**BAB III**

**GRAFIK, M-FILE DAN PEMROGRAMAN M-FILE**

1. **TUJUAN**

* Mahasiswa dapat menggunakan perintah-perintah dalam pembuatan grafik
* Dapat membuat *M-file* sederhana
* Dapat menjalankan, membuat fungsi dan menjalankan fungsi dengan *M-file*
* Dapat membuat program dalam *M-file*

# PENDAHULUAN

## Grafik

Dalam MATLAB terdapat fungsi yang digunakan untuk menampilkan sebuah grafik dari suatu fungsi menggunakan perintah *plot* (grafik 2 dimensi). Untuk menambahkan keterangan pada grafik dapat menggunakan perintah berikut:

***title*(‘teks’)** untuk menampilkan judul pada grafik

***xlabel*(‘teks’)** untuk memberi nama pada sumbu-x grafik

***ylabel*(‘teks’)** untuk memberi nama pada sumbu-y grafik

***text*(2,4,’Titik 2’)** untuk menampilkan teks‘Titik 2’ pada lokasi x=2 dan y=4

***gtext*(’Titik 3’)** untuk menampilkan teks ‘Titik 3’ dengan cara meng-*click*

kursor pada sembarang lokasi yang diinginkan

Untuk membuat grafik dalam skala logaritma atau semilogaritma, perintah plot dapat ditukar dengan *loglog* atau *semilog* dengan menggunakan cara yang sama. Jika terdapat lebih dari 1 grafik, semisal 2 grafik, maka pada layar grafik hanya muncul grafik yang kedua, demikian seterusnya. Untuk mengatasi permasalahan tersebut, dapat ditambahkan perintah *figure*(*n*) dimana *n* menunjukkan nomor grafik. Akibatnya akan muncul sebanyak *n* buah layar grafik yang baru.

Dapat juga beberapa grafik dapat ditampilkan kedalam sebuah layar saja dengan menggunakan perintah:

subplot(m,n,k) atau subplot(mnk)

dimana *m* menunjukkan baris, *n* menunjukkan kolom, dan *k* menunjukkan grafik yang ke berapa. Misalnya: subplot(1,3,1) artinya terdapat sebanyak 3 grafik dalam 1 baris dimana grafik yang dimaksud pada perintah ini diletakkan pada kolom 1. Sebagai default, MATLAB memilih *style* garis lurus serta warna biru. Pada perintah plot dapat ditambahkan tambahan argumen untuk memilih warna dan *style* untuk grafik yang akan dibuat.

Contoh: plot(x, y, ‘r+’) akan menghasilkan grafik dengan warna merah (*red*) dan style garis yang merupakan gabungan tanda +. Diantara contoh warna, penandaan, dan *style* garis yang disediakan MATLAB adalah:



## M-File

*M-file* merupakan sebuah *file* yang dapat di akses oleh MATLAB. *M-file* dapat berupa sebuah fungsi. Disebut sebagai *M-file* karena filenya berextension *m*. *M-file* merupakan sederetan perintah yang disimpan dalam bentuk skrip teks. Suatu *M-file* dapat dijalankan/dieksekusi dengan cara mengetikkan nama *M-file* tersebut pada *command window*. Hal yang perludiperhatikan dalam *M-File*:

* + Dalam *M-file*, setiap perintah diakhiri dengan titik-koma agar hasil perhitungan setiap baris tidak ditampilkan. Kecuali pada hasil perhitungan yang akan ditampilkan, tidak diakhiri titik-koma.
  + Variabel yang didefinisikan didalam *M-file* akan disimpan oleh MATLAB ketika *M-file* telah dijalankan.

Di dalam editor *M-File*, perintah yang dituliskan akan memiliki warna tertentu:

* + hijau untuk komentar
  + hitam untuk variabel dan perintah
  + biru untuk *statement* pemrograman

## *M-File* Sebagai Fungsi

Sebagai skrip program, jika akan mengubah/mengatur parameter *input* program, maka perlu dilakukan di dalam *editor windows*. Namun seringkali program tersebut harus dijalankan berulang kali dengan menggunakan nilai masukan yang berbeda. Contoh pada kasus iterasi atau optimasi. Untuk keperluan tersebut dapat dilakukan dengan menuliskan *M-file* sebagai suatu fungsi yang spesifik. Tiga unsur yang terdapat dalam fungsi:

1. Parameter masukan; disebut sebagai “argumen input”. Jumlah parameter (argumen) tersebut bisa berapapun (satu, dua, sepuluh, atau tidak ada argumen input). Jenis argumen pun bebas (variabel, bilangan ataupun teks).
2. Proses di dalam sebuah program; berupa sederetan perintah untuk menjalankan suatu algoritma.
3. Parameter keluaran; atau “argumen output” dengan jumlah dan jenisnya yang bebas.

Deklarasi fungsi di *M-file* harus dilakukan pada baris awal dengan sintaks:

**function [argumen output] = nama\_fungsi(argumen input)**

## Sistem Kendali

Seperti halnya bahasa program pada umumnya, program dapat diarahkan dengan berbagai cara, seperti: percabangan yang merupakan arah program yang berdasarkan pada kondisi tertentu, ataupun loop (perulangan) digunakan untuk melakukan iterasi.

### Pernyataan *if* … *elseif* … *else* … *end*

Merupakan *statement* untuk percabangan yang berdasarkan pada satu/beberapa keadaan tertentu. Perintah yang digunakan dalam MATLAB sebagai berikut:

**if keadaan**

**Perintah yang dikerjakan ketika keadaan terpenuhi**

**end**

**if keadaan**

**Perintah yang dijalankan ketika keadaan dipenuhi**

**else**

**Dijalankan apabila keadaan tidak dipenuhi**

**end**

**if keadaan1**

**Perintah yang dijalankan jika keadaan1 dipenuhi**

**elseif keadaan2**

**Dijalankan jika kondisi2 dipenuhi**

**elseif keadaan3**

**Dijalankan jika keadaan3 dipenuhi**

**elseif ...**

**...dan seterusnya...**

**else**

**Dijalankan jika keadaan manapun tidak dipenuhi**

**end**

**if keadaan1**

**perintah1**

**if keadaanA**

**perintahA**

**else**

**perintahB**

**end**

**else**

**perintah2**

**end**

### Pernyataan *Switch* … *case*

*Statement* yang digunakan dalam perintah *Switch* … *case* adalah:

**switch nama\_variabel**

**case{keadaan1, keadaan2,...}**

**Dijalankan jika keadaan1 atau keadaan2 dst... dipenuhi**

**case{ keadaanA, keadaan B,...}**

**Dijalankan jika keadaanA atau keadaanB dst... dipenuhi**

**case{ keadaanX, keadaanY,...}**

**Dijalankan jika keadaanX atau keadaan Y dst...**

**dipenuhi**

**case{...}**

**...dst...**

**default**

**Dijalankan jika keadaan manapun tidak dipenuhi**

**end**

### Pernyataan *for* … *end*

Pernyataan ini digunakan untuk *loop*/perhitungan berulang. Pernyataan yang digunakan dalam perintah *for* … *end* adalah:

**For variabel = nilai\_awal : inkremen : nilai\_akhir**

**Perintah yang dikerjakan**

**end**

### Pernyataan *while* … *end*

Pernyataan yang digunakan dalam perintah *while* … *end* adalah:

**while keadaan**

**perintah yang dikerjakan apabila keadaan dipenuhi**

**end %keluar dari perulangan jika keadaan tidak dipenuhi**

### Pernyataan *Break* dan *Return*

Dalam sebuah *loop*, dapat keluar dengan *break* tanpa menunggu nilai\_akhir tercapai, atau tanpa menunggu kondisi *loop* tidak dipenuhi lagi. Sementara, *return* digunakan untuk keluar dari sebuah fungsi yang sedang berjalan.

### Pernyataan *Continue*

Pernyatan *continue* digunakan untuk memaksa program menuju iterasi berikutnya dalam sebuah *loop*, tanpa mengerjakan perintah yang masih ada di bawahnya.

## Operator Perbandingan dan Logika

Perbandingan dua variabel (sama atau tidak, lebih besar atau lebih kecil), mengevaluasi apakah suatu variabel memenuhi satu dari sejumlah syarat yang telah ditentukan, dan sebagainya.

Operasi dua variabel menggunakan operator berikut:

< > lebih kecil, lebih besar

<= >= lebih kecil atau sama dengan, lebih besar atau sama dengan

== ~= sama dengan, tidak sama dengan

Operasi logika

*and*(*A*,*B*) atau *A* *&* *B* operasi logika AND antara A dan B

*or*(*A*,*B*) atau *A* | *B* operasi logika OR

*xor*(*A*,*B*) operasi logika XOR

*not*(*A*) atau *~A* operasi logika NOT pada A

# LANGKAH PRAKTIKUM

## Grafik 2 Dimensi

>>x = [ 1 2 3];

>>y = [2 4 9];

>>plot(x,y)

>> x = 1:8; y=[20 22 25 30 28 25 24 22];

>> plot(x,y)

Menampilkan grafik pada kurva y= x3 pada rentang x= -3 hingga x= +3.

>> clear

>> x=-3:0.1:3; %inkremen=0.1 agar kurva terlihat mulus

>> y=x.^3;

>> plot(x,y)

>> xlabel('Sumbu X'), ylabel('Sumbu Y')

>> title('Kurva Y=X^3')

>> grid on

Menampilkan beberapa kurva dalam satu gambar

>> clear

>> x=linspace(0,5,500);

>> y1=exp(-x); plot(x,y1);

>> grid on

>> hold on

>> y2=exp(-0.5\*x); plot(x,y2);

>> y3=exp(-0.25\*x); plot(x,y3);

>> y4=exp(-0.1\*x); plot(x,y4);

>> xlabel('sumbu-x'), ylabel('sumbu-y')

>> title('Perbandingan fungsi eksponensial ... negatif')

Penggunaan warna dalam grafik

>> clear

>> x=linspace(0,5,500);

>> y1=exp(-x); y2=exp(-0.5\*x); y3=exp(-0.25\*x);

>> y4=exp(-0.1\*x);

>> plot(x,y1,x,y2,x,y3,x,y4)

>> grid on

>> xlabel('sumbu-x'), ylabel('sumbu-y')

>> title('Kurva y = exp(-Ax)')

>> legend('A=1','A=0.5','A=0.25','A=0.1')

Menampilkan empat buah grafik pada satu jendela

>> figure

>> t=0:0.05:10;

>> sinus=sin(2\*pi\*0.25\*t);

>> cosinus=cos(2\*pi\*0.25\*t);

>> kotak=square(2\*pi\*0.25\*t);

>> gigi=sawtooth(2\*pi\*0.25\*t);

>> subplot(2,2,1);

>> plot(t,sinus), title('sinus 1/4 Hz')

>> subplot(2,2,2);

>> plot(t,cosinus), title('cosinus 1/4 Hz')

>> subplot(2,2,3);

>> plot(t,kotak), title('kotak 1/4 Hz')

>> subplot(2,2,4);

>> plot(t,gigi), title('gigi gergaji 1/4 Hz')

Menggambar grafik suatu fungsi ρ= sin2(3θ) dalam koordinat polar.

>> figure

>> theta=linspace(0,2\*pi,500);

>> rho=(cos(theta.\*3)).^2;

>> polar(theta,rho);

## Grafik 3 Dimensi

Menampilkan grafik garis

>> X = [10 20 20 10 10];

>> Y = [5 5 15 15 5];

>> Z = [0 0 70 70 0];

>> plot3(X,Y,Z); grid on;

>> xlabel('sumbu X'); ylabel('sumbu Y');

>> zlabel('sumbu Z');

>> title ('Contoh plot 3-D');

>> axis([0 25 0 20 0 80])

Contoh menggambar *Helix*

>> t=0:0.1:25;

>> X=sin(t); Y=cos(t); Z=0.5\*t;

>> plot3(X,Y,Z)

>> xlabel('sumbu X'); ylabel('sumbu Y');

>> zlabel('sumbu Z');

>> title ('Helix');

Penggunaan perintah *mesh* dan *surf* untuk menggambar fungsi dua variabel z = x2+ y2

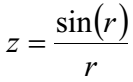
>> batas\_x = -10:1:10; batas\_y = -10:4:10;

>> [X,Y] = meshgrid(batas\_x,batas\_y);

>> Z = X.^2 + Y.^2;

>> mesh(X,Y,Z);

>> surf(X,Y,Z);

Mengambar fungsi 3 Dimensi untuk  dengan rentang

>> x = linspace(-10,10,40); y = x;

>> [X,Y] = meshgrid(x,y);

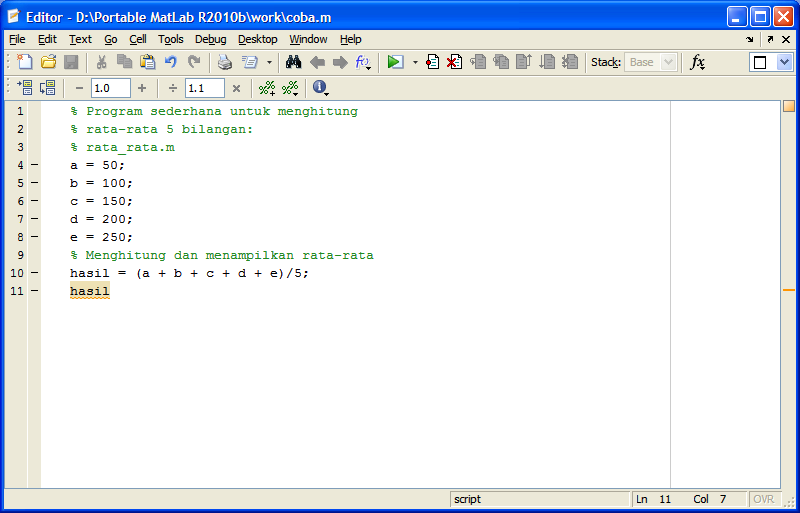
>> R = sqrt(X.^2+Y.^2);

>> Z = sin(R)./(R+eps);

>> surf(X,Y,Z);

## Membuat M-File

1. Untuk membuat *M-file* baru dapat dilakukan dengan menekan tombol File-New-Script Atau Ctrl + N.
2. Kemudian muncul Tampilan seperti berikut ini



1. Untuk menyimpan tekan File-Save
2. Tuliskan Program berikut ini

% Program sederhana untuk menghitung

% rata-rata 5 bilangan:

% rata\_rata.m

a = 50;

b = 100;

c = 150;

d = 200;

e = 250;

% Menghitung dan menampilkan rata-rata

hasil = (a + b + c + d + e)/5;

hasil

Teks yang diawali tanda “%” menunjukkan komentar, dan tidak akan dieksekusi oleh MATLAB.

1. Simpan dalam folder yang aktif dengan nama rata2.
2. Untuk mejalankan gunakan perintah berikut

>> rata2

hasil =

150

## M-File Sebagai Fungsi

Contoh fungsi untuk menghitung sisi miring, luas, dan keliling segitiga

%Fungsi untuk menghitung segitiga siku-siku: segitiga.m

%Untuk menghitung sisi miring, luas, dan keliling

function [Sisi\_miring,Luas,Keliling] = segi3(Sisi\_A,Sisi\_B,Sisi\_C)

% Menghitung sisi miring

Sisi\_miring = sqrt(Sisi\_A^2 + Sisi\_B^2);

% Menghitung luas segitiga

Luas = 1/2\* Sisi\_A \* Sisi\_B;

% Menghitung keliling

Keliling = Sisi\_A + Sisi\_B + Sisi\_C;

Simpan dengan nama segi3.m

Panggil fungsi tersebut dengan menggunakan perintah berikut:

>> [sisi\_miring,luas,keliling]=segi3(12,13,5)

## Display dan Input

Tulis program berikut

% Program menghitung kombinasi : Jumlah\_kombinasi.m

% untuk menghitung jumlah kombinasi

% dari sejumlah populasi

% Menampilkan judul program

clc;

disp('Menghitung Jumlah Kombinasi');

disp('---------------------');

% Meminta masukan dari user

n = input('Berapa mahasiswa yang ada? : ');

r = input('Berapa jumlah dalam satu team? : ');

% Menghitung kombinasi

faktor\_n=1;

for i=1:1:n

faktor\_n=faktor\_n\*i;

end

m=n-r;

faktor\_m=1;

for i=1:1:m

faktor\_m=faktor\_m\*i;

end

faktor\_r=1;

for i=1:1:r

faktor\_r=faktor\_r\*i;

end

kombinasi = faktor\_n/(faktor\_m\*faktor\_r);

% Menampilkan keluaran

disp(['Jumlah kombinasi yang ada = ',num2str(kombinasi)]);

Jalankan program:

>> Jumlah\_kombinasi

## Statemen Kendali

Fungsi IF

nilai = input('Masukkan Nilai ; ')

if nilai > 60

disp('Selamat anda Lulus')

else

disp('Maaf anda tidak Lulus')

end

Simpan dengan nama ujian1.m

nama = input('Nama Siswa = ', 's');

N1 = input('Nilai Ujian 1 = ');

N2 = input('Nilai Ujian 2 = ');

NR = (N1+N2)/2;

if NR > 60

ket = 'lulus';

else

ket = 'gagal';

end

disp(ket)

Simpan dengan nama ujian2.m

## Statement For

Menampilkan bilangan kelipatan 3 dari 30 sampai 100.

for k = 30:3:100

k

end

Menampilkan bilangan bulat dari –10 hingga 10 dan menyimpannya dalam satu vektor

Vektor=[];

for k = -10:10 %dalam hal ini inkremen = 1

Vektor = [Vektor k];

end

Vektor

Menampilkan kurva parabola: y = Ax2

figure;

x = linspace(-4,4,500); % mendefinisikan nilai x

A = 0.5:0.5:2; % mendefinisikan vektor A

for i = 1:length(A)

y = A(i)\* x.^2;

plot(x,y);

hold on;

end

grid on;

## Switch-case

clc

disp('1. Metoda Substitusi Berurut')

disp('2. Metoda Newton-Raphson')

disp('3. Metoda Tali Busur')

n = input('Metoda yang dipilih = ');

switch n

case (1), disp('Metoda Substitusi Berurut')

case (2), disp('Metoda Newton-Raphson')

case (3), disp('Metoda Tali Busur')

otherwise

disp('Metoda tidak termasuk dalam daftar')

end

Simpan dengan nama swc.m

## Statement While

Manampilkan fungsi akar kuadrat: y = B x1/2

figure;

x=linspace(0,4,500);

A=0.5:0.5:2;

i=1;

while i <= length(A)

y = A(i)\* x.^(1/2);

plot(x,y);

hold on;

i=i+1;

end

grid on;

Contoh program untuk memeriksa suatu bilangan apakah bilangan prima atau bukan.

% Fungsi untuk menentukan bilangan prima dari suatu bilangan:

% apa\_prima.m

%

function bilprima(N)

% N : bil. bulat positif yang dimasukkan oleh user

if N <= 0 %Jika N bilangan negatif

disp('Masukan harus bilangan bulat positif');

return; %Perhitungan tidak dilanjutkan

end

% Membulatkan N kalau-kalau N bukan bil. bulat

N = round(N);

prima = 1; %Kita anggap N bil prima pd awal program

%flag 'prima' kita set jadi satu.

for i = 2:floor(N/2)

if rem(N,i) == 0

prima=0;

% N bukan bilangan prima,

% flag 'prima' diset jadi nol

break; % Keluar dari loop

end

end

% Menampilkan hasil:

if prima == 0

disp(N), disp('bukan bilangan prima!');

else

disp(N), disp('adalah bilangan prima!');

end

Simpan dengan nama bilprima.m pada direktori kerja. Jalankan program dengan perintah berikut:

>> bilprima(7)

>> bilprima(4)

## Operator Perbandingan dan Logika

Jalankan perintah berikut:

>> A = [1 2 0 -1 -2]; B = [1 0 0 0 5];

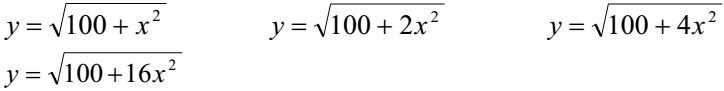
>> C = and(A,B)

>> D=A|B|C

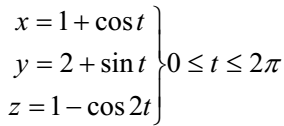
>> E = xor(~A,B)

**IV. TUGAS PRAKTIKUM**

1. Buatlah kurva y= x4−9x2 pada rentang –6 ≤ x ≤6. Buatlah inkremen x cukup kecil sehingga kurva terlihat halus.
2. Buatlah kurva berikut pada rentang -10 ≤ x ≤ 10 dalam satu gambar!



1. Buatlah kurva 3 Dimensi dari fungsi berikut:



1. Buatlah program dengan M-file untuk menghitung volume dan luas permukaan balok bila diketahui: panjang = 5, lebar = 3, tinggi = 6,5. Beri nama program ini dengan program\_balok.m
2. Untuk absen Ganjil

Buatlah suatu fungsi dengan M-file untuk menghitung volume dan luas permukaan balok dengan spesifikasi:

masukan fungsi : panjang, lebar, dan tinggi balok

keluaran fungsi : volume, dan luas permukaan balok.

Beri nama fungsi ini dengan hitung\_balok.m

Untuk absen Genap:

Buatlah suatu fungsi dengan M-file untuk menghitung volume dan luas permukaan dari suatu prisma segiempat dengan spesifikasi:

masukan fungsi : panjang dan lebar alas prisma, serta tinggi prisma

keluaran fungsi : volume, dan luas permukaan prisma

Beri nama fungsi ini dengan hitung\_prisma.m

1. Buatlah suatu program untuk menampilkan segitiga Pascal.

Pengguna harus memasukkan jumlah level segitiga yang ingin ditampilkan melalui perintah input. Jika pengguna menginginkan segitiga 4 level maka akan tampil keluaran:

1

1 1

1 2 1

1 3 3 1

Beri nama program ini dengan program\_pascal.m

1. Buatlah sebuah fungsi untuk menghitung jumlah hari di antara dua tanggal. Spesifikasi dari fungsi tersebut ialah:

masukan : tanggal, bulan, dan tahun awal, serta tanggal, bulan, dan tahun akhir.

keluaran : jumlah hari di antara dua tanggal tersebut.

Beri nama fungsi ini dengan hitung\_hari.m.

Misalkan akan menghitung jumlah hari antara 2 Januari 2004 hingga 5 November 2006, maka tulis:

>> jml\_hari = hitung\_hari(2,1,2004,5,11,2006)

jml\_hari =

1038