MODUL MATLAB

PROGRAM STUDI TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS ISKANDAR MUDA
2016

PRAKATA

Alhamdulillah puji syukur kepada Allah SWT, atas Rachmat dan Karunia-Nya bahwa Modul Matlab bagi mahasiswa Teknik Elektro Universitas Iskandar Muda dapat diselesaikan.

Buku Modul Matlab ini disusun sebagai acuan bagi mahasiswa dalam melakukan komputasi Numerikal dan pemrogram komputer, sehingga mahasiswa selain dapat menghitung secara manual juga dapat mahir menggunakan aplikasi komputer dengan menggunakan program Matlab.

Menyadari adanya kekurangan dalam penyusunan modul ini, maka saran yang sifatnya membangun dari berbagai pihak sangat diharapkan demi kesempurnaan penyusunan modul ini untuk waktu berikutnya.

Semoga keberadaan modul ini dapat bermanfaat khususnya bagi mahasiswa Teknik Elektro

Penyusun,
Dosen

Safrizal, S.T, M.T

BAB 1
Pendahuluan

Matlab merupakan sebuah singkatan dari Matrix Laboratory, yang pertama kali dikenalkan oleh University of New Mexico dan University of Stanford pada tahun 1970. software ini pertama kali memang digunakan untuk keperluan analisis numerik, aljabar linier dan teori tentang matriks. Saat ini, kemampuan dan fitur yang dimiliki oleh Matlab sudah jauh lebih lengkap dengan ditambahkannya toolbox-toolbox yang sangat luar biasa. Beberapa manfaat yang didapatkan dari Matlab antara lain:

- Perhitungan Matematika
- Komputasi numerik
- Simulasi dan pemodelan
- Visualisasi dan analisis data
- Pembuatan grafik untuk keperluan sains dan teknik
- Pengembangan aplikasi, misalnya dengan memanfaatkan GUI.

Matlab dapat dipadang sebagai sebuah kalkulator dengan fitur yang lengkap. Kita pernah menggunakan kalkulator dengan degan fasilitas minimal, misalnya hanya terdapat fasilitas penambahan, pengurangan perkalian dan pembagian. Kalkulator yang lebih lengkap lagi adalah kalkulator scientific dimana fasilitas yang diberikan tidak hanya yang disebutkan di atas, melainkan sudah ada fungsi-fungsi trigonometri, bilangan kompleks, akar kuadrat dan logaritma. Nah, Matlab mirip dengan kalkulator tersebut, tetapi dengan fitur-fitur yang lengkap diantaranya dapat digunakan untuk memprogram, aplikasi berbasis GUI dan lengkap dengan toolbox yang dapat dimanfaatkan untuk memecahkan masalah sains dan teknik.

Dokumentasi Matlab

Matlab memberikan kemudahan bagi para pengguna untuk

menemukan bantuan sehubungan dengan semua fasilitas yang diberikan oleh Matlab. Misalnya, bantuan tentang bagaimana memulai Matlab pertama kali, trik pemrograman, membuat grafik 2 dan 3 dimensi, menggunakan tool akuisisi data, pengolahan sinyal, penyelesaian persamaan diferensial parsial.

Untuk memperoleh bantuan tersebut, kita dapat memilih **MATLAB Menu** dari menu **Help.** Untuk bantuan tentang Matlab sendiri, dibagi atas beberapa bagian antara lain

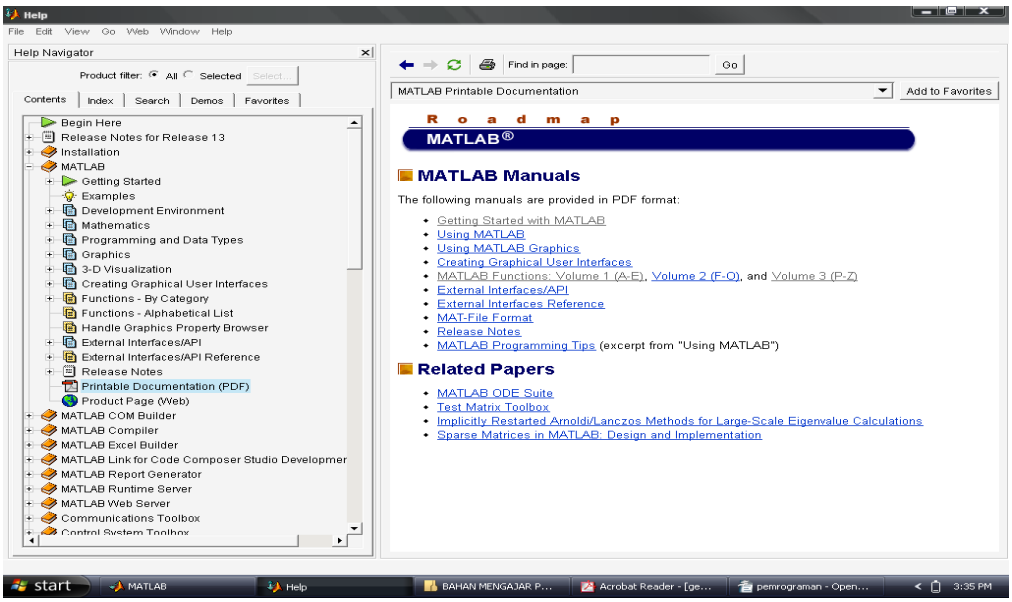
- **Development Environment,** bagian ini akan memberikan informasi yang lengkap mengenai desktop dari Matlab.
- **Mathematics,** bagian yang menjelaskan bagaimana menggunakan fitur yang dimiliki oleh Matlab untuk dalam mengolah data matematis dan statistik. Isi dalam bantuan ini dicakup antara lain: Matrks dan aljabar linier, polinomial dan interpolasi, analisis data dan statistik, fungsi function, matriks jarang (*sparse matrix).*
- **Programming and data type**, bagian ini menjelaskan bagaimana membuat *script* dan fungsi dengan menggunakan Matlab. Bantuan ini mencakup pemrograman M-File, larik, larik multidimensi, optimalisai performance Matlab, tip pemrograman Matlab.
- **Graphics,** bagian ini menjelaskan tentang bagaimana membuat atau mengeplot grafik dari data yang kita miliki. Yang termasuk dalam bagian ini antara lain, dasar-dasar pengeplotan, format grafik, membuat grafik khusus misalnya grafik dalam bentuk

bar, histogram, contour dan lain-lain

- **3-D Visualization,** bagian ini menjelaskan dengan tuntas bagaimana menampilkan data yang kita miliki dalam grafik 3 dimensi, termasuk didalamnya membuat grafik 3D, menentukan tampilan objek, transparansi objek, lighting dan lain-lain.
- **Creating Graphical User Interfaces,** bagian ini menjelaskan bagaimana kita dapat membuat GUI (Graphical User Interface) berbasis Matlab.

Disamping bagian-bagian yang sudah disebutkan di atas, disini juga disertakan beberapa bagian tambahan yang ikut melengkapi dokumentasi penjelasan tentang Matlab, diantaranya *function-By cattegory, function-Alphabetical List, handle graphic property browser, external interfaces/API, external interfaces/API references* dan lain-lain.

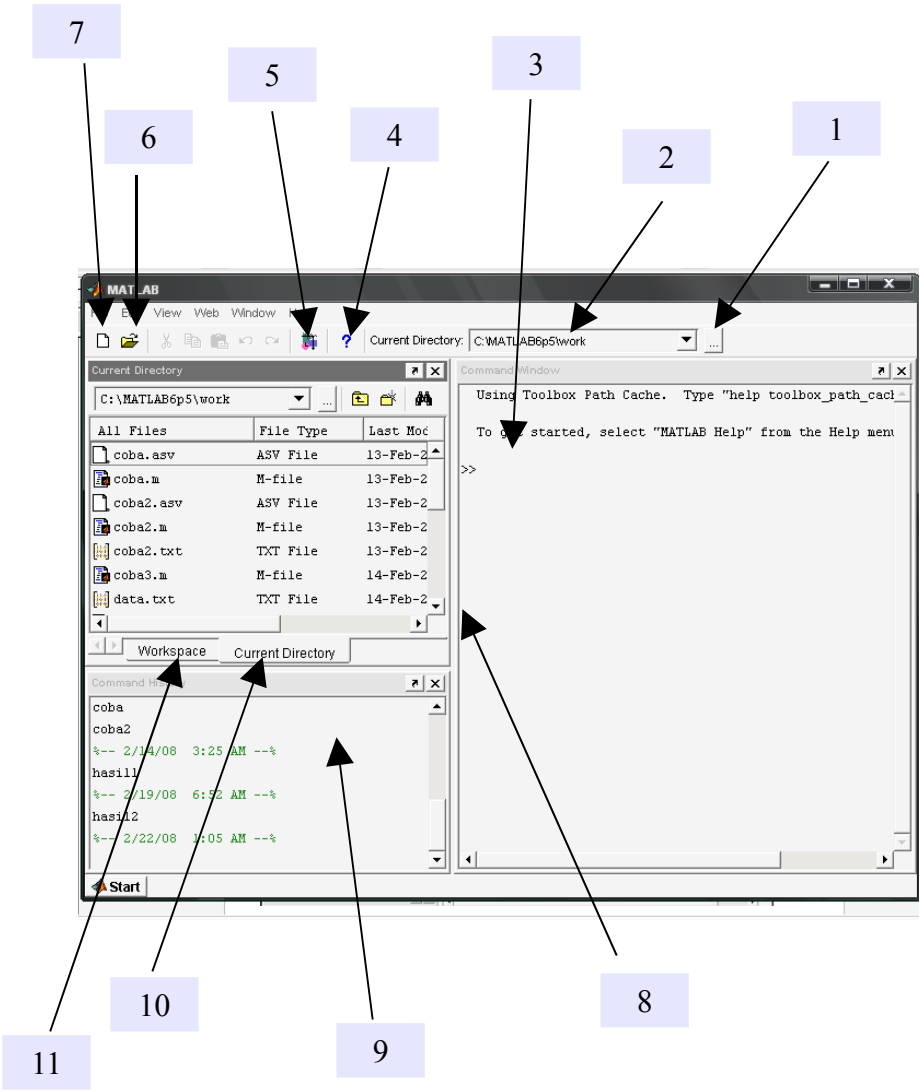
Dibawah ini diperlihatkan bagian online-help yan dapat diakses dengan cara pilih **Menu -> MATLAB Help -> Matlab.**



Gambar 1.1 . Daftar bantuan yang disediakan Matlab

Desktop Matlab

Ketika kita pertama kali menjalankan Matlab, maka tampilan pertama yang kita temui ini dikenal sebagai Desktop Matlab. Dalam desktop ini terdapat tool-tool yang berfungsi untuk manajemen file, variabel dan aplikasi yang berkaitan dengan Matlab. Dibawah ini ditunjukkan desktop Matlab versi 6.5.



Gambar 2. Tool yang disertakan pada Matlab 6.5

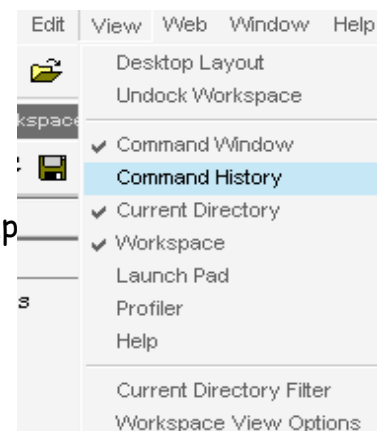
Keterangan

1. tool untuk browse direktori aktif. Dari tool ini kita dapat mengeset direktori mana yang aktif. Direktori aktif berarti bahwa

direktori inilah yang siap untuk diakses file didalamnya atau tempat yang siap untuk digunakan sebagai penyimpan data.

2. Tool yang menampilkan direktori aktif. Dari tool ini kita dapat melihat direktori mana yang aktif. Sebagai default direktori aktif Matlab adalah C:\MATLAB6p5\work, jika Matlab diinstal di direktori C:\ , kalau disimpan di D:\ maka direktori aktif defaultnya D:\MATLAB6p5\work, begitu juga di E:\ atau dimana saja.
3. Jendela ini disebut disebut sebagai Command Window. Dari jendela ini kita dapat memasukkan perintah Matlab. Disamping itu kita juga dapat menjalankan atau mengeksekusi program yang sudah kita buat di editor window dan disimpan di direktori aktif.
4. Tool yang digunakan untuk mendisplay bantuan pada Matlab.
5. Tool yang dapat digunakan untuk menuju ke **Simulink Library Browser.**
6. Tool untuk membuka file yang ada di direktori aktiv.
7. Tool untuk membuat file baru dengan format M-File.
8. Tool untuk mengatur ukuran jendela.
9. Tool untuk melihat perintah apa saja yang pernah kita jalankan melalui command window. Tool ini diberi nama **command history.**
10. Tool untuk mendisplay isi file apa saja yang terdapat di direktori aktif.
11. Tool untuk mendisplay nama variabel, ukuran, bytes dan classnya.

Tool-tool yang sudah disebutkan di atas dapat diatur kemunculannya melalui menu **View.** Misalnya, kita tidak menginginkan tampilnya jendela **command history,** maka kita kita harus menghilangkan tanda cek yang ada pada submenu command hstory. Lihat gambar 3.



Gambar 3. Menonaktifkan jendela command history

BAB 2
Memulai dengan Matlab

Ketika kita pertama kali menjalankan Matlab, maka yang tampil

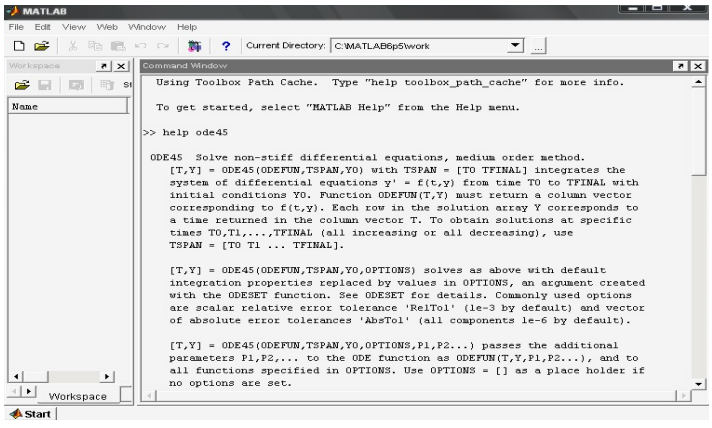
adalah Desktop Matlab seperti yang telah disinggung di atas. Diantara jendela yang yang ada pada desktop Matlab adalah command window. Di jendela inilah segala macam aktivitas berkaitan dengan perintah maupun eksekusi program berada. Perintah atau eksekusi program dapat dilakukan setelah prompt atau tanda " >> ". Sebagai contoh

```
>> x=6;
>> y=7;
>> z=x*y
z =
   42
```

A. Bantuan Matlab

Kadang-kadang, pada saat membuat program komputer kita lupa akan satu perintah atau belum tahu deskripsi atau pengguanaan dari perintah tersebut. Nah, disamping kita dapat meminta bantuan lewat tool bantuan (Help), kita dapat pula meminta bantuan lewat command window, caranya ketik **help perintah_yang dimaksud.** Misalnya, kita ingin tahu bagaimana deskripsi dan cara menggunakan fungsi ode45. Caranya cukup dengan mengetik

```
>> help ode45
```



Gambar 2.1. Mencari bantuan tentang ode45

Pada level dasar Matlab dapat dipandang sebagai sebuah kalkuklator hitung yang canggih. Contoh

```
>> Nesya = 8*100;
>>NEsya= Nesya/5;
```

```
>> NESYA=exp(NEsya*3)
>> clc;
>> pwd;
>> who
>> whos
```

Pada contoh diatas, variabel Nesya, NEsya, NESYA dipandang sebagai variabel berbeda. Variabel Nesya tidak sama dengan NEsya tidak sama pula dengan variabel NESYA. Ini berarti bahwa antara huruf besar dengan huruf kecil dibedakan. Oleh sebab itu, Matlab adalah *case sensitive.*

Operator dasar aritmatik antara lain adalah +, -, *, / da ^. Simbol ^ digunakan untuk menyatakan pangkat, misalnya

```
>> a=10
a =
       10
>> a^2
ans =
       100
>> a^3
ans =
       1000
>> 1+2*4/3
ans =
       3.6667
>> 1+2/4*3
ans =
       2.5000
```

Tetapi, coba kita lihat contoh ke-4 dan ke-5, yaitu bagaimana urutan operasi pada angka-angka tersebut. Untuk bentuk yang lebih jelas operasi 1+2*4/3 dapat dituliskan sebagai

```
1+((2*4)/3) = 1+8/3
            = 1+ 2.667
            = 3.667
```

Sedangkan operasi 1+2/4*3 dapat dituliskan sebagai

```
1+2/4*3 = 1+(2/4)*3
```

= 1 + 0.5 *3

= 2.5000

Jadi, dalam mengeksekusi sebuah operasi matematika, Matlab mengikuti aturan-aturan sebagai berikut:

- Matlab memprioritaskan operasi yang berada di dalam kurung
- operasi yang melibatkan operator * dan / (dapat * / atau / *) bekerja dari kiri ke kanan
- operasi matematika yang melibatkan operator + dan – (dapat + - atau - +) juga bekerja dari kiri ke kanan.

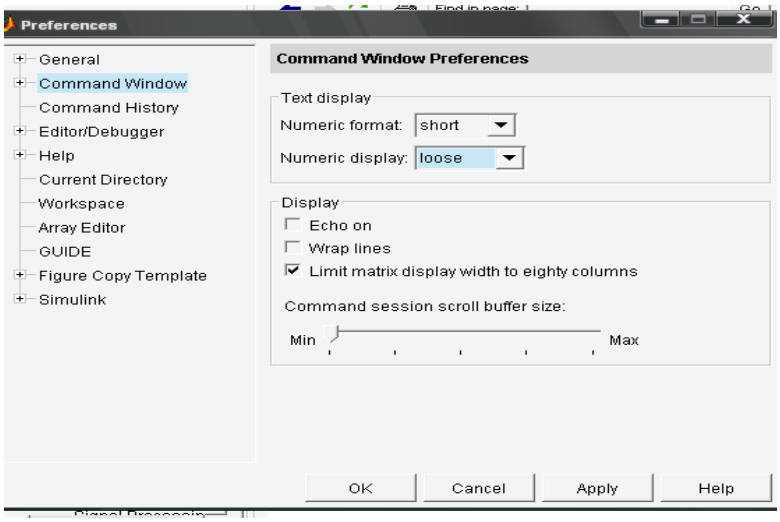
B. Cara membuat variable

Matlab tidak memerlukan deklarasi variabel atau pernyataan dimensi pada setiap variabel yang akan digunakan dalam sebuah program komputer. Variabel dalam Matlab secara otomatis akan dibuat dan disimpan setiap kali Matlab menemukan nama variabel baru. Disamping itu, hal yang sangat penting untuk diingat adalah banwa variabel Matlab bersifat **case sensitif,** artinya antara huruf besar dengan huruf kecil dibedakan. Dibawah ini diberikan beberapa aturan penulisan variabel dalam Matlab,

1. Variable tidak boleh diawali dengan angka, misalnya 2abc, 45y, 3ok43
2. Variabel dapat merupakan kombinasi antara huruf dan angka, misalnya ok45, ok45ok, abc432 dsb
3. Variabel tidak boleh menggunakan karakter khusus Matlab, misalnya %, #, - , +, = , dsb. Misalnya %ok, net-cost, %x, @sign dsb.
4. Hindari memberikan nama variabel dengan nama-nama khusus yang ada di Matlab, misalnya hindari memberikan nama variabel dengan nama pi, eps, i, j. Karena pi=22/7, eps=2^{-54}, i dan j memiliki harga $\sqrt{-1}$.

C. Format penulisan angka

Di dalam Matlab dikenal beberapa format penulisan angka yang kelak akan sangat berguna. Format penulisan angka dapat diatur melalui desktop Matlab, caranya pilih menu **File > Preferences > Command Window** , kemudian pilih format yang diinginkan.



Gambar 2.2. Mencari bantuan tentang ode45

Secara default, format penulisan angka di Matlab adalah **format short** seperti yang dapat dilihat pada gambar 2.2. Untuk mengubah ke bentuk format penulisan angka yang lain dapat dilakukan dengan menuliskan perintah

>> format format_numerik_yang_diinginkan

misalnya

>> format long

Tabel 1 diberikan contoh perintah untuk mengubah format penulisan angka yang diinginkan. Ada 8 (delapan) format penulisan angka yang dikenal dalam Matlab ditambah dengan beberapa perintah untuk mengubah bentuk real menjadi integer.

Tabel 2.1 Format penulisan angka.

No	perintah	Contoh keluaran

1	>> format short	3.1429 (4 angka di belakang koma)
2	>> format long	3.14285714285714
3	>> format short e	3.1429e+000
4	>> format long e	3.142857142857143e+000
5	>> format rational	22/7
6	>> format short g	3.14286
7	>> format long g	3.14285714285714
8	>> format bank	3.14

Beberapa perintah Matlab untuk membulatkan angka antara lain

- ceil() : perintah untuk membulatkan angka ke bil integer di atasnya (arah tak berhingga)
- floor(): perintah untuk membulatkan angka ke bil integer di bawahnya (arah minus tak berhingga)
- fix() : perintah untuk membulatkan angka ke bil integer ke atas atau ke bawah menuju arah nol
- round(): perintah untuk membulatkan angka ke bil integer ke arah lebih dekat.

Perintah tambahan yang berguna untuk pemrograman

1. clc : menghapus layar di command window
2. close all : menghapus semua gambar yang tampil sebelumnya.
3. clear : perintah untuk menghapus data di memori Matlab
4. cd : perintah untuk mengubah direktori
5. pwd : perintah untuk mengetahui kita berada di direktori mana pada saat ini.
6. dir : perintah untuk mengetahui file apa saja yang ada di current directory
7. mkdir : perintah untuk membuat direktori dibawah current direktori
8. delete : perintah untuk menghapus file
9. who : menampilkan semua variabel saat ini.
10. whos : menampilkan semua variabel saat ini bersama

dengan informasi tentang ukuran, bytes, class dll

11.what : menampilkan semua file dengan ekstensi .M (M-File)

12.lookfor : `perintah untuk mencari file dengan katakunci

D. Menampilkan/Menyembunyikan Output

Kadang-kadang ada alasan tertentu kita ingin menampilkan harga dari sebuah variabel atau mungkin menyembunyikan saja. Untuk tujuan itu, kita dapat menggunakan notasi **titik koma** (semi colon) , contoh

```
>> x=2.1; y=3*x, z=x*y
y =
    6.3000
z =
   13.2300
```

Perhatikan contoh di atas, bahwa harga variabel x tidak ditampilkan, sedangkan variabel y dan z ditampilkan. Tentunya kita dapat memahami, karena setelah variabel x diikuti tanda titik koma, sedangkan y dan z tidak diikuti titik koma.

E. Fungsi Bawaan Matlab (Built-In Functios)

E.1 Fungsi Trigonometri

Ada beberapa fungsi trigonometri yang kita kenal dalam matematika. Fungsi-fungsi tersebut masuk ke dalam fungsi bawaan Matlab. Fungsi-fungsi trigonometri tersebut antara lain: sin(), cos(), tan(), sinh(), cosh(), tanh(), asin(), acos(), atan(), asinh(), acosh()dan atanh(). Yang penting untuk diingat bahwa argumen untuk fungsi trigonometri ini adalah mode radian. Contoh

```
>> sin(pi/3),cos(pi/3),tan(pi/3)
ans =
   0.8660
ans =
   0.5000
ans =
   1.7321
>> asin(0.88),acos(0.88),atan(0.88)
ans =
```

```
    1.0759
ans =
    0.4949
ans =
    0.7217
>> sinh(pi/3),cosh(pi/3),tanh(pi/3)
ans =
    1.2494
ans =
    1.6003
ans =
    0.7807
>> asinh(1.22),acosh(1.22),atanh(1.22)
ans =
    1.0287
ans =
    0.6517
ans =
   1.1558 + 1.5708i
```

E.2 Fungsi Dasar Matlab

disamping fungsi trigonometri, fungsi-fungsi dasar juga penting. Beberapa fungsi dasar tersebut antara **abs(), sqrt(), exp(), log(), log10(), log2().** Untuk lebih jelasnya, lihat tabel dibawah ini

Tabel 2.2 Fungsi dasar Matlab

No	Nama variabel	Keterangan
1	abs()	Menyatakan harga mutlak, misal $\|x\|$
2	sqrt()	Menyatakan akar pangkat dua, misal $\sqrt{x}$
3	exp()	Menyatakan harga eksponensial, misal e^x
4	log()	Menyatakan harga ln, misal ln(x)
5	log10()	Menyatakan harga logaritma basis 10, misal log(x)
6	log2()	Menyatakan harga logaritma basis 2, misal $^2\log(x)$

Contoh

```
>> z = 3+4i;
```

```
>> abs(z)
ans =
     5
>> a=100;
>> sqrt(a)
ans =
    10
>> log(a)
ans =
    4.6052
>> log10(a)
ans =
     2
>> log2(a)
ans =
    6.6439
>> exp(log10(a))
ans =
    7.3891
```

E.3 Konstanta Khusus Matlab

Di pasal terdahulu kita sudah menyinggung beberapa konstanta khusus yang mana sebaiknya dihindari untuk didefinisikan kembali sebagai konstanta. Beberapa konstanta khusus tersebut antara lain

Tabel 2.3 Konstanta khusus

No	Konstanta	Keterangan
1	pi	3.14159265...
2	i	Unit imajiner, $\sqrt{-1}$
3	j	Sama dengan i
4	eps	Ketelitian relatif floating-point
5	realmin	Bilangan floating-point terkecil
6	realmax	Bilangan floating-point terbesar
7	inf	Bilangan tak hingga
8	NaN	Not-a-Number

BAB 3
Matriks

A. Macam-Macam Matriks dan Operasi Elementer

Istilah matriks dan array sebenarnya mempunyai arti yang agak

berbeda. Matriks adalah susunan angka-angka yang membentuk baris dan kolom hingga bentuknya menjadi dua dimensi, seperti

$$A=\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Sedangkan array merupakan bentuk khusus dari matriks karena bentuknya berupa larik membentuk baris atau kolom, dengan kata lain array merupakan matriks satu dimensi. Array sering juga disebut vektor dan bentuknya seperti

$$A=\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & \cdots \end{bmatrix}$$

atau

$$A=\begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ a_{41} \\ \vdots \end{bmatrix}$$

Kemudahan yang luar biasa diberikan oleh Matlab diantaranya adalah penulisan dan operasi Matlab yang sangat mudah. Lain halnya dengan bahasa pemrograman murni seperti halnya bahasa C, fortran atau lainnya yang begitu sulitya melakukan operasi antar matriks. Sekarang marilah kita menulis matriks misalnya

$$A=\begin{bmatrix} 2 & 1 & 12 \\ 4 & 2 & -1 \\ 5 & 12 & 4 \end{bmatrix}$$

Matriks seperti di atas dapat kita tuliskan dengan cara

```
>> A=[ 2  1  12; 4  2  -1; 5  12  4]
```

atau

```
>> A=[ 2,  1,  12; 4,  2,  -1; 5,  12,  4]
```

Jadi, antara elemen satu dengan elemen yang lain dalam satu baris

dapat dipisahkan dengan spasi atau tanda koma (,)Sedangkan tanda semikolon (titik koma) digunakan untuk memisahkan antara baris satu dengan baris lainnya.

Untuk mengakses salah satu elemen dalam matriks A dapat dilakukan dengan perintah **A(elemen_baris_ke, elemen_kolom_ke).** Contoh, misalnya elemen pada baris pertama dan kolom ketiga yang berharga 12 akan dikalikan dengan 11, maka perintah yang dapat dilakukan adalah

```
>> A(1,3)*11
```

Matlab memiliki banyak fungsi untuk men*create* matriks, antara lain

1. Untuk membuat matriks simetri secara otomatis, Matlab memiliki fungsi pascal()

```
>> pascal(4)
   ans =
         1     1     1     1
         1     2     3     4
         1     3     6    10
         1     4    10    20
```

2. Untuk membuat matriks tak simetri, Matlab memiliki fungsi magic().

```
>> magic(4)
ans =
        16     2     3    13
         5    11    10     8
         9     7     6    12
         4    14    15     1
```

3. Untuk membuat matriks yang mana elemen-elemennya bilangan random dalam ranah $0 < x < 1$ dapat dibuat dengan fungsi **rand(baris,kolom).** Sedangkan apabila kita menginginkan

elemen-elemennya berada dalam ranah $0 < x < a$, maka fungsi pembangkitannya berbentuk **a*rand(baris,kolom).** Satu lagi, elemen-elemen fungsi tadi semuanya bertipe real, kalau kita menginginkan bertipe integer maka haru ditambahi fungsi **round(), ceil(), fix()** atau **floor().** Lihat contoh

- **>>** rand(3,4)

  ans =

  0.0153 0.9318 0.8462 0.6721
  0.7468 0.4660 0.5252 0.8381
  0.4451 0.4186 0.2026 0.0196

- >> 10*rand(3,4)

  ans =

  6.8128 5.0281 3.0462 6.8222
  3.7948 7.0947 1.8965 3.0276
  8.3180 4.2889 1.9343 5.4167

- >> round(10*rand(3,4))

  ans =

  2 9 5 6
  7 9 9 8
  4 6 8 7

- >> ceil(10*rand(3,4))

  ans =

  4 6 9 8
  3 8 6 6
  4 4 4 5

4. Vektor kolom adalah bentuk matriks dengan orde $n \times 1$, vektor baris merupakan bentuk matriks orde $1 \times n$, sedangkan skalar adalah matriks orde 1×1 . Pernyataan

   $u = [1,2,3]$

   $v = [2;3;4]$

$s = 10$

menghasilkan vektor baris, vektor kolom dan sekalar.

```
u =
     1     2     3
v =

     2
     3
     4
s =
    10
```

5. **Penambahan dan Pengurangan.** Operasi penambahan dan pengurangan dua buah matrika atau lebih dapat dilakukan apabila matriks pesertanya adalah seorde.

- ```
  >> x=pascal(4);
  >> y=magic(4);
  >> z1=x+y
  z1 =
     17     3     4    14
      6    13    13    12
     10    10    12    22
      5    18    25    21
  >> z2=x-y
  z2 =
    -15    -1    -2   -12
     -4    -9    -7    -4
     -8    -4     0    -2
     -3   -10    -5    19
  ```

Apabila matriks pesertanya tidak seorde, maka akan terjadi pesan kesalahan

- ```
  >> x=pascal(4);
  >> y=magic(3);
  >> z1=x+y
  ```

??? Error using ==> +

Matrix dimensions must agree.

6. Perkalian vektor. Misalkan kita memiliki dua buah vektor $u=[1,2,3]$ dan $v=[2;3;4]$, perkalian vektor *u* dan *v* tersebut dapat menghasilkan skalar maupun vektor. Untuk memperoleh hasil skalar maka $u*v$, sedangkan untuk memperoleh hasil berbentuk matriks $v*u$.

```
>> u=[1,2,3];
>> v=[2;3;4];
>> u*v
      ans =
           20
>> v*u
ans =
    2    4    6
    3    6    9
    4    8   12
```

7. Transpose matriks. Transpose matriks berarti menukarkan elemen-elemen matriks yang berada pada baris i ke kolom j atau dapat dinyatakan dalam notasi $A_{ij}^{T}=A_{ji}$. Matlab hanya membutuhkan tanda apostrop (') untuk melakukan transpose matriks A.

```
>> z=v*u
      z =
          2    4    6
          3    6    9
          4    8   12
>> z'
      ans =
          2    3    4
```

```
     4     6     8
     6     9    12
```

Tetapi hati-hati apabila vektor atau matriks yang kita miliki adalah kompleks artinya bahwa elemen-elemennya bahwa elemen-elemennya merupkan bilangan kompleks. Hal ini karena penambahan apostrop saja berarti kita melakukan transpose konjugat kompleks matriks/vektor tersebut. Sedangkan, kalau kita hanya ingin melakukan transpose saja, maka perlu digunakan tanda titik dikuti tanda apostrop (**.'**)

```
>> A=[2i+1,i-2,2;i+2,3,3i-5;2,3i+1,3]
A =
   1.0000 + 2.0000i  -2.0000 + 1.0000i   2.0000
   2.0000 + 1.0000i   3.0000            -5.0000 + 3.0000i
   2.0000             1.0000 + 3.0000i   3.0000
>> A'
ans =
   1.0000 - 2.0000i   2.0000 - 1.0000i   2.0000
  -2.0000 - 1.0000i   3.0000             1.0000 - 3.0000i
   2.0000            -5.0000 - 3.0000i   3.0000

>> A.'
ans =
   1.0000 + 2.0000i   2.0000 + 1.0000i   2.0000
  -2.0000 + 1.0000i   3.0000             1.0000 + 3.0000i
   2.0000            -5.0000 + 3.0000i   3.0000
```

8. Perkalian Matriks. Masalah matriks memang banyak digunakan dalam penyelesaian kasus matematika. Persamaan linier simultan merupakan salah satu masalah yang dapat dipecahkan dengan matriks. Perkalian matriks sendiri membutuhkan syarat yakni *jumlah kolom matriks kiri harus samadengan jumlah baris*

matriks kanan. Oleh sebab itu, perkalian matriks tidak bersifat komutatif atau $AB \neq BA$.

```
>> A=[1,2,3;2,3,4;3,4,5]
A =
     1     2     3
     2     3     4
     3     4     5
>> B=[3,2,1;4,3,2;6,5,4]
B =
     3     2     1
     4     3     2
     6     5     4
>> A*B
ans =
    29    23    17
    42    33    24
    55    43    31
>> B*A
ans =
    10    16    22
    16    25    34
    28    43    58
```

10. Matriks identitas. Matriks identitas disebut pula matriks satuan. Sebagaimana operasi perkalian memiliki angka satuan 1, sedemikian hingga $a \times 1 = a$, operasi penambahan/ pengurangan memiliki angka identitas 0, hingga $a + 0 = a$ atau $a - 0 = a$. Matriks juga memiliki identitas matriks yang biasa disimbulkan dengan I, sehingga $AI = IA = A$. Dalam Matlab, untuk membuat matriks identitas cukup menggunakan fungsi eye(m,n)

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

```
A =
     1     2     3
     2     3     4
     3     4     5
>> I=eye(3)
I =
     1     0     0
     0     1     0
     0     0     1
>> A*I
ans =
     1     2     3
     2     3     4
     3     4     5
>> I*A
ans =
     1     2     3
     2     3     4
     3     4     5
```

11. Matriks–matriks khusus. Matlab memiliki matriks khusus, yang mana dua diantaraya sudah disebutkan di atas yaitu rand(m,n) dan eye(m,n). Disamping kedua matriks tersebut ada lagi matriks khsuus lainnya yaitu ones(m,n), zeros(m,n) dan randn(m,n)

```
>> ones(3,4)

ans =
     1     1     1     1
     1     1     1     1
     1     1     1     1
>> zeros(3,4)
```

```
ans =
     0     0     0     0
     0     0     0     0
     0     0     0     0
>> randn(3,4)
ans =
   -0.4326    0.2877    1.1892    0.1746
   -1.6656   -1.1465   -0.0376   -0.1867
    0.1253    1.1909    0.3273    0.7258
```

BAB 4
Menggambar Grafik Fungsi

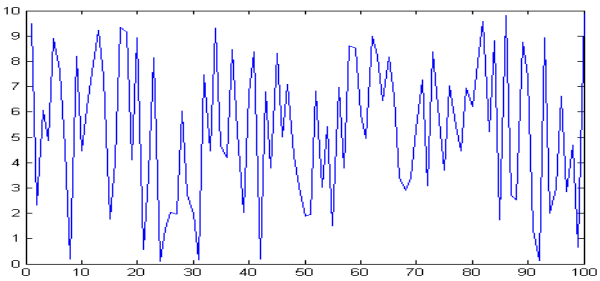
A. Membuat Grafik Garis

Peranan grafik dalam bidang sains dan teknik adalah sangat penting. Grafik dapat digunakan untuk menampilkan hasil suatu hasil penelitian maupun observasi lapangan. Dengan menampilkan dalam

sebuah grafik, pembaca akan dengan mudah memahami atau masalah tertentu. Dapat dibayangkan, misalnya kita memiliki data penelitan sebanyak 10.000 titik data dan semua data disajikan dalam bentuk tabel, sudah tentu kita akan pusing dibuatnya. Lain halnya, jika data tersebut disajikan dalam bentuk grafik, maka dengan mudah kita dapat memahami hasil penelitian tersebut.

Untuk membuat sebuah grafik garis, fungsi yang kita gunakan adalah *plot.* Fungsi ini memiliki bentuk berbeda tergantung pada argumen input yang kita berikan. Sebagai contoh, misalnya kita memiliki data dalam bentuk array dan kita simpan dalam vektor y, maka *plot(y)* akan ditampilkan grafik elemen-elemen y terhadap indeks elemen-elemen tersebut. Sedangkan, jika kita menentukan dua argumen x dan y maka plot(x,y) akan ditampilkan grafik y versus x. contoh

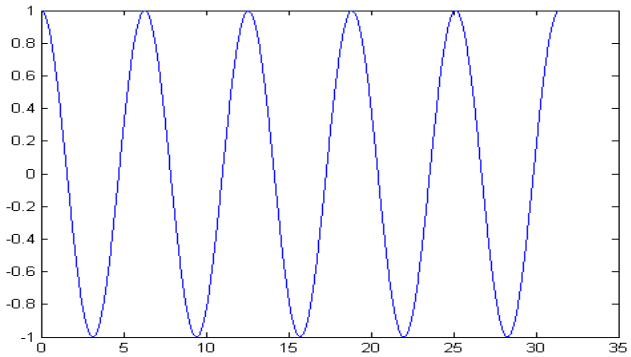
```
y=10*rand(100,1);
plot(y)
```



Gambar 4.1 Tampilan grafik y vs indeks y

Jika kita memiliki dua buah argumen x dan y, dimana $0 \leqslant x \leqslant 10\pi$ dan $y = \cos(x)$ maka grafik y vs x dapat dilihat pada gambar 4.2.

```
x=0:pi/200:10*pi;
y=cos(x);
plot(x,y)
```



Gambar 4.2 Tampilan grafik y vs x

Kita juga dapat menggunakan perintah *linspace* untuk menentukan domain fungsi, sehingga script di atas dapat dituliskan kembali menjadi

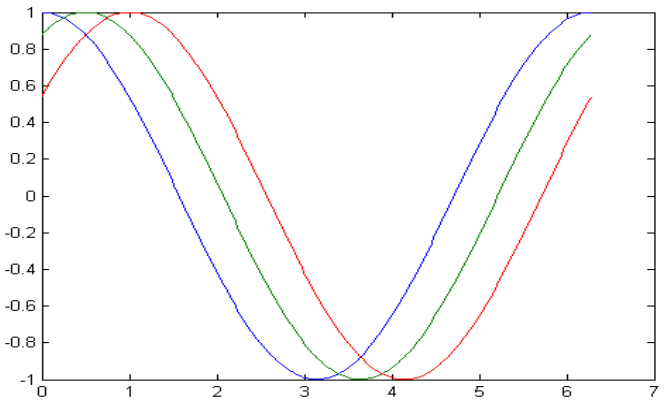
```
x=linspace(0,10*pi,200);
y=cos(x);
plot(x,y)
```

Secara umum, penggunaan perintah *linspace* mempunyai rumus

linspace(awal,akhir, jumlah_langkah)

Kita juga dapat membuat beberapa grafik dalam satu frame. Matlab secara otomatis akan membedakan grafik-grafik tersebut dengan warna yang berbeda-beda. Plot tiga grafik dalam satu frame dapat dilihat pada gambar 4.3

```
x=linspace(0,2*pi,200);
y1=cos(x);
y2=cos(x-0.5);
y3=cos(x-1.0);
plot(x,y1,x,y2,x,y3)
```

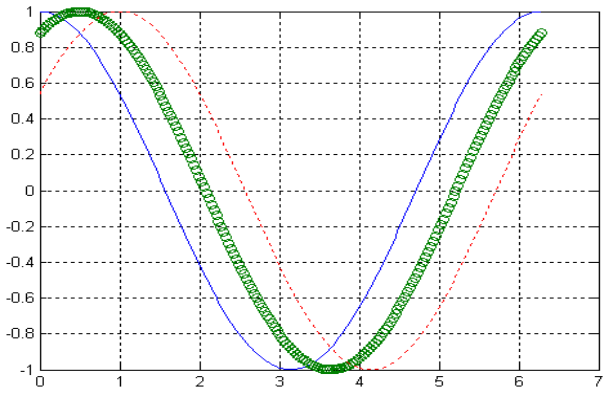


Gambar 4.3 Tampilan tiga grafik dalam satu frame x-y.

A.1 Menentukan Jenis Garis dan Jaring

Kita dapat menentukan jenis garis untuk menampilkan grafik yang kita miliki, misalnya garis putus-putus, titik-titik, kombinasi garis dan titik dan lain-lain. Sedangkan untuk menampilkan jaring-jaring pada frame, kita dapat menggunakan perintah *grid.* Lihat contoh dibawah ini

```
x=linspace(0,2*pi,200);
y1=cos(x);
y2=cos(x-0.5);
y3=cos(x-1.0);
plot(x,y1,'-',x,y2,'o',x,y3,':')
grid
```



Gambar 4.4. Menampilkan grafik dengan style garis berbeda

A.2 Warna, Jenis Garis dan Penanda

Fungsi plot dapat menerima argumen bewujud karakter maupun string yang menyatakan warna, jenis garis dan penanda. Secara umum, bentuk umum

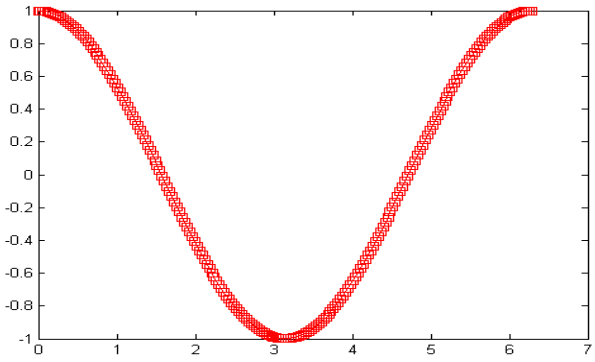
plot(x,y,'JenisGaris_Penanda_Warna')

Argumen JenisGaris_Penanda_Warna berbentuk string dan diapit oleh tanda petik tunggal.

- **Jenis garis** dapat berupa garis putus-putus (dash), titik-tiitk (dot) dan lain-lain.
- **Penanda** dapat berupa tanda bintang (*), kotak (square), bulatan (o), diamond, tanda silang (x) dan lain-lain.
- **Warna** dapat berupa merah (r), kuning (y), hijau (g), cian (C), hitam (b) dan lain-lain.

Sebagai contoh perintah plot(x,y,'-squarer') akan menampilkan grafik vs x dengan jenis grafik dash (-), penanda kotak (square) dan warna merah.

```
x=linspace(0,2*pi,200);
y=cos(x);
plot(x,y,'-squarer')
```



Gambar 4.5. Menampilkan grafik dengan style garis diikuti kotak berwarna merah

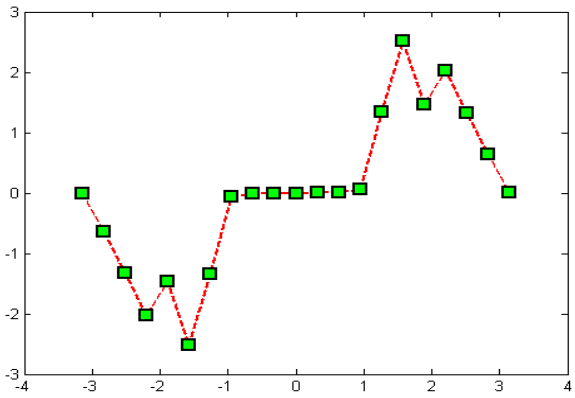
Kita juga dapat menentukan warna dan ukuran garis grafik melalui perintah-perintah

- LineWidth: menentukan ketebalan garis,
- MarkerEdgeColor: menentukan warna penanda atau warna tepian penanda masif.
- MarkerFaceColor: menentukan warna muka penanda masif.
- MarkerSize: menentukan ukuran penanda.

```
x = -pi:pi/10:pi;
y = tan(sin(x)) - sin(tan(x));
plot(x,y,'--rs','LineWidth',3,...
                'MarkerEdgeColor','k',...
                'MarkerFaceColor','g',...
                'MarkerSize',5)
```

Script di atas akan menghasilkan grafik y vs x dengan

- Jenis garis putus-putus berwarna merah dengan penanda berbentuk kotak ('--rs'),
- Tebal garis sama dengan 3
- Tepian penanda berwarna hitam (k),
- Muka penanda berwarna hijau (g),
- Ukuran penanda 5

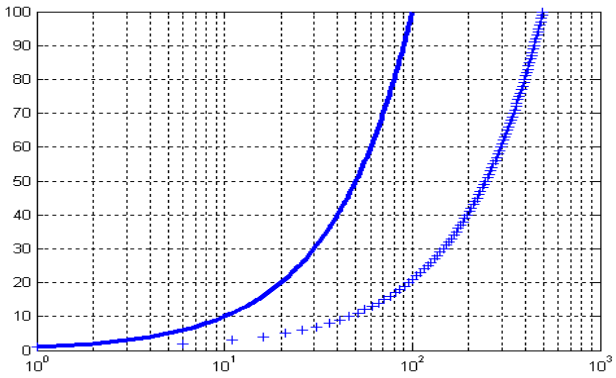


Gambar 4.6. Menampilkan grafik dengan style garis berukuran 3 diikuti kotak dengan kotak warna isi hijau dan warna tepian hitam dengan ukuran kotak 5

A.3 Menambahkan Plot Grafik Baru pada Grafik yang ada

Kita dapat menambahkan grafik baru pada grafik sebelumnya dengan perintah *hold*. Perintah ini akan aktif saat di *on*-kan atau *hold on* dan tidak aktif saat diberi perintah *hold off*. Contoh

```
clear; close all;
x=1:100;
semilogx(x,'-','LineWidth',3);
hold on;
plot(1:5:500,1:100,'+');
hold off
```

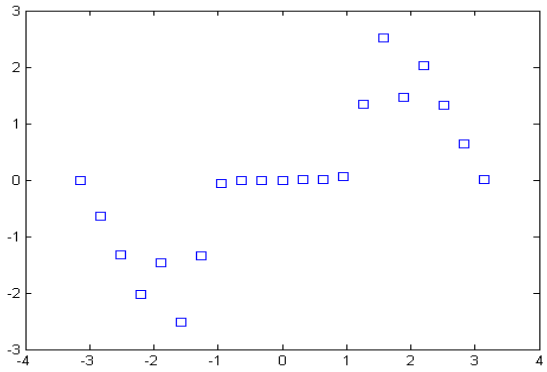


Gambar 4.7. Menambahkan grafik pada grafik terdahulu

A.4 Menggambar Titik-Titik Data

Kadang-kadang kita tidak ingin menghubungkan antar titik-titik data yang ada dengan berbagai alasan. Untuk tujuan tersebut, maka atribut atau property gambar hanya disertakan penanda (*marker)* saja.

```
clear; close all;
x = -pi:pi/10:pi;
y = tan(sin(x)) - sin(tan(x));
plot(x,y,'square')
```

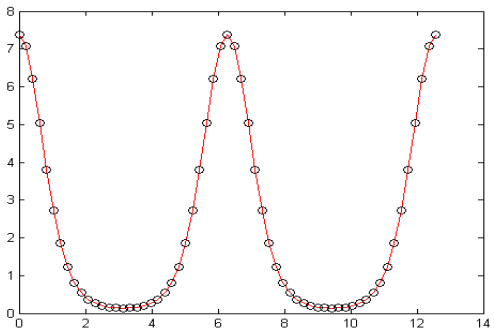


Gambar 4.8. Menggambar titik-titik data

A.5. Menggambar dengan Penanda dan Garis

Untuk mengeplot grafik dengan garis dan penanda saja, dapat dilakukan dengan mudah. Contoh,

```
x = 0:pi/15:4*pi;
y = exp(2*cos(x));
plot(x,y,'-r',x,y,'ok')
```

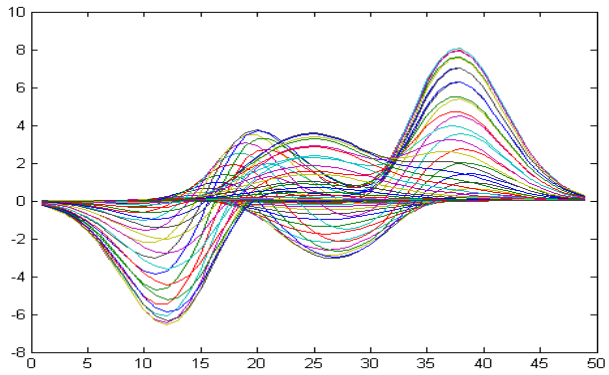


Gambar 4.9. Menggambar dengan garis dan penanda

A.6 Plot Grafik untuk Data Berbentuk Matriks

Ketika kita memanggil perintah plot untuk mengeplot grafik data bentuk vektor atau array, maka hasilnya dapat kita lihat seperti contoh-contoh di atas. Nah, bagaimana jika data yang kita miliki berbentuk matriks. Untuk memecahkan masalah ini, marilah kita ingat kembali bahwa vektor memiliki ukuran $1\times m$ atau $m\times 1$ Perintah *plot(y)* akan menampilkan setiap elemen dalam kolom atau baris y terhadap indeks elemen vektor tersebut. Demikian pula, perintah plot untuk data Y yang berbentuk matriks akan menampilkan elemen pada setiap kolom atau baris matriks terhadap indeks elemen. Jadi seandainya kita memiliki Y berukuran 10×10 , maka kita memiliki 10 (sepuluh) grafik. Sebagai contoh, fungsi *peaks* merupakan fungsi dua variabel yang menghasilkan matriks 2 dimensi dengan ukuran 49 x 49.

```
y=peaks;
plot(y)
```



Gambar 4.10 Contoh plot data berbentuk matriks

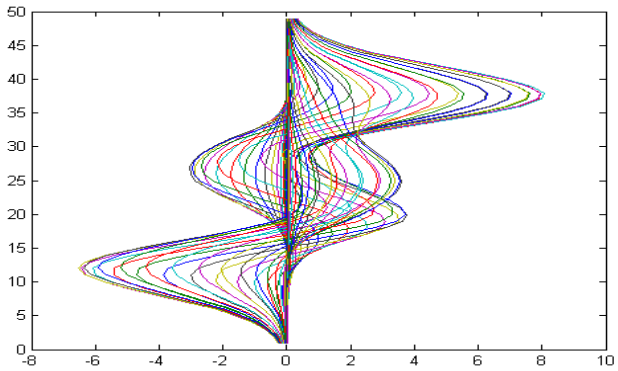
Jika kita perhatikan, grafik yang terbentuk terdiri atas 49 buah dengan warna berbeda-beda. Secara umum, jika perintah plot digunakan untuk mengeplot grafik fungsi dengan dua argumen dimana data yang terbentuk berupa matriks 2 dimensi, maka

1. jika y adalah matriks dan x berupa vektor, maka perintah plot (x,y) akan menampilkan grafik elemen pada kolom/baris matriks

y terhadap elemen vektor x.

2. jika x adalah matriks dan y berupa vektor, maka plot(x,y) akan menampilkan grafik elemen matriks pada tiap-tiap kolom/baris matriks x terhadap elemen vektor y. Contoh,

```
x=1:length(peaks);
plot(peaks,x)
```



Gambar 4.11 Contoh plot data berbentuk matriks

A.7 Memberikan Label, Legend dan Judul Grafik

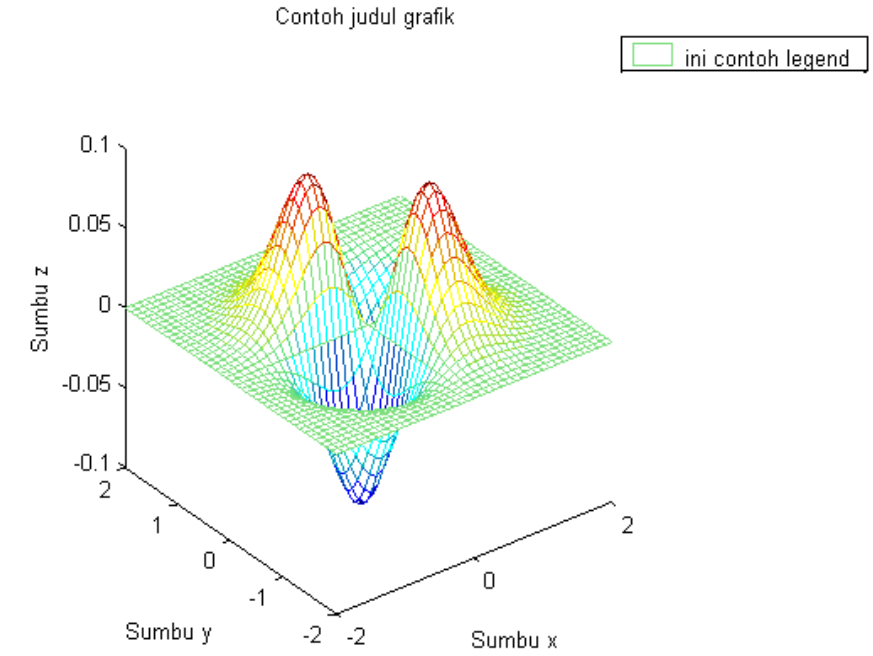
Pemberian label pada sumbu-sumbu grafik sangat penting untuk memudahkan pemahaman terhadap makna grafik itu sendiri. Perintah yang digunakan untuk tujuan tersebut antara lain

- xlabel : untuk memberikan label pada sumbu x
- ylabel : untuk memberikan label pada sumbu y
- zlabel : untuk memberikan label pada sumbu z
- tittle : memberikan judul garfik
- legend : untuk memberikan keterangan grafik

untuk lebih mudahnya perhatikan contoh di bawah ini,

```
clear; close all;
x=-2:0.1:2;
y=-2:0.1:2;
[X,Y]=meshgrid(x,y);
f=-X.*Y.*exp(-2*(X.^2+Y.^2));
```

```
mesh(X,Y,f);
xlabel('Sumbu x');
ylabel('Sumbu y');
zlabel('Sumbu z');
title('Contoh judul grafik');
legend('ini contoh legend')
```



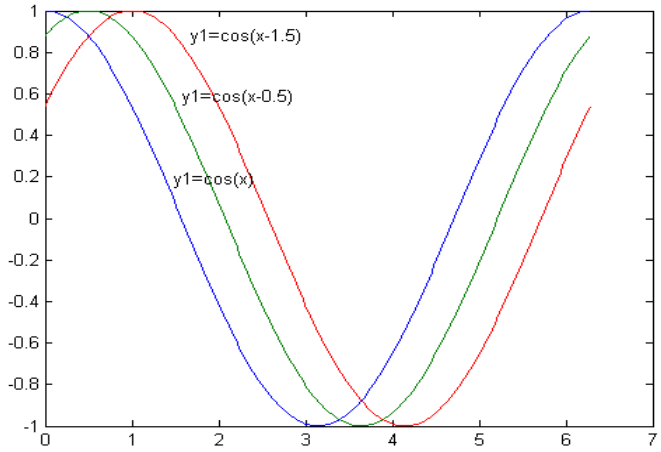
Gambar 4.12 Contoh penggunaan label, legend dan judul grafik

A.8 Memberikan Teks Tambahan Pada Grafik

Teks tambahan kadang-kadang penting diberikan apabila grafik yang digambar lebih dari satu. Untuk memberikan teks tambahan ini dapat dilakukan dengan memberikan perintah *gtext()*. Sebagai contoh

```
clear; close all;
x=linspace(0,2*pi,200);
y1=cos(x);
y2=cos(x-0.5);
y3=cos(x-1.0);
```

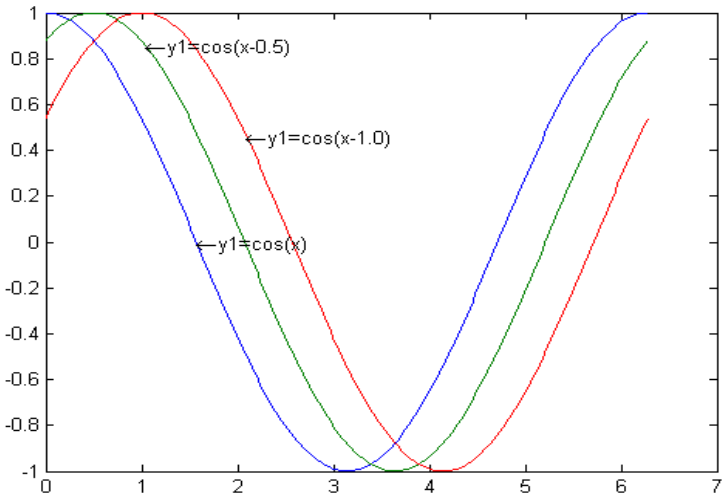
```
plot(x,y1,x,y2,x,y3);
gtext('y1=cos(x)');gtext('y1=cos(x-0.5)');
gtext('y1=cos(x-1.5)');
```



Gambar 4.12 Contoh penggunaan perintah gtext()

Untuk memberika teks tambahan dapat juga dilakukan dengan cara di bawah ini

```
clear; close all;
x=linspace(0,2*pi,200);
y1=cos(x);
y2=cos(x-0.5);
y3=cos(x-1.0);
plot(x,y1,x,y2,x,y3)
text(pi/2,cos(pi/2),'\leftarrowy1=cos(x)');
text(pi/3,cos(pi/3-0.5),'\leftarrowy1=cos(x-0.5)');
text(2*pi/3,cos(2*pi/3-1.0),'\leftarrowy1=cos(x-1.0)')
```



A.9 Penulisan Simbol, Huruf Yunani dan

Dibawah ini disajikan tabel tentang simbul dan cara penulisannya. Penulisan simbul dapat disertakan dalam label, title, text maupun legend.

Tabel 2. Simbul dan cara penulisannya

Cara penulisan	Simbul	Cara penulisan	simbul
\alpha		\upsilon	
\beta		\phi	
\gamma		\chi	
\delta		\psi	
\eta		\Gamma	
\zeta		\omega	
\iota		\Delta	
\theta		\Theta	
\vartheta		\Lambda	
\kappa		\Xi	
\lambda		\Pi	
\nu		\Sigma	
\mu		\Phi	

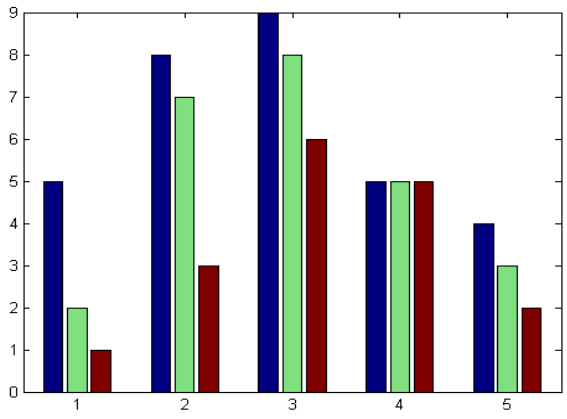
\xi		\Upsilon	
\phi		\Omega	
\rho		\Re	

A.10 Membuat Plot Khusus

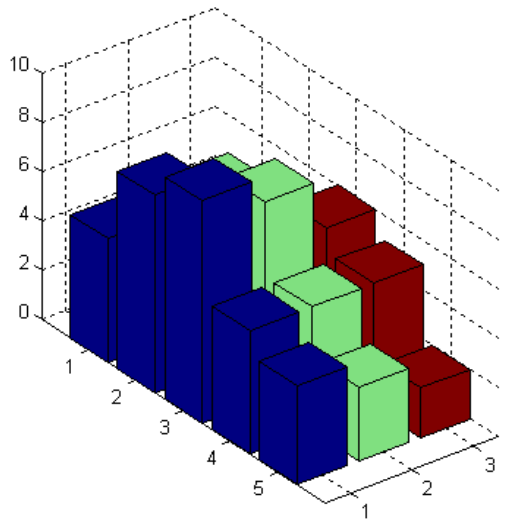
Grafik Berbentuk Bar

Matlab memiliki fungsi untuk menampilkan data dalam bentuk grafik bar. Ada dua jenis grafik bar yaitu bar horisontal dan bar vertikal. Fungsi-fungsi tersebut antara lain adalah bar, barh, bar3 dan bar3h. Contoh

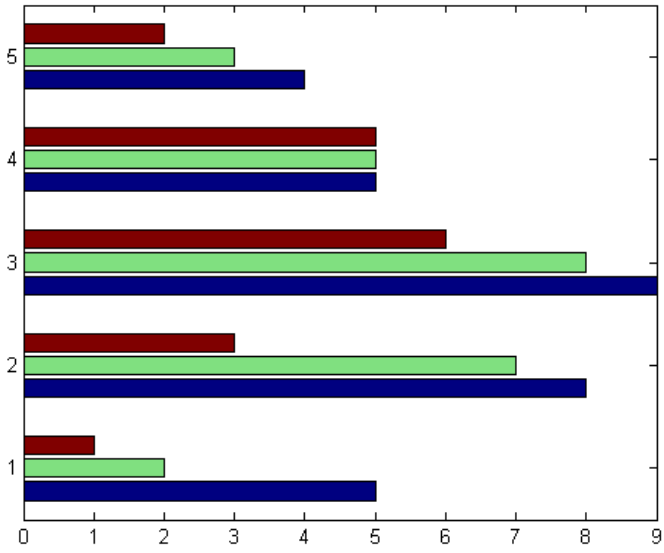
```
Y = [5 2 1
     8 7 3
     9 8 6
     5 5 5
     4 3 2];
bar(Y)
```



Gambar. Grafik berbentuk bar 2D



Gambar. Grafik berbentuk bar 3D



Gambar. Grafik berbentuk bar 2D

B.1 Membuat Plot 3 Dimensi

Tabel 4.3 dibawah ini disajikan langkah-langkah pembuatan grafik 3 dimensi dari fungsi tertentu yang memuat data grafik maupun model objek 3 dimensi.

Tabel 4.3 Langkah-langkah membuat grafik 3D

No.	Langkah-langkah	Kode
1.	Mempersiapkan data	`Z=peaks(20)`
2.	Pilih jendela dan posisi grafik dalam jendela gambar	`figure(1)` `subplot(2,1,2)`
3.	Panggil fungsi untuk menampilkan grafik	`h = surf(Z); h=mesh(Z); h=contour(Z)`

4.	Atur peta warna dan shading	`colormap hot`
		`shading interp`
		`set(h,'EdgeColor','k')`

5.	Tambahkan pencahayaan	`light('Position',[-2,2,20])`
		`lighting phong`
		`material([0.4,0.6,0.5,30])`
		`set(h,'FaceColor',[0.7 0.7 0],...`
		`'BackFaceLighting','lit')`

6.	Atur titi pandang grafik	`View([30,25])` `set(gca,'CameraViewAngleMode','Manual')`

7.	Atur batas sumbu dan *tick mark*	`Axis([5 15 5 15 -8 8])`
		`set(gca,'ZTickLabel','Negative|| ...`
		`Positive')`

8.	Atur aspek rasio	`set(gca,'PlotBoxAspectRatio',...` `[2.5 2.5 1])`

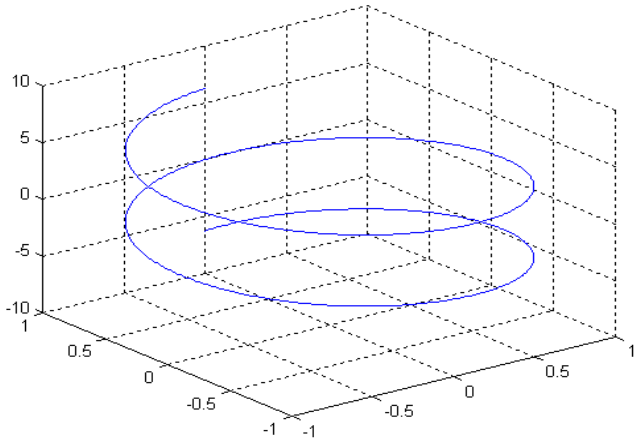
9.	Berikan tambahan keterangan pada sumbu dan judul grafik	`xlabel('X Axis')`
		`ylabel('Y Axis')`
		`zlabel('Function Value')`
		`title('Peaks')`

10.	Mencetak grafik	`set(gcf,'PaperPositionMode',...` `'auto')print -dps2`

B.2 Plot garis dari data 3D

Apabila kita memiliki dua buah fungsi dengan argumen sama, maka kita dapat menampilkan grafik 3D dari data tersebut. Demikian pula jika kita memiliki 3 buah vektor dengan *ukuran* sama, maka kita juga dapat menampilkannya menjadi grafik 3D. Contoh

```
t=-2*pi:pi/200:2*pi;
x=sin(t);
y=cos(t);
plot3(x,y,t);grid
```

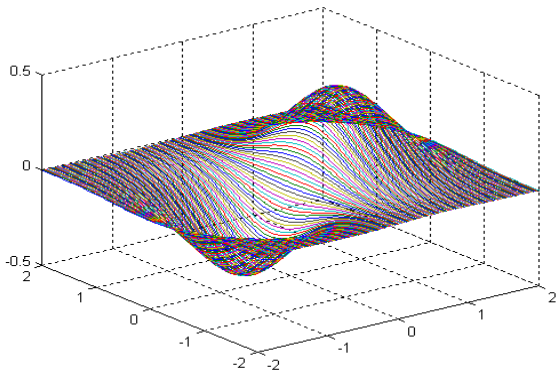


Gambar . Contoh grafik garis 3D

B.3 Plot data matriks

Apabila kita memiliki data yang sudah dibuat dalam bentuk matriks, maka kita dapat menampilkannya dalam bentuk grafik. Grafik yang terbuat merupakan plot setiap data dalam kolom matriks tersebut.

```
x=linspace(-2,2,200);
y=linspace(-2,2,200);
[X,Y] = meshgrid(x,y);
Z = X.*exp(-X.^2-Y.^2);
plot3(X,Y,Z)
grid on
```



Gambar . Contoh grafik 3D dari data matriks

Matlab juga dapat menampilkan beberapa jenis grafik 3 dimensi antara lain

No.	Fungsi	Keterangan
1.	mesh, surf	Plot grafik permukaan
2.	mesh, surfc	Plot permukaan dengan contour
3.	meshz	Plot permukaan dengan curtain plot
4.	pcolor	Plot permukaan flat
5.	surfl	
6.	surface	

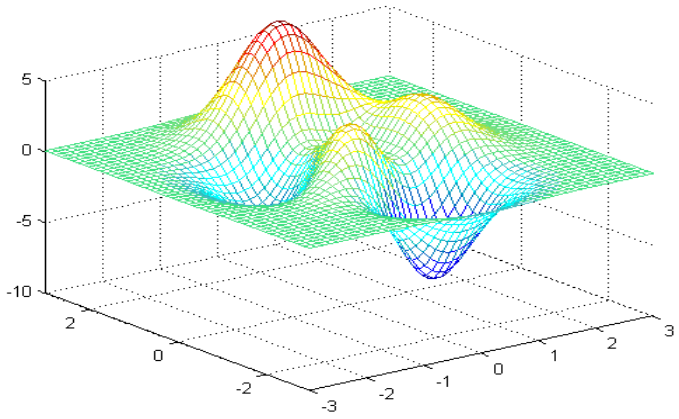
B.3 Grafik dengan mesh, surf dan pcolor

sintak untuk plot grafik dengan mesh adalah sebagai berikut,

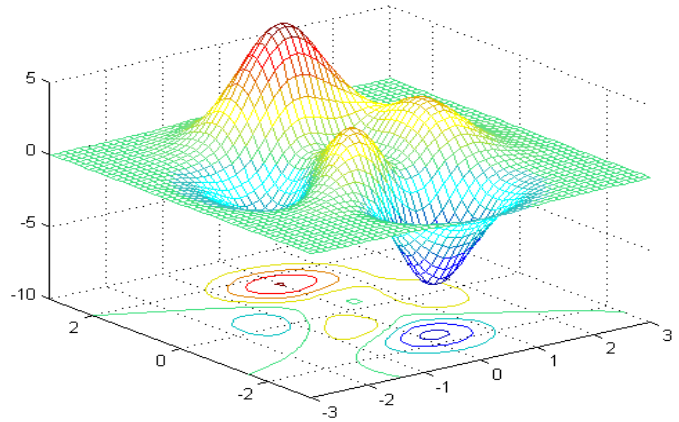
```
mesh(X,Y,Z)
mesh(Z)
mesh(...,C)
mesh(...,'PropertyName',PropertyValue,...)
meshc(...)
meshz(...)
h = mesh(...)
h = meshc(...)
h = meshz(...)
```

Contoh

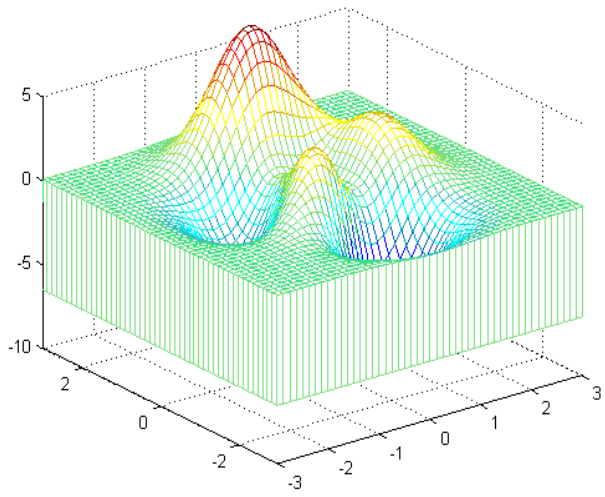
```
x=-3:.125:3;
y=-3:.125:3;
[X,Y] = meshgrid(x,y);
Z = peaks(X,Y);
mesh(X,Y,Z);
axis([-3 3 -3 3 -10 5])
```



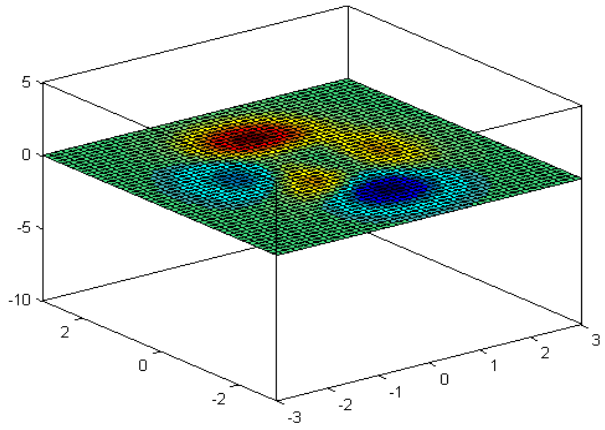
Gambar . Plot grafik 3D dengan fngsi mesh()



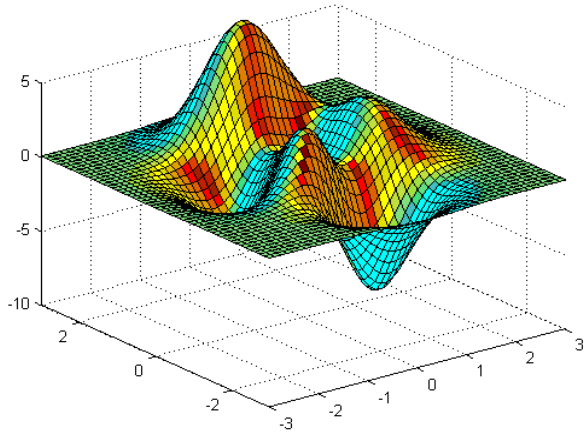
Gambar . Plot grafik 3D dengan fngsi meshc()



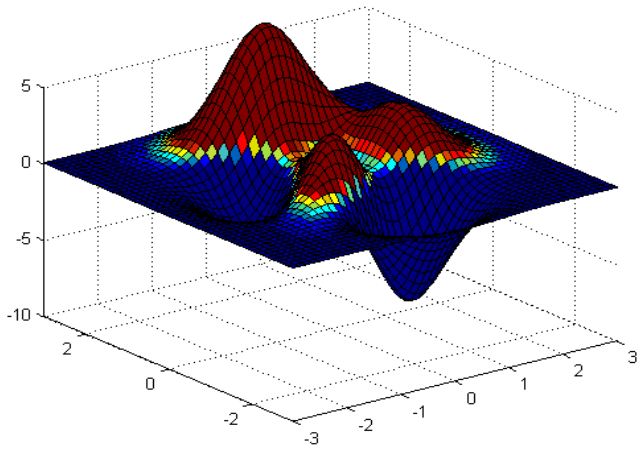
Gambar . Plot grafik 3D dengan fngsi meshz()



Gambar . Plot grafik 3D dengan fungsi pcolor()



Gambar . Plot grafik 3D dengan fngsi surfl()



Gambar . Plot grafik 3D dengan fungsi surface()

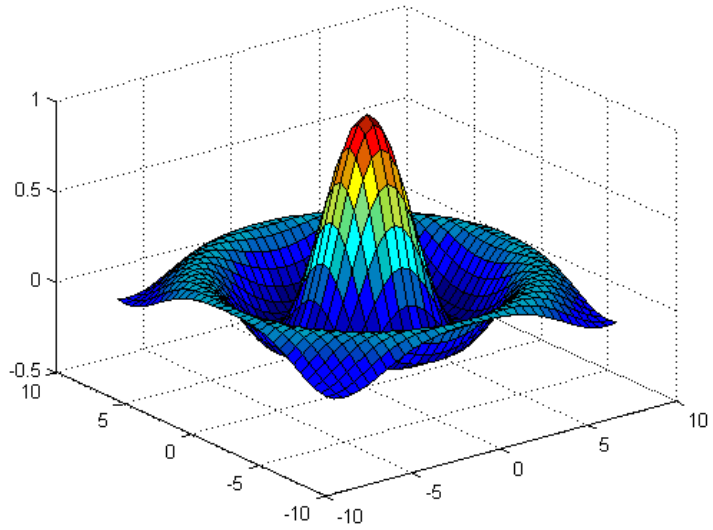
B.4 Memvisualkan fungsi dengan dua variabel

Apabila kita memiliki sebuah fungsi *f(x,y)* yakni fungsi dengan dua variabel bebas *x* dan *y*, dimana $a < x < b$ dan $a < y < b$, maka tidak serta merta dapat dibuat grafik fungsi f terhadap *x* dan *y*. Akan tetapi, ada satu langkah yang harus dilakukan yaitu membuat vektor *x* dan *y* tersebut menjadi matriks. Untuk membuat matriks vektor *x* dan *y* tersebut dapat dilakukan dengan fungsi *meshgrid.*

```
clear; close all;
x=-8:.5:8;
y=-8:.5:8;
[X,Y] = meshgrid(x,y);
R = sqrt(X.^2 + Y.^2) + eps;
Z=sin(R)./R;
surf(X,Y,Z);
grid on
```

Dibawah ini disajikan data berbentuk matriks berasal dari pengenaan fungsi *meshgrid* pada vektor *x* dan *y*.

```
x=1:3;
y=1:4;
[X,Y]=meshgrid(x,y);
X =
     1     2     3
     1     2     3
     1     2     3
     1     2     3
Y =
     1     1     1
     2     2     2
     3     3     3
     4     4     4
```

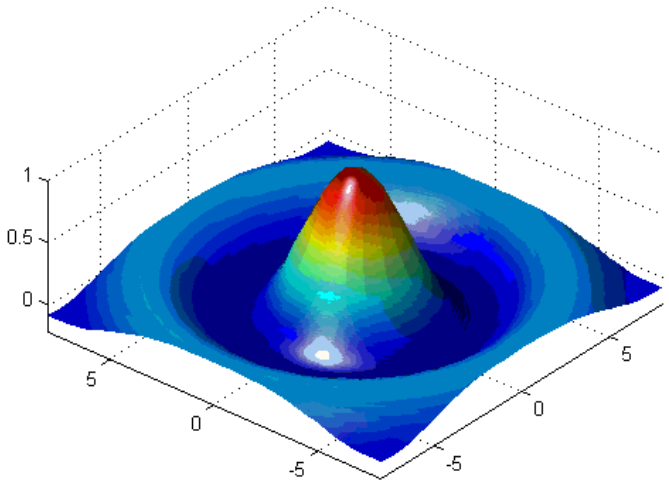


Gambar . Grafik 3D menggunakan fungsi *meshgrid* dari fungsi `Z=sin(R)./R`

Matlab menyediakan banyak teknik untuk untuk memperindah isi informasi dari grafik yang kita miliki. Di bawah ini diberikan contoh grafik dengan menggunakan pencahayaan dan titik pandang yang

sesuai menggunakan fungsi-fungsi `daspect, axis, camlight, view`

```
clear; close all;
x=-8:.5:8;
y=-8:.5:8;
[X,Y] = meshgrid(x,y);
R = sqrt(X.^2 + Y.^2) + eps;
Z=sin(R)./R;
surf(X,Y,Z,'FaceColor','interp',...
    'EdgeColor','none',...
    'FaceLighting','phong')
daspect([5 5 1])
axis tight
view(-50,30)
camlight left
```



Gambar . Grafik 3D menggunakan fungsi *meshgrid* dari fungsi `Z=sin(R)./R` dengan menambahkan teknik pencahayaan dan titik pandang yang sesuai

B.5 Plot permukaan untuk data nonuniform

Kita dapat menggunakan fungsi meshgrid untuk membuat

jaring-jaring dari titik-titik data sampel uniform. Selanjutnya, Matlab membangun plot permukaan dengan cara menghubungkan elemen-elemen matriks yang berdekatan untuk membuat sebuah mesh kuadrilateral. Untuk membuat plot permukaan dari data yang nonuniform, pertama kita harus menggunakanfungsi *griddata* untuk menginterpolasi nilai-nau surf seperti basanya.

Dibawah ini diberikan contoh menampilkan data nonuniform dengan plot permukaan. Contoh ini akan mengevaluasi fungsi sinc untuk data yang dibangkitkan dari fungsi random (acak) untuk range tertentu, dan kemudian membangkitkan data sample uniform untuk ditampilkan sebagai plot permukaan.

Proses untuk membuat tampilan data nonuniform tersebut meliputi tiga proses antara lain:

- Menggunakan linspace untuk mengenerate data pada range tertentu.
- Menggunakan fungsi meshgrid untuk mengenerate ploting grid dengan ouput dari linspace.
- Menggunakan fungsi griddata untuk menginterpolasi data yang beraturan menjadi data yang teratur.

BAB 5
PEMROGRAMAN DAN TIPE DATA

Matlab memberikan kemudahan di dalam penulisan pernyataan-pernyataan atau perintah-perintah. Pernyataan/perintah tersebut dapat dituliskan di dalam editor tertentu dan disimpan eksetensi **.m**. Dengan memberikan nama file dengan ekstensi tersebut, maka akan dapat deksekusi di dalam lingkungan Matlab. Juga, dengan ekstensi tersebut file yang kita buat bernama M-File.

M-File dapat berupa skrip maupun fungsi. M_File dalam bentuk skrip dapat dieksekusi secara mudah dengan menuliskan nama skrip tersebyut di command window, atau cukup menekan tombol F5 dengan catatan skrip sudah diberikan nama. M-File dalam bentuk fungsi terdiri atas argumen yang harus diberikan input sebelum dieksekusi dan menghasilkan ouput. Apabila M-File kita berupa fungsi, maka langkah eksekusi file adalah sebagai berikut.

• Membuat M-File dengan teks editor.	`function c = fungsiku(a,b)` `c = sqrt(a.^2 + b.^2);`

• Memanggil M-File melalui command window	`>> a=2.4;`
	`>> b= 3.1;`
	`>> c=fungsiku(a,b)`
	
	`c =`

	3.9205

Perbedaaan antara M-File dalam bentuk skrip dan fungsi antara lain

M-File dalambentuk skrip	M-File dalam bentuk fungsi
1. Tidak menerima input atau return argumen keluaran. 2. Menjalankan data yang berada di ruang kerja (workspace) 3. Sangat berguna apabila kita membutuhkan data sewaktu-waktu, karena data tersimpan dalam ruang kerja.	2. Menerima argumen input dan mereturn argumen output. 3. Tidak menjalankan data yang ada di ruang kerja, melainkan dari variabel internal yang secara default merupakan variabel lokal fungsi. 4. Sangat berguna apabila di lain waktu kita membutuhkan fungsi tersebut, karena fungsi tersimpan di current directory.

Dibawah ini diberikan contoh M-File dalam bentuk fungsi yang sangat berguna bagi pemrograman komputer.

```
function f = fact(n) % baris untuk definisi fungsi
% FACT Factorial.    % baris H1
% FACT(N) mereturn factorial dari N, H! % teks bantuan
% biasanya dilambangkan  N!
% FACT(N) hasilnya sama dengan PROD(1:N).
f = prod(1:n);       % body fungsi
```

Fungsi ini memeiliki beberapa bagian, sebagaimana daapt ditemukan di dalam fungsi-fungsi di Matlab.

- Baris definisi fungsi. Baris ini mendefinisikan nama fungsi dan argumen input dan output. Dalam contoh tersebut nama fungsinya adalah fac, argumen input adalah a dan b, serta

argumen output adalah c.

- Baris bantuan. H1 singkatan dari baris “help 1”. Matlab akan menampilkan baris H1 ketika kita mencarinya dengan perintah lookfor.
- Body fungsi. Bagian ini memuat code yang akan melakukan komputasi dan menyimpan harga untuk kepentingan setiap argumen output.

Skrip

Skrip merupakan bentuk sederhana dari M-File, karena tidak memiliki argumen input maupun output. Bentuk ini sangat berguna untuk otomatisasi serangkaian perintah-perintah Matlab, misalnya perhitungan yang berulang-ulang dengan data yang sama. Dengan menggunakan tombol down arrow atau up arrow, maka kita dapat menemukan kembali data atau perintah yang pernah kita eksekusi. Hal ini karena data atau perintah tadi terseimpan di dalam ruang kerja. Dibawah ini diberikan contoh skrip M-File sederhana

```
% Skrip M-file untuk menghasilkan   % baris komentar
% plot "flower petal"
theta = -pi:0.01:pi;                % perhitungan
rho(1,:) = 2*sin(5*theta).^2;
rho(2,:) = cos(10*theta).^3;
rho(3,:) = sin(theta).^2;
rho(4,:) = 5*cos(3.5*theta).^3;
for k = 1:4
    polar(theta,rho(k,:))           % keluaran grafik
    pause
end
```

Fungsi

Fungsi menerima argumen input dan menjalankan (return)

argumen output. Fungsi melakukan operasi dengan data yang ada di dalam workspace fungsi itu sendiri, bukan yang tersimpan di dalam workspace yang ada di desktop Matlab. Di bawah ini diberikan fungsi sederhana bernama *rerata* yang akan menghitung elemen dalam sebuah vektor.

```
function y = rerata(x)
% RERATA adalah fungsi untuk menghitung
% rerata dari elemen dari sebuah vektor
% RERATA(X), dimana X adalah vektor,
% merupakan rerata dari elemen sebuah vektor
% Jika inputan bukan berupa vektor maka
% akan error
[m,n] = size(x);
if (~((m == 1) | (n == 1)) | (m == 1 & n == 1))
    error('Inputan harus berupa vektor')
end
y = sum(x)/length(x);       % perhitungan
```

Untuk menjalankan fungsi ini, pertama kita harus memberi nama terlebih dahulu misalnya bernama rerata.m. Fungsi rerata hanya menerima inputan tunggal berupa vektor .

```
>> X=1:6;
>> rerata(X)
ans =
    3.5000
```

Argumen fungsi baik argumen input maupun output dapat lebih dari satu. Pada fungsi *rerata.m* di atas, argumen input hanya satu yaitu vektor *x*, sedangkan argumen output juga hanya satu yaitu *y*. Pertanyaannya, apakah argumen dapat lebih dari satu. Jawabnya

adalah bisa! Sebagai contoh

```
function [Luas,Keliling] = kotak(p,l);
%KOTAK adalah fungsi untuk menentukan
%luas dan keliling sebuah kotak
if (p<=0 | l<=0)
    fprintf(' Inputan salah p dan l harus >0');
end
Luas = p*l;
Keliling=2*p*l;
```

Jika dijalankan

```
>> p=3;l=5;
>> kotak(p,l)
ans =
    15
>> [L,K]=kotak(p,l)
L =
    15
K =
    30
```

Sub fungsi

Fungsi M-File dapat terdiri dari lebih dari satu fungsi. Fungsi pertama disebut sebagai fungsi utama, sedangkan fungsi berikutnya disebut sebagai subfungsi.

```
function [avg,med] = newstats(u) % fungsi utama
% NEWSTATS untuk memperoleh mean dan median.
n = length(u);
```

```
avg = mean(u,n);
med = median(u,n);

function a = mean(v,n)              % Subfungsi
% menghitung rerata.
a = sum(v)/n;

function m = median(v,n)            % Subfungsi
% menghitung median.
w = sort(v);
if rem(n,2) == 1
    m = w((n+1)/2);
else
    m = (w(n/2)+w(n/2+1))/2;
end
```

Operator Aritmatik

Matlab menyediakan operator aritmatika seperti terlihat pada tabel di bawah ini.

Operator	Deskripsi
+	Penambahan
-	Pengurangan
.*	Perkalian
./	Bagi kanan
.\	Bagi kiri
:	Operator kolon
.^	Pangkat
.'	Transpose
'	Transpose konjugat kompleks
*	Perkalian matriks

/	Bagi kanan matriks
\	Bagi kiri matriks
^	Pangkat matriks

Operator relasi

Matriks juga menyediakan operator relasi sebagai berikut

Operator	Deskripsi
<	Kurang dari
<=	Kurang dari atau sama dengan
>	Lebih dari
>=	Lebih dari atau sama dengan
==	Sama dengan
~=	Tidak sama dengan

Operator relasi berfungsi membandingkan elemen-elemen dari suatu lmatriks (larik) terhadap elemen-elemen seletak dari matriks (larik) lain. Contoh

```
>>P=[2,3,4;3,4,5;5,6,7];
>> Q=[3,2,4;2,4,5;5,1,2];
>> P==Q
ans =
     0     0     1
     0     1     1
     1     0     0
>> P<Q
ans =
     1     0     0
     0     0     0
     0     0     0
>> P<=Q
ans =
```

```
     1     0     1
     0     1     1
     1     0     0
>> P>Q
ans =
     0     1     0
     1     0     0
     0     1     1
>> P>=Q
ans =
     0     1     1
     1     1     1
     1     1     1
```

BAB 6
KONTROL ALIRAN

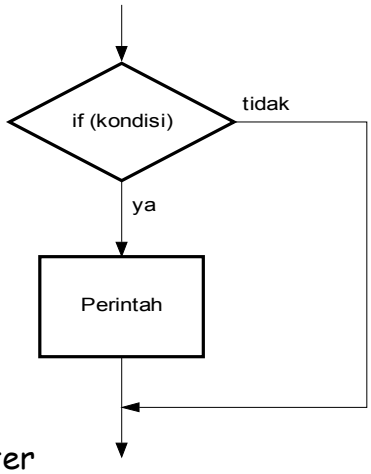
Ada delapan pernyataan kontrol kendali yang disediakan di dalam Matlab. Kedelapan pernyataan tersebut antara lain

1. **if**, termasuk di dalamnya pernyataan else dan elseif. Pernyataan ini menjalankan kelompok pernyataan berdasarkan pada syarat logika.
2. **switch**, termasuk di dalamnya adalah case dan otherwise. Statement ini mengeksekusi kelompok pernyataan berbeda bergantung pada harga syarat kondisi.
3. **while**, menjalankan group pernyataan dengan jumlah iterasi tak terbatas berdasarkan pada syarat logika.

4. **for** menjalankan group pernyataan dengan jumlah iterasi telah ditentukan.
5. **continue** melewatkan kendali ke iterasi berikutnya untuk loop for atau while.
6. **break** berfungsi menghentikan eksekusi looping for atau while.
7. **try...catch** mengubah kendali aliran apabila ditemukan kesalahan selama proses eksekusi.
8. **return** menyebabkan eksekusi kembali ke fungsi invoking. Semua aliran membangun pemnggunaan end untuk menunjukkan akhir dari blok kontrol aliran.

1. Pernyataan Bersyarat if, else, dan **elseif**

Pernyataan if digunakan untuk menyeleksi suatu kondisi yang memungkinkan dua atau lebih pilihan. Bila proses yang diseleksi terpenuhi atau bernilai benar, maka pernyataan yang ada di dalam blok if akan diproses dan dikerjakan. Jika digambarkan dalam diagram alir, maka percabangan *if* dapat digambarkan sebagaimana terlihat pada gambar 4.4



Gambar 4.4 Diagram alir percabangan *if*

Bentuk umum struktur kondisi if adalah :

pernyataan;

```
if (kondisi)
    pernyataan/perintah
end
```

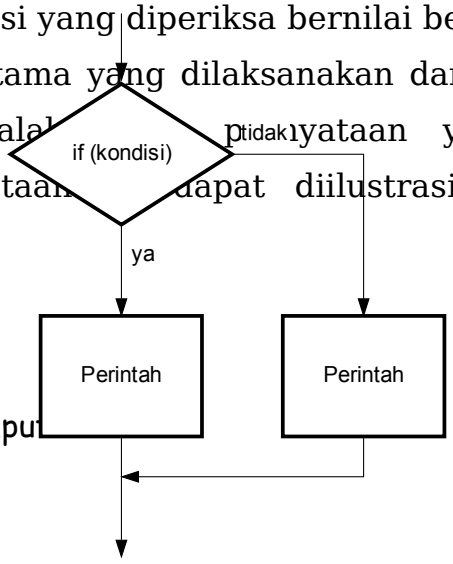
Apabila pernyataan logika berharga benar (true), maka seluruh pernyataan atau perintah yang berada diantara if dan end akan dijalankan. Sebaliknya, jika pernyataan logika berharga salah (false), maka seluruh perintah/pernyataan yang berada diantara if dan end tidak akan dijalankan karena Matlab akan langsung menuju end.

contoh:

```
if rem(a,2) == 0
   disp('a adalah bilangan genap')
   b = a/2;
end
```

Pernyataan Bersyarat if-else

Dalam pernyataan bersyarat **if-else** paling tidak terdapat dua pernyataan. Jika kondisi yang diperiksa bernilai benar atau terpenuhi maka pernyataan pertama yang dilaksanakan dan jika kondisi yang diperiksa bernilai salah maka pernyataan yang kedua yang dilaksanakan. Pernyataan ini dapat diilustrasikan seperti pada gambar 4.5.



Gambar 4.5 Diagram alir percabangan if…else

Bentuk umumnya adalah sebagai berikut :

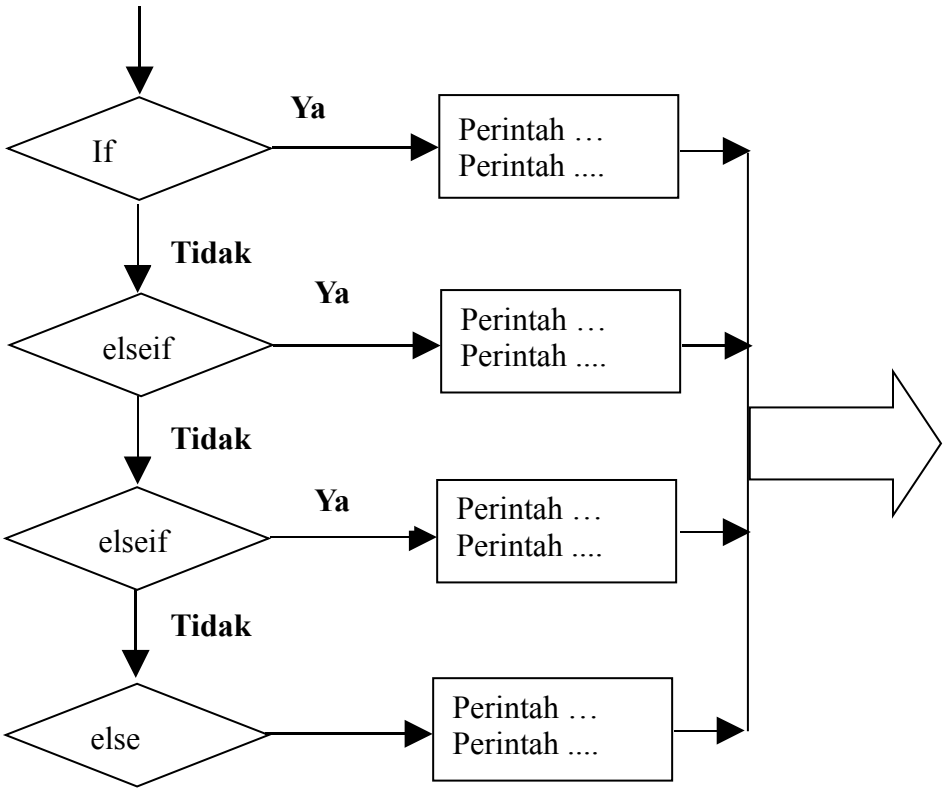
```
if(kondisi)
      perintah_1
else
      perintah_2
end
```

```
%Contoh program if-else
     x=input(Masukkan harga x:');
     if (x>3)
         disp('Pernyataan benar.');
         disp('Pernyataan benar sekali..');
     else
        disp('Pernyataan salah.');
         disp('Pernyataan salah sekali..');
     end
```

Pernyataan Bersyarat elseif

Pernyaan elseif akan dieksekusi apabila syarat yang dberikan pada if sebelumnya (atau elseif sebelumnya) berharga salah (0). Pernyaaan ini selanjutnya akan mengeksekusi perintah/pernyataan di

dalamnya apabila syarat logikanya berharga benar (1). Pernyataan ini dapat digambarkan dengan diagram dibawah ini



Gambar 6.6 Diagram alir pernyataan elseif

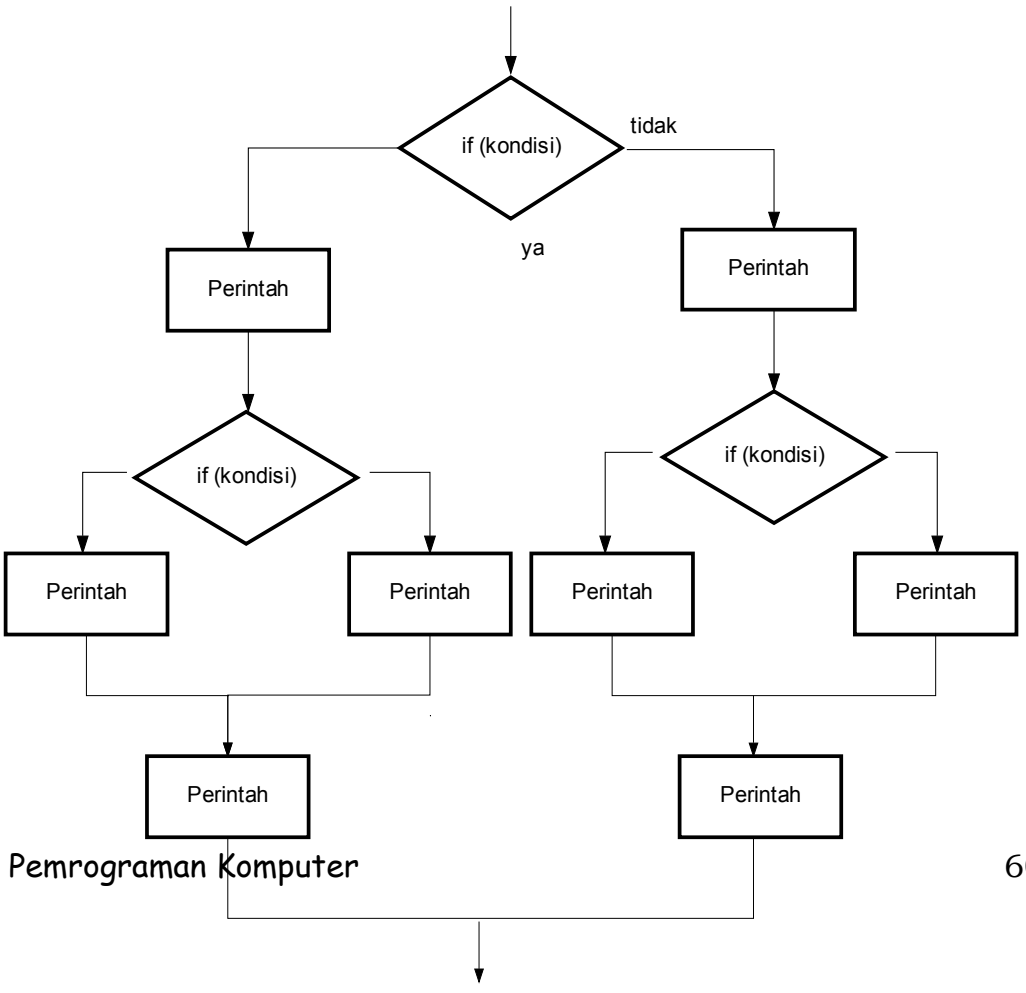
```
 %contoh penggunaan elseif
n=input('Masukkan harga n:')
if n < 0     % jika n negatif, ada pesan salah.
  disp('Masukan harus bilangan positif');
elseif rem(n,2) == 0 % jika n posisif dan bulat,
                     % kemudian bagi dengan 2
   A = n/2;
else
   A = (n+1)/2;     % jika n positif dan ganjil
                    % tambahkan 1,kemudian bagi
                    % dg 2.
end
```

Pernyataan bersyarat if bersarang

Pernyataan ini sangat penting untuk masalah-masalah yang memiliki lebih dari dua cabang. Mengapa pernyataan ini disebut

pernyataan bersarang, karena di dalam pernyataan if ada pernyataan if lagi. If yang kedua ini dapat berada di dalam if sendiri atau berada di dalam else. Untuk lebih jelasnya lihatlah diagram alir 6.5. Bentuk umum dari pernyataan if bersarang adalah

```
if (syarat/kondisi)
        perintah/pernyataan
        if (kondisi)
                perintah/pernyataan
        else
                perintah/pernyataan
        end
else
        perintah/pernyataan
        if (kondisi)
                perintah/pernyataan
        else
                perintah/pernyataan
        end
end
```



Gambar 6.5 Diagram Alir Pernyataan If Bersarang

switch

Pernyataan switch akan mengeksekusi sekelompok perintah/pernyataan berdasarkan pada harga variabel atau ekspresi. Bentuk dasar dasi switch adalah

```
switch ekspresi (skalar atau string)
   case nilai1
      pernyataan % dieksekusi jika ekspresi adalah
nilai1
   case nilai2
      pernyataan % dieksekusi jika ekspresi adalah
nilai2
   .
   .
   .
   otherwise
     pernyataan % dieksekusi jika ekspresi tidak cocok
                  % dengan case manapun
```

Dari bentuk umum penggunaan switch di atas, maka pernyataan switch terdiri atas beberapa blok antara lain:

- switch diikuti oleh ekspresi yang mana dapat berupa skalar (bilangan) maupun string.
- Sejumlah grup **case** yang masing-masing diikuti oleh nilai ekspresi. Nilai ekspresi ini berada dalam satu baris dengan case. Baris dibawahnya berisi pernyataan-pernyataan yang akan dijalankan apabila ekspresi nilai sesuai dengan ekspresi pada switch. Blok case akan berakhir saat ditemukan **blok otherwise.**
- Blok otherwise. Blok ini ada untuk mengantisipasi apabila nilai eksepresi pada switch tidak sama dengan nilai ekspresi pada case manapun. Pernyataan-pernyataan yang berada dalam blok ini akan berakhir saat bertemu dengan pernyataan **end.**
- Pernyataan end.

Pernyataan switch bekerja dengan cara membandingkan masukan (input) ekspresi dengan harga pada tiap-tiap peryataan case. Untuk ekspresi numerik, pernyataan case bernilai benar apabila (nilai=ekspresi). Untuk ekspresi string, pernyataan case bernilai benar apabila strcmp(nilai,ekspresi).

Dibawah ini diberikan sebuah contoh penggunaan pernyataan switch-case. Pernyataan switch akan mengecek variabel masukan untuk nilai tertentu. Apabila variabel masukan adalah 1, 2 atau 3 , maka pernyataan case akan menampilkan luas masing-masing untuk kubus, persegi panjang dan lingkaran. Sedangkan, apabila variabel masukan bukan 1,2 atau 3, maka switch akan langsung menuju ke otherwise.

```
% contoh menghitung luas bangun
fprintf('Pilih salah satu program..\n');
fprintf('1:  menghitung luas bujur sangkar: ');
fprintf('2:  menghitung luas persegi panjang: ');
```

```
fprintf('3:  menghitung luas lingkaran: ');

no=input('Anda ingin menghitung luas 1,2 atau 3 \n');

switch no

  case 1
   a=input('masukkan panjang sisi bujur sangkar:');
   luas=a*a;
    fprintf('Luas bujur sangkar adalah  %f \n',luas);

  case 2
    a=input('masukkan panjang persegi panjang: ');
    b=input('masukkan lebar persegi panjang: ');
    luas=a*b;
    fprintf('Luas persegi panjang adalah  %f',luas);
  case 3
    a=input('masukkan jari-jari lingkaran :');
    luas=3.14*pow(r,2);
    fprintf('Luas lingkaran adalah  %f ',luas);
  otherwise
        fprintf('Ulangi pilihan yang ada...!!')
end
```

Pernyataan switch juga dapat digunakan untuk menghandle multi syarat dalam satu case.

```
% contoh menghitung luas bangun
fprintf('Pilih salah satu program..\n');
fprintf('1:  menghitung luas bujur sangkar: ');
fprintf('2:  menghitung luas persegi panjang: ');
fprintf('3:  menghitung luas persegi panjang: ');
fprintf('4:  menghitung luas persegi panjang: ');
no=input('Anda ingin menghitung luas 1,2 atau 3 \n');
switch no
   case 1
   a=input('masukkan panjang sisi bujur sangkar:');
   luas=a*a;
    fprintf('Luas bujur sangkar adalah  %f \n',luas);
 case {2,3,4}
    a=input('masukkan panjang persegi panjang: ');
```

```
    b=input('masukkan lebar persegi panjang: ');
    luas=a*b;
    fprintf('Luas persegi panjang adalah  %f',luas);
 otherwise
        fprintf('Ulangi pilihan yang ada...!!')
end
```

4. while

Pernyataan while akan menjalankan pernyataan atau sekelompok pernyataan secara berulang-ulang selama kontrol perulangan berharga benar (1). Secara umum, bentuk dari switch adalah

```
while expression
        statements
 end
```

Program di bawah ini akan mencari n bilangan pertama perjumlahan bilangan interger yang berharga <100.

```
% contoh mencari n bilangan pertama pernjumlahan
jum=100;
n = 1;
while sum(1:n) < jum
  disp(n)
   n = n + 1;
end
```

```
%Program mencetak deret bilangan
i=1;
while(i<=6)
x=1;
   while(x<=i)
      fprintf('%i',x);
      x=x+1;
   end
fprintf('\n');
i=i+1;
```

```
end
```

5. Perulangan for

Struktur perulangan for biasa digunakan untuk mengulang suatu proses yang telah diketahui jumlah perulangannya. Dari segi penulisannya, struktur perulangan for tampaknya lebih efisien karena susunannya lebih simpel dan sederhana. Secara default, penambahan indeks adalah 1, tetapi kita dapat mengeset sendiri misalnya penambahannya 2, 3 atau berapapun bahkan dapat pula negatif. Untuk penambahan positif, perulangan akan berakhir manakala indeks sudah lebih dari yang tentukan. Sedangkan untuk penambahan negatif, perulangan berakhir saat indeks lebih kecil dari yang ditentukan. Secara umum, sintaks for adalah sebagai berikut

```
for indeks = mulai:penambahan:akhir
        pernyataan/perintah
end
```

Contoh program di bawah ini akan mencetak sebuah kalimat berbunyi “Aku mahasiswa semester 5” sebanyak lima kali dengan mengunakan pernyataan perulangan for.

```
%contoh perulangan for
N=5;
for i=1:N
   fprintf('%i. Aku mahasiswa semester 5 \n',i);
end
```

6. **Pernyataan continue**

Pernyataan continue berfungsi melewatkan kendali ke iterasi

berikutnya di dalam perulangan for maupun while. Apabila pernyataan continue berada di dalam loop bersarang, maka pernyataan continue akan melewatkan kendali ke iterasi berikutnya. Sebagai contoh, perhatikan di bawah ini

```
N=20;
for i=1:N
   if (i<5)
      continue;
   end
   fprintf('%i. Aku mahasiswa semester 5 \n',i);
end
```

Apabila skrip program di atas dieksekusi, maka yang keluar adalah tampilan tulisan “Aku mahasiswa semester 5” sebanyak 16 buah mulai dari nomor 5 hingga nomor 20. Dengan kata lain, kalimat dari nomot 1 hingga 4 tidak ditampilkan. Hal ini disebabkan perintah continue yang berada di dalam loop for.

8. **Pernyataan break**

Pernyataan break berfungsi untuk menghentikan eksekusi loop for maupun loop while. Apabila break ditemukan di dalam kalang for maupun while, maka eksekusi akan dihentikan apabila syarat yang diberikan terpenuhi. Untuk lebih jelasnya perhatikan contoh berikut ini

```
N=20;
for i=1:N
   if (i>5)
      break;
   end
   fprintf('%i. Aku mahasiswa semester 5 \n',i);
end
```

Program di atas akan ditampilkan lima buah kalimat “Aku

mahasiswa semester 5” mulai dari nomor 1 hingga 5. Setelah iterasi ke 5, program akan dihentikan karena pernyataan break di dalam pernyataan bersyarat dimana syaratnya telah terpenuhi. Dengan kata lain, kalimat pada nomor 6 hingga nomor 20 tidak akan dicetak 9.

Persamaan Linier Simultan

Salah satu masalah di dalam sains dan teknik adalah masalah penyelesaian persamaan linier simultan. Misalkan kita memiliki 3 buah persamaan simultan berbentuk

$$a_{11}x_1+a_{12}x_2+a_{13}x_3=b_1$$
$$a_{21}x_1+a_{22}x_2+a_{23}x_3=b_2$$
$$a_{31}x_1+a_{32}x_2+a_{33}x_3=b_3$$

Tiga persamaan linier tersebut dapat kita nyatakan dalam wujud matriks berbentuk

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Selanjutnya, berapakah harga-harga $x_1, x_2 \text{ dan } x_3$. Dengan Matlab kita dapat dengan mudah memperolehnya. Dimisalkan

$$A=\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix},\ X=\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \text{ dan } b=\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Untuk memperoleh harga $x_1, x_2 \text{ dan } x_3$ digunakan perintah **X=A\b.**

```
A=[1,-3,4;2,3,-4;1,2,3];
b=[3;10;1];
X=A\b
X =
   4.3333
  -0.5490
  -0.7451
```

Matriks A di atas termasuk ke dalam matriks non singular,

karena $det(A) \neq 0$, sehingga kita dapat memperoleh harga $x_1, x_2 \text{dan } x_3$. Apabila $det(A) = 0$, maka penyelesaian untuk persamaan tersebut tidak ada atau tidak unik.