Cepat Mahir Matlab

Andry Pujiriyanto
andrypuji@hmgm.geoph.itb.ac.id

Bab 1
Memulai Menggunakan Matlab

Matlab merupakan bahasa canggih untuk komputansi teknik. Matlab merupakan integrasi dari komputansi, visualisasi dan pemograman dalam suatu lingkungan yang mudah digunakan, karena permasalahan dan pemecahannya dinyatakan dalam notasi matematika biasa. Kegunaan Matlab secara umum adalah untuk :

- Matematika dan Komputansi
- Pengembangan dan Algoritma
- Pemodelan,simulasi dan pembuatan prototype
- Analisa Data,eksplorasi dan visualisasi
- Pembuatan apilikasi termasuk pembuatan *graphical user interface*

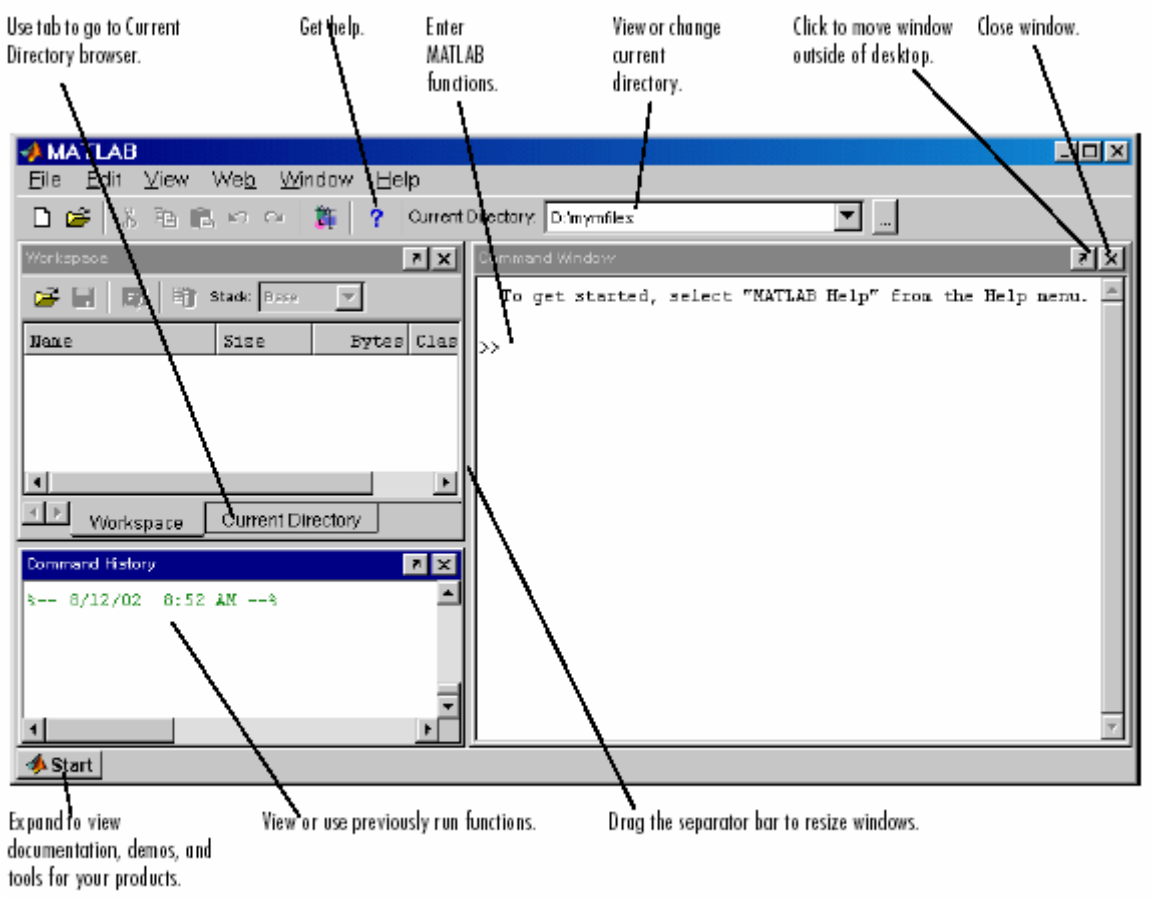
Matlab adalah sistem interaktif dengan elemen dasar array yang merupakan basis datanya. Array tersebut tidak perlu dinyatakan khusus seperti di bahasa pemograman yang ada sekarang. Hal ini memungkinkan anda untuk memecahkan banyak masalah perhitungan teknik, khususnya yang melibatkan matriks dan vektor dengan waktu yang lebih singkat dari waktu yang dibutuhkan untuk menulis program dalam bahasa C atau Fortran. Untuk memahami matlab, terlebih dahulu anda harus sudah paham mengenai matematika terutama operasi vektor dan matriks, karena operasi matriks merupakan inti utama dari matlab. Pada intinya matlab merupakan sekumpulan fungsi-fungsi yang dapat dipanggil dan dieksekusi. Fungsi-fungsi tersebut dibagi-bagi berdasarkan kegunaannya

yang dikelompokan didalam toolbox yang ada pada matlab. Untuk mengetahui lebih jauh mengenai toolbox yang ada di matlab dan fungsinya anda dapat mencarinya di website http://www.mathworks.com, atau anda dapat membuka cd dokumentasi matlab.

I.1. Desktop Matlab

Ketika anda mulai membuka program Matlab, akan muncul desktop Matlab yang berisi tools (Graphical user interface) untuk mengatur file, variabel dan aplikasi yang berhubungan dengan Matlab.

Sebagai ilustrasi dibawah ini digambarkan *desktop* yang pertama muncul di Matlab 6.5.



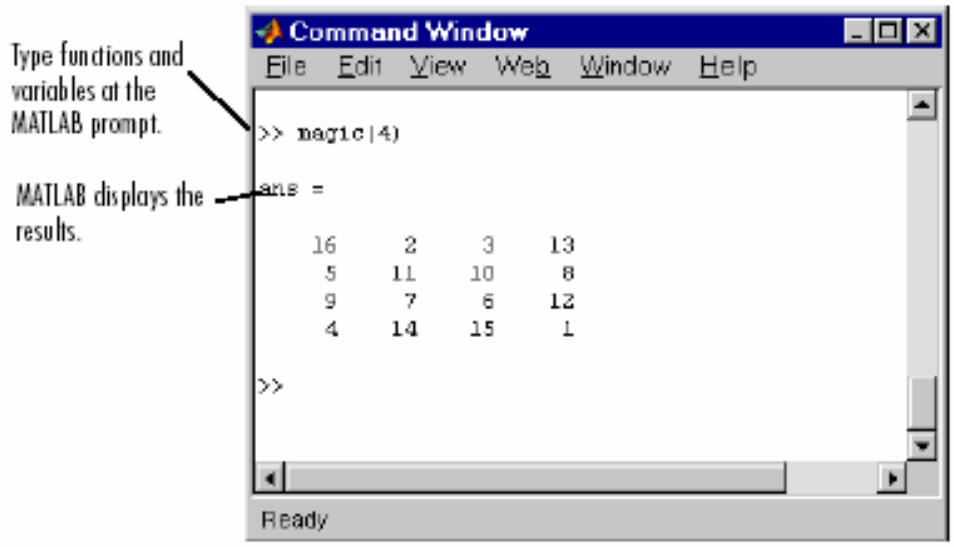
Gambar 1. Desktop Matlab versi 6.5.

Desktop Tools

Pada bagian ini diperkenalkan beberapa desktop tools yang ada pada Matlab

Command Window

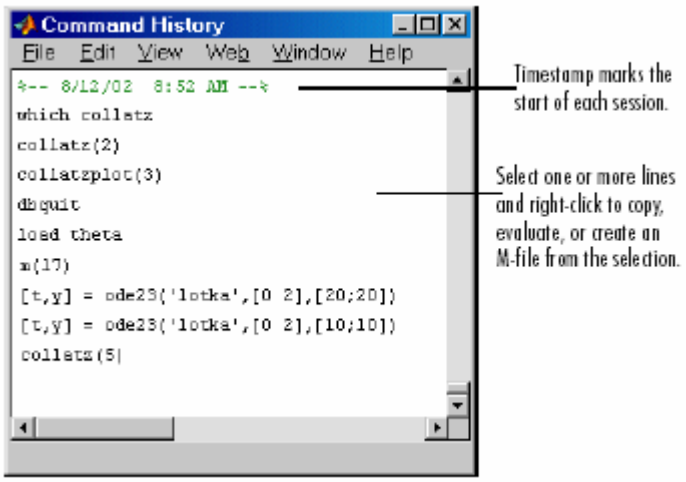
Gunakan command window untuk memasukan variabel dan menjalankan function atau M-files. Setiap perintah yang ditulis di command window langsung ditampilkan. Bila perintah anda salah akan keluar pesan error. Sebagai ilustrasi dapat dilihat pada gambar 2 dibawah ini, disini dieksekusi perintah magic(4) yang artinya kita membuat "matriks ajaib" ukuran 4 x 4. Prinsip dasar untuk memulai menggunakan Matlab, anda anggap Matlab adalah sebuah kalkulator. Untuk itu coba dengan mengerjakan operasi matematika sederhana layaknya sebuah kalkulator di command window.



Gambar 2. Command window.

Command History

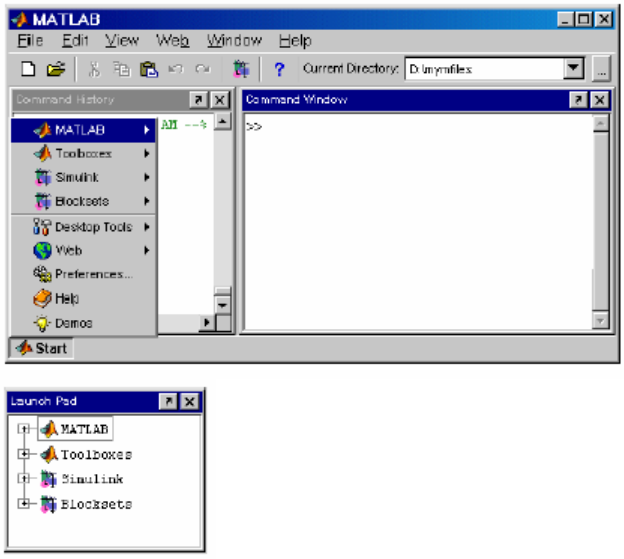
Statemen yang anda buat di command window tersimpan semuanya di command history. Di command history anda dapat melihat statemen yang lalu dan mengkopi lalu mengeksekusi statemen yang dipilih.



Gambar 3. Command History.

Tombol Start dan Launch Pad

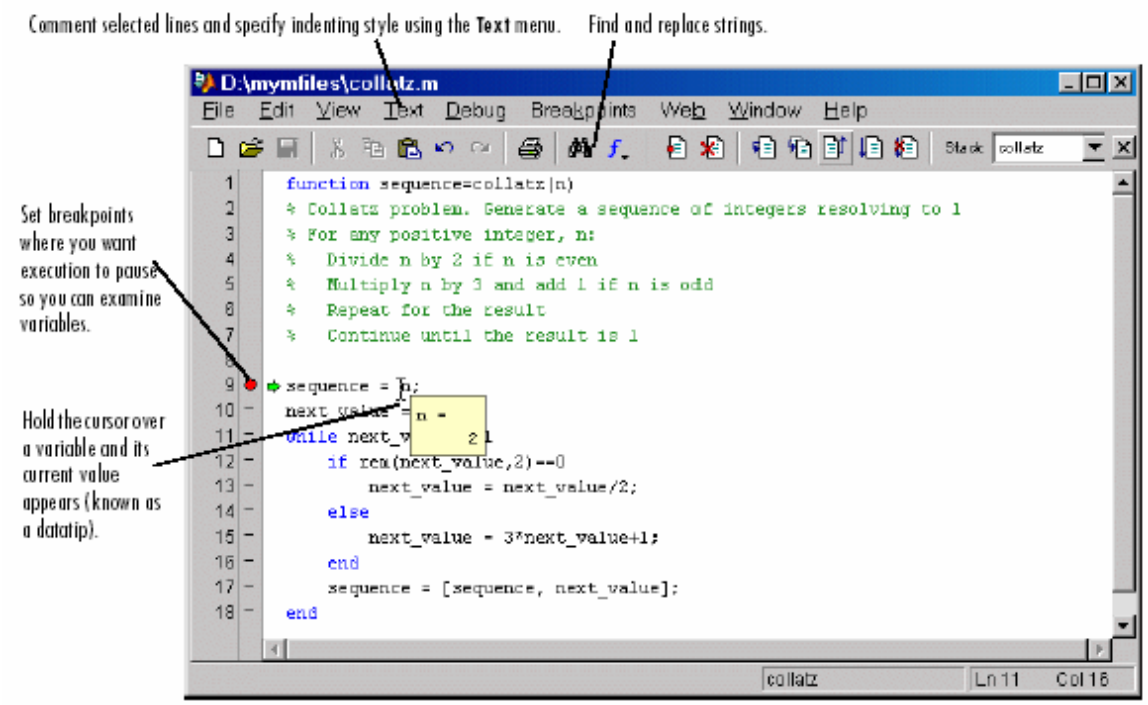
Tombol start memudahkan akses ke tools,demo dan dokumentasi ,anda hanya tinggal mengklik tombol untuk melihat pilihannya.



Gambar 4. Launch pad menggambarkan beberapa akses dengan *tree view*.

Teks Editor

Gunakan teks editor untuk membuat dan menjalankan M-files.



Gambar 5. Text Editor.

I.2. Ruang Kerja Matlab

Saat anda bekerja di command window semua perintah, variable dan data yang disimpan berada di dalam ruang kerja Matlab. Ruang kerja "default" dari Matlab yaitu di folder work di dalam folder Matlab. Apabila kita menginstal Matlab versi 6.1 di C maka folder work akan berada di C:/Matlab6p1/work. Untuk merubah ruang kerja lakukan di Command Window, seperti anda merubah direktori di DOS.

Coba anda ketik **`tes=2`** pada command window, maka akan keluar output sebagai berikut :

```
tes =
     2
```

Ini berarti variable tes telah tersimpan di dalam ruang kerja kita.

Untuk melihat data yang telah tersimpan coba anda ketik **`tes`** pada commands window.

```
tes =
     2
```

Jika anda tidak dapat mengingat nama setiap variable, maka anda dapat meminta Matlab untuk menampilkan namanya, menggunakan perintah who atau whos.

```
whos
```

```
  Name        Size                    Bytes  Class

  tes         1x1                         8  double array
  tes2        1x1                         8  double array

Grand total is 2 elements using 16 bytes
```

Untuk mengetahui isi variabel tersebut anda harus memasukkan nama variabelnya dalam command window.

Untuk memanggil perintah sebelumnya, di Matlab menggunakan tombol panah pada keyboard anda (←↑↓→) .

Untuk menghapus semua semua variabel yang sudah kita masukkan digunakan perintah **clear all**

Untuk melihat keterangan dari function di Matlab atau program yang kita buat digunakan perintah : ' **help function** ' , sebagai contoh :

```
help plot
```

```
 PLOT   Linear plot.
    PLOT(X,Y) plots vector Y versus vector X. If X or Y is a matrix,
    then the vector is plotted versus the rows or columns of the
matrix,
    whichever line up.  If X is a scalar and Y is a vector, length(Y)
    disconnected points are plotted.

    PLOT(Y) plots the columns of Y versus their index.
    If Y is complex, PLOT(Y) is equivalent to PLOT(real(Y),imag(Y)).
    In all other uses of PLOT, the imaginary part is ignored.

    Various line types, plot symbols and colors may be obtained with
    PLOT(X,Y,S) where S is a character string made from one element
    from any or all the following 3 columns:

           b     blue          .     point              -     solid
           g     green         o     circle             :     dotted
           r     red           x     x-mark             -.    dashdot
           c     cyan          +     plus               --    dashed
           m     magenta       *     star
           y     yellow        s     square
           k     black         d     diamond
                               v     triangle (down)
                               ^     triangle (up)
                               <     triangle (left)
                               >     triangle (right)
                               p     pentagram
```

```
                                h     hexagram

    For example, PLOT(X,Y,'c+:') plots a cyan dotted line with a plus
    at each data point; PLOT(X,Y,'bd') plots blue diamond at each data
    point but does not draw any line.

    PLOT(X1,Y1,S1,X2,Y2,S2,X3,Y3,S3,...) combines the plots defined by
    the (X,Y,S) triples, where the X's and Y's are vectors or matrices
    and the S's are strings.

    For example, PLOT(X,Y,'y-',X,Y,'go') plots the data twice, with a
    solid yellow line interpolating green circles at the data points.

    The PLOT command, if no color is specified, makes automatic use of
    the colors specified by the axes ColorOrder property.  The default
    ColorOrder is listed in the table above for color systems where the
    default is blue for one line, and for multiple lines, to cycle
    through the first six colors in the table.  For monochrome systems,
    PLOT cycles over the axes LineStyleOrder property.

    PLOT returns a column vector of handles to LINE objects, one
    handle per line.

    The X,Y pairs, or X,Y,S triples, can be followed by
    parameter/value pairs to specify additional properties
    of the lines.

    See also SEMILOGX, SEMILOGY, LOGLOG, PLOTYY, GRID, CLF, CLC, TITLE,
    XLABEL, YLABEL, AXIS, AXES, HOLD, COLORDEF, LEGEND, SUBPLOT, STEM.

 Overloaded methods
    help cfit/plot.m
    help fints/plot.m
    help idmodel/plot.m
    help iddata/plot.m
    help cgrules/Plot.m
    help xregtwostage/plot.m
    help xregtransient/plot.m
    help xregmodel/plot.m
    help localmod/plot.m
    help sweepset/plot.m
    help mdevtestplan/plot.m
    help cgdatasetnode/plot.m
    help cgdatadisplay/plot.m
    help ntree/plot.m
    help dtree/plot.m
    help wvtree/plot.m
    help rwvtree/plot.m
    help edwttree/plot.m
```

I.3. Struktur File

Tipe file yang sering dipakai di Matlab terdiri dari *.mat dan *.m. File dengan ekstensi *.mat biasanya untuk menyimpan workspace yang kita kerjakan di command window, sedangkan file dengan ekstensi *.m biasanya untuk menyimpan program dan disebut "m-file". Untuk "m-file" akan dibahas selanjutnya.

Untuk meyimpan pekerjaan anda pilih **File ➔ Save Workspace As ➔ ketik work_1.mat**, maka anda telah meyimpan pekerjaan anda di file work_1.mat. File ini hanya bisa di buka lagi di command window.

Untuk membuka pekerjaan anda kembali, coba :

```
clear all
load work_1.mat
whos
```

```
  Name      Size                    Bytes  Class

  tes       1x1                         8  double array
  x         1x10                       80  double array

Grand total is 11 elements using 88 bytes
```

Keterangan : Untuk Bab ini dan selanjutnya terdapat tiga tipe penulisan script, tulisan berwarna hijau menunjukan perintah yang ditulis di command window, tulisan berwarna biru merupakan output dari perintah yang ditulis dan tulisan didalam kotak menunjukkan script yang ditulis di text editor.

I.2.Bilangan dan Operator Matematika di Matlab

Terdapat tiga tipe bilangan di Matlab , yaitu :

- Bilangan bulat (integer)
- Bilangan real
- Bilangan kompleks

Contoh bilangan bulat

```
x=10
```

```
x =
    10
```

Contoh bilangan real

```
x=10.01

x =
   10.0100
```

Di dalam Matlab anda tidak perlu penanganan khusus untuk bilangan kompleks. Bilangan kompleks di beri **tanda i atau j** , contoh :

```
y=sqrt(-2)  % akat negatif 2

y =
        0 + 1.4142i

real(y)

ans =
     0

imag(y)

ans =
    1.4142

abs(y)

ans =
    1.4142

angle(y)

ans =
    1.5708
```

Matlab mempunyai variabel yang bukan merupakan bilangan yang di lambangkan dengan :

- -inf
- inf
- Nan

Daftar operasi aritmatika dasar dalam Matlab

Operators

Expressions use familiar arithmetic operators and precedence rules.

+	Addition
-	Subtraction
*	Multiplication
/	Division
\	Left division (described in "Matrices and Linear Algebra" in the MATLAB documentation)
^	Power
'	Complex conjugate transpose
( )	Specify evaluation order

Daftar konstanta yang nilainya sering digunakan

pi	3.14159265...
i	Imaginary unit, $\sqrt{-1}$
j	Same as i
eps	Floating-point relative precision, 2^{-52}
realmin	Smallest floating-point number, 2^{-1022}
realmax	Largest floating-point number, $(2-\varepsilon)2^{1023}$
Inf	Infinity
NaN	Not-a-number

Contoh 1 :

Misalnya anda mengambil kuliah sebanyak 12 SKS, yang terdiri dari seismologi 4 sks, Analisis sinyal 3 sks, Tomografi 2 sks dan Gravitasi 3 sks. Lalu pada akhir semester anda mendapat nilai sebagai berikut seismologi A, Analisis sinyal B, Tomografi C dan gravitasi A. Dengan point nilai A=4, B=3, C=2 berapa nilai IP anda ?

Untuk menyelesaikan ini digunakan pendekatan seperti perhitungan di kalkulator:

```
ip=(4*4+3*3+2*2+3*4)/(4+3+2+3)

ip =
    3.4167
```

Sebagai alternatif anda dapat menyelesaikan masalah di atas dengan terlebih dahulu menyimpan informasi yang kita punya pada variabel. Contoh :

```
seismologi=4
tomografi=2
analisis_sinyal=3
gravitasi=4


seismologi =
     4
tomografi =
     2
analisis_sinyal =
     3
gravitasi =
     4

total_sks=12

total_sks =
    12


ip=(seismologi*4+tomografi*2+analisis_sinyal*3+gravitasi*3)/total_sks

ip =
    3.4167
```

I.3. Komentar dan Tanda Baca

Semua teks sesudah tanda **%** dianggap sebagai statemen komentar, contoh:

```
semester=8   % jumlah semester S 1

semester =
     8
```

Variabel semester diisi dengan nilai 8 dan satetemen sesudah **tanda %** di anggap sebuah komentar. Statemen ini berguna untuk mendokumentasikan apa yang sudah anda kerjakan.

Tanda **titik koma (;)** dalam Matlab berguna untuk mencegah menampilkan hasil, contoh :

```
semester=8;
```

I.4.Fungsi-Fungsi Matematika Umum

Matlab mempunyai berbagai fungsi matematika umum yang biasa di gunakan dalam matematika. Sebagian besar fungsi tersebut hampir sama dengan bila anda menuliskannya secara matematis.

Sebagai contoh :

```
pi

ans =
    3.1416
y=sin(pi/6)

y =
    0.5000
y=asin(0.5)

y =
    0.5236
```

Fungsi Trigonometri	Deskripsi
sin, asin, sinh, asinh	sinus, anti sinus, hiperbolik sinus, hiperbolik anti sinus
cos, acos, cosh, acosh	cosines, anti cosines, hiperbolik cosines, hiperbolik anti cosines
tan, atan, tanh, atanh	tangent, anti tangent, tangent hiperbolik, anti tangent hiperbolik
cot, acot, coth, acoth	cotangent, anti cotangent, cotangent hiperbolik, anti cotangent hiperbolik.
sec, asec, sech, asech	secan, antisecan, secan hiperbolik, anti secan hiperbolik
csc, acsc, csh, acsh	cosecant, anti cosecant, cosecant hiperbolik, anti cosecant hiperbolik.

Fungsi Matematika Dasar	Deskripsi
Abs	nilai absolute atau amplitudo bilangan kompleks
Angle	sudut fasa
Sqrt	akar kuadrat
Real	bagian real dari bilangan kompleks
Imag	bagian imaginer dari bilangan kompleks
Conj	konjugat bilangan kompleks
Round	pembulatan ke bilangan bulat terdekat.
Fix	pembulatan ke arah nol
Floor	pembulatan kea rah $-\infty$
Ceil	pembulatan ke arah ∞
Rem	sisa
Exp	exponensial berbasis bilangan e
Log	logaritma murni
Log10	logaritma basis 10

I.5. Contoh Penyelesaian Masalah

Contoh 1: Persamaan fungsi

Tentukan nilai fungsi dibawah ini dengan nilai-nilai t=25 , x=43 ,y=15.25,z=8.2 !

a. $M = 4x^2 + 3y + 10$
b. $N = e^{2x} + x$
c. $O = \sqrt{\frac{1}{(x+y)} + \frac{1}{(t+z)}}$
d. $P = 4e^{-x/2}\sin(\pi x)$

Untuk penyelesaian di atas adalah sbb:

Pertama kita buat dulu variabel nya

```
t=25;x=43;y=15.25;z=8.2;
```

Lalu masukkan nilai tersebut ke masing-masing fungsinya

```
M=4*x^2+3*y+10

M =
  7.4518e+003

N=exp(2*x)+x

N =
  2.2352e+037

O=sqrt((1/(x+y))+(1/(t+z)))

O =
    0.2175

P=4*(exp(-x/2))*sin(pi*x)

P =
  1.6223e-023
```

Contoh 2 : Penentuan volume silinder berlubang

Sebuah silinder mempunyai diameter luar 6.3125 dan diameter dalam 5.762 , tentukan volumenya bila diketahui rumus untuk mencari volumenya :

$$V = \frac{4}{3}\pi(RE^3 - RI^3)$$

Dimana RE adalah diameter luar dan RI adalah diameter dalam

Penyelesaian :

```
re=6.3125;ri=5.762;
v=4/3*pi*(re^3-ri^3);
disp(['Volume = ',num2str(v)])
```

```
Volume = 252.3169
```

I.6.Menyimpan dan Memanggil Data

Untuk Menyimpan dan memanggil data dari file pilih **File ➔ Save Workspace As** …

Untuk memanggil data digunakan pilihan Load WorkSpace As atau Open pada menu file.

Sedangkan untuk mengimport data , untuk Matlab versi 6 keatas pilih **file➔ Import Data .. .**

Matlab juga menyediakan dua perintah **---- save dan load -------** yang jauh lebih fleksibel.

Perintah save untuk menyimpan satu atau lebih variabel dalam file format yang sesuai dengan pilihan anda.

contoh :

```
clear all
x=1:10;y=10:10:10:100; % membuat array baru
```

```
save
```

```
Saving to: Matlab.mat
```

menyimpan semua variabel Matlab dalam format biner di file Matlab.mat

```
save data
```

menyimpan semua variabel Matlab dalam format biner di file data.mat

```
save data_x x
```

menyimpan variabel x dalam format biner di file data_x.mat

```
save data_xy.dat x  -ascii
```

menyimpan variabel x dalam format biner di file data_xy.dat dalam format ascii.

untuk membuka data digunakan perintah load, contoh;

```
load data_x.mat
```

I.7. Input dan Output di Matlab

Untuk menampilkan teks atau angka dapat digunakan **fungsi disp**. Sebagai contoh :

```
disp('Ini contoh tampilan dari text')
```

```
Ini contoh tampilan dari text
```

Syarat digunakannya disp, isi didalamnya harus merupakan strings, jadi jika kita ingin menampilkan sebuah angka terlebih dahulu dirubah kedalam bentuk strings dengan menggunakan **function num2str( ).**

Sebagai contoh :

```
nim=10499006;
disp(['Nim saya adalah ',num2str(nim)])
```

```
Nim saya adalah 10499006
```

Untuk format output yang lebih fleksibel digunakan function fprintf, dimana disini anda dapat membuat tampilan di layar atau di simpan sekehendak anda. Fungsi ini mempunyai argumen sbb:

fprintf(' nama file ' ,' format string ' , list)

dimana list adalah nama variabel yang dipisahkan dengan koma.

Untuk format string :

%P.Qe untuk eksponensial
%P.Qf untuk fixed point
/n untuk membuat baris baru

contoh :

```
x=1007.46 ; y=2.1278;k=17;
fprintf('x= %8.2f y=%8.2f k=%2.0f',x,y,k)
```

```
x=  1007.46 y=    2.13 k=17
```

I.7. Script M-file

Untuk menghadapi masalah jika harus mengetikkan perintah yang jumlahnya cukup banyak dibutuhkan suatu file script. File seperti ini di Matlab disebut M-file. Pada m-file anda diperbolehkan untuk mengetikkan deretan perintah dalam suatu teks file.
Untuk membuat M-file, buka teks editor, pilih File ➔ New ➔ M-File.

Sebagai contoh berikut ini diberikan perintah-perintah untuk menyelesaikan masalah pencarian nilai blok dalam tomografi geofisika. Yang didefinisikan dengan rumus sebagai berikut :

$$no\_blok = (k-1).nx.ny + (j-1).nx + i$$
$$i = ifix((x - xo)/dx) + 1$$
$$j = ifix((y - yo)/dy) + 1$$
$$k = ifix((z - zo)/dz) + 1$$

Dimana : x,y,z koordinat yang akan dicari nilai bloknya
dx, dy, dz adalah panjang blok dalam arah x , y dan z.
xo, yo, zo adalah koordinat awal
ifix adalah bilangan bulat.

Penyelesaiannya dalam matlab adalah sebagai berikut :

```
% blok_tomo.m script file untuk mencari nilai blok tomografi
x=110;y=10;z=175;      % koordinat titik yang akan dicari nilai bloknya
dx=1;dy=1;dz=50;       % ukuran blok
nx=65;ny=35;            % Banyak kotak kearah x dan y
x0=90;y0=-15;z0=0;     % Koordinat awal

i=fix((x-x0)/dx)+1;
```

```
j=fix((y-y0)/dy)+1;
k=fix((z-z0)/dz)+1;

no_blok=(k-1)*nx*ny + (j-1)*nx+i
disp(['no blok = ',num2str(no_blok)])
```

Untuk mengeksekusi file ini terlebih dahulu anda simpan dengan memilih File➔Save as ..,lalu beri nama blok_tomo.m. Setelah itu pilih Debug ➔ Run. Atau dengan cara mengetikkan nama file tersebut di command Window Matlab :

```
blok_tomo

no blok = 8471
```

Jika perintah matlab tidak diakhiri dengan titik koma,hasil dari perintah itu serta nama variabelnya akan ditampilkan kembali dalam command window.Supaya tamplan lebih bagus , maka untuk menampilkan nama variabel digunakan perintah disp.

Perintah **echo on** membuat perintah-perintah yang dibuat di M-file akan ditampilkan kambali di command window.

Perintah input memungkinkan anda meminta input dari pemakai saat M-file dijalankan.

Contoh : Coba anda ketikkan perintah berikut di command window

```
umur=input('masukkan umur anda sekarang')
```

I.8. Operator Logika dan Relasional

Operator	Deskripsi
<	kurang dari
<=	kurang dari sama dengan
>	lebih dari
>=	lebih dari sama dengan
==	sama dengan
~=	tidak sama dengan

Operator logika	Deskripsi
|	or
&	and
~	not

Daftar Pustaka

a) Getting Started With MATLAB , Version 6 , The MathWorks.Inc , 2002

b) MATLAB Bahasa Komputansi Teknis , Penerbit ANDI Yogyakarta , 2000

c) Numerical Methods Using Matlab , ELLIS HORWOOD , 1995

d) Mastering MATLAB 5. A Comprehensive Tutorial and reference , Prentice Hall , 1998

e) http://www.mathworks.com

Cepat Mahir Matlab

Andry Pujiriyanto
andrypuji@hmgm.geoph.itb.ac.id

Bab 2
Operasi Array

II.1. Dasar –Dasar array

Variabel dengan tipe data tunggal (skalar) hanya dapat digunakan untuk meyimpan sebuah nilai saja, sehingga untuk menyimpan beberapa nilai sekaligus dalam suatu variable khusus dibutuhkan variable array atau variable berindeks.Variabel array dapat digunakan untuk menampung banyak data yang sejenis (numeric / string) .

Misalkan anda akan menghitung nilai fungsi sinus dalam range $0 \le x \le 2\pi$ dengan interval 0.2π maka anda ketikkan di matlab sbb :

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

```
y =
  Columns 1 through 7
         0    0.3090    0.5878    0.8090    0.9511    1.0000    0.9511
  Columns 8 through 14
    0.8090    0.5878    0.3090    0.0000   -0.3090   -0.5878   -0.8090
  Columns 15 through 21
   -0.9511   -1.0000   -0.9511   -0.8090   -0.5878   -0.3090   -0.0000
```

Pada bagian diatas dibuat array x dengan nilai antara 0 sampai 2π dengan interval 0.1π lalu nilai x ini dimasukkan ke dalam fungsi sin sehingga didapat nilai sinusnya.

Sebagai alternatif lain dalam membuat array yang perlu dilakukan hanyalah mengetikkan kurung kotak kiri **" ["** , memasukkan elemen-elemen dengan dipisahkan oleh spasi atau koma, kemudian menutup array dengan kurung kotak kanan **"] "**. Apabila anda menghitung sin dari x maka akan mengikuti bentuk dari x.

```
x=[0 0.5*pi pi 1.5*pi 2*pi]

x =
         0    1.5708    3.1416    4.7124    6.2832

y=sin(x)

y =
         0    1.0000    0.0000   -1.0000   -0.0000
```

Untuk memisahkan elemen array yang satu dengan lainnya digunakan spasi atau koma atau titik koma sebagai contoh :

```
x=[0,0.5*pi,pi,1.5*pi,2*pi]

x =
         0    1.5708    3.1416    4.7124    6.2832


 x=[0;0.5*pi;pi;1.5*pi;2*pi]

x =
         0
    1.5708
    3.1416
    4.7124
    6.2832
```

Bisa dilihat bila digunakan koma maka akan menghasilkan array baris, lalu bila digunakan titik koma akan menghasilkan array kolom.

Untuk membuat array yang mempunyai banyak elemen digunakan notasi kolon, sebagai contoh :

```
x=1:10

x =
     1     2     3     4     5     6     7     8     9    10
```

```
x=1:2:10

x =
     1     3     5     7     9
```

Pada kasus yang pertama anda membuat array mulai dari 1 sampai 10 dengan interval 1, matlab akan menerjemakan intervalnya 1 bila anda buat dengan cara ini.

Pada kasus kedua anda membuat array mulai dengan 1 sampai 10 dengan interval 2.

Untuk membuat array dapat juga digunakan **fungsi linspace**. Argumen fungsi ini didefinisikan sebagai:

linspace(nilai pertama, nilai terakhir , jumlah elemen) , contoh :

```
x=linspace(1,10,5)

x =
    1.0000    3.2500    5.5000    7.7500   10.0000
```

Untuk kasus khusus dimana jarak logaritma diperlukan digunakan **fungsi logspace**.Argumen fungsi ini di definisikan sebagai :

logspace(eksponen pertama,eksponen terakhir,jumlah elemen) , contoh :

```
x=logspace(1,3,6)

x =
  1.0e+003 *
    0.0100    0.0251    0.0631    0.1585    0.3981    1.0000
```

Cara Untuk Membuat Array

x=[2 3 4]
membuat vektor baris x yang memuat elemen-elemen yang diberikan

x=awal : akhir
membuat vektor baris x dimulai dengan awal,interval satu,diakhiri dengan akhir.

x=awal:kenaikan:akhir
membuat vektor baris dimulai dengan awal,interval sebesar kenaikan ,diakhiri pada atau sebelum akhir

x=linspace(awal,akhir,n)
membuat vektor baris diawali dengan awal,berakhir dengan akhir, mempunyai n elemen

x=logspace(awal,akhir,n)
membuat vektor baris dengan interval logaritma dimulai dengan 10^{awal} diakhiri dengan 10^{akhir} dan mempunyai n elemen

II.2. Pengalamatan Array

Dalam matlab elemen-elemen array diakses menggunakan subscript .
Misalnya x(1) adalah elemen pertama x, x(2) adalah elemen kedua x , dst. Sebagai contoh :

```
x=[10 20 30 40 50 60 70 80]

x =
    10    20    30    40    50    60    70    80

x(4) % elemen keempat

ans =
    40

x(7) % elemen ketujuh

ans =
    70
```

Untuk mengambil sejumlah elemen dalam array digunakan **notasi kolon**

```
x(1:5) % mengambil elemen kesatu sampai lima

ans =
    10    20    30    40    50

x(3:end)

ans =
    30    40    50    60    70    80
```

Diambil dari elemen ketiga sampai elemen terakhir. Kata **end** berarti elemen terakhir dari array x

```
x(7:-1:2)

ans =
    70    60    50    40    30    20
```

Elemen diatas maksudnya dimulai dari elemen ketujuh mundur satu sampai elemen kedua.

```
x(1:2:6)

ans =
    10    30    50
```

Elemen diatas maksudnya dimulai dari 1 naik 2 berhenti setelah mencapai enam.

```
x([3 4 5 7 6 5])

ans =
    30    40    50    70    60    50
```

Elemen diatas maksudnya diambil elemen dengan urutan elemen 2 3 4 5 7 6 5 dari array x.

Untuk menjumlahkan elemen-elemen dalam array digunakan perintah sum , contoh:

```
jumlah_x=sum(x)

jumlah_x =
   360
```

Perintah diatas maksudnya kita menjumlahkan elemen yang ada di variabel x

II.3. Operasi Array

Dari contoh diatas anda selalu membuat array berbentuk vektor baris , dalam matlab dimungkinkan juga untuk membuat suatu array berbentuk vektor kolom. Dalam hal ini manipulasi array tidak menimbulkan perubahan, satu-satunya perbedaan hanyalah hasilnya ditampilkan sebagai kolom bukan sebagai baris.

Seperti telah dijelaskan pada bagian sebelumnya untuk membuat vektor kolom elemen-elemen array dipisahkan dengan titik koma " **;** **".**

Cara lain adalah membentuk vektor baris kemudian ditranspose mejadi vektor kolom menggunakan notasi " ' ". Sebagai contoh :

```
x=1:10

x =
     1     2     3     4     5     6     7     8     9    10
```

```
y=x'

y =
     1
     2
     3
     4
     5
     6
     7
     8
     9
    10
```

Operasi Array-Skalar

Semua operasi matematika sederhana antara skalar dan array mempunyai sifat yang sama (+ , - , : , *).Semua operasi array dengan skalar akan dikenakan pada semua elemen array.

```
y=x-2

y =
    -1     0     1     2     3     4     5     6     7     8

y=3*x/2 -5

y =
  Columns 1 through 7
   -3.5000   -2.0000   -0.5000    1.0000    2.5000    4.0000    5.5000
  Columns 8 through 10
    7.0000    8.5000   10.0000
```

Operasi Array-Array

Operasi antara array tidak sama dengan operasi diatas. Logika sederhana dalam operasi array-array adalah logika operasi matematika antara vektor atau matriks. Syarat-syarat operasi matematika antara vektor atau matriks berlaku juga pada array. Sebagai contoh

```
x=[10 20 30 10;40 50 60 20;70 80 90 30]

x =
    10    20    30    10
    40    50    60    20
    70    80    90    30
```

```
y=[50 60 70 5 ;1 2 3 6 ;40 5 20 7]

y =
    50    60    70     5
     1     2     3     6
    40     5    20     7

x+y

ans =
    60    80   100    15
    41    52    63    26
   110    85   110    37

x-y

ans =
   -40   -40   -40     5
    39    48    57    14
    30    75    70    23

x*y

??? Error using ==> *
Inner matrix dimensions must agree.


x.*y

ans =
         500        1200        2100          50
          40         100         180         120
        2800         400        1800         210
```

Dalam contoh diatas untuk perkalian menggunakan " . * " ini berarti perkalian elemen dengan elemen. Perkalian tanpa titik berarti perkalian matriks sehingga syarat-syarat perkalian matriks harus terpenuhi.

```
x^2

??? Error using ==> ^
Matrix must be square.
```

`x.^2` % memangkatkan setiap elemen array

```
ans =
         100         400         900         100
        1600        2500        3600         400
        4900        6400        8100         900
```

Daftar Operator array

Operators	
+	Addition
+	Unary plus
-	Subtraction
-	Unary minus
*	Matrix multiplication
^	Matrix power
\	Backslash or left matrix divide
/	Slash or right matrix divide
'	Transpose
.'	Nonconjugated transpose
.*	Array multiplication (element-wise)
.^	Array power (element-wise)
.\	Left array divide (element-wise)
./	Right array divide (element-wise)

II.4. Memanipulasi Array

Karena array dan matriks merupakan hal yang mendasar dalam matlab,maka terdapat banyak cara untuk memanipulasinya. Sekali matriks dibuat , matlab menyediakan cara untuk meyisipkan,mengambil dan mengatur kembali isi matriks tersebut. Penguasaan akan hal ini merupakan kunci untuk menggunakan matlab secara efisien. Sebagai contoh manipulasi matriks atau array perhatikan contoh berikut :

membuat array dengan elemen bilangan satu atau nol.

```
x=ones(3,4) % membuat array dengan ukuran 3 baris emapat kolom

x =
     1     1     1     1
     1     1     1     1
     1     1     1     1

size(x) % mengetahui ukuran x

ans =
     3     4

zeros(4)

ans =
     0     0     0     0
     0     0     0     0
     0     0     0     0
     0     0     0     0
```

Jika digunakan argumen tunggal ,ones(n) atau zeros(n) matlab akan membuat array (n X n). Jika dipanggil dengan dua argumen misal ones(r,c) maka matlab akan membuat array dengan r baris dan c kolom.

```
A=[1 2 3;4 5 6; 7 8 9] % membuat array baru

A =
     1     2     3
     4     5     6
     7     8     9

A(2,3)=0;          % mengubah elemen baris 2 kolom 3 menjadi nol
A(1,1)=100;        % mengubah elemen baris 1 kolom 1 menjadi 100

A =
   100     2     3
     4     5     0
     7     8     9
```

Untuk menghapus elemen matriks digunakan tanda " [ ] ". Tanda ini juga berguna untuk membuat matriks kosong. Contoh :

```
A(:,2)=[ ]

A =
     1     3
     4     6
     7     9
```

Argumen diatas maksudnya hapus semua elemen di kolom 2.

Untuk mencari elemen array digunakan perintah find, fungsi find mempunyai argumen sebagai berikut :
i=find(x)

menghasilkan indeks dari array x dimana elemen-elemennya tidak nol
[r,c]=find(x)
menghasilkan indeks baris dan kolom dari array x dimana elemen-elemennya tidak nol.

contoh:

```
[baris,kolom]=find(A > 3)
```
% untuk mencari indeks dari array A yang mempunyai nilai > 3.

```
baris =
     2
     3
     2
     3


kolom =
     1
     1
     2
     2
```

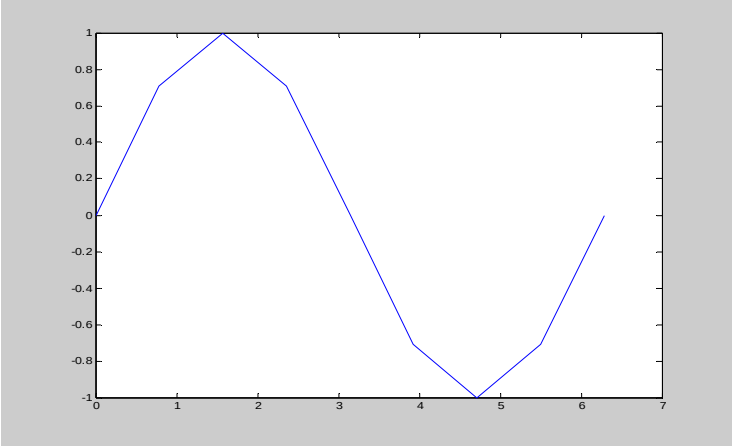
untuk melihat ukuran array biasanya digunakan perintah length atau size, biasanya size digunakan untuk melihat ukuran matriks sedangkan length untuk melihat ukuran array 1 dimensi atau vektor contoh:

```
length(baris) % melihat banyak elemen baris

ans =
     4


[baris,kolom]=size(A)

baris =
     3
kolom =
     2
```

II.5. Grafik Sederhana

Untuk melihat visualisasi dari array biasanya digunakan perintah plot, sebagai contoh:

```
x=0:0.25*pi:2*pi; % membuat vaktor baris  0 ≤ x ≤ 2π
y=sin(x);               % membuat nilai sinus
plot(x,y)               % membuat visualisasi data kita
```



Perintah plot digunakan dengan syarat jumlah elemen x dan y harus sama.

II.6. Contoh Penyelesaian Masalah

Permasalahan: Diketahui fungsi $h(k) = \frac{\omega l}{\pi} * \frac{\sin(k\pi \frac{\omega l}{\omega v})}{k\pi \frac{\omega l}{\omega v}}$, dengan nilai

$h(0) = \frac{\omega l}{\pi} = \frac{2\pi\, fl}{\pi} = 2\, fl$, $\omega v = 2\pi\, fv = \frac{\pi}{dt}$, dt=0.004 , fl=30 dan $-20 \le k \le 20$ .Buat visualisasi dari fungsi tersebut.

Penyelesaian :

Pertama dibuat array kosong hk dan nilai-nilai variabel yang diketahui

```
hk=[];  dt=0.004; fl=30; wv=pi/dt;  wl=2*pi*fl;
```

Kedua tentukan nilai h(0) dalam array kita h(0) merupakan komponen 1 dari hk maka dibuat di matlab sbb:

```
hk(1)=2*fl;
```

Ketiga buat array dengan nilai 1 sampai 20 dengan nama k

```
k=1:20;
```

Keempat masukkan nilai k ke fungsi hk beri nama hks

```
hks=hk(1)*((sin(k*pi*wl/wv))./(k*pi*wl/wv));
```

Kelima buat array hks2 untuk nilai k dari -20 sampai -1 ,bearti kita buat kebalikan dari elemen array hks

```
hks2(1:20)=hks(20:-1:1);
```

Keeenam masukkan nilai hks hk(1) dan hks kedalam array baru dengan urutan hks2 hk(1) hks beri nama array tersebut Hk.

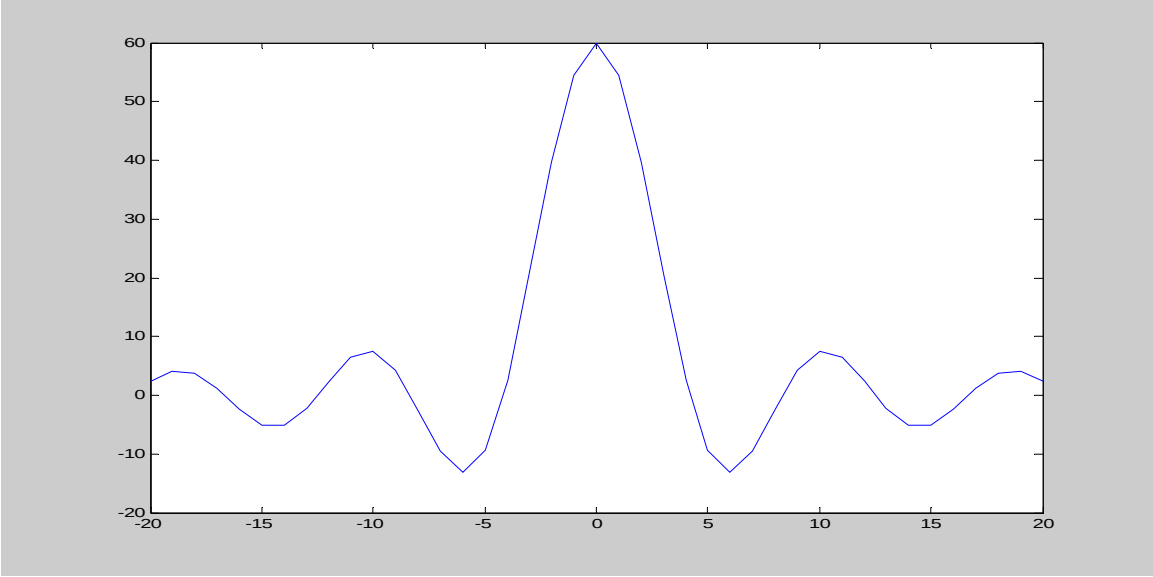
```
Hk=[hks2 hk(1) hks];
```

Ketujuh buat array dengan nilai antara -20 sampai 20.

```
k=-20:1:20;
```

Kedelapan plot variabel k dan Hk tersebut

```
plot(k,Hk)
```



Permasalahan :Buat suatu sinusoid dengan frekuensi f1=10 Hz , f2=60 Hz. , Δt=4 msec.Bila diketahui

$$y1(t) = \sin(2\pi f1 * t)$$
$$y2(t) = 0.2 * \sin(2\pi f2 * t)$$

jumlah N=250 dan t=K*Δt , $0 \le K \le N$

Penyelesaian :

Pertama masukan nilai variabel yang diketahui terlebih dahulu

```
f1=10;f2=60;dt=0.004; N=250;
```

Kedua buat array K dari nol sampai N-1 karena interval K kita harus 1 dan berjumlah 250.

```
K=0:N-1;
```

Ketiga kita buat array dengan nilai t=K*Δt

```
t=K*dt;
```

Keempat kita buat nilai y1 dan y2

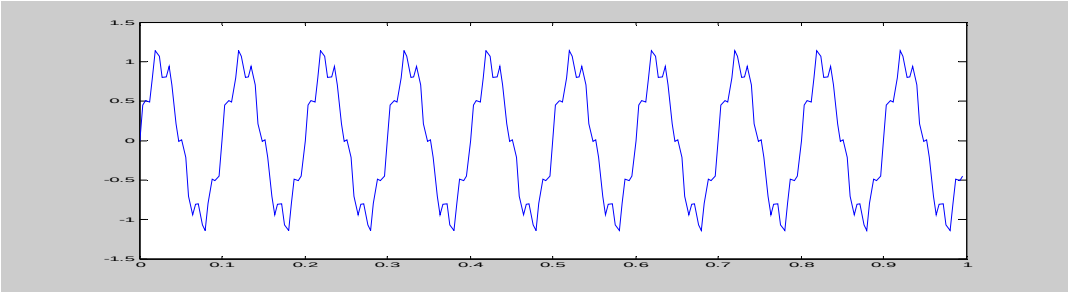
```
y1=sin(2*pi*f1*t);
y2=0.2*sin(2*pi*f2*t);
```

Kelima kita buat superposisi y1 dan y2

```
y=y1+y2;
```

Keenam kita buat visualisasinya

```
plot(t,y)
```



II.7. Latihan 1

1. Di Command Window ketik x=-1:0.1:1 , lalu coba jalankan perintah-perintah dibawah ini satu persatu
   sqrt(x)
   sin(x)
   x.^3
   plot(x,cos(x.^4))

   cos(x)
   x.^2
   plot(x,sin(x.^3))

   Perhatikan maksud setiap perintah tersebut dengan cermat

2. Jalankan Perintah-perintah dibawah ini lalu jelaskan maksudnya
   x=[2 3 4 5]
   y=-1:1:2
   x.^y
   x.*y
   x./y

3. Buat script sederhana di Matlab editor untuk membuat plot fungsi $y = x^2 \cos(x)$ dan ambil x=-2:0.1:2.

4. Jalankan perintah dicommand window sbb dan jelaskan maksudnya:

```
x=1:2:100;y=100*rand(20,1);z=rand(1,10);
v1=sum(x)
v2=sort(y)
v3=prod(1:5)
v4=[y ;  z'];
```

II.8. Latihan 2

1 . Buat suatu vektor w dengan nilai $-\pi \le x \le \pi$ dengan interval 0.004π .Lalu tentukan nilai fungsi dibawah ini :

$$X = \left| \frac{\sin(5w)}{w} \right|$$

$$Y = \cos(w + \frac{\pi}{4}) X$$

$$Z = \sin(w + \frac{\pi}{4}) X$$

Plot Y Vs Z !!

2 . Hitung nilai dari :

$$a = \sum_{n=1}^{50} 2^n + 1$$

$$b = \sum_{n=0}^{20} \frac{x^n}{n!}, x = 2$$

, Untuk faktorial gunakan function **factorial**

3 . Dengan menggunakan rumus integral tentukan suatu nilai hampiran untuk $\int_1^2 \frac{dx}{x}$

Petunjuk:

Tentukan dulu panjang selang dengan rumus $h = \frac{b-a}{n}$, a=batas bawah, b=batas akhir dan n=banyak selang.. rumus integral : $\int_a^b f(x)dx = h * \sum_a^b f(a+h)$.

4 . Dengan cara yang sama seperti diatas tentukan nilai hampiran utuk :

$$y1 = \int_{-1}^{1} x dx$$

$$y2 = \int_{-1}^{1} x^2 dx$$

$$y3 = \int_{-1}^{1} x^3 dx$$

$$y4 = \int_{-1}^{6} x^2 - 6 - 16 dx$$

5 . Buat Visualisasi dari fungsi $Hk = -\frac{\omega hp}{\pi}\frac{\sin(k\pi\frac{\omega hp}{wv})}{k\pi\frac{\omega hp}{\omega v}}$, dengan $-60 \le k \le 60$,

Δt=0.004 fp=60 Hz dan $\omega v = 2\pi fv = \frac{\pi}{dt}$ dan $Hk(0) = \frac{1}{\Delta t} - \frac{\omega hp}{\pi}$.

Daftar Pustaka

a) Getting Started With MATLAB , Version 6 , The MathWorks.Inc , 2002

b) MATLAB Bahasa Komputansi Teknis , Penerbit ANDI Yogyakarta , 2000

c) Numerical Methods Using Matlab , ELLIS HORWOOD , 1995

d) Mastering MATLAB 5. A Comprehensive Tutorial and reference , Prentice Hall , 1998

e) http://www.mathworks.com

Cepat Mahir Matlab

Andry Pujiriyanto
andrypuji@hmgm.geoph.itb.ac.id

Bab 3
Operasi Matriks

Dalam Matlab matriks merupakan sesuatu yang sangat penting , jadi pengertian dan pengelolaan data matriks di matlab merupakan hal yang paling utama. Matlab manyediakan banyak fungsi yang berguna untuk menyelesaikan masalah-masalah matriks dan aljabar linear.

III.1. Dasar-Dasar Matriks di Matlab

Sebagai dasar kita harus mengetahui notasi dan penempatan matriksnya itu sendiri.sebagai ilustrasi dibawah digambarkan sebuah matriks segiempat.

$$\begin{bmatrix} U_{1,1} & U_{1,2} & U_{1,3} & U_{1,4} \\ U_{1,2} & U_{1,1} & U_{1,1} & U_{1,1} \\ U_{1,1} & U_{1,1} & U_{1,1} & U_{1,1} \\ U_{1,1} & U_{1,1} & U_{1,1} & U_{1,1} \end{bmatrix}$$

$U_{a,b}$ ..... a = baris , b = kolom

contoh : $U_{1,1}$ artinya Matriks U baris 1 kolom 1.

$U_{2,3}$ artinya Matriks U baris 2 kolom 3.

Sebagai contoh dibawah ini akan dibuat sebuah matriks dengan elemen bilangan random dengan menggunakan function **rand**.

```
U=rand(4,4)   % membuat matriks U dengan ukuran 4 x 4.

U =
    0.9501    0.8913    0.8214    0.9218
    0.2311    0.7621    0.4447    0.7382
    0.6068    0.4565    0.6154    0.1763
    0.4860    0.0185    0.7919    0.4057
```

Untuk selanjutnya anda ambil komponen baris 3 kolom 4:

```
U(3,4)

ans =
    0.1763
```

Selanjutnya anda ambil semua komponen baris 1

```
U(1,:)

ans =
    0.9501    0.8913    0.8214    0.9218
```

Selanjutnya anda ambil semua komponen kolom 3

```
U(:,3)

ans =
    0.8214
    0.4447
    0.6154
    0.7919
```

Untuk perhitungan matematika coba anda cari inverse dan determinan dari matriks U dengan menggunakan function **inv** dan **det** .

```
inv(U)

ans =
    2.2631   -2.3495   -0.4696   -0.6631
   -0.7620    1.2122    1.7041   -1.2146
   -2.0408    1.4228    1.5538    1.3730
    1.3075   -0.0183   -2.5483    0.6344
```

```
det(U)

ans =
    0.1155
```

Syarat digunakannya perintah **inv** dan **det** ini adalah matriksnya harus berbentuk bujursangkar.

Syarat-syarat operasi matematika pada array berlaku juga pada matriks. Operasi penjumlahan, pengurangan, perkalian harus memenuhi kaidah hukum operasi matriks. Karena operasi matematika pada matriks sudah dijelaskan pada bab sebelumnya yaitu pada sub bab operasi array-array jadi pada bab ini akan lebih banyak pada aplikasi function untuk matriks di matlab.

Function diag

Function diag digunakan untuk membentuk matriks diagonal dari. Sebagai contoh :

```
d=[1 2 3 4];

D=diag(d)

D =
     1     0     0     0
     0     2     0     0
     0     0     3     0
     0     0     0     4
```

III.2. Penyelesaian Himpunan Persamaan Linear

Misalkan diketahui suatu persamaan linear

$x + 2y + 3z = 366$

$4x + 5y + 6z = 804$

$7x + 8y = 351$

Untuk mencari himpunan penyelesaian dari persamaan linear diatas maka dilakukan suatu operasi matriks dengan merubah bentuk persamaan tersebut kedalam bentuk matriks.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 0 \end{bmatrix} . \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 366 \\ 804 \\ 351 \end{bmatrix}$$, atau dalam bentuk sederhana A.x = b

Di matlab kita lakukan perintah-perintah sbb:

```
A=[1 2 3;4 5 6;7 8 0];
b=[366;804;351];
x=A\b

x =
   25.0000
   22.0000
   99.0000
```

Operator pembagian kiri \ tidak didahului oleh titik karena merupakan operasi matriks, bukan operasi elemen ke elemen suatu matriks. Penggunaan operator ini secara otomatis menemukan solusi yang memperkecil error kuadrat dalam A.x-b. Penyelesaian ini mempunyai nilai praktis yang besar dan dinamakan peneyelesaian kuadrat terkecil.

III.3. Regeresi Linear

Pada permasalahan science dan engginering dalam pemodelan data orang sering menggunakan metode pencocokan kurva . Matlab menyelesaikan masalah ini dan sekaligus menyediakan sarana untuk memanipulasi polinomial . Di dalam metode pencocokan kurva atau regresi , anda berusaha menemukan suatu kurva halus yang mendekati data tetapi tidak harus melalui setiap point data . Pada contoh dibawah dibuat suatu program untuk melakukan *curve fitting* atau regresi linear.

```
% Script fitting.m
% Aproksimasi metode least square dengan n derajat ( n > 0 ).

% membuat array data t dan y
t = linspace(0, pi/2, 10); t = t';
y = sin(2*t);

% mencocokan jumlah data dan derajat polinomial
if ( n >= length(t))
error('Degree is too big')
end

v = fliplr(vander(t));
v = v(:,1:(n+1));
c = v\y;
c = fliplr(c');
```

```
x = linspace(min(t),max(t));
w = polyval(c, x);
plot(t,y,'ro',x,w);
title(sprintf('The least-squares polynomial of degree n = %2.0f',n))
legend('data points','fitting polynomial')
```

Pada program fitting.m dibuat metode pencocokan kurva dari dua data, sebagai masukan yaitu array y dan t dan n derajat polinomial.

Function **vander(m)** membuat suatu matriks yang elemen-elemennya merupakan pangkat dari m. Dengan m adalah sebuah array. Jika ukuran array m maka function vander akan membuat matriks baru dengan ukuran m x m. sebagai contoh :

```
t=[1 5 8];

vander(t)

ans =
     1     1     1
    25     5     1
    64     8     1
```

Function **polyval(c,x)** mencari nilai polynomial derajat c dengan input data x .

Dibawah ini terdapat daftar perintah yang sering digunakan dalam operasi matriks dalam Matlab.

Daftar Function Untuk Operasi Matriks di Matlab

Generating Matrices

MATLAB provides four functions that generate basic matrices.

zeros	All zeros
ones	All ones
rand	Uniformly distributed random elements
randn	Normally distributed random elements

Function Summary

Category	Function	Description
Matrix analysis	`norm`	Matrix or vector norm.
	`normest`	Estimate the matrix 2-norm.
	`rank`	Matrix rank.
	`det`	Determinant.
	`trace`	Sum of diagonal elements.
	`null`	Null space.
	`orth`	Orthogonalization.
	`rref`	Reduced row echelon form.
	`subspace`	Angle between two subspaces.
Linear equations	`\` and `/`	Linear equation solution.
	`inv`	Matrix inverse.
	`cond`	Condition number for inversion.
	`condest`	1-norm condition number estimate.
	`chol`	Cholesky factorization.
	`cholinc`	Incomplete Cholesky factorization.
	`lu`	LU factorization.
	`luinc`	Incomplete LU factorization.
	`qr`	Orthogonal-triangular decomposition.
	`lsqnonneg`	Nonnegative least-squares.

Function Summary (Continued)

Category	Function	Description
	`pinv`	Pseudoinverse.
	`lscov`	Least squares with known covariance.
Eigenvalues and singular values	`eig`	Eigenvalues and eigenvectors.
	`svd`	Singular value decomposition.
	`eigs`	A few eigenvalues.
	`svds`	A few singular values.
	`poly`	Characteristic polynomial.
	`polyeig`	Polynomial eigenvalue problem.
	`condeig`	Condition number for eigenvalues.
	`hess`	Hessenberg form.
	`qz`	QZ factorization.
	`schur`	Schur decomposition.
Matrix functions	`expm`	Matrix exponential.
	`logm`	Matrix logarithm.
	`sqrtm`	Matrix square root.
	`funm`	Evaluate general matrix function.

Arrays and Matrices

- "Basic Information"
- "Operators"
- "Operations and Manipulation"
- "Elementary Matrices and Arrays"
- "Specialized Matrices"

Basic Information

`disp`	Display array
`display`	Display array
`isempty`	True for empty matrix
`isequal`	True if arrays are identical
`islogical`	True for logical array
`isnumeric`	True for numeric arrays
`issparse`	True for sparse matrix
`length`	Length of vector
`ndims`	Number of dimensions
`numel`	Number of elements
`size`	Size of matrix

Operators

+	Addition
+	Unary plus
-	Subtraction
-	Unary minus
*	Matrix multiplication
^	Matrix power
\	Backslash or left matrix divide
/	Slash or right matrix divide
'	Transpose
.'	Nonconjugated transpose
.*	Array multiplication (element-wise)
.^	Array power (element-wise)
.\	Left array divide (element-wise)
./	Right array divide (element-wise)

Operations and Manipulation

: (colon)	Index into array, rearrange array
blkdiag	Block diagonal concatenation
cat	Concatenate arrays
cross	Vector cross product
cumprod	Cumulative product
cumsum	Cumulative sum
diag	Diagonal matrices and diagonals of matrix
dot	Vector dot product
end	Last index
find	Find indices of nonzero elements
fliplr	Flip matrices left-right
flipud	Flip matrices up-down
flipdim	Flip matrix along specified dimension
horzcat	Horizontal concatenation
ind2sub	Multiple subscripts from linear index
ipermute	Inverse permute dimensions of multidimensional array
kron	Kronecker tensor product
max	Maximum elements of array
min	Minimum elements of array
permute	Rearrange dimensions of multidimensional array
prod	Product of array elements
repmat	Replicate and tile array
reshape	Reshape array
rot90	Rotate matrix 90 degrees
sort	Sort elements in ascending order
sortrows	Sort rows in ascending order
sum	Sum of array elements
sqrtm	Matrix square root
sub2ind	Linear index from multiple subscripts
tril	Lower triangular part of matrix
triu	Upper triangular part of matrix
vertcat	Vertical concatenation

Elementary Matrices and Arrays

: (colon)	Regularly spaced vector
blkdiag	Construct block diagonal matrix from input arguments
diag	Diagonal matrices and diagonals of matrix
eye	Identity matrix
freqspace	Frequency spacing for frequency response
linspace	Generate linearly spaced vectors
logspace	Generate logarithmically spaced vectors

Specialized Matrices

compan	Companion matrix
gallery	Test matrices
hadamard	Hadamard matrix
hankel	Hankel matrix
hilb	Hilbert matrix
invhilb	Inverse of Hilbert matrix
magic	Magic square
pascal	Pascal matrix
rosser	Classic symmetric eigenvalue test problem
toeplitz	Toeplitz matrix
vander	Vandermonde matrix
wilkinson	Wilkinson's eigenvalue test matrix

Daftar Pustaka

a) Getting Started With MATLAB , Version 6 , The MathWorks.Inc , 2002

b) MATLAB Bahasa Komputansi Teknis , Penerbit ANDI Yogyakarta , 2000

c) Numerical Methods Using Matlab , ELLIS HORWOOD , 1995

d) Mastering MATLAB 5. A Comprehensive Tutorial and reference , Prentice Hall , 1998

e) http://www.mathworks.com

Cepat Mahir Matlab

Andry Pujiriyanto
andrypuji@hmgm.geoph.itb.ac.id

Bab 4
Kontrol Program

Kontrol program sangat berguna karena memungkinkan komputansi-komputansi yang sebelumnya mempengaruhi komputansi yang akan datang. Jika anda pernah menggunakan fasilitas kontrol ini, maka bab ini bukan hal yang baru bagi anda. Namun jika kontrol program ini merupakan sesuatu yang baru bagi anda, materi ini mungkin tampak rumit.
.
Matlab menyediakan empat struktur kontrol program, yaitu loop for, loop while, kontruksi switch-case dan kontruksi if-else-end. Kontruksi-kontruksi tersebut seringkali melibatkan banyak perintah matlab, yang oleh karenanya kontruksi ini lebih banyak terdapat dalam M-file.

IV.1. Loop for

Loop for memungkinkan sekelompok perintah diulang sebanyak suatu jumlah yang tetap.Bentuk umum loop for adalah:

```
for x= array
        perintah
end
```

Keterangan: aturan pembuatan array seperti yang telah dibahas pada bab sebelumnya.

Perintah antara statemen for dan statemen end dikerjakan sekali untuk setiap kolom dalam array.Untuk tiap iterasi, x diisi dengan kolom array berikutnya, yaitu dalam iterasi ke-n dalam loop, contoh :

```
for i=1:10
      x(i)=sin(i*pi/10);
end

x

x =
    0.3090    0.5878    0.8090    0.9511    1.0000    0.9511    0.8090
0.5878    0.3090    0.0000
```

Jika kita bahas statemen diatas adalah sebagai berikut : untuk n sama dengan satu sampai sepuluh, kerjakan seluruh statemen sampai statemen end berikut. Pertama kali dieksekusi loop for untuk n=1, lalu n=2 dan seterusnya sampai n=10. Setelah n=10, loop for berhenti dan setiap perintah yang ada setelah statemen end akan dikerjakan, yang dalam contoh diatas menampilkan elemen-elemen x.

Loop for tidak dapat dihentikan dengan mengubah nilai n di tengah-tengah loop, contoh :

```
for i=1:10
      x(i)=sin(i*pi/10);
      n=10;
end

x

x =
    0.3090    0.5878    0.8090    0.9511    1.0000    0.9511    0.8090
0.5878    0.3090    0.0000
```

Statemen 1:10 adalah statemen standar pembuatan array. Setiap array Matlab yang valid dapat digunakan untuk loop for , contoh :

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


for n=data
      x=n(1)-n(2)
end
```

```
x =
    -4
x =
    -4
x =
    -4
x =
    -4
```

Biasanya loop for dapat dibuat sehingga berada dalam loop for yang lain sebanyak yang diinginkan, contoh:

```
for i=1:10
for j=1:5
U(i,j)=i^2+j^2;
end
end

U

U =
     2     5    10    17    26
     5     8    13    20    29
    10    13    18    25    34
    17    20    25    32    41
    26    29    34    41    50
    37    40    45    52    61
    50    53    58    65    74
    65    68    73    80    89
    82    85    90    97   106
   101   104   109   116   125
```

Dengan cara diatas kita membuat suatu matriks dimulai dengan membuat elemen matriks baris 1 elemen kolom 1 sampai 5 , dilanjutkan dengan membuat elemen matriks baris 2 kolom 1 sampai 5 dan seterusnya sampai baris ke 10.

IV.2. Loop While

Loop for mengerjakan sekelompok perintah pengulangan yang diulang sebanyak suatu jumlah yang kita tentukan. tetapi loop while mengerjakan sekelompok perintah yang diulang secara tidak terbatas.

while ekspresi
perintah
end

perintah yang terdapat diantara statemen while dan end dieksekusi berulang kali selama semua elemen dalam ekspresi adalah benar. Biasanya evaluasi dari ekspresi menghasilkan nilai skalar, tetapi hasil yang berupa array juga dapat diterima. Jika hasilnya adalah array, semua elemen array harus bernilai benar. Perhatikan contoh berikut.

```
num=0; eps=1;
while (1+eps)>1
eps=eps/2;
num=num+1;
end
```

III.3. Kontruksi switch-else

Bila sederetan perintah harus dikerjakan dengan didasarkan pada penggunaan berulang-ulang suatu tes dengan argumen yang sama, kontruksi switch-else akan lebih tepat digunakan. Kontruksi ini mempunyai bentuk:

switch ekspresi
case test ekspresi 1
deret perintah 1
case test ekspresi 2
deret perintah 2
otherwise
deret perintah 3
end

Pada bagian diatas ekspresi dibandingkan dengan dengan test ekspresi 1 pada statemen case pertama, jika keduanya sama maka deret perintah akan dikerjakan , dan deret statemen berikutnya yang berada sebelum statemen end diabaikan. Jika pembandingan pertama tidak benar maka akan dijalankan deret perintah pada statemen case yang kedua. Jika semua pembanding dengan case gagal akan dikerjakan deret perintah 3 yang mengikuti statemen otherwise.

Contoh sederhana dari kontruksi switch-case adalah:

```
bilangan=5;
x=rem(5,2);

switch x
      case 1
disp(['bilangan ' ,num2str(bilangan),' adalah bilangan ganjil'])
      case 0
disp(['bilangan ' ,num2str(bilangan),'adalah bilangan genap'])
otherwise
disp('Bilangan ini tidak mungkin ada')
end
```

```
bilangan 5 adalah bilangan ganjil
```

III.4. Kontruksi if-else-end

Seringkali sederetan perintah harus dikerjakan dengan didasarkan pada hasil tes rasional.Dalam bahasa pemograman , logical ini dikerjakan dengan variasi kontruksi if-else-end.Bentuk paling sederhana kontruksi if-else-end adalah:

```
if ekspresi
        perintah
end
```

Perintah diantara statemen if dan end dikerjakan jika semua elemen didalam ekspresi adalah benar, contoh:
Pada kasus dua pilihan , kontruksi if-else-end adalah :

```
if ekspresi
        perintah dikerjakan jika benar
else
        perintah dikerjakan jika salah
end
```

Pada kasus diatas sekelompok perintah yang pertama dikerjakan jika ekspresi bernilai benar, kelompok yang kedua dikerjakan jika ekspresi bernilai salah.

Jika terdapat 3 atau lebih pilihan, konstruksi if-else-end mengambil bentuk :

```
if ekspresi 1
        perintah dikerjakan jika ekspresi 1 benar
elseif ekspresi 2
        perintah dikerjakan jika ekspresi 2 benar
elseif ekspresi 3
        perintah dikerjakan jika ekspresi 3 benar
….
else
        perintah dikerjakan jika tidak ada ekspresi yang benar
end
```

Sekarang setelah anda tahu bagaimana membuat keputusan dengan struktur if-else-end ,maka terbukalah jalan untuk menunjukkan cara melompati atau keluar dari loop for dan loop while , sebagai contoh :

```
eps=1;
for num=1:1000
      eps=eps/2;
if (1+eps) <=1
eps=eps*2
break
end
end
```

```
eps =
  2.2204e-016
```

```
num
```

```
num =
    53
```

Pada contoh diatas ditunjukkan cara lain untuk mendekati eps.Dalam cara ini loop for dintruksikan untuk berulang sejumlah seribu. Konstruksi if tersebut dimaksudkan untuk mengetes jika nilai eps kecil maka break akan menghentikan loop, lalu nilai eps dikalikan 1.

Contoh Penyelesaian masalah :

Contoh 1:

Permasalahan : Dicari nilai dari persamaan $x^2 - x - 1 = 0$,Diketahui suatu persamaan iterasi untuk mencari nilai x sbb: $x_{r+1} = 1 + (1/x_r)$ untuk r = 0,1,2, …

Diberikan nilai Xo=2 ,iterasi berhenti setelah nilai $|x_{r+1} - x_r| < 0.0005$.

Penyelesaian :

```
clear all
x1=2;       % didefinisikan terlebih dahulu nilai Xo=2
err=1;            % dibuat nilai error 1 supaya lebih dari 0.0005
while err > 0.0005    % untuk nilai error lebih dari 0.0005 dilakukan
perintah seperti dibawah
    x2=1+(1/x1);     % persamaan untuk mencari nilai x r+1
    err=abs(x2-x1);  % mencari nilai error
    x1=x2;   % membuat looping agar nilai x2 yg didapat menjadi
masukkan bagi persamaan diatas
end
```

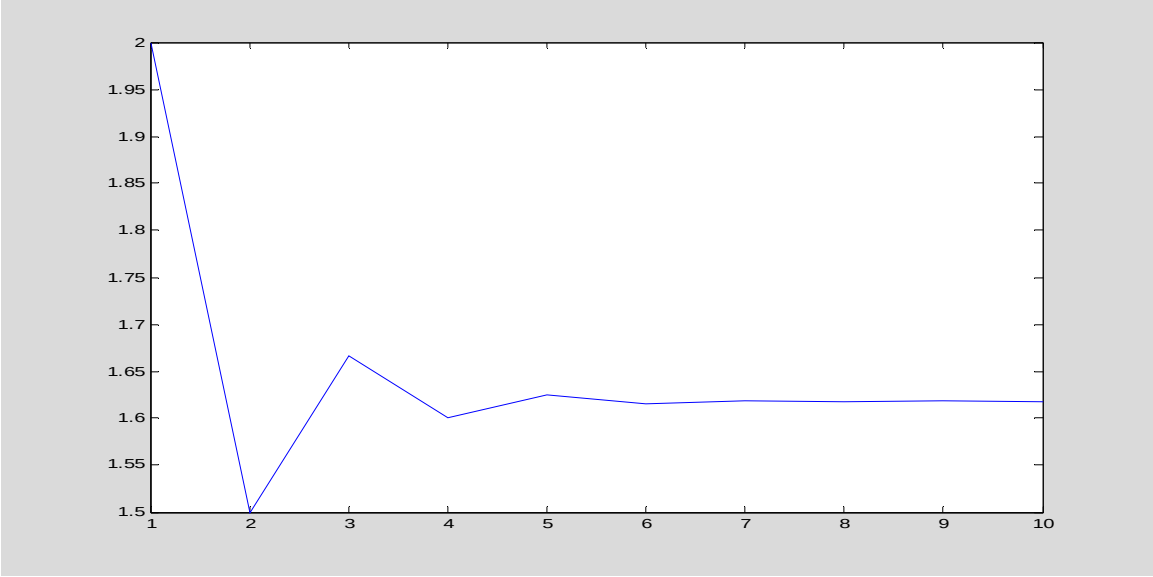
```
err

err =
  2.0429e-004
x1

x1 =
    1.6180
```

Contoh 2:

Buat grafik nilai x nya dari contoh 1 tersebut sampai didapat nilai error 0.0005!

Penyelesaian :

```
clear all
x1=2;
err=1;
i=1; % dibuat indeks kompenen 1 untuk array x
x=[ ];
x(1)=x1; % membuat elemen pertama array x == x1
while err > 0.0005
    i;
    x2=1+(1/x1);
    err=abs(x2-x1);
    x(i+1)=x2; % Untuk menyimpan nilai x yang didapat di array x
    x1=x2;
    i=i+1; % Membuat indeks 1 bertambah 1 tiap satu kali looping
end
t=1:length(x); % membuat array berdasarkan banyaknya looping
plot(t,x)
```



Contoh 3 :

Diketahui data porositas dari sebuah sumur bor

Well ID	X	Y	Porosity
1	2280	890	4
2	1240	1210	1,5
3	1651	1290	5,7
4	2169	1230	2,9
5	2059	1690	10,4
6	1722	1630	16,1
7	891	1820	1,9
8	1385	2060	7,7
9	1682	2020	15,2
10	1885	2050	7,6
11	1991	2310	11,9
12	1694	2420	15,8
13	1023	2310	6,1
14	1305	2750	4,5
15	1705	2620	12,7
16	2301	2000	12,4

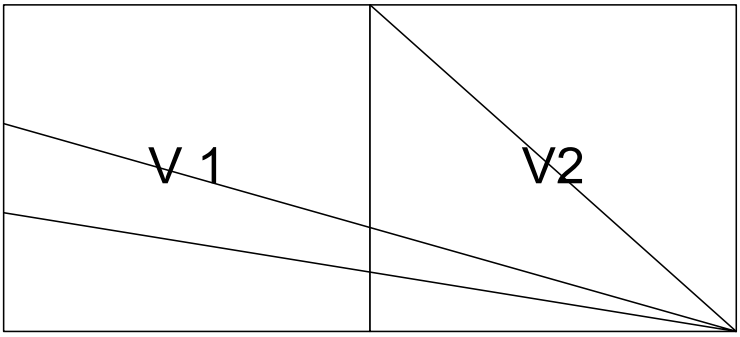
Akan dicari jarak sumur ke sumur lain :

Penyelesaian:

rumus jarak = $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$, dimana i,j = Well ID

```
% dibuat matriks segiempat yang merupakan jarak antara Well ID
n=length(x) ;% menghitung jumlah komponen x
for i=1:n
for j=1:n
d(i,j)=sqrt((x(i)-x(j))^2+(y(i)-y(j))^2); % membuat matriks d yang
berisi jarak
end
end

well=[ ]; % membuat ruang untuk matriks yang berisi no sumur
% diambil komponen dimana i <> j dan (i-j) > 0
for i=1:n
for j=1:n
if (i~=j) & (i-j) > 0
wel=[i j];
well=[well;wel];
end
end
end

m=length(well);
% membuat array d_2 yang berisi jarak yg telah ditentukan
 for i=1:m
d_2(i)=d(well(i,1),well(i,2));
end

 format short g % membuat tampilan bilangan agar berbentuk integer
% menampilkan well ID dan jaraknya
disp([well d_2' c]);
```

Contoh 4 :



Tentukan nilai V1 dan V2 dengan menggunakan inversi linear dari gambar di atas bila di ketahui waktu tempuh dari masing-masing gelombang 1.509 s, 1.447 s;0.707 s !

Penyelesaian :

Persamaan dari gambar di atas adalah :

$l_{11}^{2} * B = t_{11}$

$l_{21}^{1} * A + l_{21}^{2} * B = t_{21}$ dengan A=1/V1 dan B=1/V2

$l_{31}^{1} * A + l_{31}^{2} * B = t_{31}$

sehingga dapat dibuat persamaan matriks

$$\begin{bmatrix} 0 & -250 \\ 0 & -633 \\ 1000 & 0 \end{bmatrix} \begin{bmatrix} A \\ B \end{bmatrix} = \begin{bmatrix} 1.509 \\ 1.447 \\ 0.707 \end{bmatrix}$$

2

(0,-250)

3

(0,-633)

Maka Penyelesaian di Matlabnya adalah sbb:

```
clear all
% Input Data Koordinat source dan receiver ,t dan
s=[0,-250;0,-633;1000,0]; % source
r=[2000,-1000]; % receiver
t=[1.509;1.447;0.707]; % waktu tempuh gelombang
n=length(s);
% menghitung kecepatan awal
for i=1:n
    pj(i,1)=sqrt((r(1)-s(i,1))^2+(r(2)-s(i,2))^2);
end
v0=mean(pj);

% menghitung t calculate
```

```
for i=1:n
    t_cal(i,1)=pj(i)/v0;
end

% menghitung delta_t
for i=1:n
    del_t(i,1)=t(i)-t_cal(i);
end

% membuat matrix model
blok=2;
sumber=3;
for i=1:sumber
    for j=1:blok
        M(i,j)=pj(i)/2;
    end
end
M(3,:)=[0 ,pj(3)];
% bentuk : M*ds=del_t;
% ds=inv(M)*del_t;
% ds=inv(M'*M)*M'*del_t;

ds=inv(M'*M)*M'*del_t;

for i=1:2
    dv(i,1)=(-ds(i)*v0^2)/(1+ds(i)*v0);
end
for i=1:2
    v(i,1)=v0+dv(i);
end

fprintf('\nkecepatan V1 adalah : %5.3f\n' ,v(1));
fprintf('kecepatan V2 adalah : %5.3f' ,v(i));
```

```
kecepatan V1 adalah : 1089.443
kecepatan V2 adalah : 2000.302
```

Latihan V

1. Deret Pascal dirumuskan sebagai $(1+x)^n$, dimana n = 0,1,2 … deret pascal untuk n=2 diperlihatkan dibawah ini

   ```
       1
     1   1
   1   2   1
   ```

   Buat suatu program untuk membuat deret tersebut , sebagai contoh :

   n=2

   pa =

   ```
   1   0   0
   1   1   0
   1   2   1
   ```

   Dimana pa adalah adalah bentuk untuk deret pascal di Matlab.

2. Buat suatu program dengan input data x sebuah array dan m sebuah bilangan skalar. Pada program tersebut dibuat pengulangan array x sebanyak m, sehingga jika : **x=[1 2 3]; m=3;** maka

   y =

   ```
   1   1   1
   2   2   2
   3   3   3
   ```

3. Buat program untuk menghitung rataan terbobot dari suatu data, dimana rataan terbobot didefinisikan sebagai :

   $$rat = \frac{\sum_{k=1}^{n} w_k x_k}{\sum_{k=1}^{n} w_k}$$, disini array x berasosiasi dengan array w.

4. Buat suatu program untuk menghitung nilai dari $\sum_{k=1}^{n} w_k^{x_k}$

Daftar Pustaka

a) Mastering Matlab Web site: http://www.eece.maine.edu/mm

b) Getting Started With MATLAB, Version 6, The MathWorks.Inc, 2002

c) MATLAB Bahasa Komputansi Teknis, Penerbit ANDI Yogyakarta 2000

d) Numerical Methods Using Matlab, ELLIS HORWOOD, 1995

e) Mastering MATLAB 5. A Comprehensive Tutorial and reference, Prentice Hall, 1998

Cepat Mahir Matlab

Andry Pujiriyanto
andrypuji@hmgm.geoph.itb.ac.id

Bab 5
Fungsi M-File

Ketika anda menggunakan fungsi-fungsi di matlab seperti inv,abs,sqrt,exp , matlab menerima variabel yang anda berikan,menghitung hasil berdasarkan input dan mengembalikan hasil akhirnya pada anda sendiri.Bila kita ibaratkan fungsi adalah sebuah kotak hitam dimana yang kita lihat hanyalah apa yang masuk dan apa yang keluar.
Sifat-sifat demikian membuat fungsi dapat menjadi suatu alat yang baik untuk mengevaluasi perintah-perintah yang menggabungkan fungsi-fungsi matematika atau deretan perintah yang sering digunakan untuk memecahkan suatu masalah besar.
Untuk itu matlab menyediakan suatu struktur untuk membuat fungsi anda sendiri dalam bentuk M-file yang disimpan dalam komputer.

V.I. Aturan dan Sifat-Sifat

Fungsi M-File harus mengikuti beberapa aturan dan sejumlah sifat penting.Aturan-aturan dan sifat-sifat tersebut meliputi :

- Nama fungsi dan nama file harus identik misalnya anda membuat fungsi dengan nama pangkat maka anda memberi nama M-file anda pangkat juga.
- Baris komentar sampai dengan baris bukan komentar yang pertama adalah teks help yang ditampilkan jika anda meminta help dari fungsi yang anda buat.
- Setiap fungsi mempunyai ruang kerjanya sendiri yang berbeda dengan ruang kerja Matlab.Satu-satunya hubungan antara ruang kerja matlab dengan variabel-variabel dalam fungsi adalah variabel-variabel input dan dan output fungsi.Jika suatu fungsi mengubah nilai dalam suatu variabel input,perubahan itu hanya tampak dalam fungsi dan tidak mempengaruhi variabel yang ada dalam ruang kerja matlab.Variabel yang dibuat oleh suatu fungsi tinggal hanya dalam ruang kerja fungsi.
- Jumlah dari argumen input dan output yang digunakan jika suatu fungsi dipanggil hanya ada dalam fungsi tersebut.Fungsi ruang kerja memuat jumlah argumen input.Fungsi kerja nargout memuat jumlah argumen output.Dalam praktek,variabel-variabel nargout dan nargin biasanya digunakan untuk mengeset variabel input standar dan menentukan variabel output yang diperlukan user.

Syntax untuk membuat fungsi adalah sebagai berikut :

function y = nama_fungsi (x)

y adalah keluaran fungsi, keluaran ini dapat satu variable atau lebih dari satu variable, jika keluaran lebih dari satu variable maka mempunyai bentuk sebagai berikut :

function [y,z,a,b] = nama_fungsi (x)

x adalah masukan dari fungsi , masukan ini dapat satu variable atau lebih dari satu variable, jika masukan lebih dari satu variable maka mempunyai bentuk sebagai berikut :

function y = nama_fungsi (a,b,c,d)

Contoh 1:

Akan dibuat fungsi untuk mencari nilai y yang didefinisikan sebagai berikut:

$$f(x)= \frac{1}{(x-0.3)^2+0.01} + \frac{1}{(x-0.9)^2+0.04} - 6$$

Penyelesaiannya di matlab

Pertama buka File ➔ New ➔ M-File ketikkan :

```
% function y = humps(x)
% masukkan nilai x akan di dapat nilai y sbb
% y = 1./((x - 0.3).^2 + 0.01) + 1./((x - 0.9).^2 + 0.04) – 6

function y = humps(x)
y = 1./((x - 0.3).^2 + 0.01) + 1./((x - 0.9).^2 + 0.04) - 6;
```

kedua anda beri nama fungsi tersebut humps.m lalu simpan di folder work.

Untuk mengeksekusi fungsi dengan nilai x=2. Dapat dilakukan dengan dua cara , Lakukan di command windows atau di M-file yang baru sbb:

1.

```
y=humps(2)

y =
   -4.8552
```

2.

```
fh=@humps
feval(fh,2.0)

fh =
    @humps
ans =
   -4.8552
```

Langkah pertama merupakan hal yang umum untuk emanggil suatu fungsi dengan input x sama dengan 2.

Langkah yang kedua kita menggunakan fungsi feval untuk mengeksekusi fungsi yang kita buat.

Untuk melihat keterangan dari fungsi kita coba anda ketikkan di command window sbb:

```
help humps
```

```
  function y = humps(x)
  masukkan nilai x akan di dapat nilai y sbb
  y = 1./((x - 0.3).^2 + 0.01) + 1./((x - 0.9).^2 + 0.04) - 6
```

Inline Function

Untuk membuat fungsi hanya yang akan kita gunakan untuk saat tertentu saja, Matlab menyediakan sebuah perintah inline untuk mendifinisikan sebuah fungsi. Sebagai contoh untuk membuat fungsi $f(x,y)=\sqrt{x^2+y^2}$, maka anda buat di matlab perintah sbb:

```
f=inline ('sqrt(x^2+y^2)','x','y')
```

```
f =
     Inline function:
     f(x,y) = sqrt(x^2+y^2)
```

maka nilai untuk f(3,5) adalah

```
f(3,5)
```

```
ans =
    5.8310
```

Contoh 2

Permasalahan : Harga pembacaan gravitimeter dalah besaran skala yang harus di konversikan ke satuan percepatan gravitasi menggunakan tabel khusus untuk alat yang bersangkutan.Harga skala kelipatan 100 langsung di konversikan menjadi harga dalam miligal sesuai tabel,sisanya dikalikan dengan faktor interval dan ditambahkan sehingga menghasilkan harga pengukuran dalam satuan miligal.

Counter Read	Value In Mgals	Factor
….	….	…..
1400	1421.44	1.01552
1500	1522.99	1.01552
1600	1624.55	1.01567
….	…..	…..

Sebagai contoh jika harga skala pembacaan adalah 1510 maka berdasarkan tabel tersebut diatas gharga dalam satuan miligalnya adalah :

1500 ======= > 1522.99
10 ======= > 10 x 1.01567 =10.156
---------- +
1510 ==== > 1533.146

Coba buat fungsi skala konversi dalam matlab untuk nilai tabel yang diketahui seperti diatas.

Penyelesaian :

```
function y=kon2mgal(x)


% memisahkan bilangan ratusan dengan puluhan
% membuat kondisi untuk pemilihan nilai dalam mgal dan faktor pengali

if x >= 1400 & x < 1500
sisa=x-1400;
value= 1421.44;
faktorkali=1.01552;
elseif  x >= 1500 & x < 1600
sisa=x-1500;
value= 1522.99;
faktorkali=1.01552;
elseif  x >= 1600 & x < 1700
sisa=x-1600;
value= 1624.55;
faktorkali=1.01567;
end

% membuat hasil akhir

y=value+(sisa*faktorkali);
```

Contoh 3:

Bentuk persamaan ricker wavelet : $S(T) = \frac{R}{2b\sqrt{3}}(1-2(t/b)^2)\exp(-(t/b))^2$

dimana S(T)=amplitudo

R = konstanta
t = waktu
b = jarak antara dua titik minimum

Penyelesaian:

```
function [t,w] = genrick(freq,dt,nw);
%
% INPUTS
% freq = wavelet dominant frequency [Hz]
% dt   = sampling interval [sec]
% nw   = length of wavelet [bilangan ganjil]
%

a  = freq*sqrt(pi)/2;
nc = (nw+1)/2;
tc = (nc-1)*dt;
t  = [0:nw-1]*dt;
b  = pi*freq*(t-tc);
w  = a.*(1-2*b.^2).*exp(-b.^2);
```

Contoh 4 :

Chebyshev polynomials $T_n(x)$, n=0,1,2, ... didefinisikan sebagai rumus rekursif sbb :

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad n = 2, 3, \ldots, \quad T_0(x) = 1, T_1(x) = x.$$

Untuk mengimplementasikannya di matlab sbb:

```
function T = ChebT(n)
% Coefficients T of the nth Chebyshev polynomial of the first kind.
% They are stored in the descending order of powers.
t0 = 1;
t1 = [1 0];
if n == 0
T = t0;
elseif n == 1;
T = t1;
else
for k=2:n
T = [2*t1 0] - [0 0 t0];
t0 = t1;
```

```
t1 = T;
end
end
```

Latihan V

1. Buat suatu fungsi untuk membuat pembulatan bilangan real, sebagai contoh misalnya 5.3 bila dibulatkan sama dengan 5 dan 5.6 sama dengan 6.

2. Buat desain low pass filter yang dirumuskan sbb:

   $h_k = \frac{1}{\pi}\omega_L \frac{\sin(\omega_L k\Delta t)}{\omega_L k\Delta t}$, $k = \pm 1, \pm 2, \pm 3, \ldots$ bila Δt adalah sampling interval , makan Nyquist frequency $\omega_{Nyq} = 2\pi f_{Nyq} = 2\pi \frac{1}{2\Delta t}$

   $h_0 = \omega_L / \pi$, dengan pre-kondisi bahwa $\omega_L < \omega_{Nyq}$. Dengan bentuk fungsi **[hlp,thlp]=low_pass(fl,dt,k)** dengan fl=frekunsi , dt=waktu sampling , k=kappa

3. Buat desain high pass filter yang dirumuskan sbb:

   $h_k = \frac{1}{\pi}\omega_L \frac{\sin(\omega_L k\Delta t)}{\omega_L k\Delta t}$, $k = \pm 1, \pm 2, \pm 3, \ldots$ bila Δt adalah sampling interval , makan Nyquist frequency $\omega_{Nyq} = 2\pi f_{Nyq} = 2\pi \frac{1}{2\Delta t}$

   $h_0 = \frac{1}{\Delta t} - \frac{1}{\pi}\omega_H$, dengan pre-kondisi bahwa $\omega_H < \omega_{Nyq}$.Dengan bentuk fungsi **[hhp,thhp]=high_pass(fh,dt,k)** dengan fh=frekunsi , dt=waktu sampling , k=kappa

Daftar Pustaka

a) Mastering Matlab Web site: http://www.eece.maine.edu/mm

b) Getting Started With MATLAB, Version 6, The MathWorks.Inc, 2002

c) MATLAB Bahasa Komputansi Teknis, Penerbit ANDI Yogyakarta 2000

d) Numerical Methods Using Matlab, ELLIS HORWOOD, 1995

e) Mastering MATLAB 5. A Comprehensive Tutorial and reference, Prentice Hall, 1998

Cepat Mahir Matlab

Andry Pujiriyanto
andrypuji@hmgm.geoph.itb.ac.id

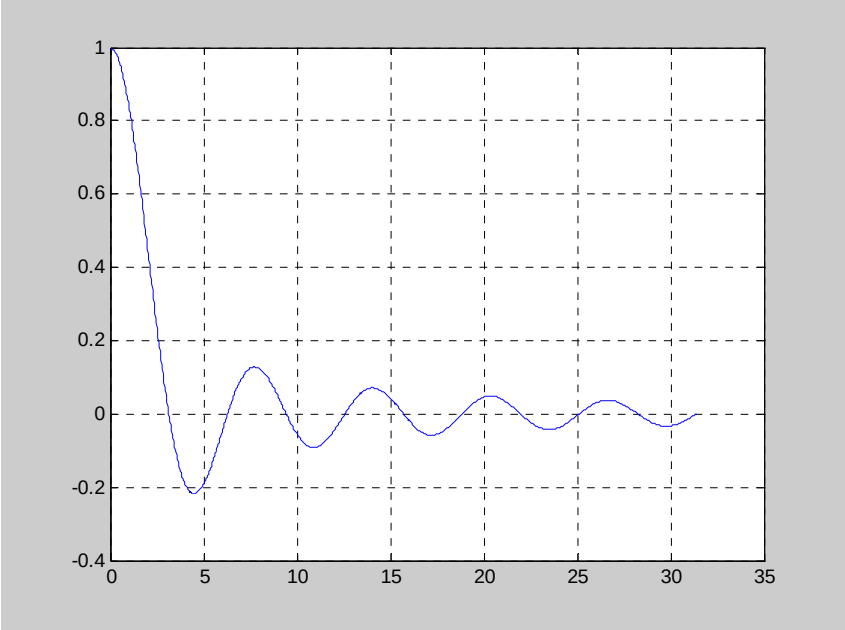
Bab 6
Membuat Visualisasi Data di Matlab

Pada bab sebelumnya telah digunakan beberapa fasilitas grafis di Matlab. Dalam bab ini akan dibahas berbagai fasilitas grafis Matlab secara lebih jelas. Matlab menyediakan berbagai fungsi untuk menampilkan data secara dua dimensi maupun tiga dimensi. Pada kasus dimana anda membuat grafik dalam tiga dimensi, anda dapat menggambar permukaan dan menempatkan binkai pada grafik tersebut. Warna digunakan untuk mewakili dimensi keempat.

Grafik Pertama

Seperti telah digunakan pada bab sebelumnya, perintah plot sering digunakan untuk menggambarkan grafik dua dimensi. Perintah plot menggambarkan data dalam array pada sumbu yang bersesuaian dan menghubungkan titik-titik tersebut dengan garis lurus. Sebagai grafik pertama anda pada bab ini akan dibuat visualisasi dari $y = \frac{\sin(x)}{x}$ dengan nilai x $\left[\frac{\pi}{100}, 10\pi\right]$,

```
x=pi/100:pi/100:10*pi;
y=sin(x)./x;
plot(x,y)
grid on
```



Fungsi dasar untuk membuat grafik dua dimensi di Matlab adalah perintah **plot** , perintah ini didasarkan dari jumlah argumen variabel input . Untuk keterangan lebih jelas mengenai fungsi plot ketik **help plot** di command window.

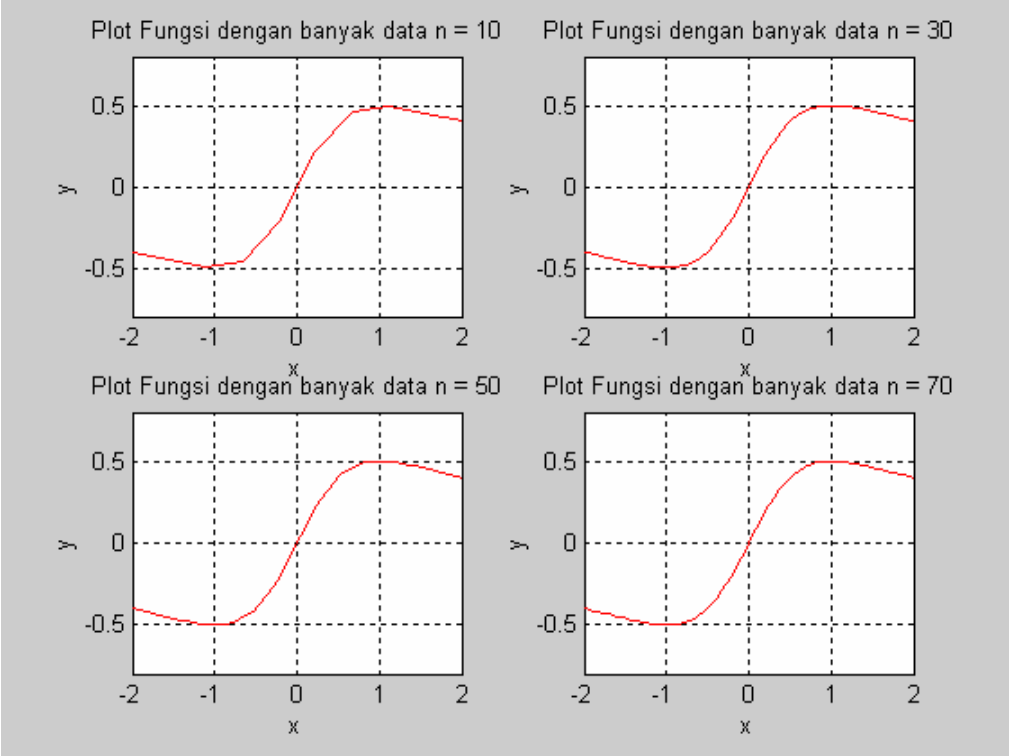
Memberi Keterangan Pada Grafik

Contoh :

Pada contoh dibawah ini dibuat grafik dari fungsi $f(x) = \frac{x}{1+x^2}$, dengan nilai x dari -2 sampai 2 menggunakan jumlah data yang berbeda.

```
% Script file graph1.
% Grafik fungsi y = x/(1+x^2)

k=0;
for n=1:2:7
n10 = 10*n;
x = linspace(-2,2,n10);
y = x./(1+x.^2);
k=k+1;
subplot(2,2,k)
plot(x,y,'r')
title(['Plot Fungsi dengan banyak data n
```

```
= ',num2str(n10)])
axis([-2,2,-.8,.8])
xlabel('x')
ylabel('y')
grid
end
```



Function **subplot** digunakan untuk membuat suatu figure dapat memuat lebih dari satu gambar. Perintah sublot didefinisikan sebagai :

subplot(n,m,i)

Perintah tersebut membagi suatu figure menjadi suatu matriks m x n area grafik dan i berfungsi sebagai indeks penomoran gambar. Subplot dinomori dari kiri ke kanan dimulai dari baris teratas.

Function **title** digunakan untuk memberi judul pada gambar. Input dari perintah title berupa string. Syntax title sebagai berikut :

title('string')

Function **xlabel** digunakan untuk memberi label sumbu pada sumbu x. Input dari perintah xlabel berupa string. Syntax xlabel sebagai berikut :

xlabel('string')

Function **ylabel** digunakan untuk memberi label sumbu y. Input dari perintah ylabel berupa string. Syntax ylabel sebagai berikut :

ylabel('string')

Function **axis** digunakan untuk mengatur nilai minimum dan maksimum dari sumbu x dan sumbu y , function axis didefinisikan sebagai :

axis([xmin xmax ymin ymax])

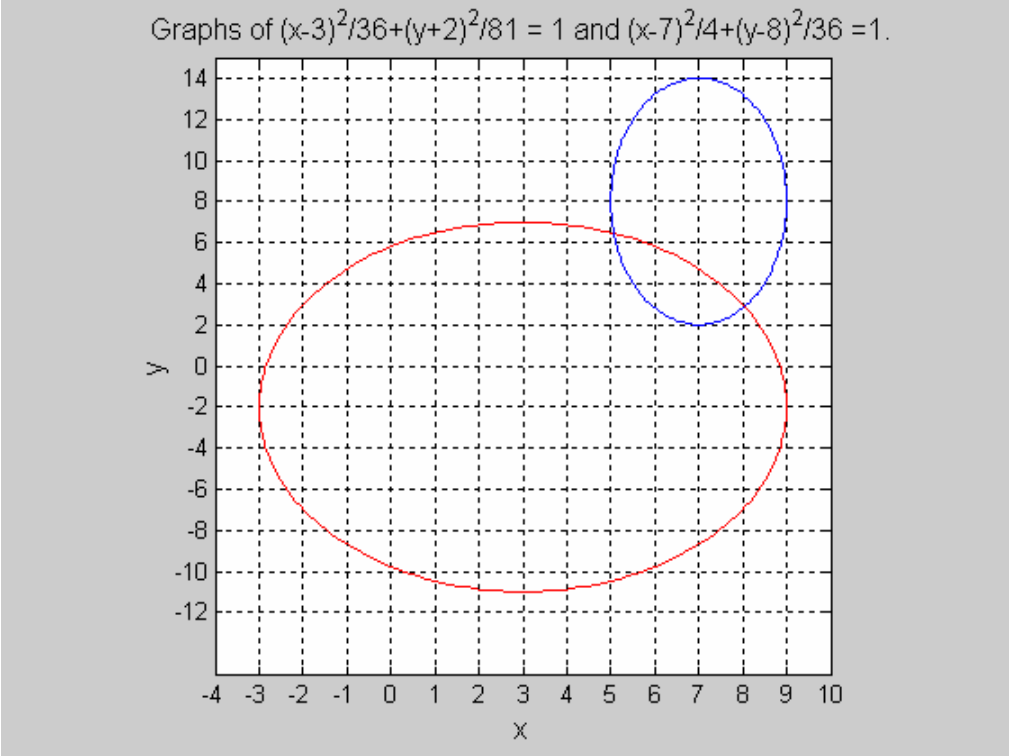
Function **grid** digunakan untuk memberi grid pada gambar kita

Membuat 2 Grafik Dalam 1 Gambar

Contoh :

Pada contoh 2 ini akan dibuat plot dari dua persamaan elips : $\frac{(x-3)^2}{36}+\frac{(y+2)^2}{81}=1$ dan $\frac{(x-7)^2}{4}+\frac{(y-8)^2}{36}=1$, menggunakan perintah plot.

```
% Script file graph2
% Grafik dari dua ellips
% x(t) = 3 + 6cos(t), y(t) = -2 + 9sin(t)
% dan
% x(t) = 7 + 2cos(t), y(t) = 8 + 6sin(t).

t = 0:pi/100:2*pi;
x1 = 3 + 6*cos(t);
y1 = -2 + 9*sin(t);
x2 = 7 + 2*cos(t);
y2 = 8 + 6*sin(t);
h1 = plot(x1,y1,'r',x2,y2,'b');
set(h1,'LineWidth',1.25)
axis('square')
xlabel('x')
h = get(gca,'xlabel');
set(h,'FontSize',12)
set(gca,'XTick',-4:10)
ylabel('y')
h = get(gca,'ylabel');
set(h,'FontSize',12)
set(gca,'YTick',-12:2:14)
title('Graphs of (x-3)^2/36+(y+2)^2/81 = 1 and (x-
7)^2/4+(y-8)^2/36 =1.')
h = get(gca,'Title');
set(h,'FontSize',12)
grid
```



Pada program graph3.m kita merubah koordinat x dan y elips dari koordinat rectangular ke dalam koordinat parametrik .
Pada program tersebut menunjukkan bahwa anda dapat menggambar lebih dari satu set data pada saat yang bersamaan hanya dengan memberikan sepasang argumen tambahan pada plot. Pada contoh diatas dibuat plot y1 terhadap x1 dan y2 terhadap x2. Plot secara otomatis menggambarkan kurva yang kedua dengan warna yang berbeda.

Pada contoh diatas dibuat pengesetan warna , huruf dan garis dari grafik plot anda dengan menggunakan perintah **`h1=plot( ……,'b') set(h1,'LineWidth',1.25) axis('square') xlabel('x')`.**

Anda dapat memilih sendiri style penandaan, warna dan bentuk garis dengan memberikan argumen ketiga pada fungsi plot untuk setiap pasangan array data. Argumen tambahan ini adalah suatu karakter string yang terdiri dari satu atau lebih karakter dari tabel di bawah ini:

Simbol	Warna	Simbol	Penandaan	Simbol	Style Garis
b	Biru	**.**	Titik	-	Gari lurus
r	Merah	**o**	Lingkaran	**:**	Garis titk-titk
g	Hijau	**x**	Tanda x	**-.**	
c	Cyan	**+**	Tanda plus		Garis terpotong &
m	Magenta	*****	Tanda	--	

y **k** **w**	Kuning Hitam Putih	**s** **d** **p** **h**	bintang Bujursangkar Diamon Pentagram Heksagram		titik Garis terpotong-potong

Sebagai contoh yang menggunakan style, garis dan penandaan pada program graph2 adalah

plot(x,y,'b:p')

Perintah tersebut memerintahkan Matlab membuat plot grafik dimana titik grafik ditandai dengan pentagram, garis mempunyai style garis titik-titik dan berwarna biru.

Perintah **`axis('square')`** memerintahkan Matlab membuat grafik yang aktif menjadi bujur sangkar bukan persegipanjang.
Pada program diatas terdapat perintah yang dimulai dengan **`h1=plot ….`** , variabel h1 menyimpan informasi mengenai grafik yang anda buat yang disebut ***handle graphics***. Perintah **`set`** pada baris selanjutnya digunakan untuk memanipulasi grafik.Harap diingat bahwa perintah ini mempunyai variabel input h1.
Perintah Matlab lain yang penting yaitu perintah **`get`** .Perintah tersebut berdasar pada parameter input yang diberi nama **`gca`** = *get current axis* .Lalu untuk memanipulasi garis yang diinginkan digunakan perintah set seperti bisa dilihat pada program graph3.

Menggunakan Perintah Hold

Anda dapat menambahkan garis pada grafik yang sudah ada dengan perintah hold. Jika anda mengeset **`hold on`** , maka Matlab tidak akan mengganti sumbu-sumbu yang sudah ada jika perintah plot yang baru diberikan. Matlab akan langsung menambahkan kurva yang baru pada grafik yang telah ada. Namun apabila data yang baru tidak mencukupi untuk batasan-batasan sumbu yang ada maka akan dilakukan penskalaan ulang. Perintah **`hold off`** akan membuat jendela figure membuat gambar yang baru. Perintah hold tanpa argumen berfungsi sebagai toggle..Sebagai contoh

Contoh :

```
% Script graph3
% Grafik dari dua ellips
% x(t) = 3 + 6cos(t), y(t) = -2 + 9sin(t)
% dan
% x(t) = 7 + 2cos(t), y(t) = 8 + 6sin(t).
% Menggunakan perintah hold

t = 0:pi/100:2*pi;
```

```
x1 = 3 + 6*cos(t);
y1 = -2 + 9*sin(t);
x2 = 7 + 2*cos(t);
y2 = 8 + 6*sin(t);
plot(x1,y1,'r') ,hold on
plot(x2,y2,'b')
```

Langkah-Langkah Untuk Memplot Data Anda

Untuk membuat suatu grafik dasar di Matlab yang dapat merepresentasikan data yang anda buat disimpulkan dalam enam tahap dalam membuat grafik pada tabel dibawah ini. Dalam langkah dibawah ini, langkah 1 sampai tiga sebenarnya sudah cukup untuk merepresentasikan data dalam sebuah grafik, langkah selanjutnya untuk membuat tampilan dari grafik lebih menarik.

Step	Typical Code

1 Prepare your data	`x = 0:0.2:12;`
	`y1 = bessel(1,x);`
	`y2 = bessel(2,x);`
	`y3 = bessel(3,x);`

2 Select a window and position a plot region within the window	`figure(1)` `subplot(2,2,1)`
3 Call elementary plotting function	`h = plot(x,y1,x,y2,x,y3);`
4 Select line and marker characteristics	`set(h,'LineWidth',2,{'LineStyle'},{'--';':';'-.'})` `set(h,{'Color'},{'r';'g';'b'})`
5 Set axis limits, tick marks, and grid lines	`axis([0 12 -0.5 1])` `grid on`

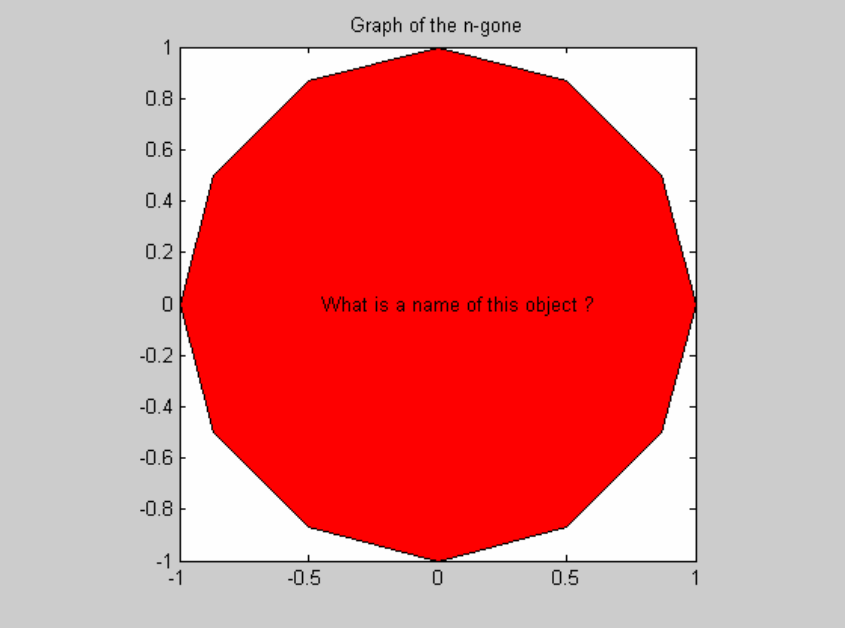
6 Annotate the graph with axis labels, legend, and text	`xlabel('Time')`
	`ylabel('Amplitude')`
	`legend(h,'First','Second','Third')`
	`title('Bessel Functions')`
	`[y,ix] = min(y1);`
	`text(x(ix),y,'First Min \rightarrow',...`
	`'HorizontalAlignment','right')`

Matlab mempunyai banyak function yang khusus untuk grafik 2 dimensi. Sebagian dari fungsi tersebut adalah **`fill, polar, bar, barh, pie, hist, compass, errorbar, stem, quiver`** dan **`feather.`**

Function fill

Pada contoh dibawah ini digunakan perintah fill untuk membuat suatu obyek.

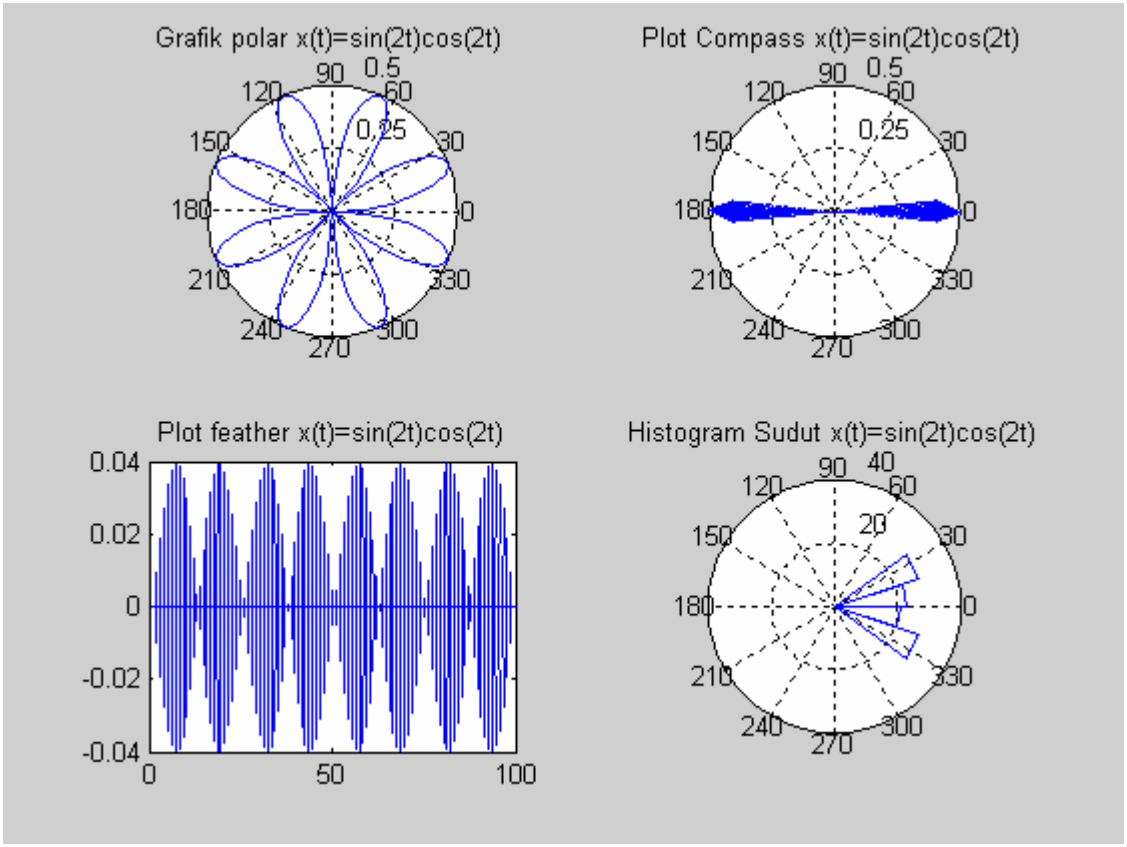
```
% Script graph4
% Menggunakan function fill

n = -6:6;
x = sin(n*pi/6);
y = cos(n*pi/6);
fill(x, y, 'r')
axis('square')
title('Graph of the n-gone')
text(-0.45,0,'What is a name of
this object ?')
```



Function polar

Grafik pada koordinat polar dapat dibuat dengan menggunakan perintah **`polar(t,r,s)`** , dengan t adalah sudut vector dalm radian, r adalah jari-jari vaktor dan s adalah karakter string yang bersifat opsional berfungsi untuk mendeskripsikan warna , symbol penandaan , dan style garis. Coba perhatikan contoh berikut :

```
% Script graph4
% membuat grafik x(t)=sin(2t)cos(2t)

t=linspace(0,2*pi);
r=sin(2*t).*cos(2*t);
polar(t,r)
subplot(2,2,1)
polar(t,r)
title('Grafikpolar x(t)=sin(2t)cos(2t)')
subplot(2,2,2)
compass(t,r)
title('PlotCompass x(t)=sin(2t)cos(2t)')
subplot(2,2,3)
feather(t,r)
title('Plotfeather x(t)=sin(2t)cos(2t)')
subplot(2,2,4)
rose(r)
title('HistogramSudut
x(t)=sin(2t)cos(2t)')
```



Function **compass** dan **feather** digunakan untuk menampilkan sudut dan besarnya elemen-elemen kompleks dalam z sebagai anak panah berasal dari pusat koordinat.

Untuk keterangan lebih detail coba gunakan perintah **help feather** , **help compass** dan **help rose**.

Function ginput

Perintah ginput merupakan cara untuk memilih titik-titik dari grafik aktif dengan bantuan mouse.**[x,y]=ginput(n)** maksudnya mengambil n titik dari sumbu aktif dan mengisikan koordinatnya dalam array kolom x dan y . Jika n tidak terisi , Matlab akan mengambil jumlah titik secara tak terbatas sampai tombol **Return** atau **Enter** ditekan. Sebagai contoh perhatikan program graph5 berikut :

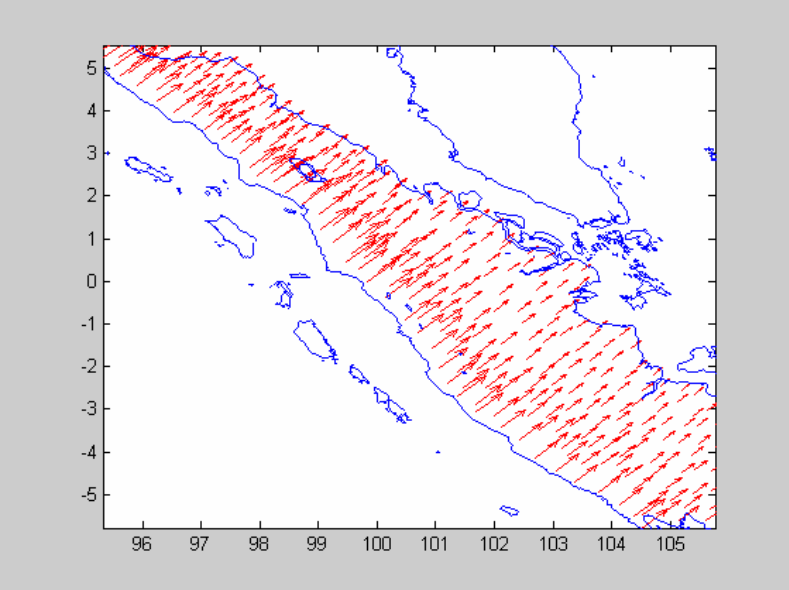
```
% Script graph5
% Loop, mengambil titik-titik pada grafik.

disp('Klik kiri untuk menentukan titik-titk yang diinginkan.')
disp('Klik kanan untuk menentukan titik terakhir')
but = 1;n=0;
while but == 1
    [xi,yi,but] = ginput(1);
    plot(xi,yi,'ro')
    n = n+1;
    xy(:,n) = [xi;yi];
end
% Interpolate with a spline curve and finer spacing.
t = 1:n;
ts = 1: 0.1: n;
xys = spline(t,xy,ts);

% Plot the interpolated curve.
plot(xys(1,:),xys(2,:),'b-');
hold off
```

Function quiver

Gambar vektor yang mengambarkan arah dan besarnya dapat divisualisasikan dengan menggunakan fungsi quiver. Misal kita mempunyai data laju perpindahan akibat pergerakan lempeng disekitar subduction zone. Data tersebut mempunyai nama subdcSMTdenseVec.dat, dan isi dari data tersebut adalah koordinat riel bumi berupa longitude (bujur), latitude (lintang), besar vektor dan kecepatan vektor.
Juga diberikan data kepulauan Indonesia di indonesia.dat. Anda akan mencoba memvisualisasikan kedua data tersebut. Dibuat script di matlab sebagai berikut :

```
% plot_vector.mat

load subdcSMTdenseVec.dat
load indonesia.dat

data=subdcSMTdenseVec; % pendefinisian data
x=data(:,1);
y=data(:,2);
U=data(:,3);
V=data(:,3);

quiver(x,y,U,V,'r'),hold on
plot(indonesia(:,1),indonesia(:,2))

axis([min(x) max(x) min(y) max(y)])
```

Apabila kita eksekusi program tersebut akan menghasilkan gambar di bawah ini :



Pada program plot_vektor.m, pertama data didefinisikan dengan menggunakan fungsi load. Lalu data tersebut disimpan di variable x, y, U dan V yang merupakan data longitude (bujur), latitude (lintang), besar vektor dan kecepatan vector.

Perintah **quiver(x,y,U,V,'r')**, membuat gambar besar dan arah vector di koordinat longitude dan latitude dengan warna merah.

Perintah **plot(indonesia(:,1),indonesia(:,2)),** memplot semua data pada kolom satu dan dua dari data indonesia.dat yang tersimpan di variable indonesia.

Perintah **axis([min(x) max(x) min(y) max(y)],** membuat batas gambar berdasarkan daerah di data vector.

Grafik Batang dan Area

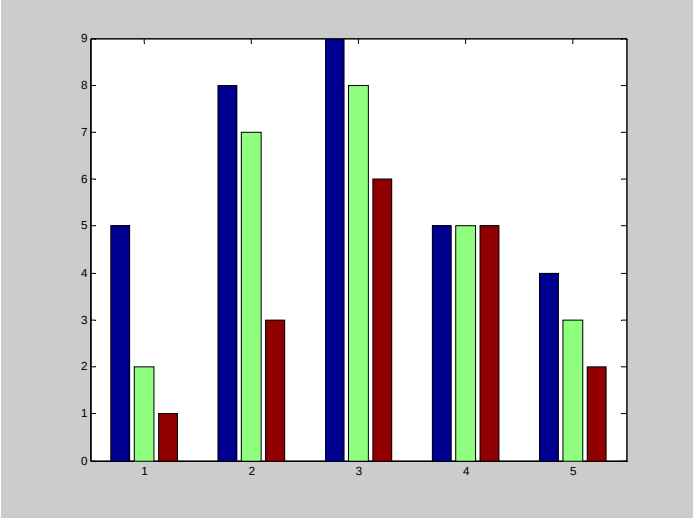
Grafik batang dapat dibuat dengan menggunakan perintah bar , bar3 , barh. Function bar dan area menampilkan data vektor dan matriks. Function jenis ini berguna untuk membandingkan beberapa data yang berbeda. Dibawah dijelaskan keterangan mengenai perintah-perintah tersebut.

Function	Description
bar	Displays columns of m-by-n matrix as m groups of n vertical bars
barh	Displays columns of m-by-n matrix as m groups of n horizontal bars
bar3	Displays columns of m-by-n matrix as m groups of n vertical 3-D bars
bar3h	Displays columns of m-by-n matrix as m groups of n horizontal 3-D bars
area	Displays vector data as stacked area plots

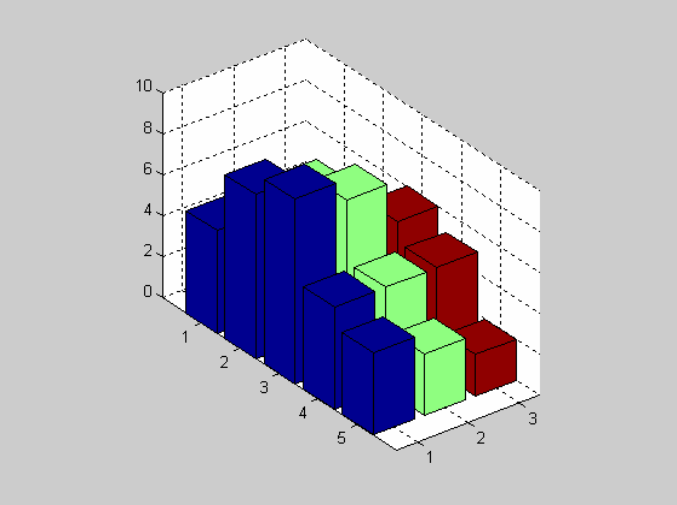
Sebagai contoh dibuat matriks Y dan digunakan function bar dan bar3 untuk matriks Y

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

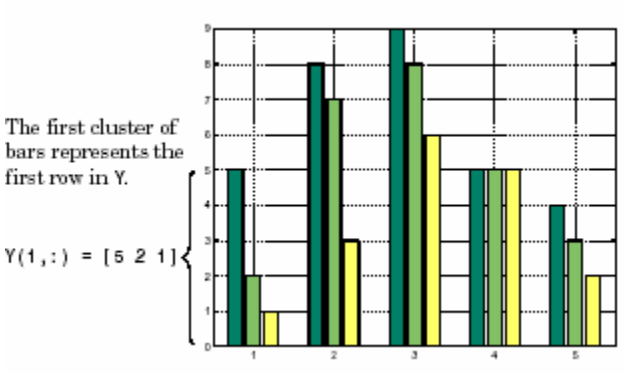
bar(Y)

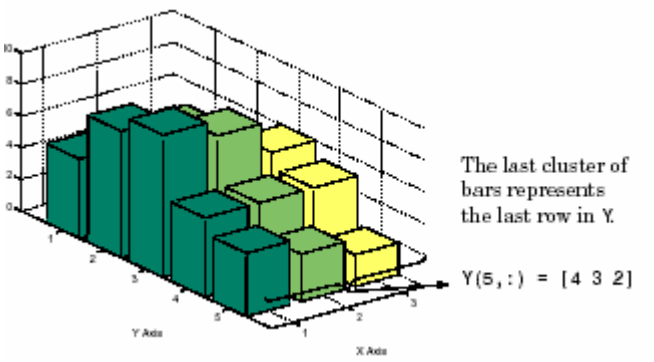


bar3(Y)



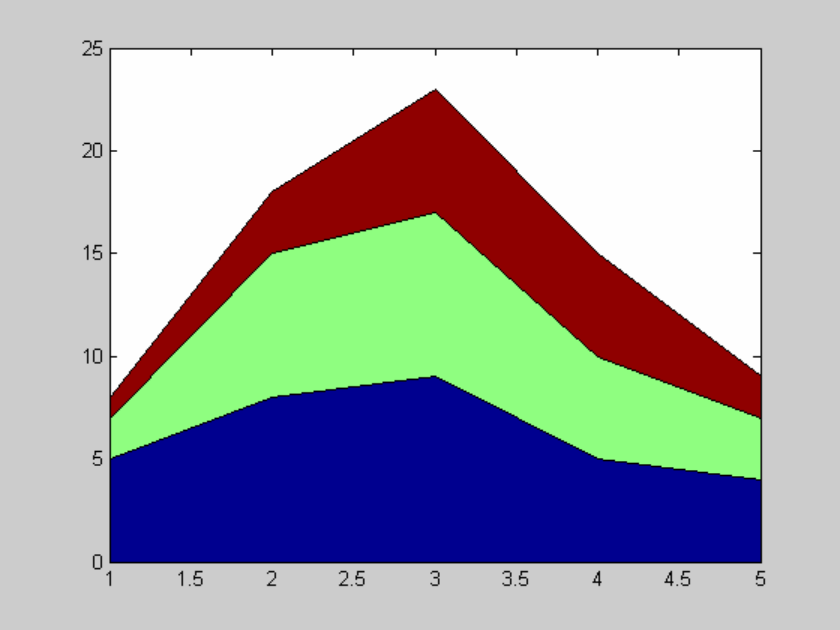
Dapat dijelaskan dari grafik diatas sebagai berikut :



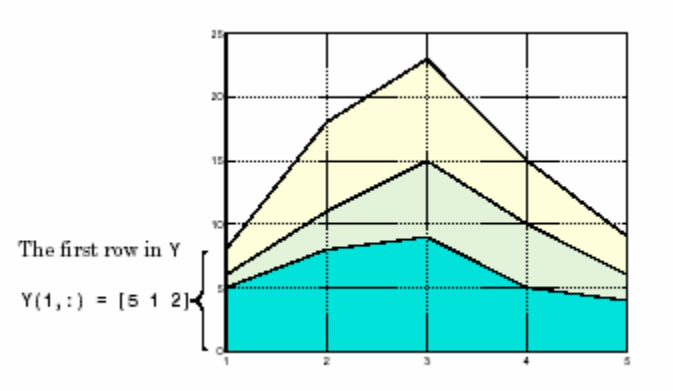


Setelah function bar selanjutnya anda coba gunakan perintah area untuk merepresentasikan matriks Y.

```
area(Y)
```



Dapat disimpulkan grafik area sebagai berikut :



Function grafik 2 D yang sering digunakan

Nama Function	Keterangan
`loglog`	Berfungsi sama dengan plot , tetapi skala yang digunakan untuk kedua sumbu adalah skala logaritma
`semilogx`	Berfungsi sama dengan plot , tetapi sumbu x menggunakan skala logaritma dan sumbu y menggunakan skala linear
`semilogy`	Berfungsi sama dengan plot , tetapi sumbu y menggunakan skala logaritma dan sumbu x menggunakan skala linear
`pie(a,b)`	Berfungsi membuat grafik lingkaran dengan a adalah suatu array dan b adalah logika opsional yang mendeskripsikan suatu jaring atau jaring yang dilepaskan dari grafik lingkaran.
`hist(y)`	Berfungsi menggambar 10 batang histogram untuk data dalam array y
`hist(y,n)`	Menggambar histogram sebanyak n , dengan n adalah bilangan skalar
`stem(z)`	Membuat suatu grafik dari titik-titik data dalam array z dihubungkan dengan sumbu mendatar oleh suatu garis.

Function imagesc

Sebuah gambar dapat diwakili oleh sebuah matriks. Oleh karena itu anda dapat menampilkan data yang berbentuk matriks ke dalam sebuah *image* dengan menggunakan perintah **`imagesc`**.syntax dari imagesc yaitu sbb:

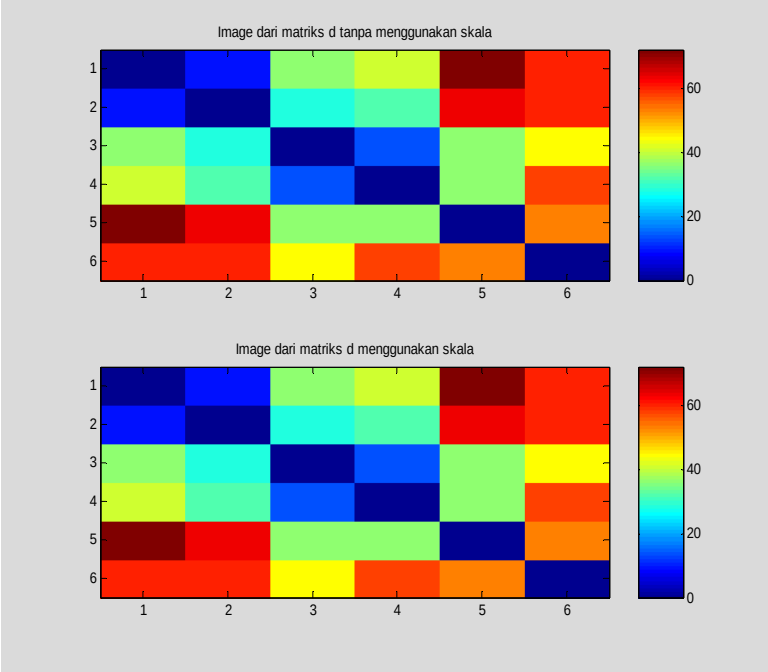
`imagesc(A)` , berfungsi menampilkan matriks A dalam sebuah image

`imagesc(x,y,A)` , berfungsi menampilkan matriks A dengan skala sumbu x dan y dimana x dan y adalah array.

sebagai contoh perhatikan contoh berikut :

```
% Script graph6
% Menggunakan function imagesc

% membuat data sebuah sumur ID,x,y

ID=[1 2 3 4 5 6];
x=[10 20 40 50 70 20];
y=[10 10 30 20 50 70];

% dicari jarak tiap titik terhadap titik lainnya
% Digunakan rumus jarak d=sqrt((x2-x1)^2+(y2-y1)^2)
```

```
n=length(x);
for i=1:n
    for j=1:n
        d(i,j)=sqrt((x(j)-x(i))^2+(y(j)-y(i))^2);
    end
end

% Menampilkan data
subplot(2,1,1)
imagesc(d)
title('Image dari matriks d tanpa menggunakan skala')
colorbar('vert')
subplot(2,1,2)
imagesc(ID,ID,d)
title('Image dari matriks d menggunakan skala')
colorbar('vert')
```



Pada program graph6 dicari jarak dari satu titik terhadap titik lainnya lalu jarak tersebut ditampilkan dalam bentuk image.

Contoh :

Script dibawah ini membuat arah sumbu y menjadi terbalik.

```
% Script flipy.m
% FLIPY: script to flip the direction of the y axis
%
% just type "flipy" at the matlab prompt

state=get(gca,'ydir');
if(strcmp(state,'normal'))
      set(gca,'ydir','reverse')
else
      set(gca,'ydir','normal')
end
```

Pada scipt flipy diatas pertama digunakan perintah **state=get(gca,'ydir'),** perintah ini untuk mendapatkan informasi dari sumbu yang ditentukan.Setelah itu digunakan **if(strcmp(state,'normal')),** untuk mngetahui apakah arah sumbu yang ditentukan tadi masih normal tau tidak. Jika arah sumbu masih normal maka arah sumbu tersebut dibalik.

Grafik 3 Dimensi

Function plot3

Dalam plot 3 Dimensi terdapat juga perintah plot3 untuk bekerja dalam tiga dimensi.Format yang digunakan sama dengan perintah plot dalam dua dimensi, kecuali data yang digunakan adalah tiga satuan, bukan sepasang. Format umum dari plot3 adalah **plot3(x1,y1,z1,S1,x2,y2,z2,S2,…)** dengan xn dan yn adalah array atau matriks sedangkan Sn adalah karakter string opsional yang mengatur warna, simbol, tanda atau style garis.

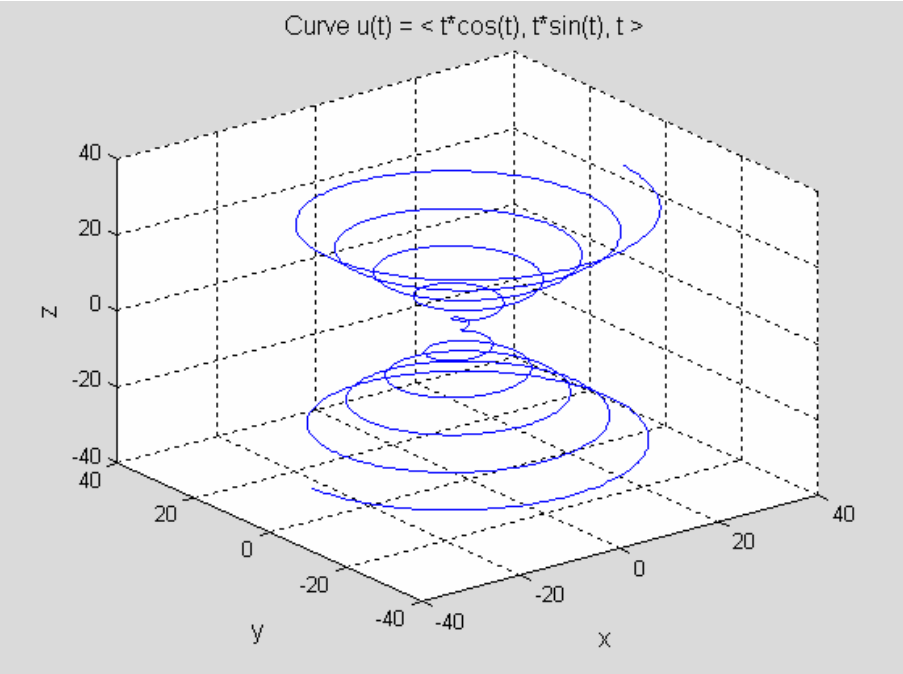
Contoh :

Akan diplot persamaan **r(t) = < t cos(t), t sin(t), t >**, dengan nilai $-10\pi \le t \le 10\pi$, maka :

```
% Script file graph7.
% Kurva r(t) = < t*cos(t), t*sin(t), t >.

t = -10*pi:pi/100:10*pi;
x = t.*cos(t);
y = t.*sin(t);
h = plot3(x,y,t);
set(h,'LineWidth',1.25)
title('Curve u(t) = < t*cos(t), t*sin(t), t >')
h = get(gca,'Title');
set(h,'FontSize',12)
xlabel('x')
```

```
h = get(gca,'xlabel');
set(h,'FontSize',12)
ylabel('y')
h = get(gca,'ylabel');
28
set(h,'FontSize',12)
zlabel('z')
h = get(gca,'zlabel');
set(h,'FontSize',12)
grid
```

Perhatikan dalam contoh diatas terdapat fungsi zlabel yang serupa dengan fungsi xlabel dan ylabel di grafik dua dimensi.Dengan cara yang sama perintah axis mempunyai bentuk tiga dimensi.Pada dasarnya konsep manipulasi grafik pada dua dimensi juga berlaku pada grafik tiga dimensi.



Grafik Jala

Matlab mendefinisikan suatu permukaan jala dengan koordinat z sebuah titik diatas grid segiempat pada bidang x-y. Perintah mesh digunakan unutk menggambarkan permukaan 3 dimensi.Sebelum anda menggunakan perintah ini sebaiknya , anda pelajar dulu perintah meshgrid . Perintah meshgrid didefinisikan sebagai berikut : **[X,Y]=meshgrid(x,y)** menciptakan suatu matriks X dengan baris-barisnya adalah duplikat dari array x dan suatu matriks Y dengan kolomnya adalah duplikat dari array y . Berikut ini contoh penggunaan function **mesh** dan **meshgrid**.

```
x = [0 1 2];
y = [10 12 14];
```

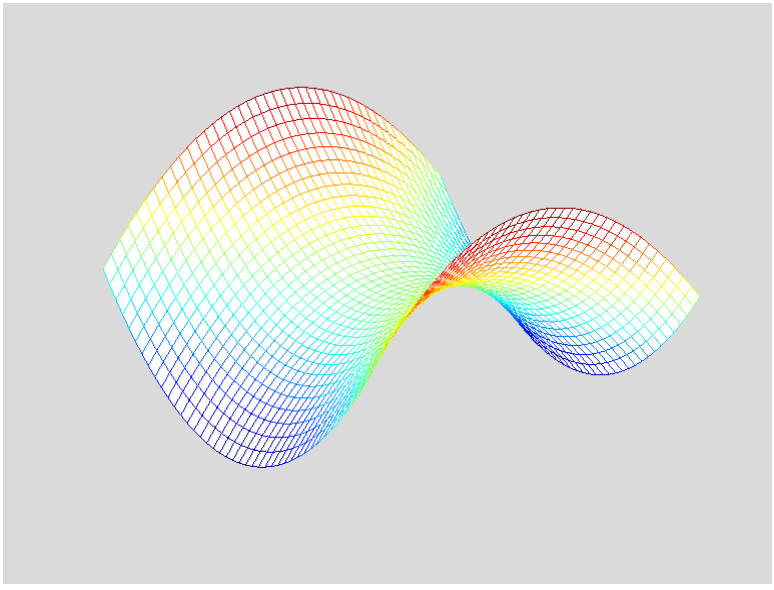
Perintah meshgrid akan membuat array x dan y menjadi dua matirks

```
[xi, yi] = meshgrid(x,y)

xi =
     0     1     2
     0     1     2
     0     1     2
yi =
    10    10    10
    12    12    12
    14    14    14
```

Bisa dilihat bahwa baris matriks xi adalah duplikat dari array x dan kolom matriks yi adalah duplikat dari array y.

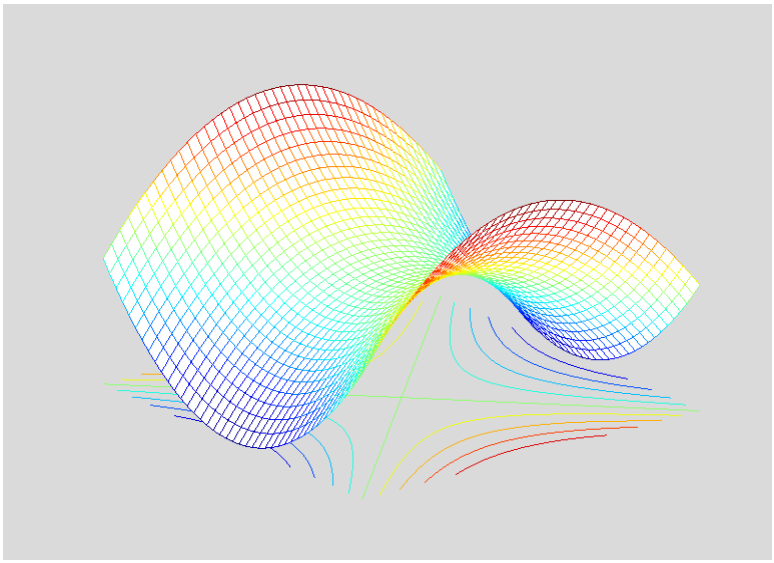
Pada contoh ini akan diplot permukaan parabola $z = y^2 - x^2$ dengan nilai $-1 \le x \le 1$ dan $-1 \le y \le 1$.

```
x = -1:0.05:1;
y = x;
[xi, yi] = meshgrid(x,y);
zi = (yi.^2) - (xi.^2);
mesh(xi, yi, zi)
axis off
```



Untuk memplot grafik mesh dengan konturnya digunakan perintah **mesc**

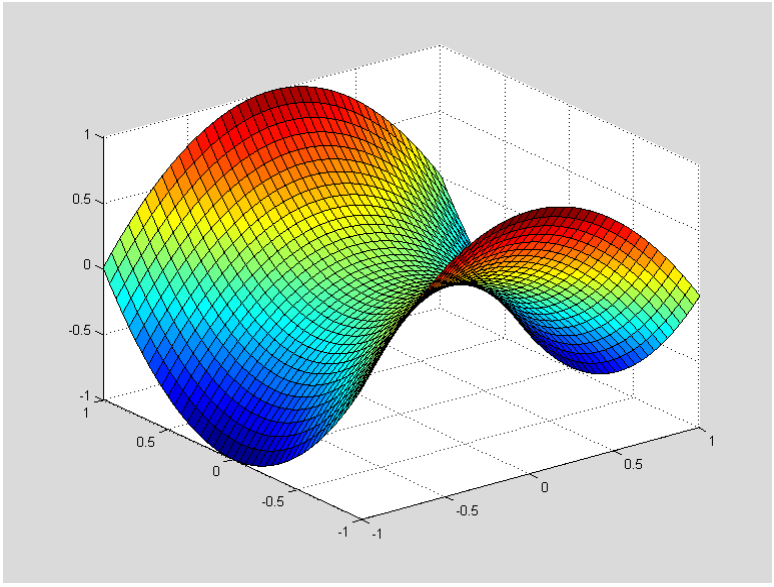
```
meshc(xi, yi, zi)
axis off
```



Dalam contoh ini mesh menggambarkan nilai-nilai elemen matriks pada titik (X_{ij}, Y_{ij}, Z_{ij}) dalam ruang tiga dimensi, mesh dapat juga menggunakan matriks tunggal sebagai argumen : mesh(Z) mengubah skala dari sumbu x dan sumbu y sebagai indeks dari matriks Z.

Matlab juga menyediakan function surf yang mempunyai susunan argumen sama dengan mesh. Berikut contoh penggunaan function surf.

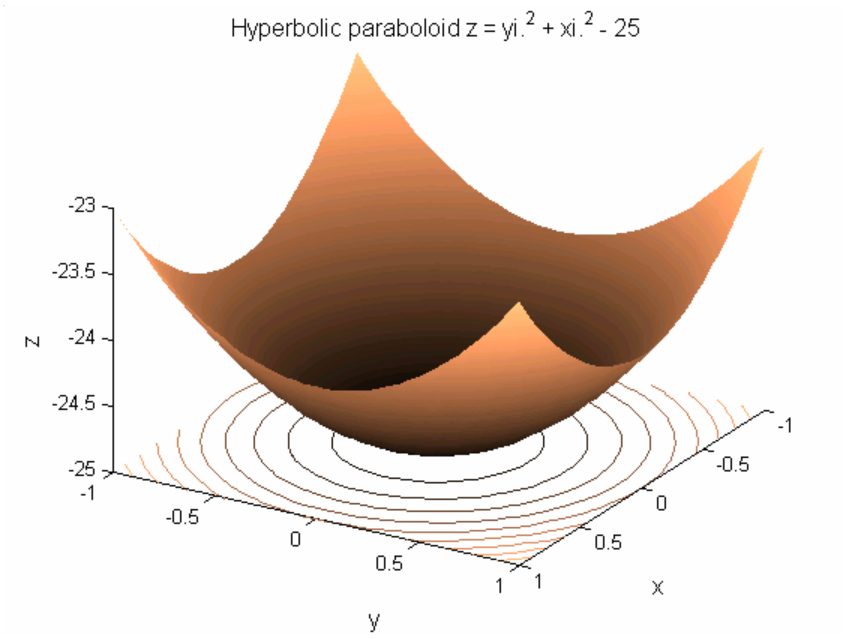
```
surf(xi,yi,zi)
```

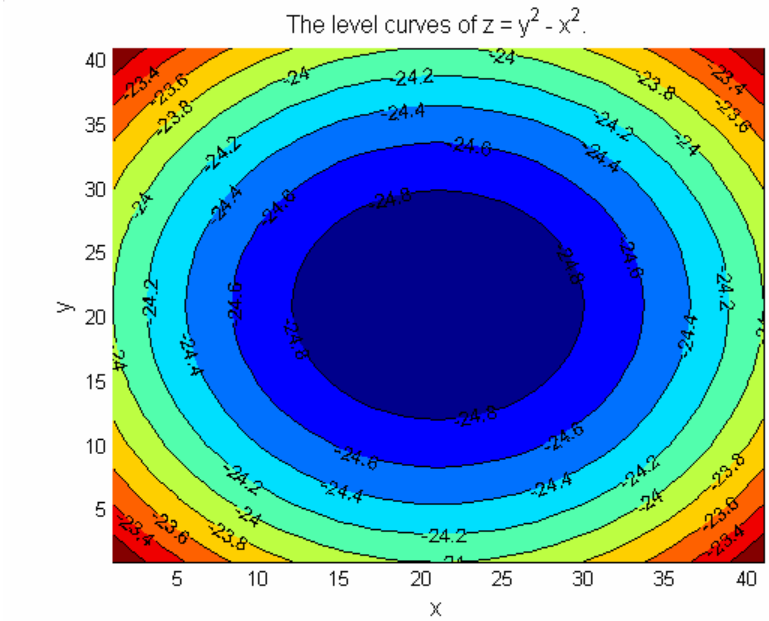


Memanipulasi Grafik

Anda dapat memanipulasi grafik supaya terlihat lebih indah, anda dapat mengeset sudut tampilan, memilih warna untuk grafik anda dan memuat shading. Sebagai contoh perhatikan program graph8.m dibawah ini :

```
% Script graph5.
% Plot permukaan zi = yi.^2 + xi.^2 - 25

x = -1:.05:1;
y = x;
[xi,yi] = meshgrid(x,y);
zi = yi.^2 + xi.^2 - 25;
figure(1)
surfc(xi,yi,zi)
colormap copper
shading interp
view([25,15,20])
grid off
title('Hyperbolic paraboloid z = yi.^2 + xi.^2 - 25')
h = get(gca,'Title');
set(h,'FontSize',12)
xlabel('x')
h = get(gca,'xlabel');
set(h,'FontSize',12)
ylabel('y')
h = get(gca,'ylabel');
set(h,'FontSize',12)
zlabel('z')
h = get(gca,'zlabel');
set(h,'FontSize',12)

figure(2)
contourf(zi), hold on, shading flat
[c,h] = contour(zi,'k-'); clabel(c,h)
title('The level curves of z = y^2 - x^2.')
h = get(gca,'Title');
set(h,'FontSize',12)
xlabel('x')
h = get(gca,'xlabel');
set(h,'FontSize',12)
ylabel('y')
h = get(gca,'ylabel');
set(h,'FontSize',12)
```





Perintah view mempunyai bentuk lain yang mungkin dalam keadaan-keadaan tertentu akan lebih berguna, **view([x y z])** menempatkan sudut pandang anda pada suatu vektor yang mempunyai koordinat Kartesius (x,y,z) dalam ruang tiga dimensi.

Dalam perinath shading, anda dapat memlih tiga cara shading untuk grafik mesh, surf, yaitu :

- **shading flat**, membuat setiap bagian garis dari jala atau jejak permukaan akan memiliki warna tetap.
- **shading faceted**, membuat shading datar dengan garis-garis jala berwarna hitam dan bertumpuk-tumpuk.
- **shading interp**, mempunyai variasi warna bagian garis secara linear.

Perintah colormap(M) menempatkan mtriks M pada peta warna untuk digunakan oleh gambar yang aktif. Sebagai contoh : colormap(cool) akan menempatkan peta warna cool.

Di bawah ini beberapa fungsi Matlab untuk menghasilkan **colormap** yang didefinisikan oleh Matlab.

hsv - hue-saturation-value color map
hot - black-red-yellow-white color map
gray - linear gray-scale color map
bone - gray-scale with tinge of blue color map
copper - linear copper-tone color map
pink - pastel shades of pink color map
white - all white color map
flag - alternating red, white, blue, and black color map
lines - color map with the line colors
colorcube - enhanced color-cube color map
vga - windows colormap for 16 colors
jet - variant of HSV
prism - prism color map
cool - shades of cyan and magenta color map
autumn - shades of red and yellow color map
spring - shades of magenta and yellow color map
winter - shades of blue and green color map
summer - shades of green and yellow color map

Contoh Penyelesaian Masalah

Data subdcSMTdenseVec.dat merupakan laju perpindahan akibat pergerakan lempeng disekitar subduction zone. Arah vektor merupakan resultan Ux dan Uy. X dan Y merupakan koordinat riel bumi berupa longitude (bujur) dan latitude (lintang). Data indonesia.dat merupakan data peta indonesia.. Estimasi nilai dilatasi pada daerah tersebut.

Strain 2-D bisa didefinisikan sebagai berikut:

$$e_{ij} = 0.5 * (U_{i,j} + U_{j,i}) = 0.5 * ((\partial U_i/\partial x_j) + (\partial U_j/\partial x_i));$$

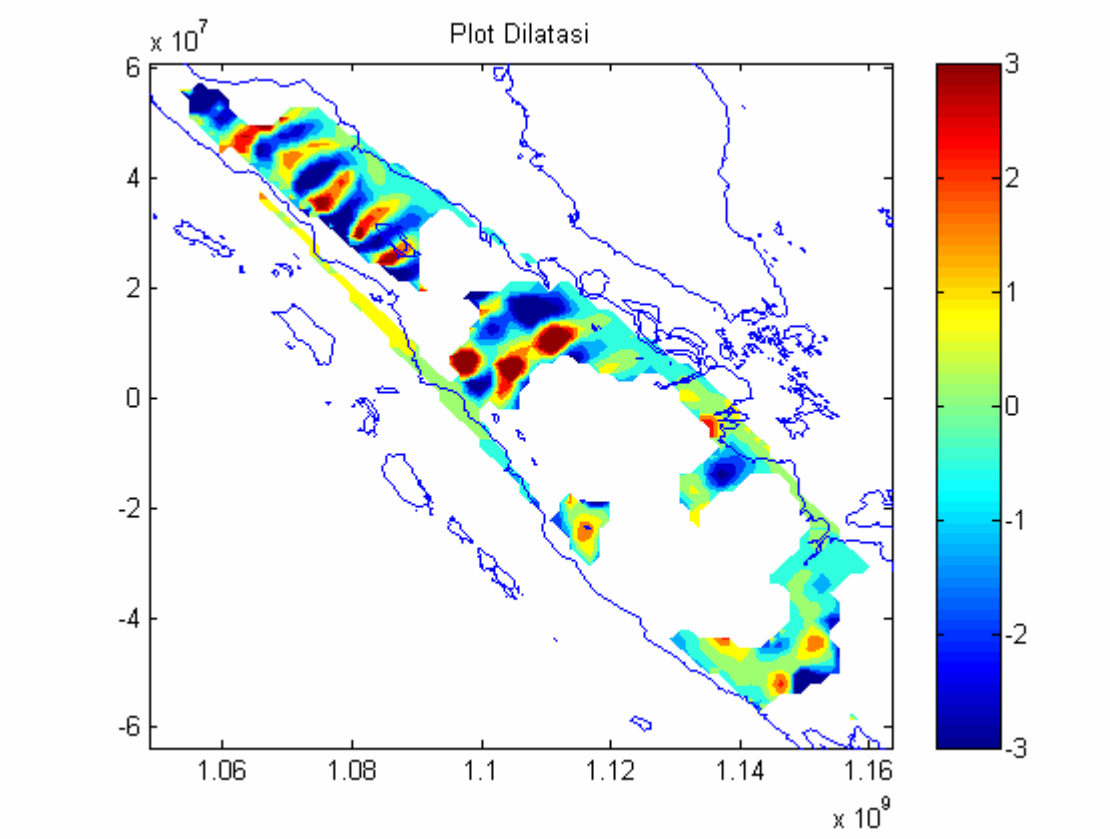
Dilatasi dapat didefinisikan dengan

$$\Delta = (e_{ii} + e_{jj})$$

Penyelesaian :

```
load subdcSMTdenseVec.dat;
data=subdcSMTdenseVec;
load indonesia.dat

xx=indonesia(:,1)*110*10^(5);
yy=indonesia(:,2)*110*10^(5);

% input data
x=data(:,1); y=data(:,2);
lonlim=[min(x) max(x)];latlim=[min(y) max(y)];
x=x*110*10^(5);
y=y*110*10^(5);
u=data(:,3); v=data(:,4);
n=length(x);
lamda=3.4*10^11;mu=lamda;

%Strain 2-D bisa didefinisikan sebagai berikut:
%eij = 0.5 * (Ui,j + Uj,i) = 0.5 * ((dUi/dxj) + (dUj/dxi));

%Strain 2-D bisa didefinisikan sebagai berikut:
%eij = 0.5 * (Ui,j + Uj,i) = 0.5 * ((dUi/dxj) + (dUj/dxi));

%Dilatasi dapat didefinisikan dengan
%delta= (eii + ejj)

%Stress diformulasikan sebagai:
%toij = lamda*ekk*deltaij + 2*myu*eij;
%lamda=myu

for i=1:n-1

   xnew(i)=0.5*(x(i+1) + x(i));
   ynew(i)=0.5*(y(i+1) + y(i));

   % itung 0.5*((dUi/dXj)+(dUj/dXi))

   du(i)=(u(i+1)-u(i)); % komponen dUi
   dv(i)=(v(i+1)-v(i)); % komponen dUj
   dx(i)=abs(x(i+1)-x(i)); % komponen dXi
   dy(i)=abs(y(i+1)-y(i)); % komponen dXj

   e11(i)=(du(i)/dx(i));
   e12(i)=(0.5*((du(i)/dy(i))+(dv(i)/dx(i))));
   e22(i)=(dv(i)/dy(i));

   % dilat=dilatasi
   dilat(i)=e11(i)+e22(i);


end
```

```
% visualisasi data perhitungan

t1=linspace(min(xnew),max(xnew),75);
t2=linspace(min(ynew),max(ynew),75);
[X,Y]=meshgrid(t1,t2); % koordinat grid

dilat_grid=griddata(xnew,ynew,dilat,X,Y,'cubic');

figure(1)
contourf(X,Y,dilat_grid,30),hold on;
title('Plot Dilatasi');caxis(10^-7*[-3 3]);shading flat
colorbar('vertical');


plot(xx,yy)

axis([min(x) max(x) min(y) max(y)])
```



Daftar Pustaka

a) Mastering Matlab Web site: http://www.eece.maine.edu/mm

b) Getting Started With MATLAB, Version 6, The MathWorks.Inc, 2002

c) MATLAB Bahasa Komputansi Teknis, Penerbit ANDI Yogyakarta 2000

d) Numerical Methods Using Matlab, ELLIS HORWOOD, 1995

e) Mastering MATLAB 5. A Comprehensive Tutorial and reference, Prentice Hall, 1998

f) Soal Ujian Tengah Semester Mekanika Sesar dan Gempa Bumi, Program studi Geofisika Departemen Geofisika & Meteorologi ITB, Bandung, 2003.