Mathematics with MATLAB

Hans-Petter Halvorsen

Mathematics with MATLAB

- MATLAB is a powerful tool for mathematical calculations.
- Type “help elfun” (elementary math functions) in the Command window for more information about basic mathematical functions.

Mathematics Topics

- Basic Math Functions and Expressions

$$z = 3x^2 + \sqrt{x^2 + y^2} + e^{\ln(x)}$$

- Statistics
  - mean, median, standard deviation, minimum, maximum and variance
- Trigonometric Functions

$$\sin(), \cos(), \tan()$$

- Complex Numbers

$$z = a + jb$$

- Polynomials

$$p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1}$$

Basic Math Functions

Create a function that calculates the following mathematical expression:

$$z = 3x^2 + \sqrt{x^2 + y^2} + e^{\ln(x)}$$

We will test with different values for x and y

We create the function:

```
function z=calcexpression(x,y)

z=3*x^2 + sqrt(x^2+y^2)+exp(log(x));
```

Testing the function gives:

```
>> x=2;
>> y=2;
>> calcexpression(x,y)
ans =
   16.8284
```

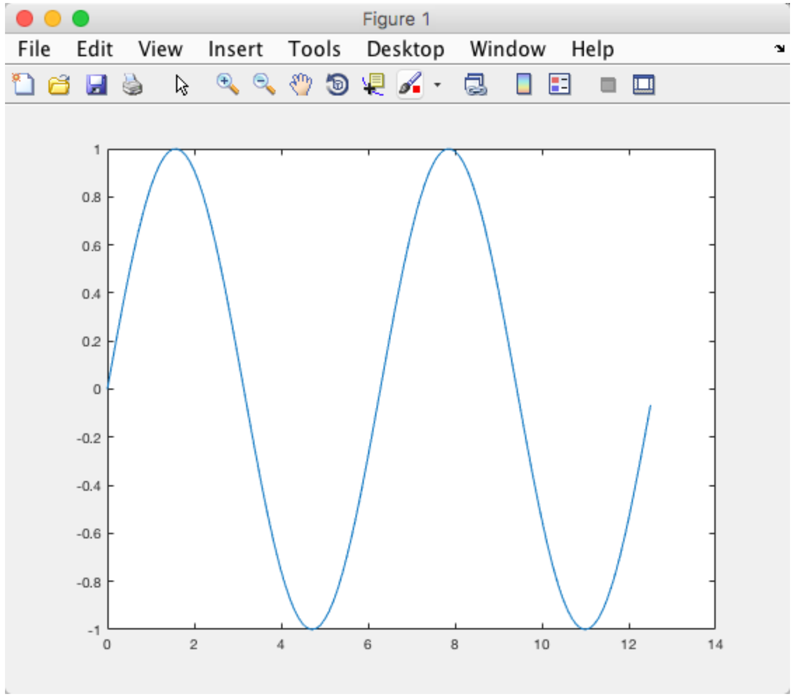
Statistics Functions

- MATLAB has lots of built-in functions for Statistics
- Create a vector with random numbers between 0 and 100. Find the following statistics: mean, median, standard deviation, minimum, maximum and the variance.

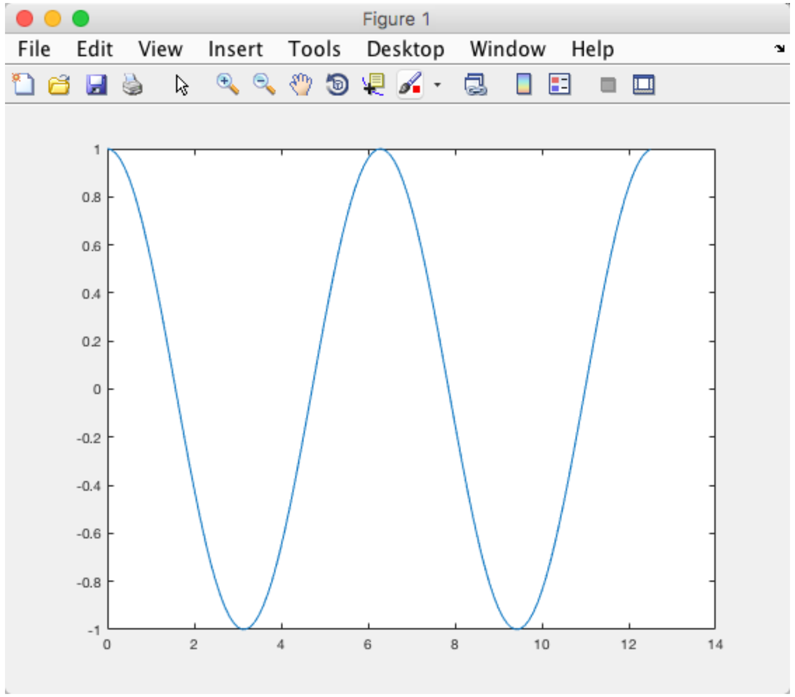
```
>> x=rand(100,1)*100;

>> mean(x)
>> median(x)
>> std(x)
>> mean(x)
>> min(x)
>> max(x)
>> var(x)
```

Trigonometric functions

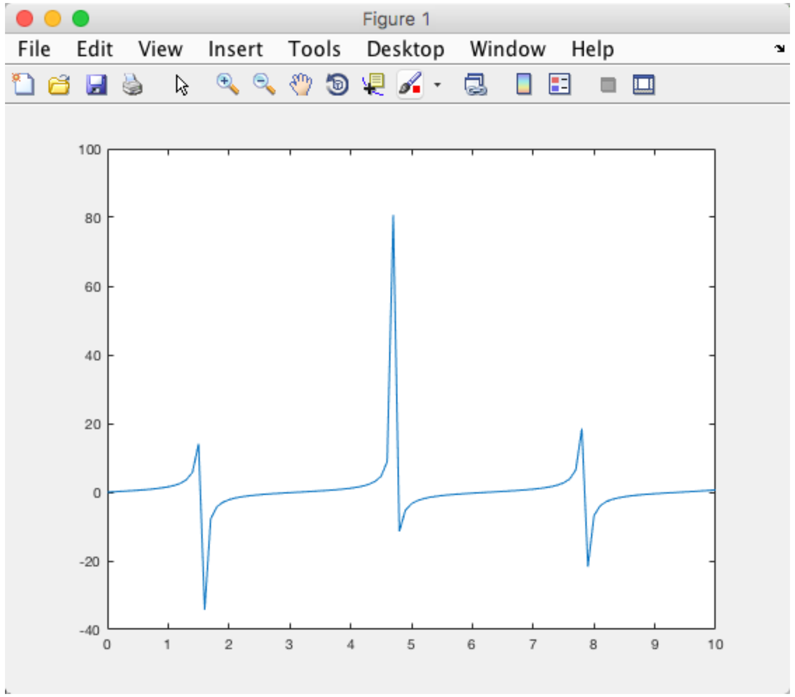
$\sin(x)$



$\cos(x)$



$\tan(x)$



Trigonometric functions

It is quite easy to convert from radians to degrees or from degrees to radians. We have that:

$$2\pi \, radians = 360 \, degress$$

This gives:

$$d\,[degrees] = \, r[radians] \cdot \left(\frac{180}{\pi}\right)$$

$$r[radians] = \, d[degrees] \cdot \left(\frac{\pi}{180}\right)$$

→ Create two functions that convert from radians to degrees (r2d(x)) and from degrees to radians (d2r(x)) respectively.

Test the functions to make sure that they work as expected.

The functions are as follows:

```
function d = r2d(r)

d=r*180/pi;
```

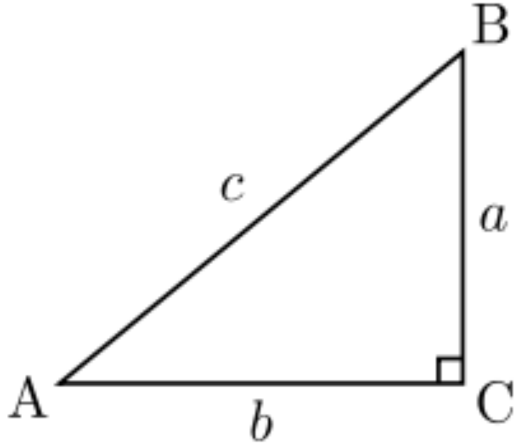
```
function r = d2r(d)

r=d*pi/180;
```

Testing the functions:

```
>> r2d(2*pi)
ans =
   360
>> d2r(180)
ans =
    3.1416
```

Trigonometric functions

Given right triangle:



- Create a function that finds the angle A (in degrees) based on input arguments (a,c), (b,c) and (a,b) respectively.
- Use, e.g., a third input “type” to define the different types above.
- Use your previous function **r2d()** to make sure the output of your function is in degrees and not in radians.
- Test the functions to make sure it works properly.

Trigonometric functions

We have that:

$$\sin A = \frac{a}{c}, A = arcsin\left(\frac{a}{c}\right)$$
$$\cos A = \frac{b}{c}, A = arccos\left(\frac{b}{c}\right)$$
$$\tan A = \frac{a}{b}, A = arctan\left(\frac{a}{b}\right)$$

The Pythagoras' theorem:

$$c^2 = a^2 + b^2$$

The function becomes as follows:

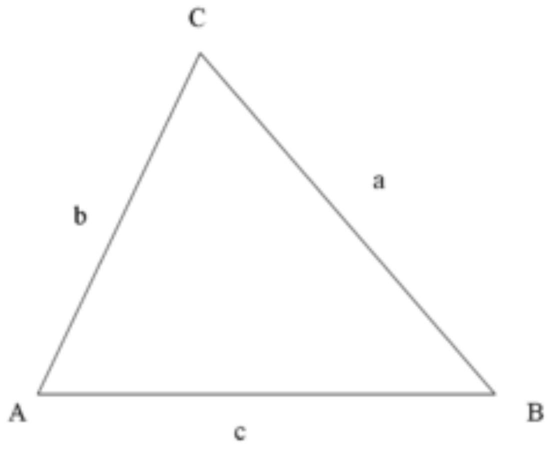
```
function angleA = right_triangle(x,y,
type)

switch type
    case 'sin'
        angleA=asin(x/y);
    case 'cos'
        angleA=acos(x/y);
    case 'tan'
        angleA=atan(x/y);
end

% Convert from radians to degrees
angleA = r2d(angleA);
```

Testing the function:

```
>> a=5
a =
     5
>> b=8
b =
     8
>> c=sqrt(a^2+b^2)
c =
    9.4340
>> right_triangle(a,c,'sin')
ans =
   32.0054
>> right_triangle(b,c,'cos')
ans =
   32.0054
>> right_triangle(a,b,'tan')
ans =
   32.0054
```

Law of cosines

Given:



Create a function where you find c using the **law of cosines**.

$$c^2 = a^2 + b^2 - 2ab\ cosC$$

The function becomes as follows:

```
function c = law_of_cosines(a,b,C)

c = sqrt(a^2 + b^2 - 2*a*b*cos(C));
```

Testing the function:

```
>> a=2;, b=3;, C=pi;,

>>law_of_cosines(a,b,C)
ans =
     5
```

Plotting Trigometric functions

- Plot $sin(\theta)$ and $cos(\theta)$ for $0 \leq \theta \leq 2\pi$ in the same plot.
- Make sure to add labels and a legend, and use different line styles and colors for the plots.

```
clf

x=0:0.01:2*pi;

plot(x, sin(x), 'c+:')
hold on

plot(x, cos(x), 'r:')
hold off

legend('sin', 'cos')
xlabel('x')
ylabel('f(x)')
```

```
clf

x=0:0.01:2*pi;

plot(x, sin(x), x, cos(x))
```

Or we can use Subplots:

```
% Define x-values
x=0:0.01:2*pi;

% subplot 1
subplot(2,1,1)
plot(x, sin(x))
title('Plotting sin(x)')
xlabel('x')
ylabel('sin(x)')

% Subplot 2
subplot(2,1,2)
plot(x, cos(x))
title('Plotting cos(x)')
xlabel('x')
ylabel('cos(x)')
```

Complex Numbers

A Complex Number is given by:

$$z = a + jb$$

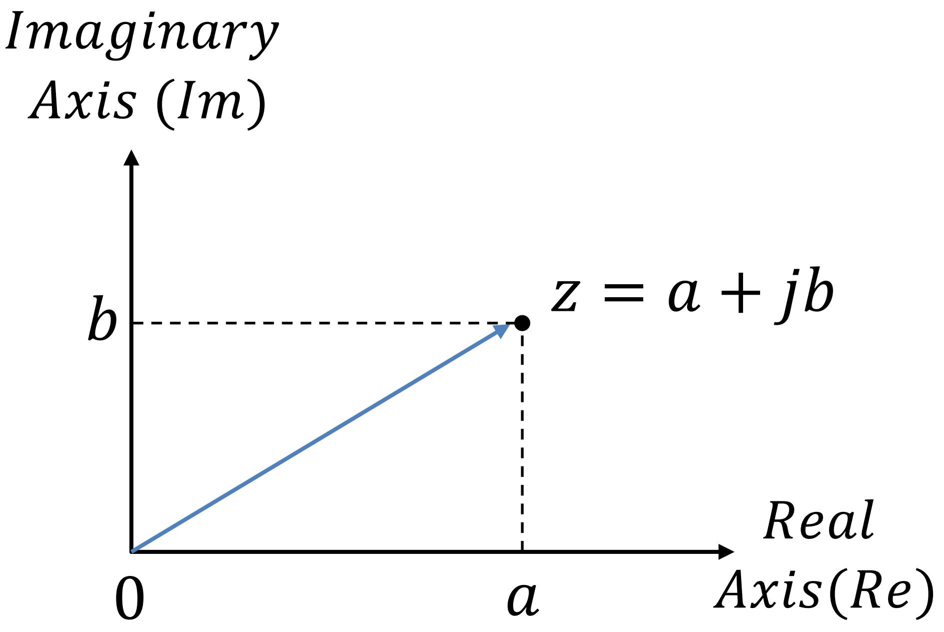
Where

$$j = \sqrt{-1}$$

We have that:

$$a = Re(z)$$

$$b = Im(z)$$



Complex Numbers

$j = \sqrt{-1}$

Polar form:

$$z = re^{j\theta}$$

Where:

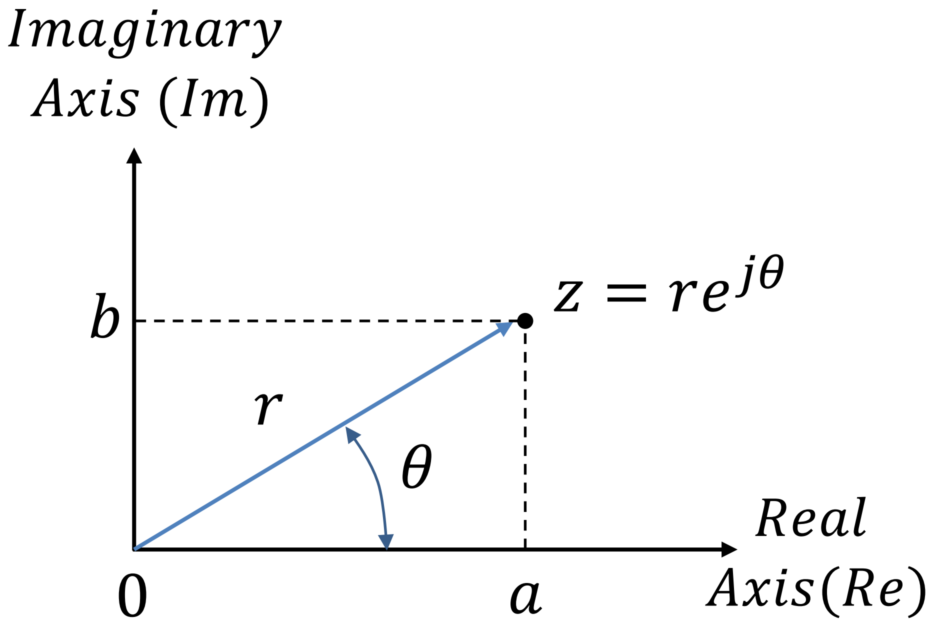
$$r = |z| = \sqrt{a^2 + b^2}$$

$$\theta = atan\frac{b}{a}$$

Note!

$$a = r\cos\theta$$

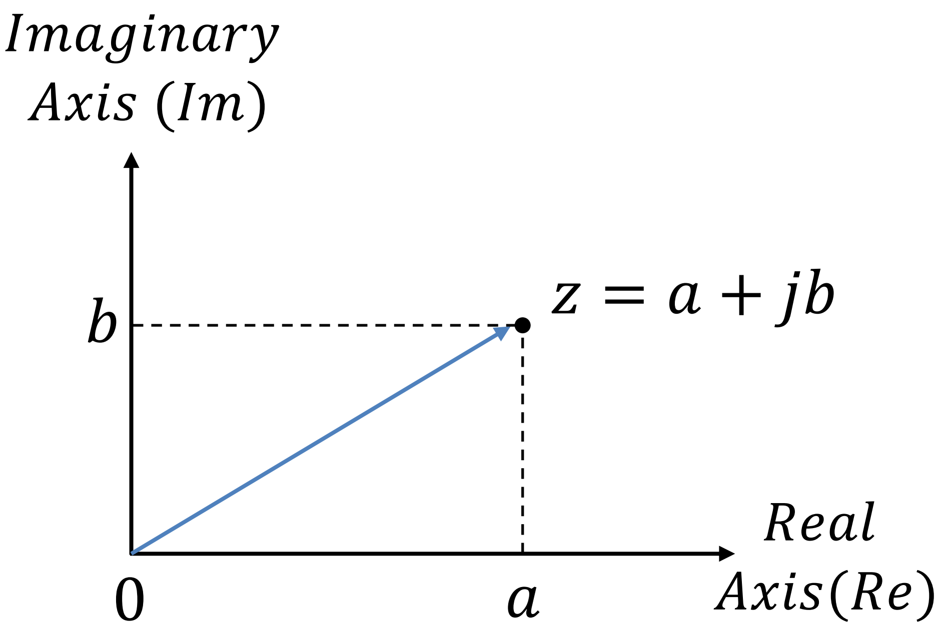
$$b = r\sin\theta$$



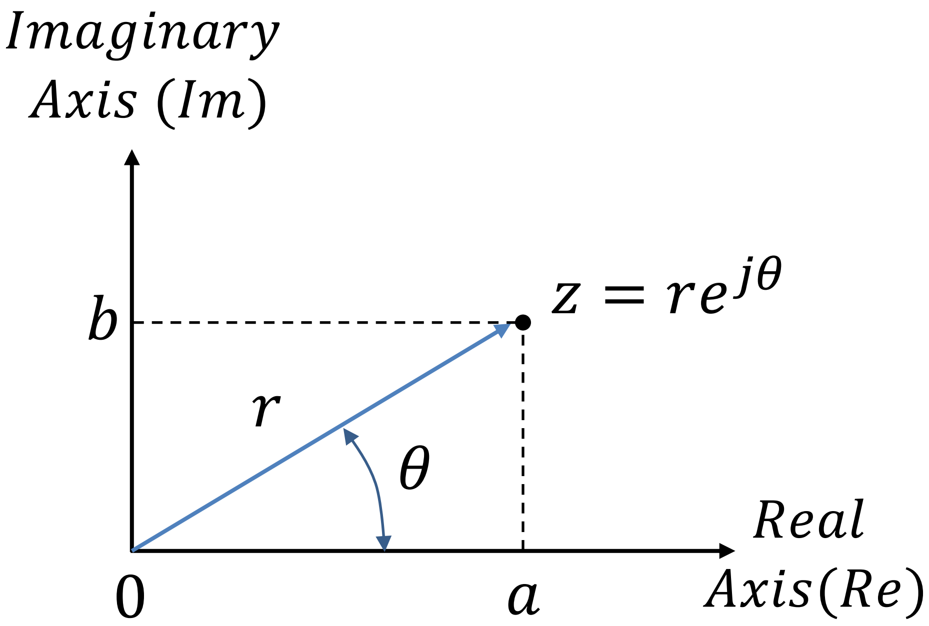
Complex Numbers

$j = \sqrt{-1}$

1 Rectangular form of a complex number



2 Exponential/polar form of a complex number



Length ("Gain"):

$$r = |z| = \sqrt{a^2 + b^2}$$

Angle ("Phase"):

$$\theta = atan\frac{b}{a}$$

Rectangular form → Exponential/polar form

Given the complex numbers (Rectangular form):

$$z = a + jb$$

Exponential/polar form:

$$z = re^{j\theta}$$

Where

$$r = \ |z| = \sqrt{a^2 + b^2}$$

```
z = a + bj
r = sqrt(a^2 + b^2)
```

or:

```
z = a + bj
r = abs(z)
```

$$\theta = atan\frac{b}{a}$$

```
z = a + bj
theta = atan(b/a)
```

or:

```
z = a + bj
theta = angle(z)
```

Rectangular form → Exponential/polar form

$$z = 5 + j3$$

```
clear
clc

a = 5;
b = 3;

% Rectangular Form:
z = a + b*i

% Polar Form:
r = sqrt(a^2 + b^2)
r = abs(z)

theta = atan(b/a)
theta = angle(z)

z = r*exp(j*theta)
```

$$z = re^{j\theta} = 5.83e^{j0.54}$$

$$z = 5 + j3$$

Complex Numbers

To add or subtract two complex numbers, we simply add (or subtract) their real parts and their imaginary parts.

Given the complex numbers:

$z_1 = a_1 + jb_1$ and $z_2 = a_2 + jb_2$

Addition:

$$z_3 = z_1 + z_2 = (a_1 + a_2) + j(b_1 + b_2)$$

Subtraction:

$$z_3 = z_1 - z_2 = (a_1 - a_2) + j(b_1 - b_2)$$

Complex Numbers

In Division and multiplication, we use the polar form.

Given the complex numbers:

$$z_1 = r_1 e^{j\theta_1} \text{ and } z_2 = r_2 e^{j\theta_2}$$

Multiplication:

$$z_3 = z_1 z_2 = r_1 r_2 e^{j(\theta_1 + \theta_2)}$$

Division:

$$z_3 = \frac{z_1}{z_2} = \frac{r_1 e^{j\theta_1}}{r_2 e^{j\theta_2}} = \frac{r_1}{r_2} e^{j(\theta_1 - \theta_2)}$$

Complex Numbers – MATLAB Functions

Function	Description	Example
i, j	Imaginary unit. As the basic imaginary unit SQRT(-1), i and j are used to enter complex numbers. For example, the expressions 3+2i, 3+2*i, 3+2j, 3+2*j and 3+2*sqrt(-1) all have the same value.	>>z=2+4i >>z=2+4j
abs	abs(x) is the absolute value of the elements of x. When x is complex, abs(x) is the complex modulus (magnitude) of the elements of X.	>>z=2+4i >>abs(z)
angle	Phase angle. angle(z) returns the phase angles, in radians	>>z=2+4i >>angle(z)
imag	Complex imaginary part. imag(z) is the imaginary part of z.	>>z=2+4i >>b=imag(z)
real	Complex real part. real(z) is the real part of z.	>>z=2+4i >>a=real(z)
conj	Complex conjugate. conj(x) is the complex conjugate of x.	>>z=2+4i >>z_con=conj(z)
complex	Construct complex result from real and imaginary parts. c = complex(a,b) returns the complex result A + Bi	>>a=2; >>b=3; >>z=complex(a,b)

Complex Numbers

Given two complex numbers:

$$c = 4 + j3, d = 1 - j$$

Find the real and imaginary part of c and d in MATLAB.

→ Use MATLAB to find $c + d, c - d, cd\ and\ c/d$.

Use the direct method supported by MATLAB and the specific complex functions **abs**, **angle**, **imag**, **real**, **conj**, **complex**, etc. together with the formulas for complex numbers .

→ Find also r and θ. Find also the complex conjugate.

```
c=5+3i;
d=1-i;

disp('c+d')
%Directly-----------------
z = c + d
%Manually-----------------
z_real = real(c) + real(d);
z_imag = imag(c) + imag(d);
z = complex(z_real,z_imag)

% r and angle + complex conungate
r=abs(z)
theta=angle(z)
complconj=conj(z)

disp('c-d')
%Directly--------------
z = c - d
%Manually--------------
z_real = real(c) - real(d);
z_imag = imag(c) - imag(d);
z = complex(z_real,z_imag)
%or: z = z_real + z_imag*i
```

```
disp('c*d')
%Directly-------------
z = c*d
%Manually-------------
z_abs = abs(c)*abs(d);
z_angle = angle(c) + angle(d);
z_real = z_abs*cos(z_angle);
z_imag = z_abs*sin(z_angle);
z = complex(z_real,z_imag)


disp('c/d')
%Directly-------------
z = c/d
%Manually-------------
z_abs = abs(c)/abs(d);
z_angle = angle(c) - angle(d);
z_real = z_abs*cos(z_angle);
z_imag = z_abs*sin(z_angle);
z = complex(z_real,z_imag)
```

This gives:

```
c+d
z =
   6.0000 + 2.0000i
z =
   6.0000 + 2.0000i
r =
    6.3246
theta =
    0.3218
complconj =
   6.0000 - 2.0000i
c-d
z =
   4.0000 + 4.0000i
z =
   4.0000 + 4.0000i
c*d
z =
   8.0000 - 2.0000i
z =
   8.0000 - 2.0000i
c/d
z =
   1.0000 + 4.0000i
z =
   1.0000 + 4.0000i
```

Complex Roots

Find the roots of the equation:

$$x^2 + 4x + 13$$

The roots are given by:

$$x^2 + 4x + 13 = 0$$

Find also the Sum and Difference of the roots.

We can e.g., use the **solveeq** function we created in a previous example:

```
function x = solveeq(a,b,c)
if a~=0
    x = zeros(2,1);
    x(1,1)=(-b+sqrt(b^2-4*a*c))/(2*a);
    x(2,1)=(-b-sqrt(b^2-4*a*c))/(2*a);
elseif b~=0
        x=-c/b;
elseif c~=0
    disp('No solution')
else
   disp('Any complex number is a solution')
end
```

```
a=1;
b=4;
c=13;

solveeq(a,b,c)
```

Or we can use the **root** function:

```
a=1;
b=4;
c=13;

p=[a,b,c]
roots(p)
```

Note! the solution is complex conjugate.

The sum of two complex conjugate numbers is always real.

In our case:

```
ans =
    -4
```

While the difference is imaginary (no real part).

```
ans =
          0 + 6.0000i
```

Polynomials

A polynomial is expressed as:

$$p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + \ p_{n+1}$$

where $p_1, p_2, p_3, \ldots$ are the coefficients of the polynomial.

Example of polynomial:

$$p(x) = -5.45x^4 + 3.2x^2 + 8x + 5.6$$

Polynomials in MATLAB

MATLAB represents polynomials as row arrays containing coefficients ordered by descending powers.

Example:

$$p(x) = -5.45x^4 + 3.2x^2 + 8x + 5.6$$

```
>> p=[-5.45 0 3.2 8 5.8]
p =
   -5.4500         0    3.2000    8.0000    5.8000
```

MATLAB offers lots of functions on polynomials, such as **conv**, **roots**, **deconv**, **polyval**, **polyint**, **polyder**, **polyfit**, etc.

→ You should look up these functions in the Help system in MATLAB.

Polynomials

Define the following polynomial in MATLAB:

$$p(x) = -2.1x^4 + 2x^3 + 5x + 11$$

→ Find the roots of the polynomial ($p(x) = 0$)

→ Find $p(x = 2)$

Use the polynomial functions listed above.

MATLAB Code:

```
p = [-2.1, 2, 0, 5, 11]

roots(p)

x = 2;
polyval(p,x)
```

This gives:

```
p =
   -2.1000   2.0000   0   5.0000   11.0000
ans =
   2.0820
  -0.0199 + 1.5193i
  -0.0199 - 1.5193i
  -1.0898
ans =
    3.4000
```

$$p(x = 2)$$

$$p(x) = 0$$

Instead of using the polyval() function we could of course also found the answer like this:

```
x = 2;
p = -2.1*x^4 + 2*x^3+5*x+11
```

We can use e.g., the polyval() function to check if the answers are correct:

```
>> x = 2.0820;
>> polyval(p,x)
```

etc.

The answers shall then of course be 0 (or at least a very small number).

Polynomials

Given the following polynomials:

$$p_1(x) = 1 + x - x^2$$
$$p_2(x) = 2 + x^3$$

→ Find the polynomial $p(x) = p_1(x) \cdot p_2(x)$ using MATLAB and find the roots

→ Find the roots of the polynomial ($p(x) = 0$)

→ Find $p(x = 2)$

→ Find the differentiation/derivative of $p_2(x)$, i.e., $p_2{}'$

We will use the polynomial functions listed above.

Note!

The polynomials may be rewritten as:

$$p_1(x) = -x^2 + x + 1$$

$$p_2(x) = x^3 + 0x^2 + 0x + 2$$

The MATLAB code becomes:

```
P1 = [-1, 1, 1];
P2 = [1, 0, 0, 2];

p = conv(p1,p2)

r = roots(p)

polyval(p,2)
```

This gives:

```
p =
     -1      1      1     -2      2      2
r =
    1.6180
    0.6300 + 1.0911i
    0.6300 - 1.0911i
   -1.2599
   -0.6180
ans =
    -10
```

The Polynomial becomes:

$$p(x) = -x^5 + x^4 + x^3 - 2x^2 + 2x + 2$$

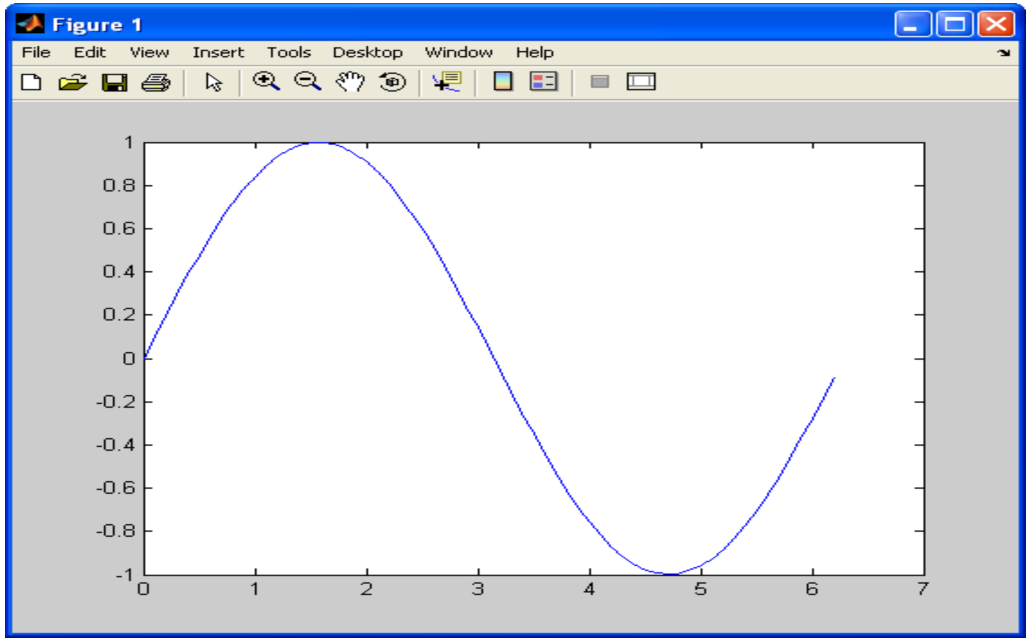
Polynomial Fitting

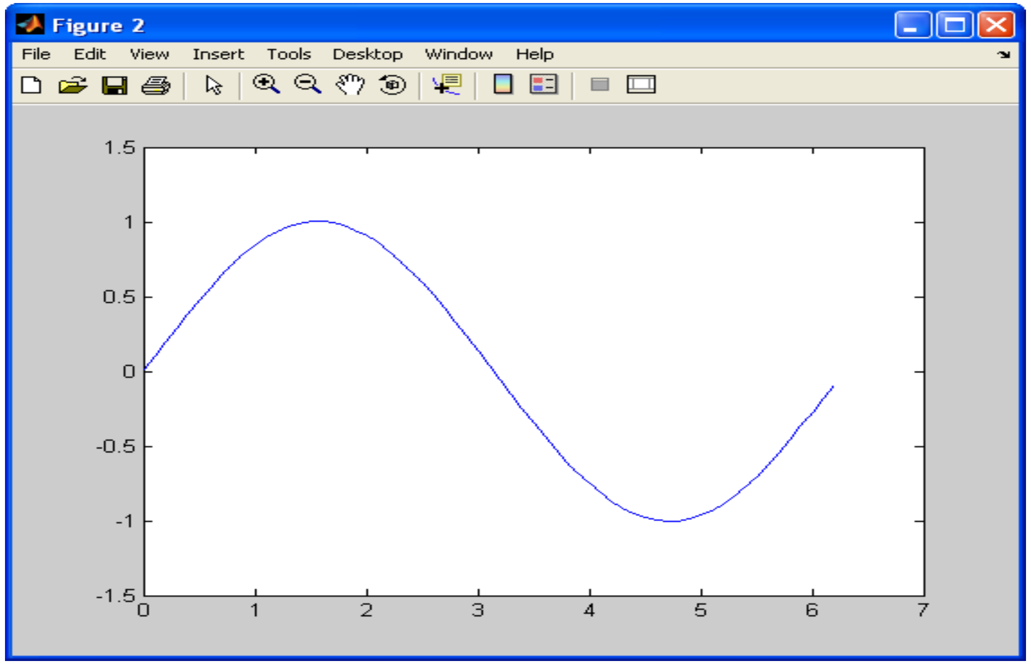
Find the 6.order Polynomial that best fits the following function:

$$y = \sin(x)$$

→ Plot both the function and the 6. order Polynomial to compare the results.

```
x=0:0.1:2*pi;
y=sin(x);

figure(1)
plot(x,y)


% Finding a 6. order polynomial
p=polyfit(x,y,6)

y2=polyval(p,x);

figure(2)
plot(x,y2)
```





Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: https://www.halvorsen.blog