

# INTRODUCTION TO MATLAB

## 1. INSTALLATION

The National University of Singapore has a Total Academic Headcount licence for MATLAB. Students may use it for academic, research, and learning. The license allows students to install MATLAB on personally-owned computers.

1. If you are NOT using NUS network, you are required to use nVPN:

- <https://webvpn.nus.edu.sg/>

Please sign in with your NUSNET ID in the format: **nusstu\nusnetid** and password.

If you are using NUS network, please proceed to Step 2.

2. Click the link <https://sm05.stf.nus.edu.sg/studentmatlab/> to download MATLAB. It is required to sign in with your NUSNET ID in the format: **nusstu\nusnetid** and password. Then follow the instructions.

3. Create a MathWorks account using your NUS email address.

- (i) <https://www.mathworks.com/mwaccount/register>

- ◊ Email address: Your NUS email account (e.g., e0012345@u.nus.edu)

- ◊ Location: Singapore

- ◊ How will you use MathWorks software? Student use

- ◊ Are you at least 13 years or older? Yes

- (ii) You will receive an email from [service@mathworks.com](mailto:service@mathworks.com) with title “Verify Email Address”. Click the link in the email to verify your account.

- (iii) Finish creating your profile. Then you should be able to see the following information:

- Your account has been created and license 40707750 has been associated with your account.

4. Click the **Download** button to download and run the installer.

- (i) When prompted, log in with your MathWorks Account (your NUS email account).

- (ii) Select your licence (40707750, Student, Academic — Total Headcount).

- (iii) Choose installation folder.

- (iv) Select products to install.

## 2. BASIC OPERATIONS

First of all, we learn some basic operations in MATLAB.

MATLAB environment behaves like a super-complex calculator. You can enter the commands at the `>>` command prompt. The answer appears by pressing Enter.

We can use the following arithmetic operators: addition `+`, subtraction `-`, multiplication `*`, division `/`, exponentiation `^`.

For example, to add two numbers 123 and 321, we simply type

```
>> 123 + 321
```

and then press Enter:

```
ans = 444
```

Similarly, we can apply other operators to use MATLAB as an ordinary calculator:

```
>> 123 - 321
```

```
ans = -198
```

```
>> 123 * 321
```

```
ans = 39483
```

```
>> 123 / 321
```

```
ans = 0.3832
```

```
>> 123 ^ 3
```

```
ans = 1860867
```

You may add a semicolon `;` at the end of the statement; then MATLAB will hide the output. For example,

```
>> a = 3;
```

```
>> a ^ 2
```

```
ans = 9
```

Note that the symbol `a` is now defined as 3. We may remove it from the memory by using `clear a` or remove all variables from the memory by `clear`. If we want to clear the command window, we can use `clc`.

## 3. BASIC FUNCTIONS

The function `sqrt(x)` computes the principle square root of the number  $x$ . For example,

```
>> sqrt(2)
```

```
ans = 1.4142
```

By default, MATLAB displays four decimal digits to its answers. But we can change the format for numeric display. (The percent symbol `%` is used for indicating a comment line.)

```
>> format long      % 16 decimal digits
>> sqrt(2)
ans = 1.414213562373095
>> format rat       % rational approximation
>> sqrt(2)
ans = 1393/985
>> format short     % four decimal digits (default)
>> sqrt(2)
ans = 1.4142
```

We can also use the following functions in MATLAB:

- (i) Trigonometric functions: `sin(x)`, `cos(x)`, `tan(x)`, `cot(x)`, `sec(x)`, `csc(x)`.
- (ii) Inverse trigonometric functions: `asin(x)`, `acos(x)`, `atan(x)`, `acot(x)`.
- (iii) Exponential and logarithmic functions: `exp(x)`, `log(x)` (base  $e$ ), `log10(x)` (base 10).

For trigonometric and inverse trigonometric functions, the angles are measured in radian ( $\pi$  radian =  $180^\circ$ ), and the constant  $\pi = 3.1415926\cdots$  is defined by `pi` (MATLAB is case-sensitive).

```
>> sin(pi/4)
ans = 0.7071
>> atan(1)
ans = 0.7854
```

#### 4. SOLVING BASIC ALGEBRAIC EQUATIONS

The command `solve` is used for solving algebraic equations. We shall

- (i) Use `syms` to declare the variables.
- (ii) Use `solve(equations, variables)` to solve the specific equations.

Note that `=` is used to assign a value to a variable; and `==` shall be used for equality.

**4.1. Single Variable Equations.** For example,  $x^2 + x - 1 = 0$ .

```
>> syms x;
>> solve(x^2 + x - 1 == 0, x)
```

```
ans = -5^(1/2)/2 - 1/2
      5^(1/2)/2 - 1/2
```

We may use `vpa(x)` or `vpa(x,d)` to evaluate each element of the symbolic input to  $d$  digits (the default is 32 digits).

```
>> vpa(ans,10)
ans = -1.618033989
      0.6180339887
```

Sometimes the exact solutions cannot be specifically displayed. For example,

$$x^4 - 7x^3 + 3x^2 - 5x + 9 = 0.$$

```
>> solve(x^4 - 7*x^3 + 3*x^2 - 5*x + 9 == 0, x)
ans = root(z^4 - 7*z^3 + 3*z^2 - 5*z + 9, z, 1)
      root(z^4 - 7*z^3 + 3*z^2 - 5*z + 9, z, 2)
      root(z^4 - 7*z^3 + 3*z^2 - 5*z + 9, z, 3)
      root(z^4 - 7*z^3 + 3*z^2 - 5*z + 9, z, 4)
```

Then evaluate using floating points to obtain the decimal expressions of the four roots:

```
>> vpa(ans,10)
ans = 1.059780463
      - 0.3450883978 - 1.077836295i
      - 0.3450883978 + 1.077836295i
      6.630396332
```

Here the symbol `i` refers to the *imaginary unit*  $i$  where  $i = e^{\pi i/2}$  such that  $i^2 = -1$ .

Alternatively, one may use `vpasolve` to get the decimal expression directly:

```
>> vpasolve(x^4 - 7*x^3 + 3*x^2 - 5*x + 9 == 0, x)
ans = 1.0597804633025896291682772499885
      6.630396332390718431485053218985
      - 0.34508839784665403032666523448675 - 1.0778362954630176596831109269793i
      - 0.34508839784665403032666523448675 + 1.0778362954630176596831109269793i
```

MATLAB can also be used to solve symbolic equations. For example,  $ax^2 + bx + c = 0$ .

```
>> syms a b c x;
>> solve(a*x^2 + b*x + c == 0, x)
ans = -(b + (b^2 - 4*a*c)^(1/2))/(2*a)
```

$$-(b - (b^2 - 4ac)^{1/2})/(2a)$$

**4.2. Multi-Variable Equations.** MATLAB can also solve multi-variable equations, for which the equations shall be put together in square brackets `[ ]`, so do the variables. Moreover, since the output consists multiple variables, we shall assign the solution as a vector. For example,

$$3x + y = 10, \quad x + y = 20.$$

```
>> syms x y;
>> [Sx,Sy] = solve([3*x+y==10, x+y==20], [x,y])
Sx =  -5
Sy =  25
```

The solution of the a general linear system in variables  $x$  and  $y$

$$\begin{cases} ax + by = e \\ cx + dy = f \end{cases}$$

can be obtained as follows:

```
>> syms a b c d e f x y;
>> [Sx,Sy] = [Sx, Sy] = solve([a*x+b*y == e, c*x+d*y == f], [x,y])
Sx =  -(b*f - d*e)/(a*d - b*c)
Sy =  (a*f - c*e)/(a*d - b*c)
```

## 5. SUMS

If we need to find the sum of a sequence, use the command `symsum`. The format is

```
>> symsum(expression, variable, [min, max])
```

For example, for the arithmetic sequence  $\{a_n\}$  given by  $a_n = a + (n-1)d$ , the sum of its first  $n$  terms is  $(2a + (n-1)d)n/2$ .

```
>> syms a d k n;
>> symsum(a+(k-1)*d, k, [1, n])
ans =  a*n - d*(n - (n*(n + 1))/2)
>> simplify(ans) % simplify the previous answer
ans =  (n*(2*a - d + d*n))/2
```

For the geometric sequence  $\{a_n\}$  given by  $a_n = ar^{n-1}$ , the sum of its first  $n$  terms is

$$S_n = \begin{cases} a(r^n - 1)/(r - 1) & \text{if } r \neq 1, \\ an & \text{if } r = 1. \end{cases}$$

```
>> syms a r k n;
>> symsum(a*r^(k-1), k, [1, n])
ans = piecewise(r == 1, a*n, r ~= 1, (a*(r^n - 1))/(r - 1))
```

Moreover, we can find the sum to infinity, which is defined by `inf` in MATLAB. Recall that for the geometric sequence  $\{a_n\}$  given by  $a_n = ar^{n-1}$ , the sum to infinity is

$$S_{\infty} = \begin{cases} a/(1-r) & \text{if } |r| < 1, \\ \text{does not exist} & \text{if } |r| \geq 1. \end{cases}$$

```
>> syms a r k;
>> assume(abs(r)<1);
>> symsum(a*r^(k-1), k, [1, inf])
ans = -a/(r - 1)
```

## 6. VECTOR

A vector  $ai + bj + ck$  or  $(a, b, c)$  can be defined as `[a, b, c]` or simply `[a b c]`.

The addition, subtraction and multiplication with numbers can be evaluated using `+`, `-` and `*` respectively.

For example, let  $u = (1, 2, 3)$  and  $v = (4, 5, 6)$ .

```
>> u = [1 2 3];
>> v = [4 5 6];
>> u + v
ans = 5 7 9
>> u - v
ans = -3 -3 -3
>> 3*u
ans = 3 6 9
```

The dot product  $u \bullet v$  of two vectors  $u$  and  $v$  is defined by `dot(u,v)`.

```
>> dot(u,v)
ans = 32
```

We can use the formula  $|v| = \sqrt{v \bullet v}$  to find the norm of  $v$ :

```
>> sqrt(dot(v,v))
ans = 8.7750
```

Alternatively, MATLAB provides a command `norm` for the norm of a vector.

```
>> norm(v)
ans = 8.7750
```

The cross product  $u \times v$  is defined by `cross(u,v)`.

```
>> cross(u,v)
ans = -3 6 -3
```

## 7. FUNCTION

**7.1. Standard Function.** To define a function, we shall (i) give the name of the function, (ii) use `@` to declare the name of the variable, (iii) provide the expression of the function. For example, define  $f(x) = x^2$ :

```
>> f = @(x) x^2;
```

Then  $f(2)$  can be evaluated by

```
>> f(2);
ans = 4
```

Another example:  $g(x) = \sin(x^3)/x^2$

```
>> g = @(x) sin(x^3)/x^2;
```

Multi-variable functions can be defined similarly by declaring more variables. For example,  $h(x, y) = \sqrt{x^2 + y^2}$ :

```
>> h = @(x,y) sqrt(x^2 + y^2);
```

To evaluate  $h(5, 12)$ :

```
>> h(5,12)
ans = 13
```

**7.2. Piecewise Function.** The absolute value function  $f(x) = |x|$  is a piecewise function:

$$f(x) = \begin{cases} x & \text{if } x \geq 0, \\ -x & \text{if } x < 0, \end{cases}$$

which can be defined in MATLAB:

```
>> syms x
>> f = piecewise(x>=0, x, x<0, -x);
```

The general format is to use the `piecewise` as

```
>> f = piecewise(condition 1, value 1, ..., condition N, value N);
```

In the example above, the first condition is `x>=0` and this means “ $x$  is greater than or equal to 0”. The value `x` is the expression of the function when the condition `x>=0` is satisfied. It defines  $f(x) = x$  when  $x \geq 0$ . The second condition `x<0` and the second value `-x` defines  $f(x) = -x$  when  $x < 0$ .

In order to evaluate the value of the function at given point, for example,  $f(-2)$ , we use

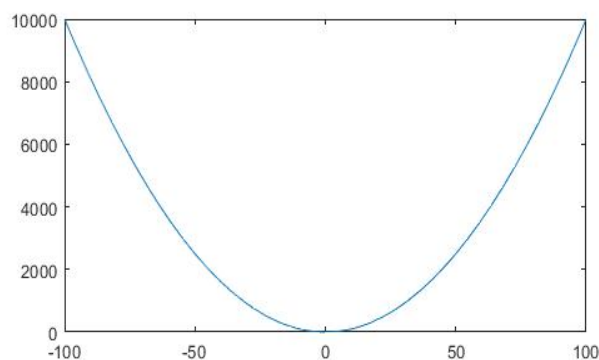
```
>> subs(f, x, -2)
ans = 2
```

## 8. CURVE PLOTTING

**8.1. Standard Function.** Once a function is defined, `fplot` can be used to plot its graph over a specified interval in the form `[xmin, xmax]`.

For example, we plot  $f(x) = x^2$  on the interval  $(-100, 100)$ :

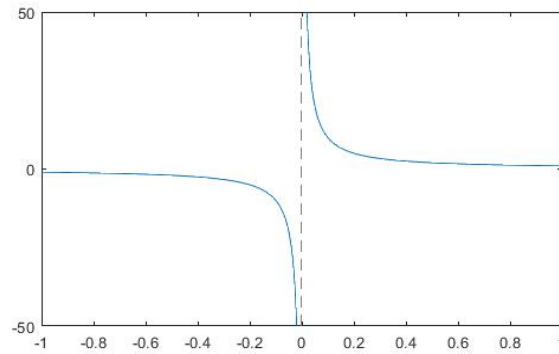
```
>> f = @(x) x^2;
>> fplot(f, [-100,100]);
```



The command can be used even if the function is undefined somewhere on the specific interval. For example,  $g(x) = 1/x$  on  $(-1, 1)$  is undefined at  $x = 0$ .

```
>> g = @(x) 1/x;
>> fplot(g, [-1,1]);
```



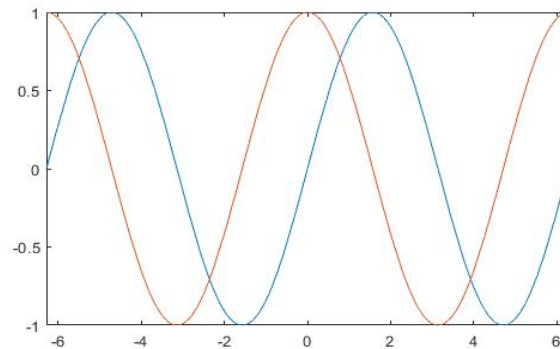


**8.2. Multiple Curves.** In order to plot more curves in the same coordinate system, we use the commands `on` and `off`. For example,

$$f(x) = \sin x \quad \text{and} \quad g(x) = \cos x, \quad -2\pi < x < 2\pi.$$

(We can use Shift + Enter to start a new line without breaking the command.)

```
>> f = @(x) sin(x);
>> g = @(x) cos(x);
>> fplot(f, [-2*pi, 2*pi]);
    hold on
    fplot(g, [-2*pi, 2*pi]);
    hold off
```



We can draw multiple graphs on the same plot. MATLAB provides some basic colour options: white `w`, black `b`, blue `b`, red `r`, cyan `c`, green `g`, magenta `m`, yellow `y` to distinguish

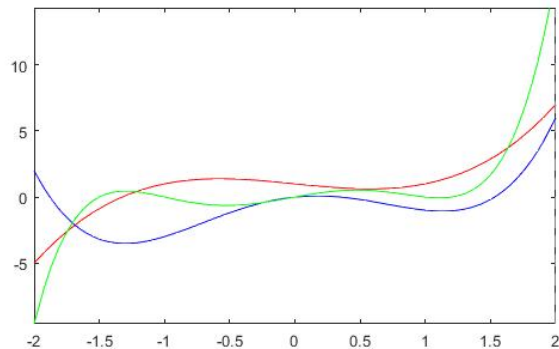
different graphs. For example,

$$f(x) = x^3 - x + 1,$$

$$g(x) = x^4 - 3x^2 + x,$$

$$h(x) = x^5 + 0.3x^4 - 2.8x^3 - 0.3x^2 + 1.8x.$$

```
>> f = @(x) x^3 - x + 1;
>> g = @(x) x^4 - 3*x^2 + x;
>> h = @(x) x^5 + 0.3*x^4 - 2.8*x^3 - 0.3*x^2 + 1.8*x;
>> fplot(f, [-2,2], 'r');
    hold on
    fplot(g, [-2,2], 'b');
    fplot(h, [-2,2], 'g');
    hold off;
```



**8.3. Parametric Equations.** A curve may be defined using a pair of parametric equations:

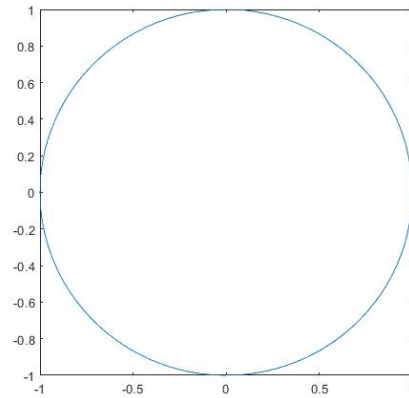
$$x = x(t) \quad \text{and} \quad y = y(t).$$

In order to use `fplot`, we shall first define the functions for the  $x$ - and  $y$ -coordinates.

Recall that a unit circle can be parameterized by

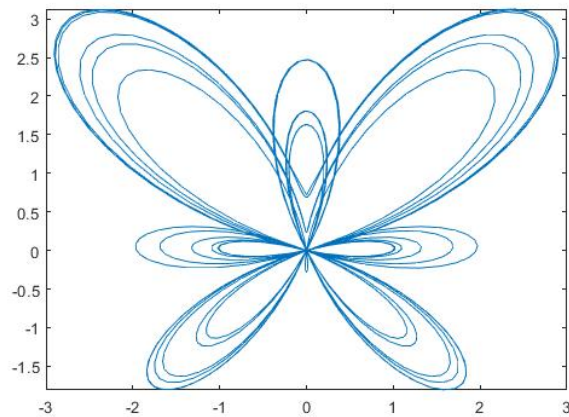
$$x = \cos t, \quad y = \sin t, \quad 0 \leq t \leq 2\pi.$$

```
>> f = @(t) cos(t);
>> g = @(t) sin(t);
>> fplot(f,g, [0,2*pi]);
```



MATLAB can draw a butterfly:

```
>> f = @(t) sin(t)*(exp(cos(t))-2*cos(4*t) - (sin(t/12))^5);
>> g = @(t) cos(t)*(exp(cos(t))-2*cos(4*t) - (sin(t/12))^5);
>> fplot(f,g, [0,12*pi]);
```

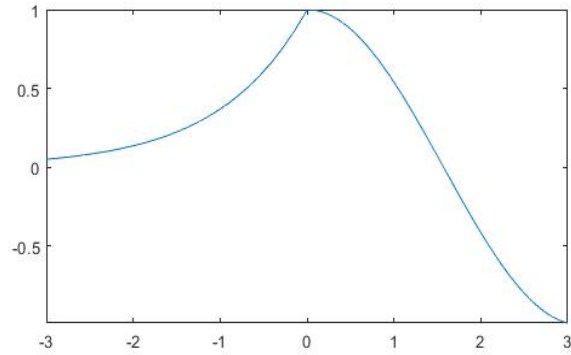


**8.4. Piecewise Function.** There are two ways to plot a piecewise function.

Once a piecewise function is already defined, `fplot` can directly plot its graph:

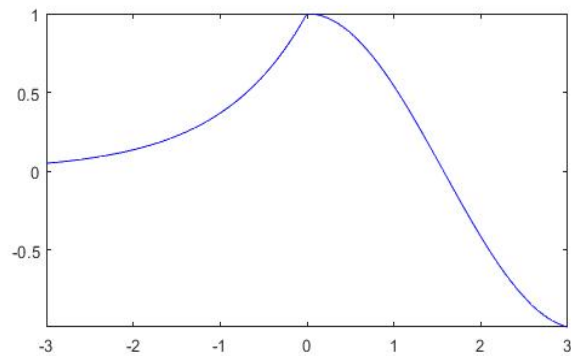
$$f(x) = \begin{cases} e^x & \text{if } x < 0, \\ \cos x & \text{if } x \geq 0. \end{cases}$$

```
>> syms x;
>> f = piecewise(x<0, exp(x), x>=0, cos(x));
>> fplot(f, [-3,3]);
```



Alternatively, we can plot different branches on different intervals using the same colour.

```
>> f1 = @(x) exp(x);
>> f2 = @(x) cos(x);
>> fplot(f1, [-3,0], 'b');
    hold on
    fplot(f2, [0,3], 'b');
    hold off;
```

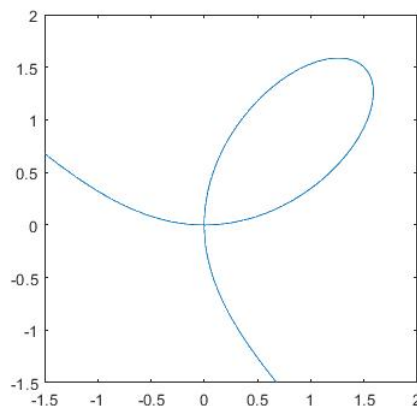


**8.5. Implicit Function.** The command `fimplicit` can be used to plot graphs defined by implicit functions, i.e.,  $f(x, y) = 0$ . The format is:

```
>> fimplicit(function in two variables, [xmin, xmax, ymin, ymax])
```

For example, to plot  $x^3 + y^3 = 3xy$ , we shall first define the function  $f(x, y) = x^3 + y^3 - 3xy$ :

```
>> f = @(x,y) x^3 + y^3 - 3*x*y;
>> fimplicit(f, [-1.5, 2, -1.5, 2]);
```



## 9. LIMITS

The limit of a function describes the behavior of a function near a point or at infinity. The limit  $\lim_{x \rightarrow a} f(x)$  can be easily evaluated with the command `limit`:

```
>> limit(f(x), x, a)
```

For example,  $\lim_{x \rightarrow 0} \frac{x}{\sqrt{1+3x}-1}$ :

```
>> syms x;
```

```
>> limit(x/(sqrt(1+3*x)-1), x, 0)
```

```
ans = 2/3
```

The one-sided limits can be evaluated by specifying the direction `left` or `right`. For example, let the floor function  $\lfloor x \rfloor$  denote the greatest integer smaller or equal to  $x$ . Then

$$\lim_{x \rightarrow 2^-} \lfloor x \rfloor = 1, \quad \lim_{x \rightarrow 2^+} \lfloor x \rfloor = 2, \quad \text{and} \quad \lim_{x \rightarrow 2} \lfloor x \rfloor \text{ does not exist.}$$

```
>> syms x;
```

```
>> limit(floor(x), x, 2, 'left')
```

```
ans = 1
```

```
>> limit(floor(x), x, 2, 'right')
```

```
ans = 2
```

```
>> limit(floor(x), x, 2)
```

```
ans = limit(floor(x), x, 2)
```

We can find the limit at infinity `inf` or negative infinity `-inf`. For example,  $\lim_{x \rightarrow \infty} \left(1 + \frac{a}{x}\right)^x$ :

```
>> syms a x;
```

```
>> limit((1+a/x)^x, x, inf)
```

```
ans = exp(a)
```

Sometimes the limit depends on the value of the parameters. For example,

$$\lim_{x \rightarrow \infty} 2^{ax} = \begin{cases} 0 & \text{if } a < 0, \\ 1 & \text{if } a = 0, \\ \infty & \text{if } a > 0. \end{cases}$$

We may use `assume` to declare the value of the parameter.

```
>> syms a x;
>> assume(a<0)
>> limit(2^(a*x), x, inf)
ans = 0
>> assume(a==0)
>> limit(2^(a*x), x, inf)
ans = 1
>> assume(a>0)
>> limit(2^(a*x), x, inf)
ans = Inf
```

## 10. DERIVATIVE

The derivative of a function  $f$  at point  $a$  is defined as the limit

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h}.$$

For example, if  $f(x) = x^2$ , then  $f'(x) = 2x$ :

```
>> syms x h;
>> f = @(x) x^2;
>> limit((f(x+h)-f(x))/h, h, 0)
ans = 2*x
```

On the other hand, MATLAB includes the command `diff` for differentiation. For example, find the derivative of  $f(x) = \cos(x^2)$  with respect to  $x$ :

```
>> syms x;
>> f = @(x) cos(x^2);
>> diff(f, x)
ans = -2*x*sin(x^2)
```

In order to differentiate  $g(x) = \frac{x - 3x\sqrt{x}}{\sqrt{x}}$  with respect to  $x$ :

```
>> syms x;
>> g = @(x) (x-3*x*sqrt(x))/sqrt(x);
>> diff(g, x)
ans = - (x - 3*x^(3/2))/(2*x^(3/2)) - ((9*x^(1/2))/2 - 1)/x^(1/2)
```

The answer seems complicated. Fortunately MATLAB provides the command `simplify` which may help in this situation:

```
>> simplify(ans)
ans = -(6*x^(1/2) - 1)/(2*x^(1/2))
```

Suppose we are looking for  $f^{(4)}(x)$ , the 4<sup>th</sup> order derivative of  $f(x)$  with respect to  $x$ , of course we may use

```
>> diff(diff(diff(diff(f, x), x), x), x)
ans = 16*x^4*cos(x^2) - 12*cos(x^2) + 48*x^2*sin(x^2)
```

However, it may be a bit cumbersome. The following shorter commands have the same effect:

```
>> diff(f, x,x,x,x)
ans = 16*x^4*cos(x^2) - 12*cos(x^2) + 48*x^2*sin(x^2)
>> diff(f, x, 4)
ans = 16*x^4*cos(x^2) - 12*cos(x^2) + 48*x^2*sin(x^2)
```

For example, we can find the local extreme values of  $f(x) = x^3 - 3x^2 + x - 2$ .

```
>> syms x;
>> f = @(x) x^3 - 3*x^2 + x - 2;
>> g = diff(f, x); % define g(x) to be f'(x)
>> h = diff(f, x, x); % define h(x) to be f''(x)
```

(i) Solve  $f'(x) = 0$  to obtain the stationary points.

```
>> solve(g(x) == 0, x)
ans = 1 - 6^(1/2)/3
      6^(1/2)/3 + 1
```

(ii) Check the sign of  $f''(x)$  at the stationary points. (Note that you can use Copy and Paste in the scripts.)

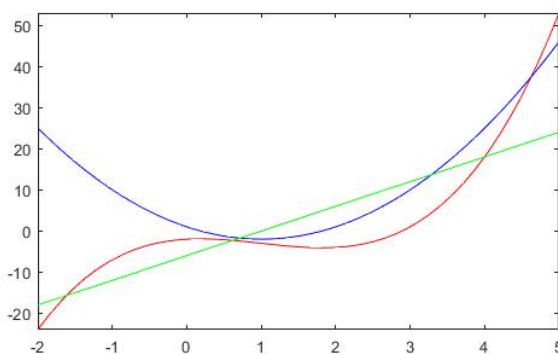
```
>> subs(h, x, 1-6^(1/2)/3)
ans = -22063042185692343/4503599627370496
```

```
>> subs(h, x, 6^(1/2)/3+1)
ans = 5515760546423085/1125899906842624
```

Hence,  $f$  has a local maximum at  $1 - \sqrt{6}/3$  and a local minimum at  $1 + \sqrt{6}/3$ .

We can plot  $f(x)$ ,  $f'(x)$  and  $f''(x)$  in the same coordinate system.

```
>> fplot(f, [-2,5], 'r');
    hold on
    fplot(g, [-2,5], 'b');
    fplot(h, [-2,5], 'g');
    hold off
```



## 11. INTEGRAL

**11.1. Indefinite Integral.** For a function  $f(x)$ , its indefinite integral is a function  $F(x)$  such that  $F'(x) = f(x)$ , denoted by

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

All indefinite integrals differ by a constant only. So we also use

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

to represent the entire family of indefinite integrals of  $f(x)$ , where  $C$  is an arbitrary constant.

In MATLAB, we can use the command `int` to find an indefinite integral.

For example,  $\int \frac{1}{(1+x^2)^2} dx = \frac{1}{2} \tan^{-1} x + \frac{x}{2(x^2+1)} + C$ .

```
>> syms x;
>> f = @(x) 1/(1+x^2)^2;
>> int(f, x)
ans = atan(x)/2 + x/(2*(x^2 + 1))
```



Note that the constant  $C$  is dropped in the answer. Another example:

$$\int \sin^6 x \, dx = \frac{5}{16}x - \frac{15}{64}\sin 2x + \frac{3}{64}\sin 4x - \frac{1}{192}\sin 6x + C.$$

```
>> syms x;
>> g = @(x) sin(x)^6;
>> int(g, x)
ans = (5*x)/16 - (15*sin(2*x))/64 + (3*sin(4*x))/64 - sin(6*x)/192
```

**11.2. Definite Integral.** The definite integral  $\int_a^b f(x) \, dx$  represents the net area bounded between the graph of  $y = f(x)$  and the  $x$ -axis from  $x = a$  to  $x = b$ . The command `int` can also be used to find definite integral by specifying the lower limit  $a$  and upper limit  $b$  in the form

```
>> int(expression, variable, [lower limit, upper limit])
```

For example,  $\int_0^1 \frac{1}{(1+x^2)^2} \, dx = \frac{\pi}{8} + \frac{1}{4}$ .

```
>> syms x;
>> f = @(x) 1/(1+x^2)^2;
>> int(f, x, [0, 1])
ans = pi/8 + 1/4
```

Another example:

$$\int_0^\pi \sin^6 x \, dx = \frac{5}{16}\pi.$$

```
>> syms x;
>> g = @(x) sin(x)^6;
>> int(g, x, [0, pi])
ans = (5*pi)/16
```

We can verify the fundamental theorem of calculus:

$$\frac{d}{dx} \int_a^x f(t) \, dx = f(x).$$

```
>> syms a t x f(t); % use f(t) to declare that f is a function in t
>> diff(int(f(t), t, [a, x]), x)
ans = f(x)
```

It is also possible to integrate over an infinite interval by specifying the lower limit and/or the upper limit as `-inf` or `inf`. For example,

$$\int_{-\infty}^{\infty} e^{-x^2} \, dx = \sqrt{\pi}.$$

```
>> int(exp(-x^2), x, [-inf, inf])
ans = pi^(1/2)
```

## 12. DIFFERENTIAL EQUATION

**12.1. General Solution.** In order to solve an ordinary differential equation of the form  $\frac{dy}{dx} = f(x, y)$ , we shall first declare that  $y$  is a function in  $x$ , and then use `dsolve`.

For example,  $\frac{dy}{dx} = 1 + x + y$ . Note that the equal sign is represented by `==`.

```
>> syms y(x);    % declare that y is a function in x
>> dsolve(diff(y, x) == 1+x+y)
ans = C1*exp(x) - x -2
```

Here `C1` refers to an undermined constant. So the general solution of the given equation is  $y = Ce^x - x - 2$ , where  $C$  is an arbitrary constant.

Ordinary differential equations with higher order can be solved similarly.

For example,  $\frac{d^2y}{dx^2} + 2\frac{dy}{dx} + 2y = e^x$ .

```
>> syms y(x);
>> dsolve(diff(y,x,x) + 2*diff(y,x) + 2*y == 0)
ans = C2*exp(-x)*cos(x) - C3*exp(-x)*sin(x)
```

The general solution is  $y = C_2 e^{-x} \sin x + C_3 e^{-x} \cos x$ , where  $C_2, C_3$  are arbitrary constants.

**12.2. Particular Solution.** Suppose an initial condition is given. We can use

```
>> dsolve(ode, condition)
For example,  $\frac{dy}{dx} = 1 + x + y$  such that  $y = 1$  at  $x = 0$ .
>> syms y(x);
>> dsolve(diff(y, x) == 1+x+y, y(0)==1)
ans = 3*exp(x) - x -2
```

For higher order differential equations, we need more initial conditions. We use

```
>> dsolve(ode, [condition 1, ..., condition N])
```

The condition  $y'(a) = b$  shall be defined as `Dy(a)=b` if we set

```
>> Dy = diff(y,x);
For example,  $\frac{d^2y}{dx^2} + 8\frac{dy}{dx} - 9y = 0$ , where  $y(1) = 2$  and  $y'(1) = 0$ .
>> syms y(x);
```

```
>> Dy = diff(y,x);
>> dsolve(diff(y,x,x) + 8*diff(y,x) - 9*y == 0, [y(1) == 2, Dy(1) == 0])
ans = (exp(-9*x)*exp(9))/5 + (9*exp(-1)*exp(x))/5
>> simplify(ans)
ans = (9*exp(x - 1))/5 + exp(9 - 9*x)/5
```

**12.3. System of Differential Equations.** The command `dsolve` can also be used for a system of differential equations. Note that the output consists multiple functions. So the output must be in vector form:

```
>> [var 1, ..., var N] = dsolve([eqn1, ..., eqn M], [cond 1, ..., cond k])
```

For example, the following linear system consists of two functions  $x(t)$  and  $y(t)$  with two equations and two initial conditions:

$$\begin{cases} x'(t) = 3x + 4y \\ y'(t) = -4x + 3y \end{cases} \quad \text{where } x(0) = 2 \quad \text{and} \quad y(0) = 3.$$

```
>> syms x(t) y(t);
>> [Sx, Sy] = dsolve([diff(x,t) == 3*x + 4*y, diff(y,t) == -4*x + 3*y], [x(0) == 2, y(0) == 3])
Sx = 2*cos(4*t)*exp(3*t) + 3*sin(4*t)*exp(3*t)
Sy = 3*cos(4*t)*exp(3*t) - 2*sin(4*t)*exp(3*t)
```

## 13. PLOTTING IN THE SPACE

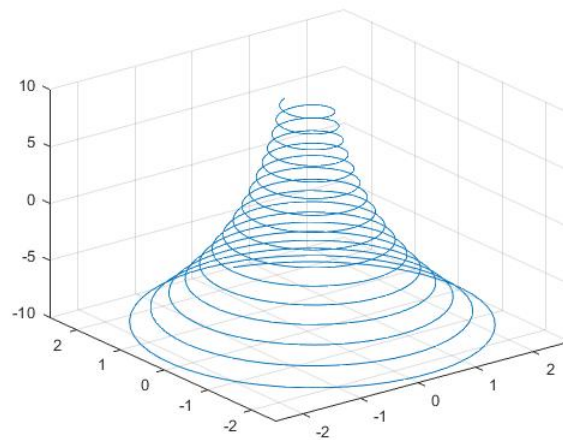
**13.1. Curve in the Space.** A curve in the  $xyz$ -space can be parametrized as

$$\mathbf{r}(t) = x(t)\mathbf{i} + y(t)\mathbf{j} + z(t)\mathbf{k}, \quad a \leq t \leq b.$$

Plotting a space curve in the  $xyz$ -space is similar to plotting a parametrized curve in the  $xy$ -plane. The only difference is to replace `fplot` by `fplot3`. For example,

$$\mathbf{r}(t) = (e^{-t/10} \sin 5t) \mathbf{i} + (e^{-t/10} \cos 5t) \mathbf{j} + t \mathbf{k}.$$

```
>> x = @(t) exp(-t/10)*sin(5*t);
>> y = @(t) exp(-t/10)*cos(5*t);
>> z = @(t) t;
>> fplot3(x,y,z, [-10,10])
```



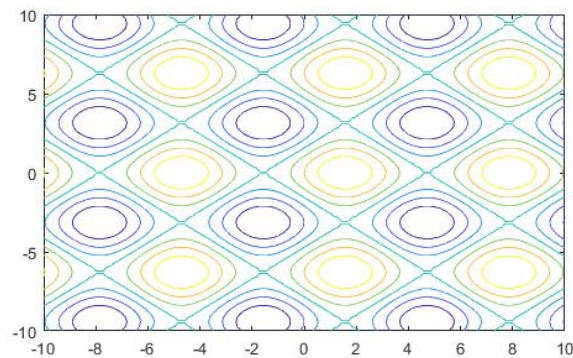
**13.2. Level Curves.** A surface is defined as a two-variable function  $z = f(x, y)$ . Its level curve at  $z = c$  is the intersection of the surface with the horizontal plane at  $z = c$ ; that is,  $f(x, y) = c$ .

The command `fcontour` plots the level curve of a two-variable function.

```
>> fcontour(function, [xmin, xmax, ymin, ymax])
```

For example,  $f(x, y) = \sin x + \cos y$ :

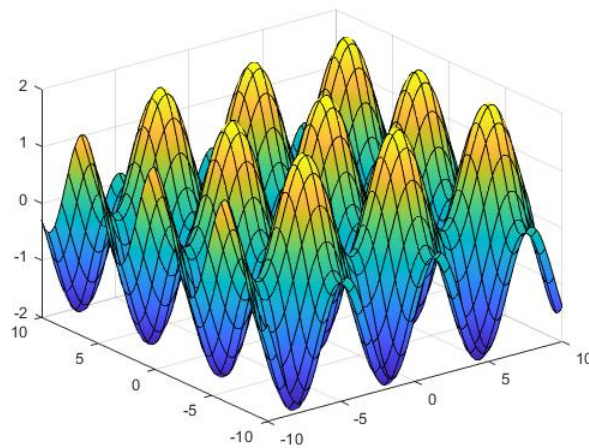
```
>> f = @(x,y) sin(x) + cos(y);
>> fcontour(f, [-10,10, -10,10])
```



It is also possible plot the level curves of two surfaces using `hold on` and `hold off`.

**13.3. Surface.** A surface  $z = f(x, y)$  in two variables can be plotted in  $xyz$ -space using `fsurf`. For example,  $f(x, y) = \sin x + \cos y$ .

```
>> f = @(x,y) sin(x) + cos(y);
>> fsurf(f, [-10,10, -10,10])
```



13.4. **Parametrized Surface.** A surface can be parametrized by two parameters  $u$  and  $v$ :

$$\mathbf{r}(u, v) = x(u, v)\mathbf{i} + y(u, v)\mathbf{j} + z(u, v)\mathbf{k}.$$

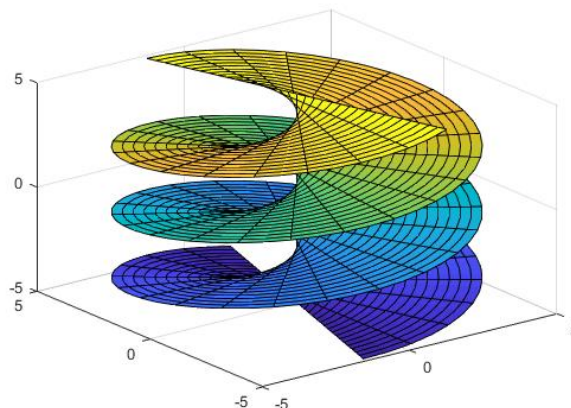
We can still use `fsurf` to plot parametrized surface in the form

```
>> fsurf(x, y, z, [umin, umax, vmin, vmax])
```

For example,

$$\mathbf{r}(u, v) = u \cos v \mathbf{i} + u \sin v \mathbf{j} + v \mathbf{k}.$$

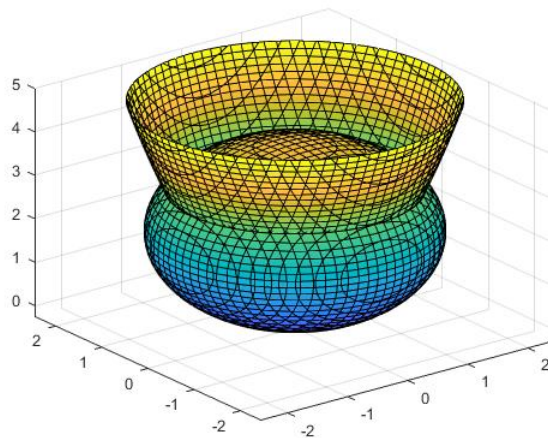
```
>> x = @(u,v) u*cos(v);
>> y = @(u,v) u*sin(v);
>> z = @(u,v) v;
>> fsurf(x,y,z, [-5,5, -5,5])
```



**13.5. Implicit Function.** A surface can also be defined by an implicit function, i.e.,  $f(x, y, z) = 0$ . It is almost the same as plotting a curve defined by implicit function in the  $xy$ -plane. Simply use `fimplicit3` instead of `fimplicit`.

For example, plot the sphere  $x^2 + y^2 + z^2 = 4z$  and the paraboloid  $z = x^2 + y^2$ .

```
>> f = @(x,y,z) x^2 + y^2 + z^2 - 4*z;
>> g = @(x,y,z) x^2 + y^2 - z;
>> fimPLICIT3(f, [-3,3, -3,3, -1,5]);
    hold on
    fimPLICIT3(g, [-3,3, -3,3, -1,5]);
    hold off
```



## 14. MULTI-VARIABLE DERIVATIVES

**14.1. Partial Derivatives.** For a multi-variable function, we can find its partial derivatives by specifying the variables to be differentiated with respect to properly.

For example, for  $f(x, y, z) = x \ln(xy^2z^3)$ , its partial derivatives  $\frac{\partial f}{\partial x}$ ,  $\frac{\partial f}{\partial y}$  and  $\frac{\partial f}{\partial z}$  are given by

```
>> syms x y z;
>> f = @(x,y,z) x*log(x*y^2*z^3);
>> diff(f, x)
ans = log(x*y^2*z^3)+1
>> diff(f, y)
ans = (2*x)/y
>> diff(f, z)
ans = 3*x/z
```

```
ans = (3*x)/z
```

The command `gradient` provides the gradient vector  $\nabla f$  of a function  $f$  so that we can obtain all the derivatives at once (the output is a column vector).

```
>> gradient(f, [x,y,z])
```

```
ans = log(x*y^2*z^3)+1
      (2*x)/y
      (3*x)/z
```

We may verify the mixed derivative theorem:  $\frac{\partial^2 f}{\partial y \partial x} = \frac{\partial^2 f}{\partial x \partial y}$ :

```
>> diff(f, x, y)
```

```
ans = 2/y
```

```
>> diff(f, y, x)
```

```
ans = 2/y
```

Recall that the directional derivative of  $f(x, y, z)$  along a vector  $\mathbf{v}$  at  $(a, b, c)$  is given by

$$\nabla f(a, b, c) \cdot \mathbf{v} / |\mathbf{v}|.$$

For example, at  $(1, 2, 3)$ , the directional derivative of  $f$  along  $\mathbf{v} = (4, 5, 6)$  is

```
>> v = [4 5 6];
>> subs(gradient(f, [x,y,z]), {x,y,z}, {1,2,3});
>> dot(ans, v)/norm(v)
ans = (77^(1/2)*(4*log(108) + 15))/77
```

**14.2. Derivative of a Curve.** For a curve  $C$  defined by

$$\mathbf{r}(t) = x(t)\mathbf{i} + y(t)\mathbf{j} + z(t)\mathbf{k},$$

its derivative vector is  $\mathbf{r}'(t) = x'(t)\mathbf{i} + y'(t)\mathbf{j} + z'(t)\mathbf{k}$ .

One may define the  $x$ -,  $y$ - and  $z$ -components separately. Alternatively, we can define  $\mathbf{r}$  as a vector function in  $t$ . For example,  $\mathbf{r}(t) = (\cos t)\mathbf{i} + (\sin t)\mathbf{j} + t\mathbf{k}$ .

```
>> r = @(t) [cos(t) sin(t) t];
```

Then its derivative vector is

```
>> syms t;
```

```
>> diff(r, t)
```

```
ans = [-sin(t), cos(t), 1]
```

## 15. INTEGRAL FOR MULTI-VARIABLE FUNCTIONS

15.1. **Double Integral.** Suppose that a region  $D$  in the  $xy$ -plane is parametrized by

$$a \leq x \leq b, \quad \alpha(x) \leq y \leq \beta(x).$$

Then for any function  $f(x, y)$ , the double integral

$$\iint_D f(x, y) dA = \int_a^b \int_{\alpha(x)}^{\beta(x)} f(x, y) dy dx,$$

is converted to two integrals.

Let  $D = \{(x, y) \mid 0 \leq x \leq 2, x^2 \leq y \leq 2x\}$ , then  $\iint_D (x^2 + y^2) dA = \int_0^2 \int_{x^2}^{2x} (x^2 + y^2) dy dx.$

```
>> syms x y;
```

```
>> f = @(x,y) x^2 + y^2;
```

We first find  $\int_{x^2}^{2x} f(x, y) dy$ :

```
>> int(f, y, [x^2, 2*x])
```

```
ans = -(x^3*(x^3 + 3*x - 14))/3
```

then integrate the result with respect to  $x$  on  $[0, 2]$ :

```
>> int(ans, x, [0, 2])
```

```
ans = 216/35
```

Note that  $D$  can also be parametrized by  $D = \{(x, y) \mid 0 \leq y \leq 4, y/2 \leq x \leq \sqrt{y}\}$ . So

$$\iint_D (x^2 + y^2) dA = \int_0^4 \int_{y/2}^{\sqrt{y}} (x^2 + y^2) dx dy.$$

```
>> int(f, x, [y/2, sqrt(y)])
```

```
ans = y^(3/2)*(y - (13*y^(3/2)))/24 + 1/3
```

```
>> int(ans, y, [0, 4])
```

```
ans = 216/35
```

15.2. **Line Integral of a Function.** Let  $C$  a curve parametrized by

$$\mathbf{r}(t) = x(t)\mathbf{i} + y(t)\mathbf{j} + z(t)\mathbf{k}, \quad a \leq t \leq b.$$

Its length is the line integral

$$\int_C 1 ds = \int_a^b |\mathbf{r}'(t)| dt.$$

For example,  $\mathbf{r} = (\cos t)\mathbf{i} + (\sin t)\mathbf{j} + t\mathbf{k}$ ,  $0 \leq t \leq 2\pi$ .

```
>> r = @(t) [cos(t) sin(t) t];
```

```
>> syms t;
```



```
>> norm(diff(f,t))
ans = (abs(cos(t))^2 + abs(sin(t))^2 + 1)^(1/2)
>> int(ans, [0, 2*pi])
ans = 2*pi*2^(1/2)
```

In general, the line integral of a function  $f$  along  $C$  is

$$\int_C f \, ds = \int_a^b f(x(t), y(t), z(t)) |\mathbf{r}'(t)| \, dt.$$

Since there is no simple way to evaluate  $f(\mathbf{r}(t))$  directly, we may need to define all the components of  $\mathbf{r}$  as functions.

For example,  $\mathbf{r} = (\cos t)\mathbf{i} + (\sin t)\mathbf{j} + t\mathbf{k}$ ,  $0 \leq t \leq 2\pi$ , and  $f(x, y, z) = x^2 + yz$ .

```
>> x = @(t) cos(t);
>> y = @(t) sin(t);
>> z = @(t) t;
>> r = @(t) [x(t) y(t) z(t)]; % define the curve
>> f = @(x,y,z) x^2 + y*z;
>> syms t;
>> dot(f(x(t),y(t),z(t)), norm(diff(r,t)));
>> int(ans, t, [0, 2*pi])
ans = -pi*2^(1/2)
```

**15.3. Line Integral of a Vector Field.** A vector field  $\mathbf{F}$  is a vector valued function in multi-variables:

$$\mathbf{F}(x, y, z) = P(x, y, z)\mathbf{i} + Q(x, y, z)\mathbf{j} + R(x, y, z)\mathbf{k}.$$

Its line integral along the curve  $C$  defined by  $\mathbf{r}(t) = x(t)\mathbf{i} + y(t)\mathbf{j} + z(t)\mathbf{k}$ ,  $a \leq t \leq b$ , is

$$\int_C \mathbf{F} \cdot d\mathbf{r} = \int_a^b \mathbf{F}(x(t), y(t), z(t)) \cdot \mathbf{r}'(t) \, dt.$$

For example,  $\mathbf{r} = (\cos t)\mathbf{i} + (\sin t)\mathbf{j} + t\mathbf{k}$ ,  $0 \leq t \leq 2\pi$ , and  $\mathbf{F}(x, y, z) = x^2\mathbf{i} + y^2\mathbf{j} + z^2\mathbf{k}$ .

```
>> x = @(t) cos(t);
>> y = @(t) sin(t);
>> z = @(t) t;
>> r = @(t) [x(t) y(t) z(t)];
>> F = @(x,y,z) x^2 + y^2 + z^2;
>> syms t;
```

```
>> dot(F(x(t),y(t),z(t)), diff(r,t));
>> int(ans, t, [0, 2*pi])
ans = (8*pi^3)/3
```

## 16. SERIES

**16.1. Taylor Series.** The Taylor series of a function  $f(x)$  at  $x = a$  is of the form

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!} (x-a)^n.$$

In particular, if  $a = 0$ , it is called the Maclaurin series of  $f(x)$ :

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!} x^n.$$

MATLAB has the command `taylor` to produce the Taylor series of a function up to certain order:

```
>> taylor(function, variable, point, 'Order', number)
```

For example, find the Taylor series of  $f(x) = \frac{x^2+3}{x+5}$  at  $x = 3$  of order  $< 6$ :

```
>> taylor((x^2+3)/(x+5), x, 3, 'Order', 6)
ans = (9*x)/16 + (7*(x - 3)^2)/128 - (7*(x - 3)^3)/1024 + (7*(x - 3)^4)/8192
- (7*(x - 3)^5)/65536 - 3/16
```

**16.2. Fourier Series.** The fourier series of a periodic function  $f(x)$  with periodic  $2L$  is given by

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos \frac{n\pi x}{L} + \sum_{n=1}^{\infty} b_n \sin \frac{n\pi x}{L},$$

where

$$a_n = \frac{2}{L} \int_{-L}^L f(x) \cos \frac{n\pi x}{L} dx \quad \text{and} \quad b_n = \frac{2}{L} \int_{-L}^L f(x) \sin \frac{n\pi x}{L} dx.$$

There is no simple command in MATLAB to produce the coefficients for fourier series. But we can define by ourself.

For example, let  $f(x) = \begin{cases} 0 & \text{if } -2 \leq x < 0, \\ x & \text{if } 0 \leq x < 2, \end{cases}$  and  $f(x) = f(x+4)$  for all  $x$ ; so  $L = 2$ .

```
>> syms x
>> f = piecewise(-2<=x<0, 0, 0<=x<2, x);
>> L = 2;
>> a = @(n) 2/L*int(f*cos(pi*n*x/L), x, [-L, L]);
>> b = @(n) 2/L*int(f*sin(pi*n*x/L), x, [-L, L]);
```

We may find  $a_0, \dots, a_6$  and  $b_1, \dots, b_6$  using simple commands:

```
>> a(0:6)
ans = [ 2, -8/pi^2, 0, -8/(9*pi^2), 0, -8/(25*pi^2), 0]
>> b(1:6)
ans = [ 0, 4/pi, -2/pi, 4/(3*pi), -1/pi, 4/(5*pi), -2/(3*pi)]
```

## 17. OPERATIONS ON MATRICES

**17.1. Basic Operations.** The entries of a matrix shall be entered row by row, while the entries in each row are separated by a space and the rows are separated by a semi-colon `;`. For example,

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}.$$

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

The matrix addition, subtraction and multiplication with scalar can be evaluated using `+`, `-` and `*` respectively. For example,

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 4 & 1 \\ 2 & 5 \end{pmatrix}.$$

```
>> A = [1 2; 3 4];
>> B = [4 1; 2 5];
>> A + B
ans = 5 3
      5 9
>> A - B
ans = -3 1
      1 -1
>> 3*A
ans = 3 6
      9 12
```

We illustrate more operations using the previously defined  $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$  and  $B = \begin{pmatrix} 4 & 1 \\ 2 & 5 \end{pmatrix}$ .

(i) Transpose  $\mathbf{A}^T$ :

```
>> A'
ans =  1  3
       2  4
```

(ii) Rank  $\text{rank}(\mathbf{A})$ :

```
>> rank(A)
2
```

(iii) Determinant  $\det(\mathbf{A})$ , provided that  $\mathbf{A}$  is a square matrix.

```
>> det(A)
ans = -2
```

(iv) Powers  $\mathbf{A}^n$ , provided that  $\mathbf{A}$  is a square matrix. If  $n < 0$ ,  $\mathbf{A}$  needs to be invertible (that is, non-singular).

```
>> A^2
ans =  7  10
      15  22
```

(v) If  $\mathbf{A}$  is invertible, its inverse can be evaluated using either `A^(-1)` or `inv(A)`.

```
>> inv(A)
ans = -2.0000  1.0000
      1.5000 -0.5000
```

(vi) The adjoint  $\text{adj}(\mathbf{A})$ , provided that  $\mathbf{A}$  is a square matrix.

```
>> adjoint(A)
ans =  4.0000 -2.0000
      -3.0000  1.0000
```

(vii) Matrix product  $\mathbf{AB}$ , provided that the sizes are matched.

```
>> A * B
ans =  8  11
      20  23
```

Moreover, we can generate special matrices using the following command:

(i) Zero matrix  $\mathbf{0}_{m \times n}$  of size  $m \times n$ : `zeros(m,n)`.

```
>> zeros(2,3)
ans =  0  0  0
```

0 0 0

(ii) Identity matrix  $I_n$  of order  $n$ : `eye(n)`.

```
>> eye(2)
ans =  1  0
       0  1
```

(iii) Diagonal matrix with diagonal entries  $a_1, \dots, a_n$ : `diag([a1 ... an])`.

```
>> diag(2,3)
ans =  2  0
       0  3
```

**17.2. Row and Column Operations.** Let  $A$  be a matrix. For example,

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 5 & 6 & 7 \\ 4 & 5 & 6 & 7 & 8 \end{pmatrix}.$$

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

(i) The size of  $A$  is given by

```
>> size(A)
ans =  4  5
```

(ii) The  $(i, j)$  entry of  $A$  is simply `A(i,j)`. For example,

```
>> A(2,5)
ans =  6
```

(iii) The  $i$ th row of  $A$ : `A(i, :)`. For example,

```
>> A(4, :)
ans =  4  5  6  7  8
```

If we need more rows, indicate the indices in square brackets. For example,

```
>> A([2,4], :)
ans =  2  3  4  5  6
       4  5  6  7  8
```

(iv) The  $i$ th column of  $A$  is given by `A(:, i)`.

```
>> A(:, 3)
ans =  3
```

4

5

6

(v) If we need more columns, indicate the indices in square brackets.

```
>> A(:, [3,4])
```

```
ans = 3 4
```

4 5

5 6

6 7

(vi) Submatrix of  $A$ . Simply indicate the required rows and columns in square brackets.

```
>> A([1,2], [3,4])
```

```
ans = 3 4
```

4 5

We can perform the elementary row operations as follows:

(i) Multiplying the  $i$ th row by a constant  $c$ :  $A(i,:) = c*A(i,:)$ .

```
>> A(1,:) = -2*A(1,:);
```

```
A = -2 -4 -6 -8 -10
```

2 3 4 5 6

3 4 5 6 7

4 5 6 7 8

(ii) Interchanging the  $i$ th and  $j$ th rows:  $A([i,j],:) = A([j,i],:)$ .

```
>> A([2,3],:) = A([3,2],:);
```

```
A = -2 -4 -6 -8 -10
```

3 4 5 6 7

2 3 4 5 6

4 5 6 7 8

(iii) Adding  $c$  times of the  $i$ th row to the  $j$ th row:  $A(j,:) = A(j,:) + c*A(i,:)$ .

```
>> A(4,:) = A(4,:) + 2*A(1,:);
```

```
A = -2 -4 -6 -8 -10
```

3 4 5 6 7

2 3 4 5 6

40   -3   -6   -9   -12

(iv) The reduced row echelon form of  $A$  is given by

```
>> rref(A)
ans =  1  0  -1  -2  -3
       0  1  2  3  4
       0  0  0  0  0
       0  0  0  0  0
```

Similarly, we can perform the elementary column operations as follows:

- (i) Multiplying the  $i$ th column by a constant  $c$ :  $A(:,i) = c*A(:,i)$ .
- (ii) Interchanging the  $i$ th and  $j$ th columns:  $A(:,[i,j]) = A(:,[j,i])$ .
- (iii) Adding  $c$  times of the  $i$ th column to the  $j$ th column:  $A(:,j) = A(:,j) + c*A(:,i)$ .

## 18. LINEAR SYSTEM

**18.1. Homogeneous Linear System.** Let  $A$  be a matrix. The solution set of the homogeneous linear system  $Ax = 0$  can be obtained by back-substitution from its reduced row echelon form.

For example, let  $A = \begin{pmatrix} 0 & 3 & -6 & 6 & 4 \\ 3 & -7 & 8 & -5 & 8 \\ 3 & -9 & 12 & -9 & 6 \end{pmatrix}$ .

```
>> A = [0 3 -6 6 4; 3 -7 8 -5 8; 3 -9 12 -9 6];
>> rref(A)
ans =  1  0  -2  3  0
       0  1  -2  2  0
       0  0  0  0  1
```

Alternatively, we note that the solution set of  $Ax = 0$  is precisely the nullspace of  $A$ . The command `null(A)` finds a basis for the nullspace of  $A$  (using column vectors).

```
>> null(A)
ans = -0.7952  0.1467
       -0.3797  0.5633
       0.2256,  0.6983
       0.4155  0.4166
       -0.0000 -0.0000
```

Let  $\mathbf{v}_1 = (-0.7952, 0.3797, 0.2256, 0.4155, -0.0000)^T$  and  $\mathbf{v}_2 = (0.1467, 0.5633, 0.6983, 0.4166, -0.0000)^T$ . Then  $\mathbf{A}\mathbf{x} = \mathbf{0}$  has a general solution  $\mathbf{x} = c_1\mathbf{v}_1 + c_2\mathbf{v}_2$ .

**18.2. Non-Homogeneous Linear System.** Let  $\mathbf{A}$  be an  $m \times n$  matrix and  $\mathbf{b}$  an  $m \times 1$  vector. Recall that the linear system  $\mathbf{A}\mathbf{x} = \mathbf{b}$  can be solved as follows:

- (i) Find a particular solution to  $\mathbf{A}\mathbf{x} = \mathbf{b}$  (if the system is consistent), say  $\mathbf{x}_p$ .
- (ii) Find the general solution to the homogeneous system  $\mathbf{A}\mathbf{x} = \mathbf{0}$ , say  $\mathbf{x}_h$ .

Then the general solution to  $\mathbf{A}\mathbf{x} = \mathbf{b}$  is  $\mathbf{x} = \mathbf{x}_p + \mathbf{x}_h$ .

One may use the back-substitution method to solve the system from the reduced row echelon form of the augment matrix  $(\mathbf{A} \mid \mathbf{b})$ .

For example, let  $\mathbf{A}$  be the matrix previously defined, and  $\mathbf{b} = \begin{pmatrix} -5 \\ 9 \\ 15 \end{pmatrix}$ .

```
>> b = [-5; 9; -15];
>> rref([A b])
ans =  1  0  -2  3  0  -24
       0  1  -2  2  0  -7
       0  0  0  0  1  4
```

Alternatively, both `linsolve(A,b)` and `A\b` gives a particular solution to the system  $\mathbf{A}\mathbf{x} = \mathbf{b}$ , provided that the system is consistent.

```
>> linsolve(A,b)
ans = -17.0000
       0
       3.5000
       0
       4.0000
```

So the system  $\mathbf{A}\mathbf{x} = \mathbf{b}$  has a particular solution  $\mathbf{v} = (-17, 0, 3.5, 0, 4)^T$ .

Recall that  $\mathbf{A}\mathbf{x} = \mathbf{0}$  has a general solution  $c_1\mathbf{v}_1 + c_2\mathbf{v}_2$ . Then  $\mathbf{A}\mathbf{x} = \mathbf{b}$  has a general solution  $\mathbf{x} = \mathbf{v} + c_1\mathbf{v}_1 + c_2\mathbf{v}_2$ , where  $c_1, c_2$  are arbitrary parameters.

**18.3. Least Squares Solution.** Sometimes the system  $\mathbf{A}\mathbf{x} = \mathbf{b}$  is inconsistent, we prefer to get the least squares solution, i.e., the solution to  $\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b}$ . A least squares solution can also be obtained by `linsolve(A,b)` or `A\b`.



For example,  $A = \begin{pmatrix} 4 & 0 \\ 0 & 2 \\ 1 & 1 \end{pmatrix}$  and  $b = \begin{pmatrix} 2 \\ 0 \\ 11 \end{pmatrix}$ .

One checks that the system is inconsistent. But we have least squares solution  $x = (1, 2)^T$ .

```
>> A = [4 0; 0 2; 1 1];
>> b = [2; 0; 11];
>> linsolve(A,b)
ans = 1.0000
      2.0000
```

## 19. MORE MATRIX OPERATIONS

**19.1. Orthonormal Basis.** Let  $V$  be a vector space spanned by vectors  $u_1, \dots, u_k$ . Using Gram-Schmidt process, one obtains an orthonormal basis for  $V$ . In MATLAB, `orth` can be used to generate orthonormal basis for  $V$ .

More precisely, `orth(A)` gives an orthonormal basis for the column space of the matrix  $A$ .

For example, let  $V$  be the vector space spanned by

$$v_1 = (-10, 2, -6, 16, 2), \quad v_2 = (13, 1, 3, -16, 1), \quad v_3 = (7, -5, 13, -2, -5), \quad v_4 = (11, 3, -3, 5, -7).$$

We shall first define a matrix  $A$  whose columns are  $v_1, v_2, v_3, v_4$ . Since it is easier to input row vectors in MATLAB, we may view each  $v_i$  as a row vector and take transpose:

$$A = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix}^T = \begin{pmatrix} -10 & 13 & 7 & 11 \\ 2 & 1 & -5 & 3 \\ -6 & 3 & 13 & -3 \\ 16 & -16 & -2 & 5 \\ 2 & 1 & -5 & -7 \end{pmatrix}.$$

```
>> A = [-10 2 -6 16 2; 13 1 3 -16 1; 7 -5 13 -2 -5; 11 3 -3 5 -7]';
>> orth(A)
ans = -0.6140 -0.5638 0.3459 0.3489
      0.0720 -0.0389 0.4409 0.2483
      -0.3406 -0.0986 -0.8114 0.3016
      0.7011 -0.6159 -0.1476 0.3117
      0.1016 0.5399 0.0764 0.7928
```

The columns of the resulting matrix is thus an orthonormal basis for  $V$ .

**19.2. Eigenvalues.** Let  $A$  be a square matrix of order  $n$ . Its characteristic polynomial is  $\det(xI_n - A)$ . Alternatively, we can use the command `charpoly`.

For example, let  $A = \begin{pmatrix} 4 & -1 & 6 \\ 2 & 1 & 6 \\ 2 & -1 & 8 \end{pmatrix}$ .

```
>> A = [4 -1 6; 2 1 6; 2 -1 8];
```

We shall first declare the variable of the polynomial, e.g.,  $x$ :

```
>> syms x;
```

Then either of the following commands evaluates the characteristic polynomial of  $A$ :

```
>> det(x*eye(3)-A)
```

```
ans = x^3 - 13*x^2 + 40*x - 36
```

```
>> charpoly(A,x)
```

```
ans = x^3 - 13*x^2 + 40*x - 36
```

The roots of the characteristic equation of  $A$  are precisely all the eigenvalues of  $A$ :

```
>> solve(charpoly(A,x) == 0, x)
```

```
ans = 2
```

```
2
```

```
9
```

Alternatively, MATLAB can evaluate the eigenvalues using `eig`.

```
>> eig(A)
```

```
ans = 9.0000
```

```
2.0000
```

```
2.0000
```

The command `[matrix1, matrix2] = eig(A)` can return the corresponding eigenvectors. Here `matrix2` is a diagonal matrix whose diagonal entries are the eigenvalues of  $A$  (counting multiplicities) and `matrix1` is a matrix whose columns are the corresponding (unit) eigenvectors of  $A$ .

```
>> [P, D] = eig(A)
```

```
P = -0.5774 -0.6122 0.3205
```

```
-0.5774 -0.7873 -0.9112
```

```
-0.5774 0.0728 -0.2587
```

```
D = 9.0000 0 0
```

```

0      2.0000      0
0      0      2.0000

```

Note that  $A$  is diagonalizable if and only if `matrix2` is invertible.

In this problem, one verifies that  $P$  is invertible and  $P^{-1}AP = D$ :

```

>> inv(P)*A*P
ans =  9.0000  -0.0000  -0.0000
       0.0000  2.0000   0
       0.0000  -0.0000  2.0000

```

**19.3. Differential Equation in Matrix Form.** A linear differential equation may be written in matrix form. For example,

$$\begin{cases} \frac{dx}{dt} = x + 2y + 1, \\ \frac{dy}{dt} = -x + y + t. \end{cases}$$

It can be expressed in matrix form:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 3 & -4 \\ 4 & -7 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 1 \\ t \end{pmatrix}.$$

Let  $x = \begin{pmatrix} x \\ y \end{pmatrix}$ ,  $A = \begin{pmatrix} 1 & 2 \\ -1 & 1 \end{pmatrix}$  and  $b = \begin{pmatrix} 1 \\ t \end{pmatrix}$ . Then the differential equation is simply  $x' = Ax + b$ .

The command `dsolve` can also be used for differential equation in matrix form:

```

>> syms x(t) y(t);
>> X = [x; y];
>> A = [3 -4; 4 -7];
>> b = [1; t];
>> [Sx, Sy] = dsolve(diff(X,t) == A*X + b)
Sx = (exp(-5*t)*(C1 + (2*exp(5*t)*(10*t - 7))/75))/2 + 2*exp(t)*(C2 + (exp(-t)*(t
- 1))/3)
Sy = exp(-5*t)*(C1 + (2*exp(5*t)*(10*t - 7))/75) + exp(t)*(C2 + (exp(-t)*(t
- 1))/3)

```