and Y.

```
%Evaluation of Volume of solid of revolution
clear all
clc
syms x
f(x)=sqrt(x);          % Given function
yr=1;                  % Axis of revolution y=yr
I=[0,4];               % Interval of integration
a=I(1);b=I(2);
vol=pi*int((f(x)-yr)^2,a,b);
disp('Volume of solid of revolution is: ');
disp(vol);
% Visualization if solid of revolution
fx=matlabFunction(f);
xv = linspace(a,b,101); % Creates 101 points from a to b
[X,Y,Z] = cylinder(fx(xv)-yr);
Z = a+Z.*(b-a);     % Extending the default unit height of the
cylinder profile to the interval of integration.
surf(Z,Y+yr,X)    % Plotting the solid of revolution about y=yr
hold on;
plot([a b],[yr yr],'-r','LineWidth',2); % Plotting the line y=yr
view(22,11);      % 3-D graph viewpoint specification
xlabel('X-axis');ylabel('Y-axis');zlabel('Z-axis');
```


Output

```
Volume of solid of revolution is:
(4*pi)/3
```



Exercise:

1. Find the area of the region bounded by the curve $y = x^2 - 2x$ and the line $y = x$.
2. To find the area of the region bounded by the curves $y^2 = x$, $y = x - 2$ in the first quadrant.
3. Find the area of the region bounded by the curves $x = y^3$ and $x = y^2$.
4. Find the volume of the solid generated by revolving about the $x-$axis the region bounded by the curve $y = \dfrac{4}{x^2+4}$, the $x-$axis, and the lines $x = 0$ and $x = 2$.

--*

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Experiment 2–B
Laplace transforms, Inverse Laplace transforms

The Laplace Transform of a function $f(t)$ is defined as

$$F(s) = L[f(t)] = \int_0^\infty e^{-st} f(t)dt$$, provided the integral exists.

MATLAB Syntax used:

Command	Purpose
`x=input('prompt')`	Displays the text in prompt and waits for the user to input a value and press the Enter key. The user can enter expressions and can use variables in the workspace.
`laplace(f)`	To find the Laplace transform of a scalar symbol f with default independent variable t. The default return is a function of s.
`laplace(f,w)`	Returns the Laplace transform of f in symbol w instead of the default s.
`laplace(f,x,w)`	Assumes f as a function of the symbolic variable x and returns the Laplace transform as a function of w.
`ilaplace(F)`	To find the inverse Laplace transform of the scalar symbolic object F with default independent variable s. The default return is a function of t.
`ilaplace(F,x)`	Returns the inverse Laplace transform of the function F as a function of x instead of the default t.
`ilaplace(F,w,x)`	Assumes F as a function of the symbolic variable w and returns the inverse Laplace transform of F as a function of x.
`heaviside(t-a)`	To input the heaviside's unit step function H(t−a).
`dirac(t-a)`	To input the dirac delta function δ(t−a).

Example 1. The following MATLAB code finds the Laplace transform of a function $f(t)$

```
clear all
clc
syms t
f=input('Enter the function of t: ');
F=laplace(f);
disp('Laplace transform of f(t) = ');
disp(F);
```

Input:
```
Enter the function of t: t^2
```
Output:
```
Laplace transform of f(t) =
2/s^3
```


Example 2. The following MATLAB code finds the Laplace transform of $f(t)$ in terms of w.

```
clear all
clc
syms t w
f=input('Enter the function of t: ');
F=laplace(f,w);
disp('Laplace transform of f(t) = ');
disp(F);
```

Input:
```
Enter the function of t: sin(t)
```

Output:
```
Laplace transform of f(t) =
1/(w^2 + 1)
```

Example 3. The following MATLAB code finds the Laplace transform of x^3e^{-3x} in terms of w.

```
clear all
clc
syms x w
f=input('Enter the function of x: ');
F=laplace(f,x,w);
disp('Laplace transform of f(t) = ');
disp(F);
```
Input:
```
Enter the function of x: x^3*exp(-3*x)
```

Output:
```
Laplace transform of f(t) =
6/(w + 3)^4
```

Example 4: The following MATLAB code computes the Laplace Transform of

$$f(t) = \begin{cases} t^2, 0 < t < 2 \\ t-1, 2 < t < 3 \\ 7, t > 3 \end{cases}$$

```
clear all
clc
syms t
f=input('Enter the function of t: ');
F=laplace(f);
F=simplify(F);
disp('Laplace transform of f(t) = ');
disp(F);
```

Input:
```
Enter the function of t: t^2*(heaviside(t)-heaviside(t-2))+
(t-1)*(heaviside(t-2)-heaviside(t-3))+7*heaviside(t-3)
```

Output:
```
Laplace transform of f(t) =
-(exp(-3*s)*(s-2*exp(3*s)+2*exp(s)+3*s^2*exp(s)+3*s*exp(s)-
5*s^2))/s^3
```

Example 5. The following MATLAB code computes the inverse Laplace transform of $F(s)$.

```
clear all
clc
syms s
F=input('Enter the function of s: ');
f=ilaplace(F);
disp('f(t) = ');
disp(f);
```

Input:

```
Enter the function of s: 6/(s^3+2*s^2-s-2)
```

Ouput:

```
f(t)=
2*exp(-2*t)-3*exp(-t)+exp(t)
```

Example 6. Write MATLAB commands to find the following:

(i) $L[\delta(t)]$ (ii) $L[\delta(t-a)]$ (iii) $L[\delta(t-a)\sin(t)]$

Solution:

(i)

```
syms t
F=laplace(dirac(t))
```

Output:

```
F =
1
```

(ii)

```
syms t a
F=laplace(dirac(t-a))
```

Output:

```
F =
piecewise(a < 0, 0, 0 <= a, exp(-a*s))
```

(iii)

```
syms t a
F=laplace(dirac(t-a)*sin(t))
```

Output:

```
F =
piecewise(a<0,0,0<=a,exp(-a*s)*sin(a))
```

Example 7. Write MATLAB commands to find (i) $L^{-1}\left[\frac{s}{s-a}\right]$ (ii) $L^{-1}\left[\frac{se^{-s}+ae^{-2s}}{s^2+a^2}\right]$

(i)

```
syms s a
f=ilaplace(s/(s-a))
```

Output

```
f=
dirac(t) + a*exp(a*t)
```


(ii)

```
syms s a
f= ilaplace((s*exp(-s)+a*exp(-2*s))/(s^2+a^2))
```

Output

```
f =
heaviside(t-1)*cos(a*(t-1))+sin(a*(t-2))*heaviside(t-2)
```

Exercise.

1. Find the Laplace transforms of the following functions:

(i) $f(t) = 1 + 2\sqrt{t} + \frac{3}{\sqrt{t}}$

(ii) $f(t) = \begin{cases} \sin t & ; \quad 0 \le t \le \pi \\ 0 & ; \quad \pi \le t \le 2\pi \end{cases}$

(iii) $f(t) = \sin^3 t$

(iv) $f(t) = \sin 2t \sin 3t$

(v) $f(t) = e^{-t} \sin^2 t$

(vi) $f(t) = \frac{\cos 2t - \cos 3t}{t}$

2. Find the inverse Laplace transforms of the following functions:

(i) $F(s) = \frac{6}{s^2 + 2s - 8}$

(ii) $F(s) = \frac{4s+5}{(s-1)^2(s+2)}$

(iii) $F(s) = \frac{s^2 + 2s - 4}{(s^2 + 2s + 5)(s^2 + 2s + 2)}$

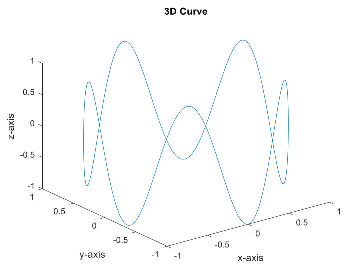
--*

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Experiment 3–A
Plotting 3D curves and surfaces, Taylor series of function of two variables

Commands	Description
`plot3(x,y,z)`	displays a three-dimensional plot of a set of data points
`comet3(x,y,z)`	displays a comet graph of the curve through the points [x(i),y(i),z(i)]
`ezplot3(funx,funy,funz)`	plots the spatial curve funx(t), funy(t), and funz(t) over the default domain 0 < t < 2π.
`ezplot3(fx,fy,fz,[tmin,tmax])`	plots the curve fx(t), fy(t), and fz(t) over the domain tmin < t < tmax
`[X,Y] = meshgrid(xgv,ygv)`	[X,Y] = meshgrid(xgv,ygv) replicates the grid vectors xgv and ygv to produce a full grid. This grid is represented by the output coordinate arrays X and Y. The output coordinate arrays X and Y contain copies of the grid vectors xgv and ygv respectively. The sizes of the output arrays are determined by the length of the grid vectors. For grid vectors xgv and ygv of length M and N respectively, X and Y will have N rows and M columns.
`surf(X,Y,Z)`	Uses Z for the colour data and surface height. X and Y are vectors or matrices defining the x and y components of a surface. If X and Y are vectors, length(X) = n andlength(Y) = m, where [m,n] = size(Z). In this case, the vertices of the surface faces are (X(*j*), Y(*i*), Z(*i,j*)) triples. To create X and Y matrices for arbitrary domains, use the meshgrid function.
`ezsurf(fun)`	creates a graph of fun(x,y) using the surf function. fun is plotted over the default domain: -2π < x < 2π, -2π < y < 2π.
`taylor(f,[x,y],[a,b],'order',n)`	To find the taylor series expansion of order n for the function $f(x,y)$ about the point (a,b).

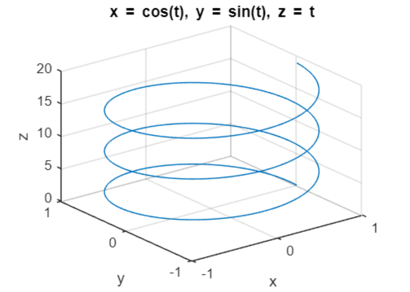
Example.1 The plotting of 3D parametric curve $x=\cos(t)$, $y=\sin(t)$, $z=\sin(5t)$ using `comet3` and `plot3` functions is given in the following code:

```
clear
clc
t=linspace(0,2*pi,500);
x=cos(t);
y=sin(t);
z=sin(5*t);
comet3(x,y,z);
plot3(x,y,z);
xlabel('x-axis');
ylabel('y-axis');
zlabel('z-axis');
title('3D Curve');
```



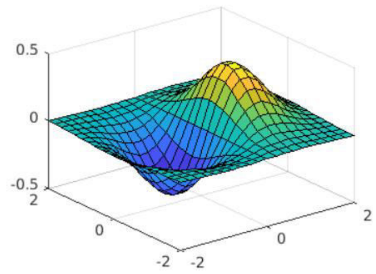
Example.2 The following code plots the helix defined by the parametric equations $x=\cos t$, $y=\sin t$, $z=t$.

```
clear
clc
syms t
x=cos(t);
y=sin(t);
z=t;
ezplot3(x,y,z,[0,6*pi])
```



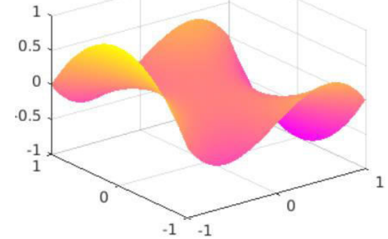
Example.3
The following code plots the surface $f(x,y)=e^{-x^2-y^2}$, $-2\le x,y\le 2$.

```
clear
clc
[x,y]=meshgrid(-2:.2:2);
f=x.*exp(-x.^2 - y.^2);
surf(x,y,f)
```



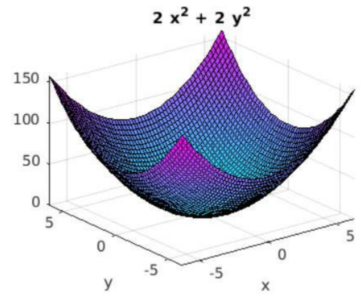
Example.4 In the following code the function $z=xy^2-x^3$ is plotted in the interval $-1\le x,y\le 1$

```
clear
clc
x=-1:.05:1;
y=-1:.05:1;
[x,y]=meshgrid(x,y);
z=x.*y.^2-x.^3
surf(x,y,z);
colormap spring
shading interp
```



Example.5 In the following code we used `ezsurf` to plot the function $f(x,y)=2(x^2+y^2)$

```
clear
clc
syms x y
f=2*(x^2+y^2)
ezsurf(f)
colormap cool
```



Taylor Series for a two variable functions:

Let $f(x,y)$ be a function of two variables x and y then the Taylor series expansion of f about the point (a,b) is given by

$$f(x,y) = f(a,b) + \left[(x-a)f_x(a,b) + (y-b)f_y(a,b)\right] + \frac{1}{2!}\left[(x-a)^2 f_{xx}(a,b) + 2(x-a)(y-b)f_{xy}(a,b) + (y-b)^2 f_{yy}(a,b)\right] + \cdots$$

Example 6. In the following code, Taylor series of the function $f(x,y) = e^x\, cosy$ is evaluated about the origin. Also the function along with its Taylor series is plotted in the corresponding figure.

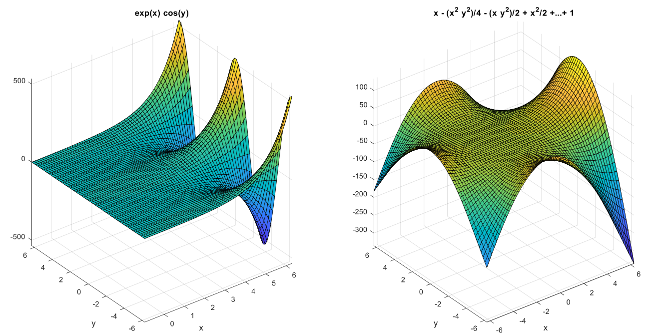
```
clear
clc
close all
syms x y
f=input('Enter the function f(x,y): ');
I=input('Enter the point[a,b] at which Taylor series is sought: ');
a=I(1);b=I(2);
n=input('Enter the order of series:');
tayser=taylor(f,[x,y],[a,b],'order',n)
subplot(1,2,1);
ezsurf(f); %Function plot
subplot(1,2,2);
ezsurf(tayser); % Taylors series of f
```

Input

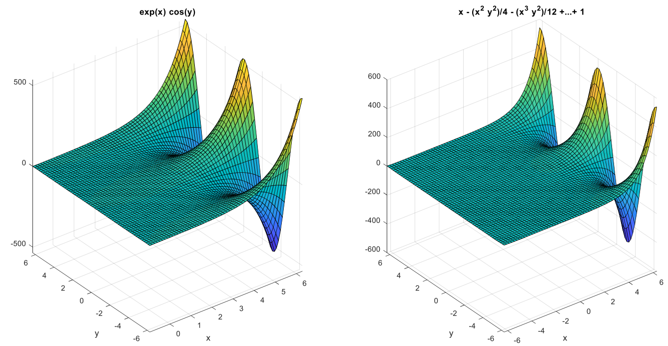
```
Enter the function f(x,y): exp(x)*cos(y)
Enter the point [a,b] around which Taylor series is sought: [0 0]
Enter the order of series:5
```

Output

```
tayser =
x^4/24 + x^3/6 - (x^2*y^2)/4 + x^2/2 - (x*y^2)/2 + x + y^4/24 - y^2/2 + 1
```



Note: The above program when executed with higher order (`n=20`), produces a better approximation than with order 5, the same is shown in the figure below.



Exercise:

1. Draw the surface of the function $f(x,y)=e^x+e^y$ using ezsurf.
2. Draw the 3-D plot for the function $f(t)=(t,\ t^2, t^3)$, where $0 \le t \le 100$.
3. Using 'surf' plot the surface $f(x,y) = x(x^2+y^2)$.
4. Expand $f(x,y) = e^x \ln(1+y)$ in terms of x and y upto the terms of 3rd degree using Taylor series.
5. Expand e^{xy} in Taylor series the neighbourhood of $(1,1)$.

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Experiment 3–B
Maxima and Minima of functions of two variables

Aim:
To find Maximum and Minimum values (Extreme values) of a function $f(x,y)$ using MATLAB.

Mathematical form:
Let $z = f(x,y)$ be the given function. Critical points are points in the xy – plane where the tangent plane is horizontal. The tangent plane is horizontal, if its normal vector points in the z – direction. Hence, critical points are solutions of the equations: $f_x(x,y) = 0$ and $f_y(a,b) = 0$.

<u>Procedure for finding the maximum or minimum values of *f*(*x*,*y*):</u>

(1) For the given function *f*(*x*,*y*) find $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}$ and equate it to zero and solve them to find the roots (x_1, y_1), (x_2, y_2),.... These points may be maximum or minimum points.

(2) Find the values $r = \frac{\partial^2 f}{\partial x^2}$, $s = \frac{\partial^2 f}{\partial x \partial y}$, $t = \frac{\partial^2 f}{\partial y^2}$ at these points.

(3) (a) If $rt - s^2 > 0$ and $r < 0$ at a certain point, then the function is maximum at that point.
(b)If $rt - s^2 > 0$ and $r > 0$ at a certain point , then the function is minimum at that point.
(c) If $rt - s^2 < 0$ for a certain point, then the function is neither maximum nor minimum at that point. This point is known as saddle point.
(d) If $rt - s^2 = 0$ at a certain point, then nothing can be said whether the function is maximum or minimum at that point. In this case further investigation are required.

MATLAB Syntax used:

`diff`	`diff(expr)` differentiates a symbolic expression expr with respect to its free variable as determined by symvar.
`solve`	Symbolic solution of algebraic equations, The input to solve can be either symbolic expressions or strings
`size`	Dimensions of data and model objects and to access a specific size output.
`figure`	Create figure graphics object, Figure objects are the individual windows on the screen in which the MATLAB software displays graphical output
`double`	Convert to double precision, `double(x)` returns the double-precision value for `X`. If `X` is already a double-precision array, double has no effect.
`sprintf`	Format data into string. It applies the *format* to all elements of array *A* and any additional array arguments in column order, and returns the results to string *str*. `sprintf('%f',var)` is used to format the floating-point number `var` into string.
`solve`	Symbolic solution of algebraic equations, The input to solve can be either symbolic expressions or strings.
`fsurf`	`fsurf(f)` creates a surface plot of the function z = f(x,y) over the default interval `[-5, 5]` for x and y.
`plot3`	The `plot3` function displays a three-dimensional plot of a set of data points.

MATLAB code:

```
clc
clear
syms x y
f(x,y)=input('Enter the function f(x,y):');
p=diff(f,x); q=diff(f,y);
[ax,ay]=solve(p,q);
ax=double(ax);ay=double(ay);
r=diff(p,x);  s=diff(p,y); t=diff(q,y);D=r*t-s^2;
figure
fsurf(f);
legstr={'Function Plot'};% for Legend
for i=1:size(ax)
T1=D(ax(i),ay(i));
T2=r(ax(i),ay(i));
T3=f(ax(i),ay(i));
if(double(T1)==0)
sprintf('At (%f,%f) further investigation is required',ax(i),ay(i))
legstr=[legstr,{'Case of Further investigation'}];
mkr='ko';
elseif (double(T1)<0)
sprintf('The point (%f,%f) is a saddle point', ax(i),ay(i))
legstr=[legstr,{'Saddle Point'}]; % updating Legend
mkr='bv'; % marker
else
if (double(T2) < 0)
sprintf('The maximum value of the function is f(%f,%f)=%f', ax(i),ay(i), T3)
legstr=[legstr,{'Maximum value of the function'}];% updating Legend
mkr='g+';% marker
else
sprintf('The minimum value of the function is f(%f,%f)=%f', ax(i),ay(i), T3)
legstr=[legstr,{'Minimum value of the function'}];% updating Legend
mkr='r*'; % marker
end
end
hold on
plot3(ax(i),ay(i),T3,mkr,'Linewidth',4);
end
legend(legstr,'Location','Best');
```

Example 1. Obtain the maximum and minimum values of $f(x,y) = 2(x^2 - y^2) - x^4 + y^4$

Solution:

S.No.	Critical Points	*r*	*D=rt–s*^{2}	Remarks
1	(0,0)	4	$-16 < 0$	Saddle Point
2	(0,1)	4	32	Minimum
3	(0, –1)	4	32	Minimum
4	(1,0)	-8	32	Maximum
5	(1,1)	-8	$-64 < 0$	Saddle Point
6	(1, –1)	-8	$-64 < 0$	Saddle Point
7	(–1,0)	-8	32	Maximum
8	(–1,1)	-8	$-64 < 0$	Saddle Point
9	(–1, –1)	-8	$-64 < 0$	Saddle Point

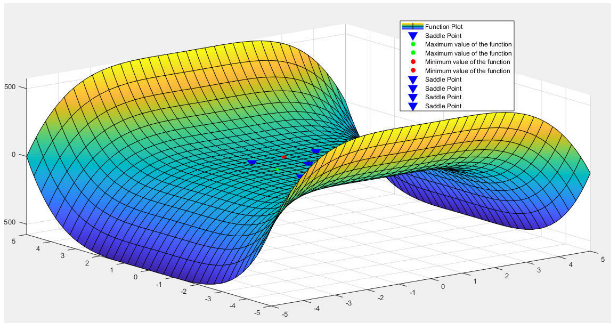
The Minimum value of *f*(*x*,*y*) is –1 at (0,1) & (0, –1) and the Maximum value for *f*(*x*,*y*) is +1 at (1,0) & (–1,0)

Input:

```
Enter the function f(x,y):2*(x^2-y^2)-x^4+y^4
```

Out Put:

```
 ans =
'The point (0.000000,0.000000) is a saddle point'
ans =
'The maximum value of the function is f(-1.000000,0.000000)=1.000000'
ans =
 'The maximum value of the function is f(1.000000,0.000000)=1.000000'
ans =
'The minimum value of the function is f(0.000000,-1.000000)=-1.000000'
ans =
 'The minimum value of the function is f(0.000000,1.000000)=-1.000000'
ans =
    'The point (-1.000000,-1.000000) is a saddle point'
ans =
    'The point (1.000000,-1.000000) is a saddle point'
ans =
    'The point (-1.000000,1.000000) is a saddle point'
ans =
    'The point (1.000000,1.000000) is a saddle point'
```



Example. 2 Four small towns in a rural area wish to pool their resources to build a television station. If the towns are located at the points $(-5,0)$, $(1,7)$, $(9,0)$ and $(0,-8)$ on a rectangular map grid, where units are in miles, at what point $S(x, y)$ should the station be located to minimize the sum of the distances from the towns?

Solution: Let $S(x,y)$ be the location where the television station is to be set up.

The location of the towns are $A(-5,0)$, $B(1,7)$, $C(9,0)$ and $D(0,-8)$.

The point $S(x,y)$ where the sum of the distances from the above points is to be minimized is the same point that minimizes the sum of the squares of the distances; namely,

$$S(x,y) = \left[(x+5)^2 + y^2\right] + \left[(x-1)^2 + (y-7)^2\right] + \left[(x-9)^2 + y^2\right] + \left[x^2 + (y+8)^2\right]$$

$S_x = 2(x+5) + 2(x-1) + 2(x-9) + 2x$.

$S_y = 2y + 2(y-7) + 2y + 2(y+8)$.

Then $S_x = 0 \Rightarrow x = 5/4$ and $S_y = 0 \Rightarrow y = -1/4$.

$r = S_{xx} = 8 > 0$, $s = S_{xy} = 0$ and $t = S_{yy} = 8 > 0$.

Hence $rt - s^2 > 0$, $r > 0$ at $(5/4, -1/4)$.

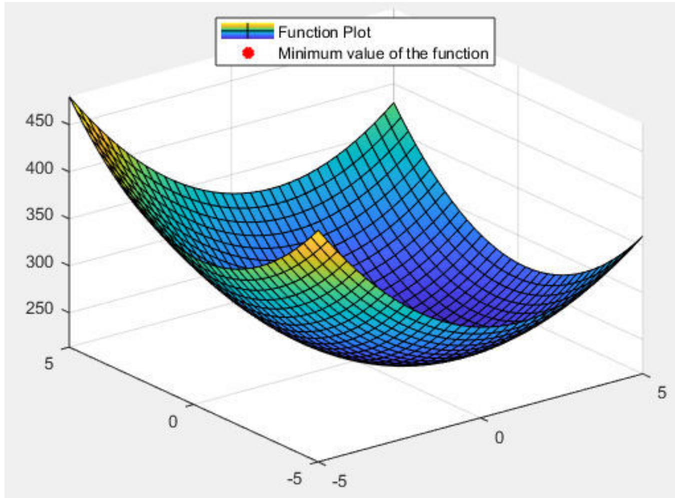
Therefore, the television station can be set up at the location $S(5/4, -1/4)$ on the rectangular map grid such that the distance from S to each of the towns is a minimum.

Input

```
Enter the function f(x,y):
(x+5)^2+y^2+(x-1)^2+(y-7)^2+(x-9)^2+y^2+x^2+(y+8)^2
```

Output

```
ans =
    'The minimum value of the function is f(1.250000,-0.250000)=
                                                     213.500000'
```



Exercise

Find the maxima and minima for the following functions

1. $f(x,y) = x^4 + y^4 - x^2 - y^2 + 1$.
2. $f(x,y) = x^3 + 3xy^2 - 15x^2 - 15y^2 + 72x$.

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Experiment 4–A
Double Integrals and change of order of integration

In this experiment, we consider a continuous function f such that $f(x,y)\geq 0$ for all (x,y) in a region R in the xy – plane, then the volume of the solid region that lies above R and below the graph of f is defined as the double integral $V=\iint_R f(x,y)dA$, where R is the region bounded by the curves $y=\phi_1(x)$ and $y=\phi_2(x)$ between $x=a$ and $x=b$.

In this case, the inner integration is with respect to y and outer integration is with respect to x. Hence

$$V=\iint_R f(x,y)dA=\int_{x=a}^{b}[\int_{y=\phi_1(x)}^{\phi_2(x)} f(x,y)dy]dx$$

MATLAB Syntax

`int(int(f, y, phi1, phi2),x,a,b)` where y is the inner variable, x is the outer variable.

When R is a region bounded by the curves $x=\psi_1(y)$ and $x=\psi_2(y)$ between $y=c$ and $y=d$, i.e., the inner integration is with respect to x and outer integration is with respect to y. Then $V=\iint_R f(x,y)dA=\int_{y=c}^{d}[\int_{x=\psi_1(y)}^{\psi_2(y)} f(x,y)dx]dy$

MATLAB Syntax

`int(int(f, x, psi1, psi2),y,c,d)` where x is the inner variable, y is the outer variable.

Supporting files required:

To visualize the surfaces two additional m-files viz., `viewSolid.m, viewSolidone.m` are required. These files are to be included in the current working directory before execution. Students are advised to upload these files (`viewSolid.m and viewSolidone.m`) to their MATLAB drive. These supporting files **should not be edited**.

Download `viewSolid.m` from the following link:
https://drive.google.com/file/d/1qEsq7VCgrmI60GI-C0bMY6kl8yRWBznk/view?usp=sharing

Download `viewSolidone.m` from the following link:
https://drive.google.com/file/d/1H1cJOfJArmUujQNVxeSeuGfBJejtbGgJ/view?usp=sharing

Syntax for visualization of the surfaces:

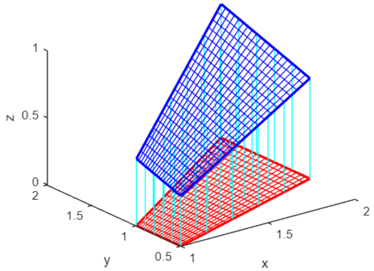
```
viewSolid(z,0+0*x+0*y,f,y,phi1,phi2,x,a,b)

viewSolidone (z,0+0*x+0*y,f,x,psi1,psi2,y,c,d)
```

Example. 1 To find $\int_1^2\int_{x/2}^{x}\frac{x+y}{4}dydx$.

```
syms x y z
int(int((x+y)/4,y,x/2,x),x,1,2)
viewSolid(z,0+0*x+0*y,(x+y)/4,y,x/2,x,x,1,2)
```

Output
```
ans =
49/96
```

In this figure the required volume is above the plane z=0 (shown in red) and above the surface $z=\frac{x+y}{4}$ (shown in green) .



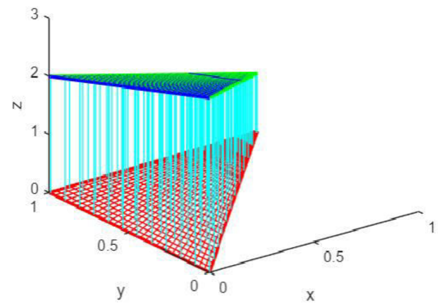
Example. 2 To find the volume of the prism whose base is the triangle in the xy – plane bounded by the x – axis and the lines $y=x$ and $x=1$ and whose top lies in the plane $z=f(x,y)=3-x-y$. The limits of integration here are $y=0$ to 1 while $x=y$ to 1.

Hence $\iint_R(3-x-y)dA=\int_0^1\int_y^1(3-x-y)dxdy$

```
syms x y z
int(int(3-x-y,x,y,1),y,0,1)
viewSolidone(z,0+0*x+0*y,3-x-y,x,y,1,y,0,1)
```

Output:
```
ans =
1
```

In this figure the triangular region on the *xy* plane is shown in red, while the plane surface *z*=3–*x*–*y* above the *xy* plane is shown in green.



Example 3 Evaluate the integral $\int_0^2\int_{x^2}^{2x}(4x+2)dydx$ by changing the order of integration.

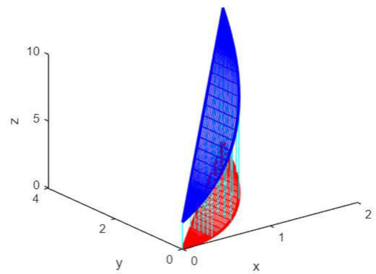
As per the given limits of integration $x=0$ to 2 while $y=x^2$ to $2x$.

MATLAB Code:
```
syms x y z
int(int((4*x+2),y,x^2,2*x),x,0,2)
viewSolid(z,0+0*x+0*y, 4*x+2,y,x^2,2*x,x,0,2)
```

Output
```
ans =

8
```

By changing the order of integration, the limits are $y=0$ to 4 while $x=\frac{y}{2}$ to $\sqrt{y}$.

```
int(int(4*x+2,x,y/2,sqrt(y)),y,0,4)

ans =

8
```

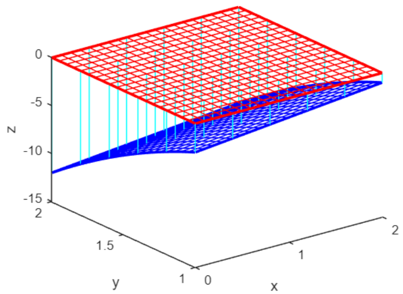


Example 4: Evaluate $\iint_R (x-3y^2)dA$ where $R=\{(x,y)\,|\,0\le x\le 2, 1\le y\le 2\}$

```
clc
clear all
syms x y z
viewSolid(z, 0+0*x+0*y, x-3*y^2,y,1+0*x, 2+0*x,x,0,2)
int(int(x-3*y^2,y,1,2),x,0,2)
```

Output:
```
>> ans
     -12
```

In this figure the required volume is below the plane $z=0$ (shown in red) and above the surface $z=x-3y^2$ (shown in blue). The reason why the answer is negative is that the surface $z=x-3y^2$ is below $z=0$ for the given domain of integration.

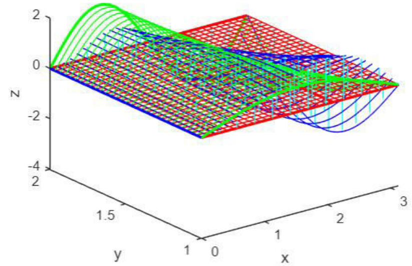


Example 5: Evaluate $\iint_R y\sin(xy)\ dA$ where $R=[1,2]\times[0,\pi]$

MATLAB Code:

```
syms x y z
viewSolidone(z, 0+0*x+0*y, y*sin(x*y),x,1+0*y, 2+0*y,y,0,pi)
int(int(y*sin(x*y),x,1,2),y,0,pi)
```

Output:
```
>>  ans
    0
```



For a function $f(x,y)$ that takes on both positive and negative values $\iint_R f(x,y)dA$ is a difference of volumes V_1-V_2, V_1 is the volume above R and below the graph of f and V_2 is the volume below R and above the graph. The integral in this example is 0 means $V_1=V_2$

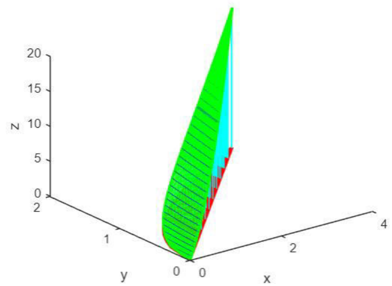
Example 6: Find the volume of the solid that lies under the paraboloid $z=x^2+y^2$ and above the region D in the xy-plane bounded by the lines $y=2x$ and the parabola $y=x^2$

MATLAB Code:

```
syms x y z
viewSolidone(z, 0+0*x+0*y, x^2+y^2,x,y/2,sqrt(y),y,0,4)
int(int(x^2+y^2,x,y/2,sqrt(y)),y,0,4)
```

Output:

```
>>  ans
    216/35
```



Exercise:

1. Find the volume of the solid S that is bounded by the elliptic paraboloid $x^2+2y^2+z=16$, the planes $x=2$ *and* $y=2$, and the three coordinate planes.
2. Evaluate $\iint_R \sin x\cos y\,dA$ where $R=[0,\pi/2]\times[0,\pi/2]$

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Experiment 4–B
Triple Integrals

Triple integrals enable us to solve more general problems such as to calculate the volumes of three – dimensional shapes, the masses and moments of solids of varying density, and the average value of a function over a three –dimensional region.
The triple integral of $f(x,y,z)$ over the region D is given by

$$\iiint_D f(x,y,z)dV$$

where the region D is bounded by the surfaces $x=a$, $x=b$, $y=\psi_1(x)$ to $y=\psi_2(x)$, $z=\phi_1(x,y)$ to $z=\phi_2(x,y)$.
Hence

$$\iiint_D f(x,y,z)dV=\int_a^b\int_{\psi_1(x)}^{\psi_2(x)}\int_{\phi_1(x,y)}^{\phi_2(x,y)} f(x,y,z)dzdydx.$$

Similarly when the region D is bounded by the surfaces $y=c$, $y=d$, $x=\psi_1(y)$ to $x=\psi_2(y)$, $z=\phi_1(x,y)$ to $z=\phi_2(x,y)$.
Hence

$$\iiint_D f(x,y,z)dV=\int_a^b\int_{\psi_1(y)}^{\psi_2(y)}\int_{\phi_1(x,y)}^{\phi_2(x,y)} f(x,y,z)dzdxdy.$$

Volume using Triple Integral

The volume of a closed, bounded region D in space is given by

$$V=\iiint_D dV$$

Syntax for evaluation of triple integral:

```
int(int(int(f,z,za,zb),y,ya,yb),x,xa,xb)
```

or

```
I=int(int(int(f,z,za,zb),x,xa,xb),y,ya,yb)
```

Syntax for visualization of region bounded by the limits of integration:

```
viewSolid(z,za,zb,y,ya,yb,x,xa,xb)
```

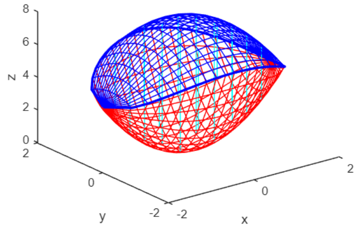
```
viewSolidone(z,za,zb,xa,xb,y,ya,yb)
```


Example 1. Find the volume of the region D enclosed by the surfaces $z=x^2+3y^2$ and $z=8-x^2-y^2$.

```
clear
clc
syms x y z
xa=-2;
xb=2;
ya=-sqrt(2-x^2/2);
yb=sqrt(2-x^2/2);
za=x^2+3*y^2;
zb=8-x^2-y^2;
I=int(int(int(1+0*z,z,za,zb),y,ya,yb),x,xa,xb)
viewSolid(z,za,zb,y,ya,yb,x,xa,xb)
```

Output

```
I =
8*pi*2^(1/2)
```

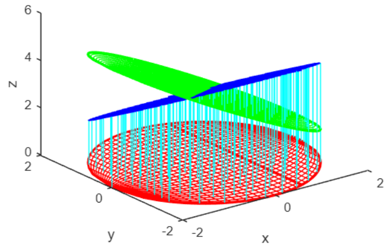


Example 2. Find the volume of the region cut from the cylinder $x^2+y^2=4$ by the plane $z=0$ and the plane $x+z=3$.
The limits of integration are $z=0$ to $3-x$, $x=-\sqrt{4-y}$ to $\sqrt{4-y}$, $y=-2$ to 2.

```
clear
clc
syms x y z
ya=-2;
yb=2;
xa=-sqrt(4-y^2);
xb=sqrt(4-y^2);
za=0+0*x+0*y;
zb=3-x-0*y;
I=int(int(int(1+0*z,z,za,zb),x,xa,xb),y,ya,yb)
viewSolidone(z,za,zb,x,xa,xb,y,ya,yb)
```

Output

```
I =
12*pi
```



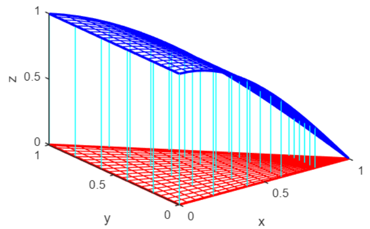
Example 3. Find the volume of the region in the first octant bounded by the coordinate planes, the plane $y = 1 - x$, and the surface $z = \cos(\pi x/2)$, $0 \le x \le 1$.

The limits of integration are $z = 0$ to $\cos(\pi x/2)$, $y = 0$ to $1 - x$, $x = 0$ to 1.

```
clear
clc
syms x y z real
xa=0;
xb=1;
ya=0+0*x;
yb=1-x;
za=0*x+0*y;
zb=cos(pi*x/2)+0*y;
I=int(int(int(1+0*z,z,za,zb),y,ya,yb),x,xa,xb)
viewSolid(z,za,zb,y,ya,yb,x,xa,xb)
```

Output.

```
I =

4/pi^2
```



Exercise.

1. Find the volume of the region bounded between the planes $x + y + 2z = 2$ and $2x + 2y + z = 4$ in the first octant.
2. Find the volume of the region cut from the solid elliptical cylinder $x^2 + 4y^2 \le 4$ by the $xy-$ plane and the plane $z = x + 2$.
3. The finite region bounded by the planes $z = x$, $x + z = 8$, $z = y$, $y = 8$ and $z = 0$.

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Experiment 5–A
Divergence, Curl and Gradient and visualization of vector field

Aim:

To write Matlab codes to visualize the vector field of 2-Dimensions as well as 3-Dimensions.
To find and visualize the gradient of scalar function, divergence and curl of a vector function.

Gradient vector of a scalar function $f(x, y, z)$

The vector function ∇f is defined as the gradient of the scalar function f and written as *grad f.*

$$grad\ f = \nabla f = \frac{\partial f}{\partial x}\hat{i} + \frac{\partial f}{\partial y}\hat{j} + \frac{\partial f}{\partial z}\hat{k}.$$

Divergence of a vector $\vec{F}$

Divergence of a continuously differentiable vector point function $\vec{F}$ is denoted by $div\,\vec{F}$

$$div\,\vec{F} = \nabla \bullet \vec{F} = \left(\frac{\partial}{\partial x}\hat{i} + \frac{\partial}{\partial y}\hat{j} + \frac{\partial}{\partial z}\hat{k}\right) \bullet \left(F_1\hat{i} + F_2\hat{j} + F_3\hat{k}\right) = \frac{\partial f_1}{\partial x} + \frac{\partial f_2}{\partial y} + \frac{\partial f_3}{\partial z}$$

Curl of a vector $\vec{F}$

Curl of a continuously differentiable vector point function $\vec{F}$ is denoted by $curl\,\vec{F}$

$$curl\,\vec{F} = \nabla \times \vec{F} = \left(\frac{\partial}{\partial x}\hat{i} + \frac{\partial}{\partial y}\hat{j} + \frac{\partial}{\partial z}\hat{k}\right) \times \left(F_1\hat{i} + F_2\hat{j} + F_3\hat{k}\right) = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ F_1 & F_2 & F_3 \end{vmatrix}.$$

MATLAB Syntax used:

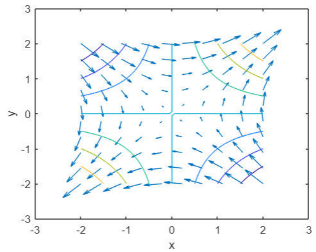
`quiver(x,y,u,v)`	Displays velocity vectors as arrows with components (u,v) at the points (x,y)
`quiver3(x,y,z,u,v,w)`	Plots vectors with components (u,v,w) at the points (x,y,z)
`gradient(f,v)`	Finds the gradient vector of scalar function `f` with respect to vector `v` in Cartesian coordinates.
`divergence(f,v)`	Finds the divergence of vector field `f` with respect to vector `v` in Cartesian coordinates.
`curl(V,X)`	Finds the curl of vector field `f` with respect to vector v in Cartesian coordinates.
`pcolor(x,y,C)`	When x,y and C are matrices of the same size, pcolor(x,y,C) plots the colored patches of vertices (x(i,j), y(i,j)) and color C(i,j).

Example 1: Find the Gradient of the function $f = 2xy$.

```
clear
clc
syms x y
f=input('Enter the function f(x,y):');
grad=gradient(f,[x,y])
P(x,y)=grad(1);Q(x,y)=grad(2);
x=linspace(-2,2,10);y=x;
[X,Y]=meshgrid(x,y);
U=P(X,Y); V=Q(X,Y);
quiver(X,Y,U,V,1)
axis on
xlabel('x'); ylabel('y')
hold on
fcontour(f,[-2,2])
```

Input:
```
Enter the function f(x,y):2*x*y
```

Output:
```
grad =
 2*y
 2*x
```



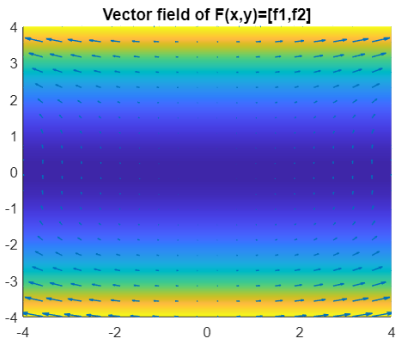
Example 2: Find the divergence of the vector field $\vec{F} = xy^2\hat{i} + x^2\hat{j}$ and visualize it.

```
clear
clc
syms x y
f=input('Enter the 2D vector function in the form [f1,f2]:');
div(x,y)=divergence(f,[x,y])
P(x,y)=f(1);Q(x,y)=f(2);
x=linspace(-4,4,20);y=x;
[X,Y]=meshgrid(x,y);
U=P(X,Y);V=Q(X,Y);
figure
pcolor(X,Y,div(X,Y));
shading interp
hold on;
quiver(X,Y,U,V,1)
axis on
hold off;
title('Vector field of F(x,y)=[f1,f2]');
```

Intput:
```
Enter the 2D vector function in the form [f1,f2]:
[x*y^2, x^2]
```

Output:

```
div(x,y)=
y^2
```

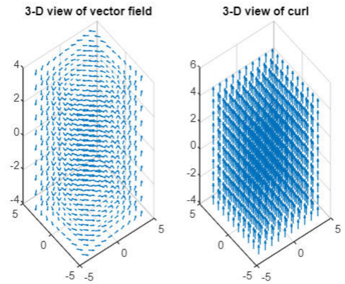


Example 3. Find and visualize the curl of a vector function $\overline{F} = -yi + xj$.

```
clear
clc
syms x y z
f=input('Enter the 3D vector function in the form [f1,f2,f3]:');
P(x,y,z)=f(1);Q(x,y,z)=f(2);R(x,y,z)=f(3); %Components of vector f
crl=curl(f,[x,y,z]) %Calculating curl
C1(x,y,z)=crl(1);C2(x,y,z)=crl(2);C3(x,y,z)=crl(3);%Components of curl(f)
x=linspace(-4,4,10);y=x;z=x;
[X,Y,Z]=meshgrid(x,y,z);
U=P(X,Y,Z);V=Q(X,Y,Z);W=R(X,Y,Z);
CR1=C1(X,Y,Z);CR2=C2(X,Y,Z);CR3=C3(X,Y,Z);
figure;
subplot(1,2,1);
quiver3(X,Y,Z,U,V,W);
title('3-D view of vector field');
subplot(1,2,2);
quiver3(X,Y,Z,CR1,CR2,CR3);
title('3-D view of curl');
```

Input:
```
Enter the 3D vector function in the form [f1,f2,f3]:
[-y,x,0]
```

Output
```
crl =
0
0
2
```



Exercise:

1. Draw the two dimensional vector field for the vector $2xi + 3yj$.
2. Find the Gradient of the function $f = x^2y^3 - 4y$.
3. Find the divergence of a vector field $f = [xy, x^2]$.
4. Visualize the curl of a vector function $f = [yz, 3zx, z]$.

Department of Mathematics
School of Advanced Sciences
MAT 1011 – Calculus for Engineers (MATLAB)
Experiment 5–B
Line integral and work done

Aim:

To write MATLAB codes to find the work done by a force $\vec{F}$ and visualize the force field with the path.

Mathematical form:

Let the given function be $\vec{F} = F_1(x,y,z)\hat{i} + F_2(x,y,z)\hat{j} + F_3(x,y,z)\hat{k}$, where (x,y,z) given in parametric form $\bar{r} = x(t)\hat{i} + y(t)\hat{j} + z(t)\hat{k}$, $a < t < b$. Then

$$\int_a^b \bar{F}\cdot d\bar{r} = \int_a^b \left(\bar{F}(r(t))\cdot\frac{d\bar{r}}{dt}\right)dt$$

Example 1. Finding the line integral $\int_C \bar{F}\cdot d\bar{r}$ along the given curve C given by $x(t) = t + \sin(\pi t/2)$, $y(t) = t + \cos(\pi t/2)$, $0 \le t \le 1$, where $\bar{F} = xy^2\hat{i} + x^2y\hat{j}$.

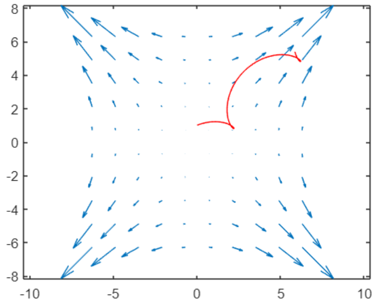
```
clc
clear all
syms x y t
f=input('Enter the components of 2D vector function [u,v] ');
r=input('Enter x,y in parametric form');
I=input('Enter the limits of integration for t in the form [a,b]');
a=I(1);b=I(2);
dr=diff(r,t);
F=subs(f,{x,y},r);
Fdr=sum(F.*dr);
I=int(Fdr,t,a,b)
P(x,y)=f(1);Q(x,y)=f(2);
x1=linspace(-2*pi,2*pi,10); y1=x1;
[X,Y] = meshgrid(x1,y1);
U=P(X,Y); V=Q(X,Y);
quiver(X,Y,U,V,1.5)
hold on
t1=linspace(0,2*pi);
x=subs(r(1),t1);y=subs(r(2),t1);
plot(x,y,'r')
```

Input

```
Enter the components of 2D vector function [u,v]:
[x*y^2 x^2*y]
Enter x(t) and y(t) in parametric form:
[t+sin((pi*t)/2) t+cos((pi*t)/2)]
Enter the limits of integration for t in the form [a,b]:
[0,1]
```


Output

```
I =
2
```



Example 2. Evaluate $\int_C \bar{F}\cdot d\bar{r}$ along the given curve C given by $r:[t,t^2,t^3]$, $0 \le t \le 1$, where $\bar{F} = xy\hat{i} + yz\hat{j} + zx\hat{k}$.

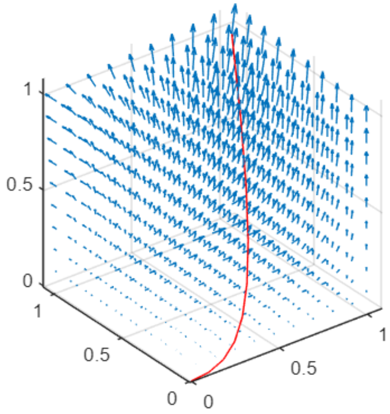
```
clc
clear all
syms x y z t
f=input('Enter the components of 3D vector function [u,v,w] ');
r=input('Enter x,y,z in parametric form');
I=input('Enter the limits of integration for t in the form [a,b]');
a=I(1);b=I(2);
dr=diff(r,t);
F=subs(f,{x,y,z},r);
Fdr=sum(F.*dr);
I=int(Fdr,t,a,b)
P(x,y,z)=f(1);Q(x,y,z)=f(2); R(x,y,z)=f(3);
x1=linspace(0,1,10); y1=x1; z1=x1;
[X,Y,Z] = meshgrid(x1,y1,z1);
U=P(X,Y,Z); V=Q(X,Y,Z); W=R(X,Y,Z);
quiver3(X,Y,Z,U,V,W,1.5)
hold on
t1=linspace(0,1,10);
x=subs(r(1),t1);y=subs(r(2),t1);z=subs(r(3),t1);
plot3(x,y,z,'r')
```

Input

```
Enter the components of 2D vector function [u,v,w]
[x*y y*z z*x]
Enter r in parametric form [x(t) y(t) z(t)]
[t t^2 t^3]
Enter the limits of integration for t in the form [a,b]
[0,1]
```

Output

```
27/28
```



Exercise:

1) Find the work done for the force $\vec{F}(x,y,z) = yz\vec{i} + xz\,\vec{j} + (xy+2z)\,\vec{k}$ along the line segment from (1,0,-2) to (4,6,3).
2) Find the work done for the force $\vec{F}(x,y,) = x^2\vec{i} + y^2\,\vec{j}$ along the arc of the parabola $y = 2x^2$ from (-1,2) to (2,8).