UMSIDA PRESS
Jl. Mojopahit 666 B Sidoarjo

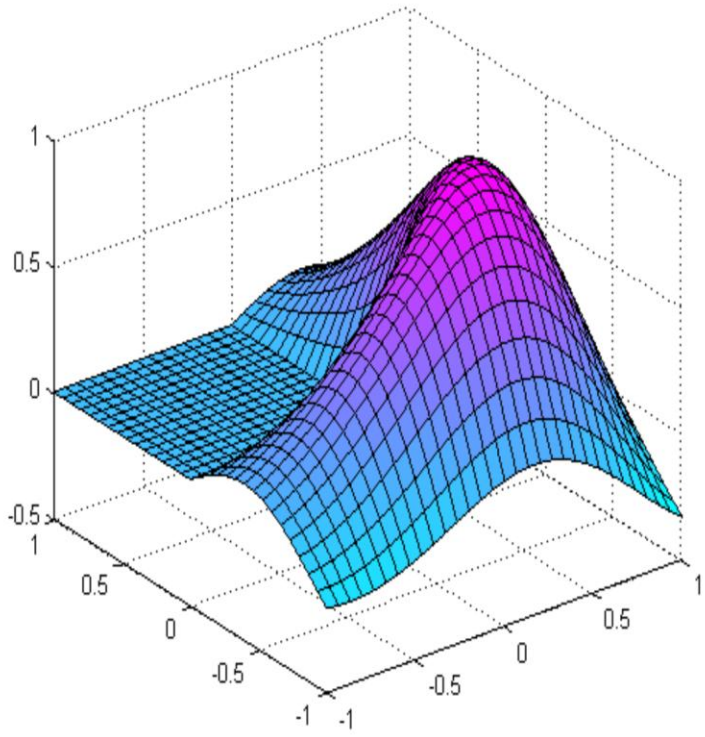
9 789793 401591
ISBN : 978-979-3401-59-1

PANDUAN APLIKASI MATLAB TEKNIK MESIN

EDI WIDODO, ST., MT
ROHMAN DIJAYA, M.Kom

# PANDUAN APLIKASI MATLAB TEKNIK MESIN

## PANDUAN PRAKTIS BELAJAR MATLAB




UNIVERSITAS MUHAMMADIYAH SIDOARJO 2017

# PANDUAN
# APLIKASI MATLAB TEKNIK MESIN

**Penulis**

**Edi Widodo, ST, MT**

**Rohman Dijaya, S.Kom, M.Kom**

Diterbitkan oleh

**UMSIDA PRESS**

Jl. Mojopahit 666 B Sidoarjo

ISBN: 978-979-3401-59-1

**PANDUAN**
**APLIKASI MATLAB TEKNIK MESIN**

**Penulis :**
Edi Widodo, ST, MT
Rohman Dijaya, S.Kom, M.Kom

**ISBN :**
978-979-3401-59-1

**Editor :**
Septi Budi Sartika, M.Pd

M. Tanzil Multazam , S.H., M.Kn.

**Copy Editor :**
Fika Megawati, S.Pd., M.Pd.

**Design Sampul dan Tata Letak :**
Mochamad Nashrullah, S.Pd

**Penerbit :**
UMSIDA Press

**Redaksi :**
Universitas Muhammadiyah Sidoarjo
Jl. Mojopahit No 666B
Sidoarjo, Jawa TImur

**Cetakan pertama, Agustus 2017**

# KATA PENGANTAR

Alhamdulillah, puji syukur kehadirat Allah SWT, atas limpahan rahmat dan hidayahNya, penulis dapat memulai menyusun panduan aplikasi Matlab khususnya untuk aplikasi teknik mesin. Mata kuliah algoritma pemrograman untuk teknik mesin memang memiliki porsi yang sedikit dalam struktur kurikulum. Hal ini disebabkan salah satunya adalah begitu luasnya kajian dalam bidang teknik mesin, atau lebih dikenal dengan istilah mechanical engineering. Namun demikian, kehadiran mata kuliah algoritma pemrograman, tetap diperlukan, mengingat untuk dapat menyelesaikan permasalahan teknik dalam mekanika, diperlukan suatu piranti yang mumpuni dan akurat. Kehadiran komputer sebagai alat hitung tidak terpisahkan dalam perkembangan pengetahuan dan teknologi. Pemrograman Matlab, atau dikenal dengan Matrik Laboratory, sudah dikenal memiliki kemampuan dalam perhitungan matematik yang tidak diragukan dan luas dipergunakan dalam penelitian maupun industri. Untuk dapat menjembatani mahasiswa dalam penguasaan program Matlab untuk aplikasi dalam teknik mesin, dibutuhkan metoda dan panduan yang memudahkan untuk diaplikasikan dalam menyelesaikan masalah dan melakukan perhitungan. Kehadiran modul ajar ini diharapkan dapat memberikan panduan sederhana dan mudah dipahami serta mudah diaplikasikan dengan konsep-konsep dasar penyelesaian persamaan matematis. Modul ini memberikan peletakan dasar pemahaman dalam penguasaan program matlab. Untuk dapat menguasai lebih jauh dalam aplikasi program untuk masalah yang lebih kompleks, dibutuhkan kajian yang lebih mendalam dengan referesensi yang spesifik

dan relevan. Pada akhirnya penulis berharap modul yang akan disusun ini mampu memberikan pengantar dasar penguasaan dan pemahaman bagi mahasiswa, terutama mereka yang memulai dari nol untuk pembelajaran algoritma dan pemrograman.

Sidoarjo, 30 September 2017 Penulis

Edi Widodo

# DAFTAR ISI

# Bab 1 Pengenalan Matlab

Tujuan instruksional umum

Mahasiswa dapat memahami dan mengenal karakter pemrograman Matlab

Tujuan instruksional khusus

Setelah mempelajari bab 1, diharapkan mahasiswa dapat mencapai kompetensi berikut:

1. Memahami cakupan algoritma pemrograman, meliputi pengertian, contoh aplikasi, tujuan dari mempelajari algoritma dan memberi contoh sederhana dari sebuah algoritma
2. Mampu menjelaskan Pemrograman Matlab sebagai sebuah aplikasi untuk membuat algoritma.
3. Mampu melakukan instalasi salah satu bahasa pemrograman
4. Mampu mengoperasikan matlab.
5. Mampu mengaplikasikan bahasa matlab untuk menyelesaikan permasalahan dalam bidang ilmu teknik mesin

## 1.1. Sepintas tentang Matlab

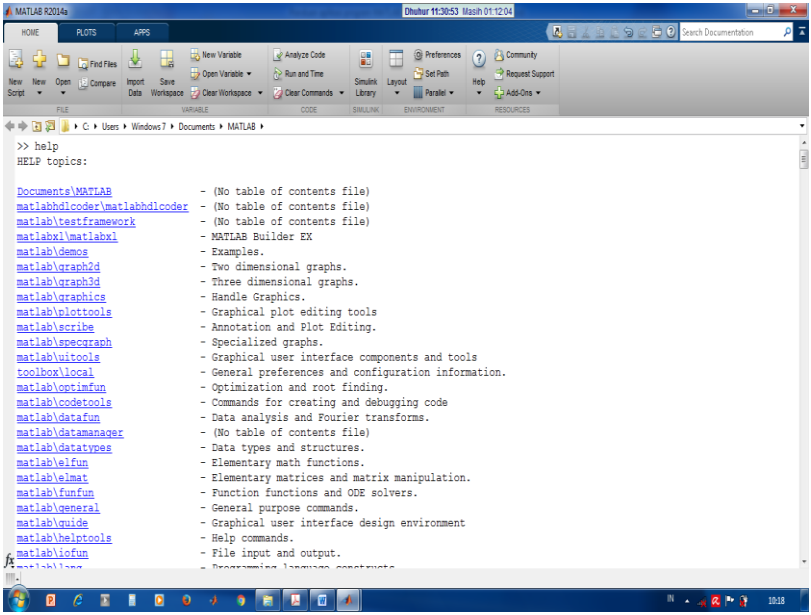
Matlab adalah software komputasi teknik yang bersifat komersial dibawah naungan Mathworks, Inc., USA. Merupakan bahasa pemrograman yang terintegrasi meliputi grafik antar muka dan memiliki toolbox khusus untuk untuk aplikasi spesifik dengan cakupan yang sangat luas. Meliputi semua aplikasi perhitungan

matematik, fisika, kimia , statistik, bahkan perhitungan keuangan dapat difasilitasi oleh Matlab. Hal ini membuat Matlab menjadi sangat populer dipakai baik oleh kaum akademisi maupun dalam aplikasi bidang rekayasa.

Sesuai dengan namanya, Matlab merupakan singkatan dari MATrik LABoratory, artinya semua proses algoritma pada dasarnya merupakan operasi matrik, walaupun pada perkembangannya seolah –olah tidak terlihat harus selalu dalam bentuk penulisan matrik. Program ini memiliki kemampuan komputasi yang sudah diakui baik di kalangan pendidikan, penelitian, maupun dunia industri. Kemampuan Matlab dalam memvisualisasikan persamaan / data-data penelitian menjadi salah satu alasan utama mengapa program ini merupakan pilihan terbaik dibanding dengan program-program komputasi sejenis.

## 1.2. Instalasi Matlab

Untuk melakukan instalasi diperlukan CD instaler yang siap untuk di-run di program windows atau PC (personal computer). Untuk instalasi versi terbaru dari Matlab dibutuhkan spesifikasi komputer yang cukup agar dapat menampilkan interface secara sempurna. Spesifikasi minimal yang diperlukan dapat dicek pada web resmi dari matlab, yaitu www.Mathworks.com.




Gambar 1.1. Tampilan muka matlab

## 1.3. Memulai Matlab

Setelah komputer/laptop terinstall maka dapat dimulai eksekusi program. Yang pertama muncul dari dekstop adalah tampilan command windows. Dalam command window semua perintah dapat diketikkan untuk didefiniskan sebagai sebuah command yang akan langsung dieksekusi oleh matlab. Sebagai contoh, kita dapat mengetikkan help untuk mendapatkan menu help dalam matlab :

>> help

Maka akan ditampilkan help topics

>> help

HELP topics:

| | |
|---|---|
| Documents/MATLAB | - (No table of contents file) |
| matlab/general | - General purpose commands. |
| matlab/ops | - Operators and special characters. |
| matlab/lang | - Programming language constructs. |
| matlab/elmat | - Elementary matrices and matrix manipulation. |
| matlab/elfun | - Elementary math functions. |
| matlab/specfun | - Specialized math functions. |

....

Dengan menu help ini, kita mendapatkan informasi yang menyeluruh semua hal tentang Matlab. Mulai dari cara instalasi, fungsi-fungsi dalam matlab, contoh program sampai dengan demo-demo yang sangat atraktif dari program/algoritma yang dapat dieksekusi dengan Matlab.

Misal kita ingin mencari fungsi-fungsi persamaan dasar trigonometri, eksponensial, bilangan komplek dan lain-lain, kita dapat memilih perintah help elfun

>> help elfun

Kemudian setelah dienter maka akan ditampilkan fungsi-fungsi yang kita cari

Elementary math functions.

Trigonometric.

| | |
|---|---|
| sin | - Sine. |
| sind | - Sine of argument in degrees. |
| sinh | - Hyperbolic sine. |
| asin | - Inverse sine. |
| asind | - Inverse sine, result in degrees. |
| asinh | - Inverse hyperbolic sine. |
| cos | - Cosine. |
| cosd | - Cosine of argument in degrees. |
| cosh | - Hyperbolic cosine. |
| acos | - Inverse cosine. |
| acosd | - Inverse cosine, result in degrees. |
| acosh | - Inverse hyperbolic cosine. |
| tan | - Tangent. |
| tand | - Tangent of argument in degrees. |
| tanh | - Hyperbolic tangent. |
| atan | - Inverse tangent. |
| atand | - Inverse tangent, result in degrees. |

atan2 - Four quadrant inverse tangent.

atan2d - Four quadrant inverse tangent, result in degrees.

atanh - Inverse hyperbolic tangent.

sec - Secant.

secd - Secant of argument in degrees.

sech - Hyperbolic secant.

asec - Inverse secant.

asecd - Inverse secant, result in degrees.

asech - Inverse hyperbolic secant.

csc - Cosecant.

cscd - Cosecant of argument in degrees.

csch - Hyperbolic cosecant.

acsc - Inverse cosecant.

acscd - Inverse cosecant, result in degrees.

acsch - Inverse hyperbolic cosecant.

cot - Cotangent.

cotd - Cotangent of argument in degrees.

coth - Hyperbolic cotangent.

acot - Inverse cotangent.

acotd - Inverse cotangent, result in degrees.

acoth - Inverse hyperbolic cotangent.

hypot - Square root of sum of squares.

Exponential.

exp - Exponential.

expm1 - Compute exp(x)-1 accurately.

log - Natural logarithm.

log1p - Compute log(1+x) accurately.

log10 - Common (base 10) logarithm.

log2 - Base 2 logarithm and dissect floating point number.

pow2 - Base 2 power and scale floating point number.

realpow - Power that will error out on complex result.

reallog - Natural logarithm of real number.

realsqrt - Square root of number greater than or equal to zero.

sqrt - Square root.

nthroot - Real n-th root of real numbers.

nextpow2 - Next higher power of 2.

Complex.

abs - Absolute value.

angle - Phase angle.

complex - Construct complex data from real and imaginary parts.

conj - Complex conjugate.

imag - Complex imaginary part.

real - Complex real part.

unwrap - Unwrap phase angle.

isreal - True for real array.

cplxpair - Sort numbers into complex conjugate pairs.

Rounding and remainder.

fix - Round towards zero.

floor - Round towards minus infinity.

ceil - Round towards plus infinity.

round - Round towards nearest integer.

mod - Modulus (signed remainder after division).

rem - Remainder after division.

sign - Signum.

Kemudian kita dapat mengetik clc untuk membersihkan kembali layar pada command windows

>> clc

Dengan menekan enter maka layar akan bersih kembali

Kita dapat mengetikkan date untuk mendapatkan tanggal secara real time

>> date

Setelah kita enter maka hasilnya

>> date

ans =

27-Mar-2017

Setiap instruksi atau command yang diketikkan, setelah ditekan perintah eksekusi (enter ) maka selalu dimulai dengan script ans. Ans artinya adalah jawaban dari setiap intruksi yang diberikan untuk dijalankan dalam command windows.

## 1.4. Menampilkan demo

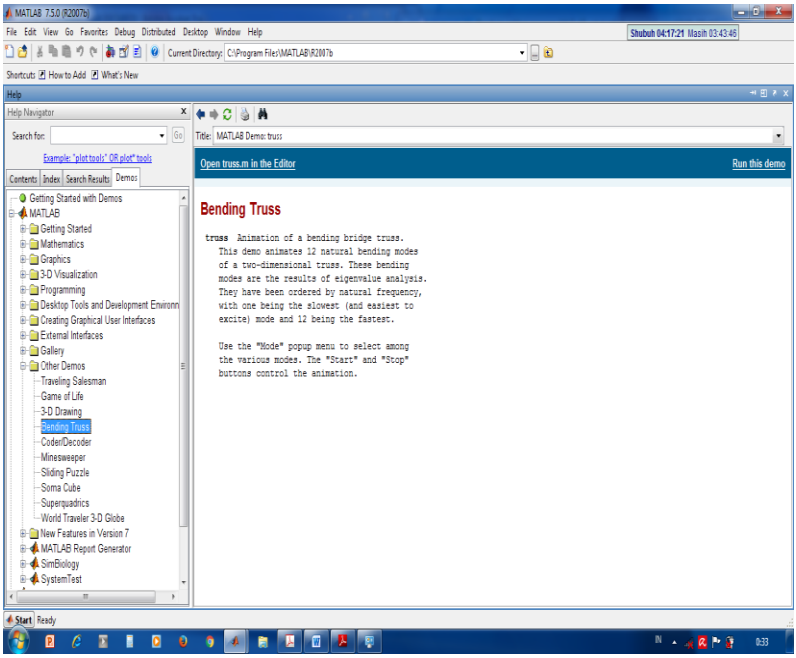
Untuk melihat hasil-hasil eksekusi yang dapat dilakukan dengan menggunakan matlab, kita dapat melihatnya pada menu demo. Atau dapat juga kita ketikkan “demo” pada command windows. Dari demo ini, matlab telah menyediakan contoh-contoh aplikasi program yang luas dari penggunaan matlab. Serta memberikan

gambaran bagaimana matlab mampu memberikan solusi yang kompleks dari hampir semua disiplin ilmu teknik rekayasa. Mulai dari aplikasi perhitungan, simulasi serta grafik-grafik menarik dan atraktif yang dapat dihasilkan dengan matlab.

>> demo

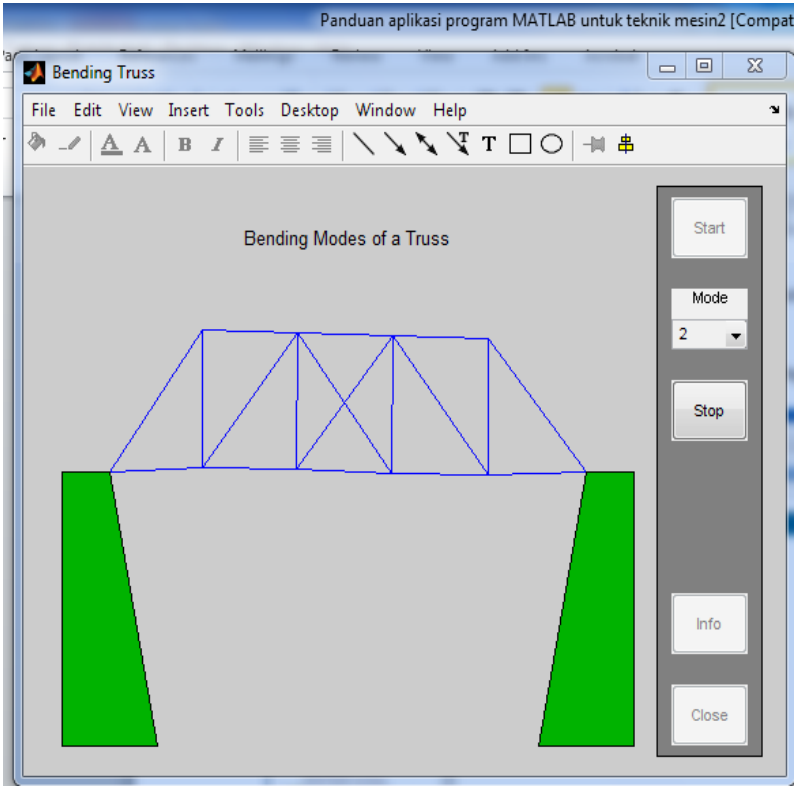
Dari demo tersebut kita dapat memilih atau melihat hasil eksekusi dari program-program yang telah dibuat.

Misalkan kita ingin melihat demo bending truss maka akan ditampilkan sebagai berikut




Gambar.1.2 Tampilan Demo

Dengan memilih run demo maka akan kita lihat demo dari bending truss




Gambar 1.3.Demo Truss struktur

# Bab 2 Operasi Dasar

Tujuan instruksional umum

Mahasiswa mampu memahami perintah dasar mengoperasikan Matlab

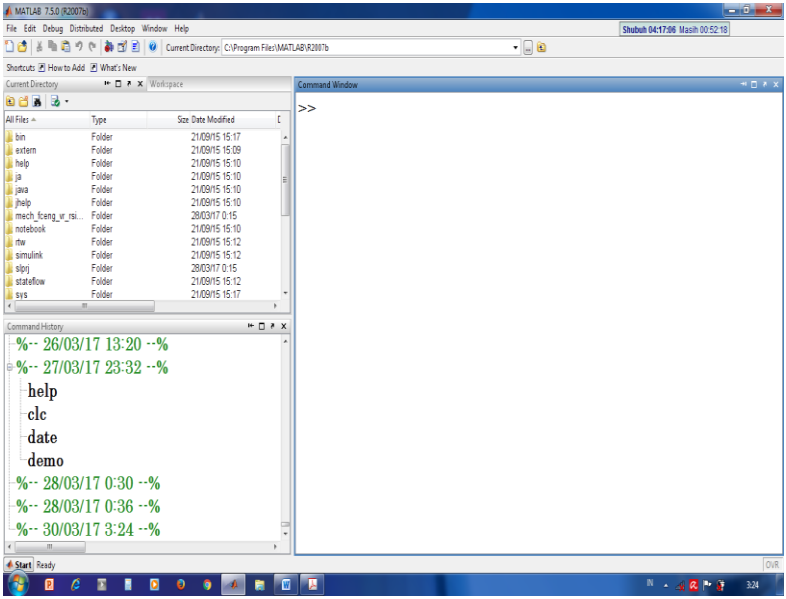
Tujuan instruksional khusus

Setelah mempelajari bab 2 Operasi dasar pemrograman matlab, mahasiswa mampu mencapai kompetensi berikut:

1. Mahasiswa mampu menggunakan command windows Matlab
2. Mahasiswa mampu mengoperasikan matlab sebagai kalkulator sederhanan untuk menyelesaikan persamaan matematika,
3. Mahasiswa mampu menciptkan variabel untuk menyelesaikan operasi perhitungan
4. Mahasiswa mampu menyelesaikan fungsi matematika

## 2.1. Command windows

Commands window pada matlab berfungsi untuk memberikan perintah/instruksi untuk dieksekusi oleh program. Semua fungsi dapat dioperasikan melalui command windows. Pada tampilan muka ketika Matlab dijalankan, commands window menjadi jendela yang akan tampil dominan untuk dapat langsung dilajalankan.




Gambar 2.1 Tampilan dekstop

Dalam tampilan standar/ default, akan ditampilkan menu-menu standar, jendela current directory,workspace, command history dan command windows, serta icon-icon penting lainnya. Command history akan memberikan list command/instruksi yang pernah atau barusan dijalankan sejak terakhir dibuka sampai dengan bahkan beberapa kali terakhir program dijalankan. Current directory memberikan informasi directory-directory yang pernah diaktifkan , serta command windows untuk instruksi / program yang akan dijalankan.

## 2.2. Perhitungan matematika sederhana

Secara sederhana matlab merupakan alat bantu hitung, yang digunakan untuk memberikan support dalam menyelesaikan problem-problem kalkulasi secara matematik. Dengan masalah matematika mulai dari operasi dasar meliputi penjumlahan, pengurangan, perkalian, dan pembagian sampai dengan penyelesaian persamaan komplek yang rumit, matlab hadir untuk memberikan solusi mudah dan cepat untuk membantu memecahkan kalkulasi. Adapun operator aritmatika yang dipergunakan, sesuai dengan perintah dasar yang umum dipakai:

Penambahan : +

Pengurangan : -

Perkalian : *

Pembagian : /

Pangkat : ^

Pembagian terbalik : \

Dengan hierarki operasi aritmatik sebagaimana umumnya :

a. Operasi di dalam tanda kurung " (,)" akan diselesaikan lebih dahulu

b. Operasi pangkat

c. Operasi perkalian dan pembagian

d. Operasi penjumlahan dan pengurangan

Berikut contoh operasi dasar aritmatika

```
>> 2 + 4 - 5

ans =

    1

>> 3 * 4 /(3 - 6)^2

ans =

  1.3333
```

Ans (answer) merupakan jawaban dari instruksi yang diberikan. Setiap kali kita mengeksekusi sebuah statement (instruksi) maka jawaban atau hasil dari command yang diketikkan akan didahului dengan ans, sebagai penanda hasil dari instruksi yang diberikan.

```
>> 3 * 4 /(3 - 6)^2 , 17 / 3 - (4 + 3) * (49)^0.5

ans =

   1.3333

ans =

 -43.3333
```

Pada contoh diatas, matlab bisa menyelesaikan dua persamaan sekaligus tanpa harus menunggu hasil instruksi yang diberikan. Dengan menambahkan operator koma (colon) maka operasi beberapa persamaan akan diselesaikan secara bersamaan.

## 2.3. Membuat variabel

Di dalam program ini, kita bisa membuat variabel yang tersimpan dan dapat dipanggil atau dieksekusi kapan saja, asalkan variabel tersebut telah disimpan dan tidak dihapus.

Membuat variabel

```
>> a =  20

a =   20

>> b = 30

b =    30
```

mengoperasikan variabel

```
>> c = a + b

c  =    50

>> d= (c / a)* 2

d  =   5
```

jadi dengan menggunakan variabel kita dapat menyimpan suatu statemen, persamaan dan lain dengan nama tertentu, yang dapat dipanggil sewaktu-waktu. Adapun aturan penulisan variabel sebagai berikut:

a. Boleh berupa huruf besar, kecil, atau gabungan dari huruf besar dan kecil. Misal : A,a, b, C, AA, Ab,

b. Boleh diberi angka dengan syarat tidak berdiri sendiri, yaitu bersambung dengan karakter yang lain. Misal A1, b2, Gaya1, Suhu_akhir, dll.

c. Tidak diizinkan menggunakan variabel operasi dasar yang dipergunakan dalam program matlab misal ( +, -, /, ans, dll)

d. Matlab peka terhadap huruf kapital (case sensitif) sehingga program akan membedakan antara huruf kapital dan huruf kecil. Misal AA dengan Aa dibaca sebagai dua variabel yang berbeda.

## 2.4. Variabel yang umum dalam matlab

Variabel yang terdefinisi di dalam program matlab adalah variabel yang umum dipakai sehingga pengguna tidak perlu membuat variabel baru.

Variabel terdefinisi

ans : answer, digunakan untuk menyimpan hasil perhitungan terakhir

eps : bilangan sangat kecil mendekati nol

pi : konstanta $\pi = 3.14$

inf : infinity ,bilangan positif tak berhingga

NaN : not a number. Merupakan hasil perhitungan yang tidak terdefinisi. Misal : 2/0, dan lain-lain

i,j : unit imajiner, menyatakan bilangan kompleks

## 2.5. Fungsi-fungsi matematika

Fungsi-fungsi standar yang umum dipakai dalam perhitungan matematis telah disediakan di dalam program Matlab. Fungsi-fungsi tersebut dipanggil dengan menggunakan command skript yang umum sebagaimana penulisan fungsi dalam perhitungan matematika. Fungsi perhitungan eksponensial, fungsi turunan dan integral, fungsi trigonometri serta fungsi matematika yang lain telah disediakan di dalam matlab.

Contoh fungsi matematik

Fungsi eksponen

sqrt(x) : akar kuadrat x

exp(x) : pangkat natural dari x, yaitu $e^x$

log(x) : logaritma natural dari x, yaitu ln x

log10(x) : logaritma basis 10 dari x, yaitu $\log_{10} x$

log2(x) : logaritma basis 2 dari x , yaitu $\log_2 x$

fungsi trigonometri

sin(x), cos(x), tan(x), cot(x), sec(x), csc(x)

asin(x), acos(x), atan(x), acot(x), asec(x), acsc(x)

fungsi trigonometri hiperbolik

sinh(x), cosh(x), coth(x), asinh(x), acosh(x),atanh(x),

asinh(x),acosh(x), acoth(x), acsch(x),asech(x),atanh(x)

Fungsi pembulatan

round(x), floor(x), ceil(x), fix(x), rem(x)

Fungsi bilangan komplex:

real(x) : menghitung komponen riil dari bilangan komplek x

imag(x) : menghitung komponen imajiner dari bilanan komplek x

abs(x) : menghitung magnitude dari bilangan komplek x

angle(x) : menghitung argumen dari bilangan komplek x

conj(x) : menghitung konjugasi dari bilangan komplek x

complex(a,b) : menghitung bilangan komplek dari a+bi

## 2.6. Aplikasi

Untuk lebih mudah memahami, berikut contoh yang dapat dijalan dalam command windows dan dapat diamati hasil eksekusi dari command berikut:

```
>> a=100;        b=8;      c=10000;        d=pi/2; e=2 + 3i;

>> sqrt(a)

>> log10(a)

>> log2(8)

>> log10(c)

>> sin(d/3); cos(d*(2/3));tan(d/2);

>>asin(d/3); acos(d*(2/3)); atan(d/2);

>>real(e)
```

```
>>imag(e)

>>angle(e)

>>abs(e)
```

# BAB 3 Pengenalan Awal

Pendahuluan

Matlab adalah interactive program untuk numerical computation dan data visualization; digunakan secara extensif oleh control engineers untuk analysis dan design. Terdapat banyak toolboxes yang tersedia yang terdiri dari basic functions di Matlab dalam aplikasi yang berbeda. Ide pada tutorialini adalah pengguna dapat melihat Matlab pada satu window ketika menjalankan Matlab di Window yang lain. Pengguna dapat membuat plot dan menggunakan program yang tersedia dalam m-file.

## 3.1. Vektor

Didalam matlab untuk menjalankan perintah vector dapat dilakukan dengan perintah berikut

*nama variable = [elemen vector]*

a = [1 2 3 4 5 6 9 8 7]

maka matlab akan menjalankan :

a = 1 2 3 4 5 6 9 8 7

Jika anda membuat vector dengan elemen 0 dan 30 dengan kenaikan 2 (metode ini digunakan untuk menciptakan vector waktu) dapat digunakan perintah berikut:

v = 0:2:30

maka matlab akan menjalankan

v = 0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30

Manipulasi vectors sering digunakan untuk system operasi. Misalkan anda ingin menambahkan 2 untuk setiap elemen 'a'. Persamaan menjadi :

b = a + 2

maka matlab akan menjalankan

b = 3 4 5 6 7 8 11 10 9

Jika ingin menambah 2 vektor secara bersamaan dengan panjang yang sama :

c = a + b

c = 4 6 8 10 12 14 20 18 16

Pengurangan vector dengan panjang yang sama juga dapat dilakukan dengan metode yang sama.

## 3.2. Fungsi

Matlab memiliki banyak fungsi standar. Setiap fungsi akan mempunyai tugas yang berbeda. Matlab berisi functions standard seperti sin, cos, log, exp, sqrt, dan fungsi lainnya. Secara umum fungsi konstanta seperti pi, dan i atau j atau akar -1, juga tersedia di Matlab. Sebagai contoh :

sin(pi/4)

*ans = 0.7071*

Untuk menentukan kegunaan setiap fungsi, ketik *help**[nama fungsi]*** *di command window Matlab.* Matlab juga mengijinkan anda menulis fungsi sendir dengan perintah *function;* pelajari bagaimana anda membuat program sendiri dan lihat fungsi yang tersedia di Matlab

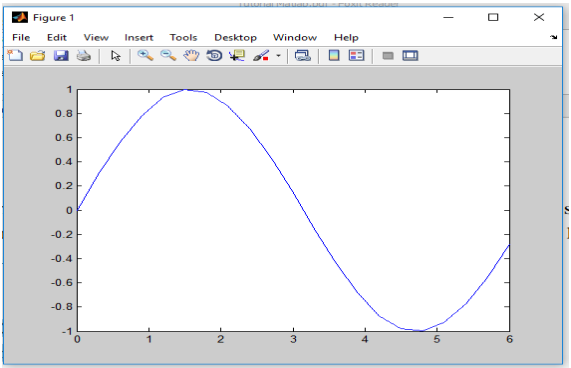
Plot

Sangat mudah membuat plots di Matlab. Misalkan andaingin memplot sebuah gelombang sinus sebagai fungsi waktu. Pertama buat vector waktu, dan kemudian hitung nilai sin untuk setiap vector waktu seperti pada gambar 1 :

t=0:0.30:6;

y = sin(t);

plot(t,y)




Gambar 3.1. Plot sinus

## 3.3. Polinomials

Di Matlab, sebuah polynomial diwakilkan oleh sebuahvektor. Untuk menciptakan polynomial di Matlab, masukkan coefficient polynomial kedalam vector dalam orde yang menurun. Misalkan polynomial berikut:

$$s^4 + 4s^3 - 12s^2 + 3s + 7$$

Didalam matlab perintah yang digunakan yaitu

x = [1 4 -12 -3 7]

sehingga nilai yang ditampilkan matlab adalah berikut

x = 1 4 -12 -3 7

Matlab dapat menginterpretasikan sebuah panjang n+1sebagai nth order polynomial. Jika polynomial missing pada coefficients, anda harus memasukkan nilai nol kedalam tempat yang bersesuaian di dalam vector. Sebagai contoh,

$$s^4 + 1$$

ditulis di Matlab sebagai:

y = [1 0 0 0 1]

Didalam matlab mencari nilai polynomial menggunakan fungsi polyval. Sebagai contoh, untuk mencari nilai polynomial pada s=3,

z = polyval([1 0 0 0 1],3)

z = 82

fungsi akar untuk ekstrak polynomial menggunakan fungsi *root*, misalkan pada persamaan

$$s^4 + 4s^3 - 12s^2 + 3s + 7$$

Perintah matlab yang digunakan yaitu

```
r= roots([1 4 -12 -3 7])

r =

   -5.9105

    1.9639

   -0.8037

    0.7503
```

JIka anda ingin mengalikan hasil 2 polynomials lakukan dengan convolution dari coefficients. Fungsi conv dapat digunakan. Seperti pada perintah berikut

```
x = [5 7];

y = [2 6 9];

z = conv(x,y)
```

matlab akan menghasilkan output sebagai berikut

```
z =10    44    87    63
```

Untuk membagi 2 polynomials dapat dilakukan dengan fungsi deconv. Misalkan z dibagi

y dengan hasil x.

[xx, R] = deconv(z,y)

xx =

1 2

R =

0 0 0 0

Jika anda ingin menambah 2 polinomial secara bersamaan dengan orde yang sama,

buatlah z=x+y akan berhasil (vectors x dan y harus mempunyai panjang yang sama).

Secara umum, anda dapat mendefinisikan fungsi, polyadd..

z = polyadd(x,y)

x =

1 2

y =

1 4 8

z =

1 5 10

Matriks

Masukkan matriks ke dalam Matlab seperti vector, kecuali penggunaan (,).

```
B = [1 2 3 4;5 6 7 8;9 10 11 12]

    B =

        1     2     3     4

        5     6     7     8

        9    10    11    12

B = [ 1   2   3   4

     5   6   7   8

     9 10 11 12]

    B =

        1     2     3     4

        5     6     7     8

        9    10    11    12
```

Matriks di Matlab dapat dimanipulasi dengan banyak cara. Misalkan dengan membuat transpose dengan menggunakan perintah ' :

```
C = B'

    C =

        1    5     9
```

2 6 10

3 7 11

4 8 12

Sekarang anda dapat mengalikan kedua matriks B dan C secara bersamaan.

D = B * C

D =

30 70 110

70 174 278

110 278 446

D = C * B

D =

107 122 137 152

122 140 158 176

137 158 179 200

152 176 200 224

Manipulasi matrix lain adalah dengan menggunakan operator .* .

E = [1 2;3 4]

F = [2 3;4 5]

```
G = E .* F

    E =

        1    2

        3    4

    F =

        2    3

        4    5

    G =

        2    6

       12   20
```

Jika anda ingin membuat pangkat dari tiap elemen matriks, gunakan fungsi berikut .^

```
E^3

    ans =

       37     54

       81    118
```

Jika anda ingin membuat pangkat dari tiap elemen matriks, gunakan fungsi berikut .^

```
E.^3

    ans =
```

1 8

27 64

Anda juga dapat menghitung inverse sebuah matrix:

X = inv(E)

X =

-2.0000 1.0000

1.5000 -0.5000

atau nilai eigen matriks:

eig(E)

ans =

-0.3723

5.3723

Untuk mendapatkan coefficients characteristic polynomial sebuah matrix. Gunakan

fungsi "poly" :

p = poly(E)

p =

1.0000 -5.0000 -2.0000

Ingat eigenvalues sebuah matrix adalah sama sepertiakar polynomial karakteristik :

```
roots(p)

    ans =

      5.3723

     -0.3723
```

# Bab 4 Matrik

Tujuan instruksional khusus

Setelah mempelajari bab 3 Matrik, mahasiswa mampu mencapai kompetensi berikut:

1. Mahasiswa mampu membuat fungsi skalar, vektor dan matrik
2. Mahasiswa mampu mendefinisikan ukuran matrik, mengenali dan membuat matrik khusus
3. Mahasiswa mampu memanipulasi indeks matriks, membuat deret dan membentuk ulang matrik

## 4.1. Matrik , vektor, dan skalar

Dalam program matlab, setiap input yang dimasukkan dapat dikategerikan atau dibaca sebagai sebuah matrik. Dalam mendefinisikan matrik, dikenal istilah skalar, dan vektor.

a. Skalar merupakan bilangan tunggal

b. Vektor merupakan bilangan yang disusun dalam bentuk baris dan kolom, dimana disebut matrik baris jika hanya terdiri dari satu baris dan memiliki n kolom, dan disebut matrik kolom jika memiliki n baris dan satu kolom

c. Matrik merupakan bilangan yang disusun dalam bentuk m baris dan n kolom. Secara umum matrik dapat ditulis Matrik m x n. Artinya matrik yang terdiri dari m baris dan n kolom.

Didalam matlab, untuk menuliskan matrik maupun vektor, cukup dengan menambahkan kurung siku [ ]. Untuk memisahkan kolom cukup dengan menekan spasi atau menambahkan koma “ , ”, sedangkan untuk memisahkan baris dapat dilakukan dengan dua cara, yaitu menekan enter atau menambahkan titik koma “ ; “ .

Untuk mendefinisikan skalar, dapat menggunakan tanda kurung siku, maupun langsung memasukkan input tanpa kurung siku, yang akan dibaca sebagai bilangan tunggal (skalar).

```
>> 2

ans =

    2

>> bilangan=1

bilangan =

    1
```

Dari contoh diatas, angka 2 dibaca sebagai sebuah bilangan tunggal (skalar), sedangkan bilangan dibaca sebagai sebuah variabel tunggal (juga didefinisikan skalar) yang memiliki nilai 1. Sehingga jika dipanggil variabel bilangan maka akan dieksekusi sebagai sebuah skalar.

```
>> bilangan

bilangan =

    1
```

Untuk mendefinisikan vektor baris

```
>> vektor=[1 2 4]

vektor =

    1     2     4

>> vektor2=[3,4,5,]

vektor2 =

    3     4     5
```

Sedangkan untuk membuat vektor kolom sebagai berikut

```
>> vektor3=[2;5;7;8]

vektor3 =

    2

    5

    7

    8

>> vektor4=[7

0

4]

vektor4 =

    7

    0
```

4

Dengan cara yang sama, matrik dengan ukuran m baris n kolom, dilakukan sebagai berikut:

Misal membangun matrik ukuran empat baris empat kolom

>> Matrik=[2 4 6 7;3 5 7 9;5 8 0 1;1 4 7 9]

Matrik =

2 4 6 7

3 5 7 9

5 8 0 1

1 4 7 9

Untuk matrik ukuran 3x3

>> Matrik2=[3 4 5

4 2 3

2 1 3]

Matrik2 =

3 4 5

4 2 3

2 1 3

Jadi untuk memisahkan baris dapat menggunakan tanda " ; " atau langsung dengan menekan enter.

Kita juga dapat membuat matrik dengan menggunakan indek matriknya. Misal membuat matrik ukuran 2 x 3 ( matrik dua baris tiga kolom) :

```
>> Matrik3(1,1)=2;

>> Matrik3(1,2)=4;

>> Matrik3(1,3)=6;

>> Matrik3(2,1)=3;

>> Matrik3(2,2)=5;

>> Matrik3(2,3)=7;
```

Kemudian kita panggil matrik yang telah dibuat

```
>> Matrik3

Matrik3 =

   2    4    6

   3    5    7
```

Untuk membangun suatu bilangan random dilakukan dengan perintah rand. Jika ditulis rand(m,n), maka bilangan random yang ditampilkan dalam bentuk matrik ukuran m baris dan n kolom. Bilangan random ini merupakan bilangan acak antara 0 sampai dengan 1.

```
>> rand(3,3)

ans =
```

0.8147 0.9134 0.2785

0.9058 0.6324 0.5469

0.1270 0.0975 0.9575

Misalkan bilangan random sebagai matrik ukuran 2 x 4 ( atau dengan kata lain, kita membangun bilangan random dalam bentuk matrik ukuran 2 baris empat kolom) :

>> rand(2,4)

ans =

0.9157 0.9595 0.0357 0.9340

0.7922 0.6557 0.8491 0.6787

Dengan perintah rand(m) artinya bilangan random dengan ukuran m baris m kolom (matrik persegi)

>> rand(4)

ans =

0.4387 0.1869 0.7094 0.6551

0.3816 0.4898 0.7547 0.1626

0.7655 0.4456 0.2760 0.1190

0.7952 0.6463 0.6797 0.4984

## 4.2. Operasi matrik

Dari matrik yang telah dibuat, dapat dilakukan operasi aljabar matrik, meliputi penjumlahan, pengurangan, perkalian dan semua operasi matrik (transpose,invers).

```
>> a=[2 3 4;3 2 1;3 2 2]

a =

   2   3   4

   3   2   1

   3   2   2

>> b=[4 5 3; 3 2 1;5 3 1]

b =

   4   5   3

   3   2   1

   5   3   1

>> x=a+b

x =

   6   8   7

   6   4   2

   8   5   3

>> y=b-a
```

y =

2 2 -1

0 0 0

2 1 -1

Untuk perkalian matrik dapat dilakukan dengan sangat mudah, misal d = a * b, dapat dilakukan sebagai berikut:

>> d=a*b

d =

37 28 13

23 22 12

28 25 13

Yang harus diingat, perkalian matrik harus memenuhi kaidah perkalian baris dan kolom. Artinya jumlah baris pengali harus sama dengan jumlah kolom yang dikalikan. Untuk contoh diatas tidak menimbulkan masalah karena jumlah baris dan kolom dari dua matrik tersebut sama. Misalnya sebuah matrik dengan ukuran 3 x 4 hanya bisa dikalikan dengan matrik yang memiliki 4 baris. Misal matrik M 4x1, M 4x2 , M 4x...

Demikian juga matrik ukuran 2x3 hanya bisa dikalikan dengan matrik yang memiliki 3 baris, yaitu M 3x1, M3x2, M3x...

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f & g \\ h & i & j \end{bmatrix} = \begin{bmatrix} ae + bh & af + bi & ag + bj \\ ce + dh & cf + di & cg + dj \end{bmatrix}$$

Perhatikan contoh matrik a yang sebelumnya telah dipraktekkan:

```
>> a

a =

    2    3    4

    3    2    1

    3    2    2
```

Misal matrik a dikalikan dengan sebuah matrik dengan ukuran 3x1

```
>> e

e =

    2

    3

    4

>> f = a * e

f =

   29

   16

   20
```

Atau dengan matrik ukuran 3x2.

```
>> h=[2 3;4 5 ;5 6]

h =
```

```
    2    3

    4    5

    5    6

>> g = a * h

g =

   36   45

   19   25

   24   31
```

Atau dengan matrik ukuran 3x3. Misal kita membuat matrik j, yang merupakan bilangan random dengan ukuran 3 baris dan 3 kolom.

```
>> j=rand(3)

j =

   0.7922    0.0357    0.6787

   0.9595    0.8491    0.7577

   0.6557    0.9340    0.7431
```

Maka hasil perkalian dari matrik a dan j adalah sebagai berikut:

```
>> h=a*j

h =

   7.0859    6.3548    6.6032
```

4.9513 2.7394 4.2948

5.6071 3.6734 5.0380

## 4.3. Manipulasi khusus

Matrik khusus dapat diperoleh dengan menggunakan perintah yang juga disediakan matlab, diantaranya membuat deret, membuat matrik khusus serta manipulasi matrik. Berikut contoh command untuk membuat matrik khusus:

Ones(a) membuat matriks satuan dengan ukuran a x a,

Ones(a,b) membuat matriks satuan dengan ukuran a x b

Zeros(a,b) membuat matrik nol dengan ukuran a x b

Eye(a) membuat matrik identitas dengan ukuran a x a

Rand(a) membuat matrik random dengan ukuran a x a yang berupa bilangan terdistribusi

normal dengan mean = 0 dan varians = 1.

[] membuat matrik kosong

Selanjutnya berikut contoh penggunaan perintah matrik tersebut

```
>> ones(4)

ans =

   1   1   1   1

   1   1   1   1
```

```
   1    1    1    1

   1    1    1    1

>>matrik_1=4*ones(4)

matrik_1 =

   4    4    4    4

   4    4    4    4

   4    4    4    4

   4    4    4    4

>> matrik_2=eye(4)

matrik_2 =

   1    0    0    0

   0    1    0    0

   0    0    1    0

   0    0    0    1

>> matrik_3=[ones(3,3) eye(3,3)]

matrik_3 =

   1    1    1    1    0    0

   1    1    1    0    1    0

   1    1    1    0    0    1
```

Membangun matrik bisa juga dengan menggunakan operator titik dua (:)

## 4.4. Membuat deret

Untuk membuat deret dengan matlab dapat dilakukan dengan mudah. Misal kita akan membuat deret bilangan 1 sampai dengan 10

```
>> deret=1:10

deret =

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

Jika deret tersebut ingin dibuat dengan interval 0,5, maka tinggal menyisipkan interval diantara bilangan deret yang akan dibuat. (contoh berikut dibatasi antara 1 sampai dengan 5)

```
>> deret_2=1:0.5:5

deret_2 =

   1.0000    1.5000    2.0000    2.5000    3.0000    3.5000    4.0000
4.5000    5.0000
```

Demikian seterusnya, kita dapat membuat deret dengan menggunakan selang / interval berapapun sesuai dengan yang diiinginkan.

Selain menggunakan operator titik dua, deret dapat dibangun dengan menggunakan perintah linspace.

Misal akan membangun deret bilangan dari 1 sampai 10 sebanyak 2 bilangan

```
>> deret_3=linspace(1,10,2)

deret_3 =

    1    10
```

Juga jika sebanyak 3 bilangan

```
>> deret_3=linspace(1,10,3)

deret_3 =

   1.0000    5.5000   10.0000
```

Dan seterusnya sesuai dengan jumlah bilangan yang diinginkan. Kelebihan dari deret ini adalah berapapun jumlah bilangan yang akan dibuat bisa ditampilkan tanpa kita menentukan berapa intervalnya. Interval akan diketahui setelah deret yang diinginkan sudah ditampilkan.

## 4.5. Menyelesaikan persamaan linear

Persamaan linear dengan beberapa variabel dapat diselesaikan dengan mudah baik dengan metode eleminasi maupun substitusi. Disamping itu dapat dilakukan dengan menggunakan metode matrik. Jika sebuah persamaan linear memiliki banyak varibel dapat diselesaikan dengan mudah dengan menggunakan matlab, asalkan kita dapat mengubah bentuk persamaannya menjadi matrik.

Misal sebuah persamaan 3 variabel

x + 2y + z = 6

x + 3y + 2z = 9

2x + y + 2z = 12

Maka persamaan linear tersebut dapat diselesaikan dengan mengubah menjadi bentuk perkalian matrik

Jika persamaan di atas diubah dalam bentuk perkalian matrik, kita dapat membuat sebuah matrik A sebagai koefisien variabel, Matrik X sebagai variabel x, y, dan z (yang dicari penyelesaiannya), serta matrik B sebagai hasil dari perkalian Matrik A dan X. Secara matematis dapat ditulis:

A . X = B

$$\begin{bmatrix} 1 & 2 & 1 \\ 1 & 3 & 2 \\ 2 & 1 & 2 \end{bmatrix} . \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 6 \\ 9 \\ 12 \end{bmatrix}$$

Untuk mencari penyelesaian dari X, kita mendapatkan dengan menggunakan persamaan :

$X = A^{-1} . B$

Dimana $A^{-1}$ adalah invers dari A.

Dalam matlab, invers matrik didapatkan cukup dengan menggunakan fungsi inv(x).

$A^{-1}$ = inv(A)

>> A

A =

```
    1    2    1

    1    3    2

    2    1    2

>> B

B =

    6

    9

   12

>> X=inv(A)*B

X =

   3.0000

        0

   3.0000
```

Sehingga himpunan penyelesaian dari persamaan linear tersebut seperti hasil eksekusi dari matrik X yaitu :

$$X = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 3 \end{bmatrix}$$

## 4.6. Transposisi

Transposisi artinya memindahkan posisi dari matrik yang telah disusun yaitu baris matrik diubah menjadi kolom dan kolom diubah menjadi baris. Untuk mendapatkan perubahan ini, menggunakan operator petik tunggal. ( ‘ ).

Untuk bilangan kompleks, petik tunggal ini akan memberikan nilai bilangan kompleks beserta konjugasinya. Sedangkan titik petik ( . ‘ ) akan memberikan nilai bilangan kompleks tanpa disertai bilangan konjugasi.

Untuk matrik bilangan riil, perhatikan contoh berikut:

```
>> matrik_contoh=[2 4 6]

matrik_contoh =

    2     4     6

>> matrik_transpose=matrik_contoh'

matrik_transpose =

    2

    4

    6

>> matrik_contoh2=[3 4 ; 5 7 ; 7 2]


matrik_contoh2 =

    3     4
```

```
    5    7

    7    2

>> matrik_transpose2=matrik_contoh2'

matrik_transpose2 =

    3    5    7

    4    7    2
```

Dari contoh tersebut, dapat dilihat perubahan matrik yang di transpose.

Untuk bilangan komplek, perhatikan contoh berikut

```
>> matrik_kompleks=[2 + 2*i 3]

matrik_kompleks =

  2.0000 + 2.0000i   3.0000 + 0.0000i

>> matrik_kompleks=[2+2*i 3]

matrik_kompleks =

  2.0000 + 2.0000i   3.0000 + 0.0000i

>> matrik_kompleks2=[2+2*i 3; 2 3+2i]

matrik_kompleks2 =

  2.0000 + 2.0000i   3.0000 + 0.0000i

  2.0000 + 0.0000i   3.0000 + 2.0000i
```

```
>> matrik_konjugasi=matrik_komplek2.'

Undefined function or variable 'matrik_komplek2'.

Did you mean:

>> matrik_konjugasi=matrik_kompleks2.'

matrik_konjugasi =

  2.0000 + 2.0000i   2.0000 + 0.0000i

  3.0000 + 0.0000i   3.0000 + 2.0000i

>> matrik_konjugasi=matrik_kompleks2'

matrik_konjugasi =

  2.0000 - 2.0000i   2.0000 + 0.0000i

  3.0000 + 0.0000i   3.0000 - 2.0000i
```

## 4.7. Operasi elemen per elemen

Perkalian matrik merupakan hasil operasi perkalian baris dan kolom, sebagaimana pada contoh bab sebelumnya. Berbeda dengan operasi penjumlahan dan pengurangan, yang merupakan operasi yang dilakukan pada masing-masing elemen matrik yang dijumlahkan / dikurangkan. Namun dengan menggunakan matlab, perkalian matrik dapat dilakukan elemen per elemen sebagaimana operasi penjumlahan / pengurangan matrik. Cukup menambahkan operator titik pada matrik yang akan dikalikan maka operasi yang dilakukan adalah operasi elemen per elemen matrik. Tentu saja dua matrik yang dikalikan adalah matrik yang memiliki ukuran yang sama. Di samping itu, operasi perpangkatan juga bisa

menggukan operator titik pada matrik yang akan didefinisikan. Berikut contohnya:
operasi penjumlahan matrik

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

>> B=[3 5 2; 2 3 4];

>> A

A =

   2   3   4

   3   2   1

>> B

B =

   3   5   2

   2   3   4

>> C=A+B

C =

   5   8   6

   5   5   5

>> D=A.*B

D =

   6  15   8
```

```
   6    6    4
```

Untuk pangkat

```
>> E=A.^2

E =

   4    9   16

   9    4    1
```

Untuk pembagian

```
>> F=D./3

F =

   2.0000    5.0000    2.6667

   2.0000    2.0000    1.3333
```

Pembagian terbalik

```
>> G=2.\E

G =

   2.0000    4.5000    8.0000

   4.5000    2.0000    0.5000
```

# BAB 5 PLOT GRAFIK

Tujuan instruksional umum

Mahasiswa mampu mengoperasikan Matlab membuat plot grafik

Tujuan instruksional khusus

Setelah mempelajari bab 5 Membuat Grafik 2D dan 3D, mahasiswa mampu mencapai kompetensi berikut:

1. mahasiswa mampu membuat plot grafik 2 dimensi untuk sebuah persamaan
2. mahasiswa mampu membuat grafik 3 Dimensi meliputi plot garis, plot permukaan, dan plot kontur

Maltab memiliki fasilitas yang bagus dan unggul dalam menampilkan grafik. Grafik yang dibuat dengan menggunakan perintah/ command yang sederhana. Grafik yang ditampilkan bisa merupakan grafik sederhana dua dimensi maupun grafik plot tiga dimensi.

## 5.1. Plot grafik dua dimensi

Perintah dasar dalam membuat grafik adalah dengan perintah plot. Dengan hanya mendefinisikan plot untuk sumbu x dan sumbu y , kita sudah dapat membuat sebuah plot grafik.

Misalkan secara sederhana untuk pemahaman pembuatan grafik pasangan x dan y, jika kita memiliki bilangan x = 0 sampai dengan 10. Dengan menggunakan deret kita bisa mendefinisikan x sebagai berikut:

```
>> x=0:10
```

Maka bilangan x yang dimaksud adalah

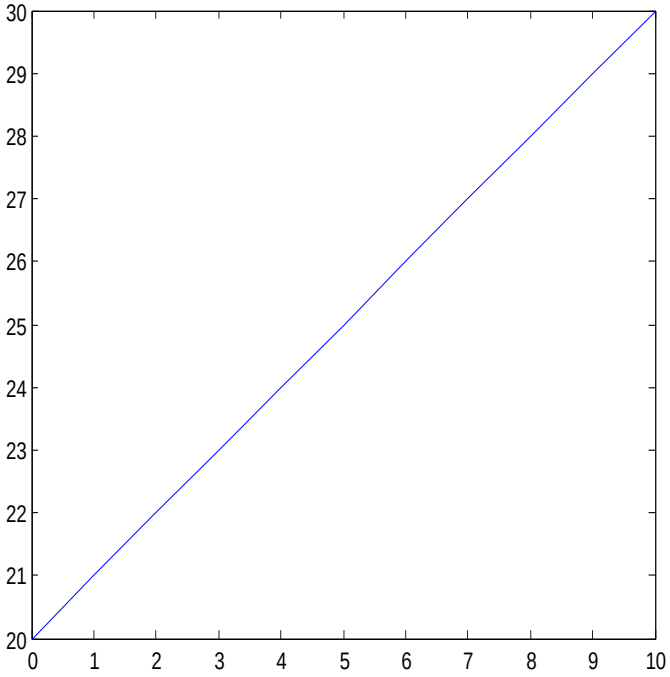
```
x =

   0    1    2    3    4    5    6    7    8    9   10
```

Kemudian berpasangan dengan y = 20 sampai dengan 30. Dengan deret juga kita dapat mendefinisikan y

```
>> y=20:30

y =

  20   21   22   23   24   25   26   27   28   29   30
```

Kedua variabel x dan y tersebut dapat dipasangkan dengan menggunakan koordinat berpasangan dan akan membentuk grafik dengan perintah plot.

```
>> plot(x,y)
```




Gambar 5.1. Plot (x,y)

Setelah itu dapat ditambah kan judul grafik, keterangan untuk sumbu x dan sumbu y

>> title('grafik linear x dan y');

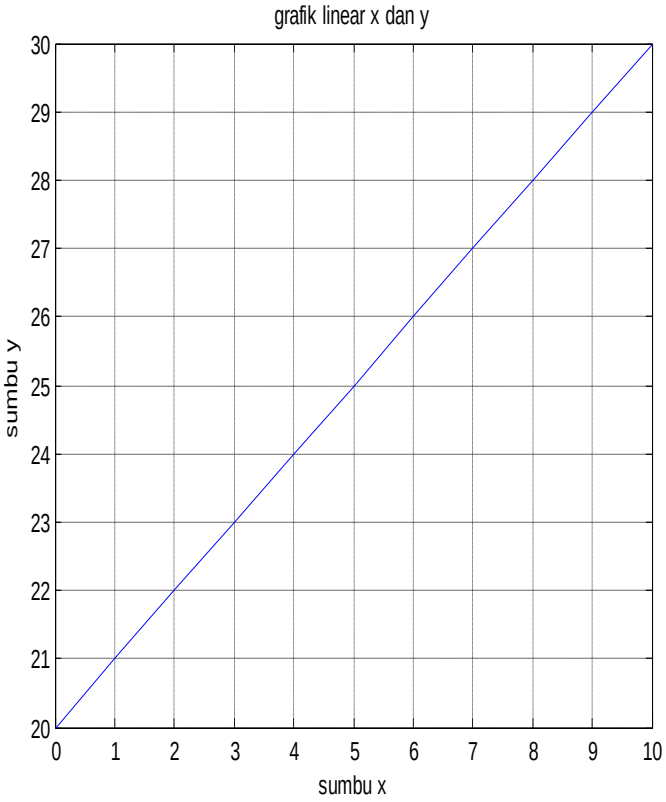
>> xlabel('sumbu x');

>> ylabel('sumbu y');

Untuk menambahkan grid dapat dilakukan dengan menambahkan perintah grid on

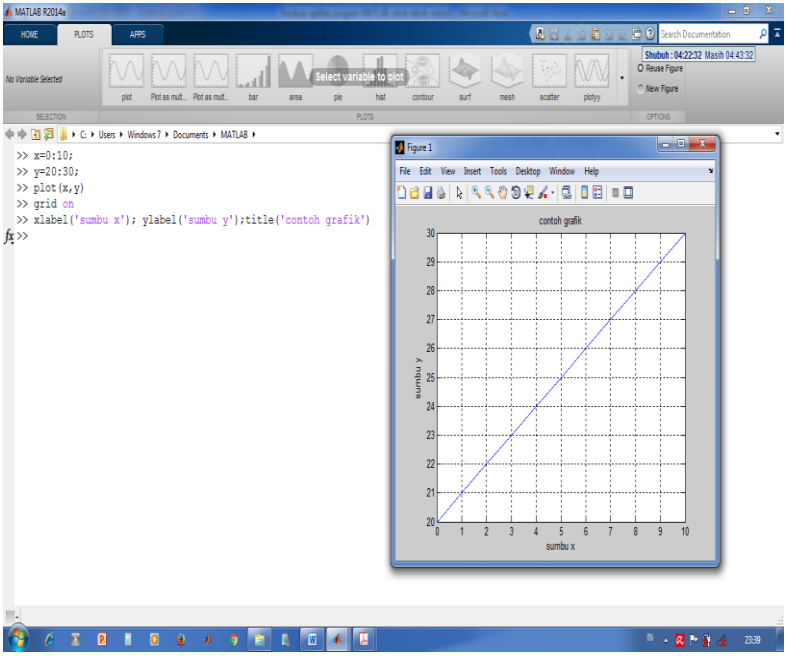
>> grid on

Maka didapat grafik sebagai berikut:




Gambar 5.2. Menampilkan judul tabel

Dari perintah diatas didapatkan hasil sebagai berikut:




Gambar 5.3. Menampilkan figure

Sebagai contoh yang lain, misalnya membuat grafik dari persamaan kuadrat : y= $3x^2$ + 3x -15, pada interval nilai x antara -5 sampai dengan x= 5.

Maka langkah pertama adalah mendefinisikan nilai x sesuai dengan intervalnya dan fungsi persamaan yang di berikan.

Untuk menentukan nilai x antara -5 sampai dengan 5 dibuat selang yang kecil supaya kurva grafik yang dihasilkan menjadi lebih halus, karena titik grafiknya memiliki nilai selang yang kecil. Jika selangnya besar maka kurva yang dihasilkan menjadi kurang halus.

Untuk menentukan nilai x, dapat dilakukan dengan membuat deret antar nilai -5 sampai dengan 5 dan diambil selang sebesar 0,01

```
>> x=-5;0.01:5;
```

Kemudian dimasukkan nilai persamaan untuk mendapat nilai y:

```
>> y=3*x.^2+3*x-5;
```

Untuk mendapatkan grafiknya, digunakan perintah plot

```
>> plot(x,y);
```

Maka akan didapat grafik persamaannya. Untuk menambahkan judul digunakan perintah title

```
>> title('grafik y = 3x^2 + 3x - 5');
```

Untuk menambahkan label pada sumbu x dan sumbu y digunakan perintah label :

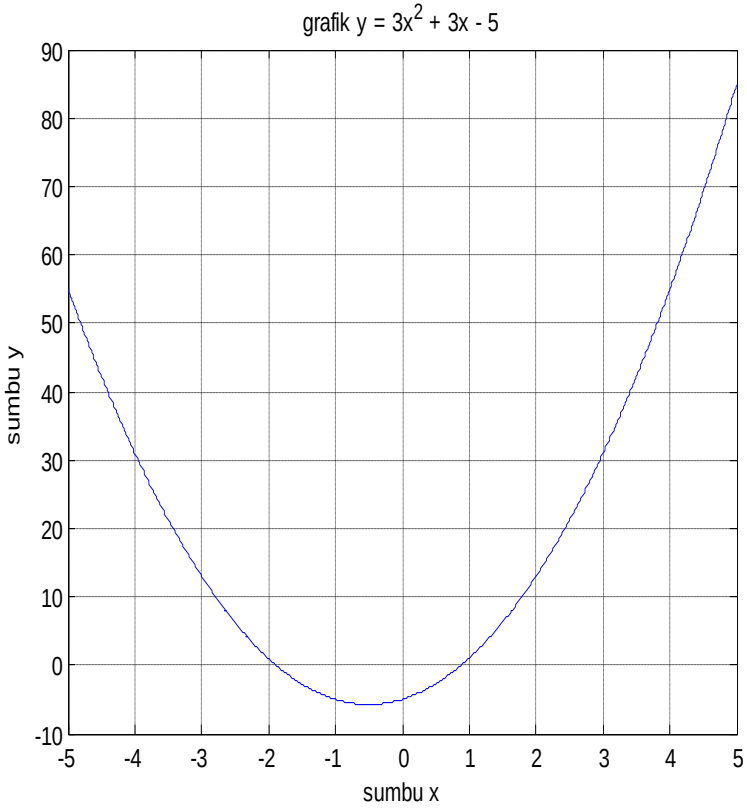
```
>> xlabel('sumbu x');
```

```
>> ylabel('sumbu y');
```

Untuk menambahkan grid ditambahkan perintah grid on

```
>> grid on
```

Berikut grafik yang kita dapatkan




Gambar 5.4. Grafik fungsi kuadrat

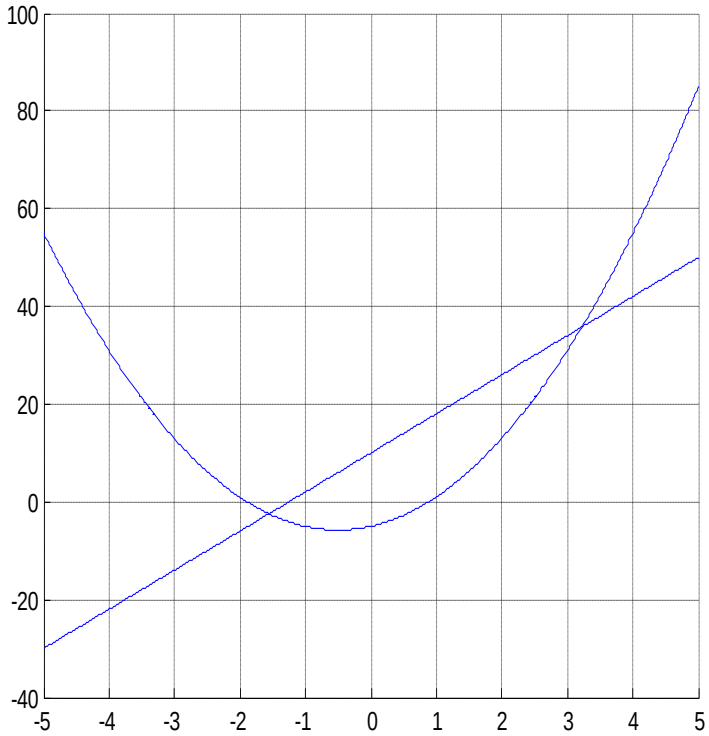
Kitapun dapat menambahkan grafik dengan persamaan yang lain. Misalkan kita ingin menambahkan sebuah persamaan linear y = 8x + 10, maka terlebih dahulu ditambahkan perintah hold on, artinya grafik sebelumnya akan tetap ditampilkan (tidak dihapus). Jika digunakan langsung perintah plot, maka otomatis grafik sebelumnya akan dihapus. Untuk lebih jelasnya berikut contohnya:

```
>> hold on
```

Selanjutnya kita buat variabel baru untuk persamaan linear yang diinginkan, misalkan yL , (persamaan y linear).

```
>> yL=8*x+10;
```

Selanjutnya dibuat plot dari persamaan linear tersebut




Gambar 5.5. Perintah Hold On

Kita tinggal menambahkan judulnya dengan perintah title dan label (untuk sumbu x dan sumbu y)

Disamping itu, properti titik dari grafik yang dibuat dapat dirubah dengan menambahkan perintah string, dengan command sebagai berikut:

Plot(x,y,'string')

Misal untuk membuat kurva grafik berwarna merah kita tinggal menambahkan r setelah plot x dan y.

```
>> plot(x,y,'r')
```

Berikut tabel untuk mengubah properti dari kurva yang dibuat

| Warna | | Jenis garis | | Jenis point | |
|---|---|---|---|---|---|
| b | biru | - | utuh | . | titik |
| g | hijau | : | titik-titik | o | lingkaran |
| r | merah | -. | titik strip | x | tanda x |
| c | biru muda | -- | putus-putus | + | tanda + |
| m | ungu | | | * | tanda * |
| y | kuning | | | s | bujur sangkar |
| k | hitam | | | d | permata |
| w | putih | | | v | segitiga ke bawah |
| | | | | ^ | segitiga ke atas |

| | | |
|---|---|---|
| | | < segitiga ke kiri |
| | | '> segitiga ke kanan |
| | | p segilima |
| | | h segienam |

Sebagai contoh yang lain sebagai berikut

Misalkan kita akan membuat sebuah grafi k sinus (trigonometri). Jika diberikan persamaan y=sin x , diminta gambar grafiknya antara sudut 0 sampai dengan $360^0$.

Maka langkah pertama yang dilakukan adalah menyatakan nilai x sebagai variabel persamaan y. Derajat dalam matlab kita rubah dengan menggunakan π, dimana $\pi = 180^0$ . Dalam contoh ini kita akan mendefinisikan x dengan sebuah deret, dengan perintah linspace.

```
>> x=linspace(0,2*pi,100);
```

Linspace artinya kita membuat deret antara bilangan 0 sampai dengan 2 x pi sebanyak 100 titik bilangan.

Kemudian kita masukkan persamaan yang diinginkan

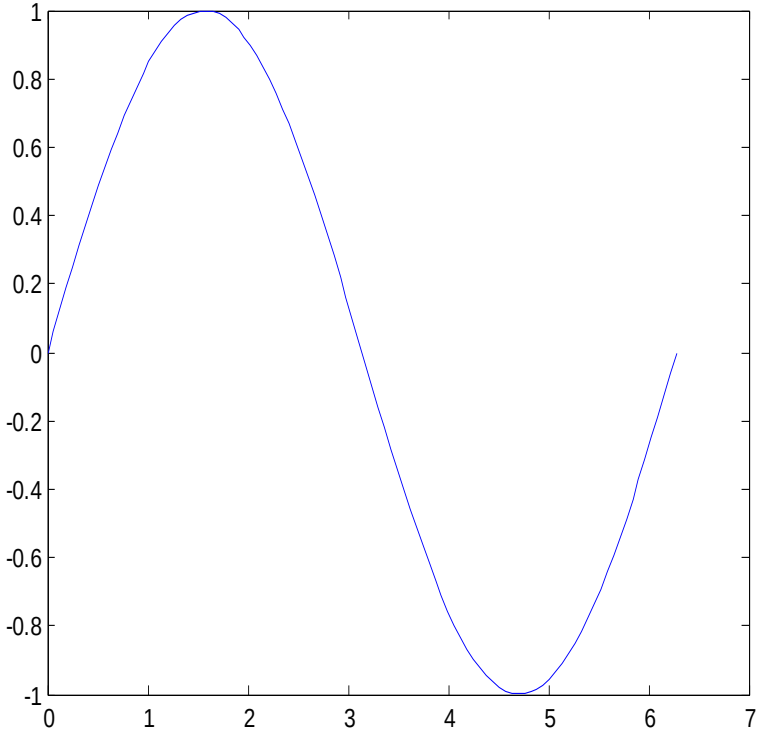
```
>> y=sin(x);
```

Selanjutnya kita buat grafiknya dengan perintah plot. Untuk mendapatkan jendela grafik, kita dapat menggunakan perintah figure.

```
>> figure
```

```
>> plot(x,y);
```

Maka akan didapat grafik sebagai berikut




Gambar 5.6. Grafik fungsi sinus

Sebagaimana keterangan sebelumnya, kita dapat mengubah grafik dalam figure tersebut dengan memasukkan persamaan yang baru. Maka otomatis grafik sebelumnya akan terhapus diganti yang baru.

Namun jika kita ingin membuat jendela yang baru, tanpa menghapus yang lama, kita tinggal mengetikkan figure untuk menambah jendela grafik.

Misal kita akan membuat grafik tangga dari persamaan sinus tersebut, dengan membuat selang/interval 20 titik dari 0 sampai dengan 2π, maka kita bisa membuat interval sudut x yang baru:

```
>> x=linspace(0,2*pi,20);

>> y=sin(x);
```

Kemudian kita akan memplot grafiknya, dengan membuat jendela figure yang baru:

```
>> % membuat jendela figure

>> figure

>> stairs(x,y);
```

Kita dapat menggabungkan beberapa grafik dalam satu figure, dengan cara menuliskan persamaan yang diinginkan sekaligus.

Misal kita akan menampilkan grafik sinus dan kosinus dari sudut x yang telah didefinisikan;

```
>> x=linspace(0,2*pi,100);

>> y1=sin(x);
```

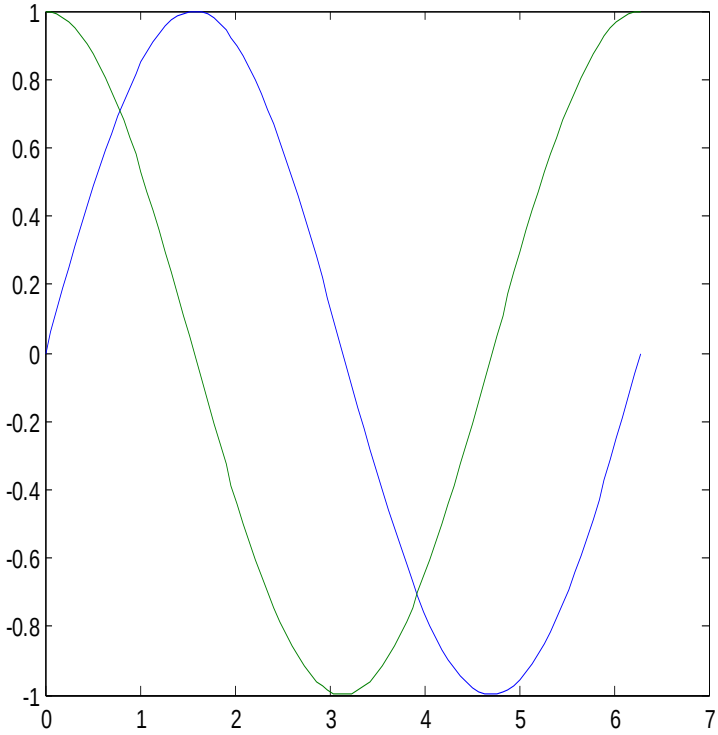
>> y2=cos(x);

Kemudian kita membuat plot grafiknya

>> figure

>> plot(x,y1,x,y2);

Maka didapatkan

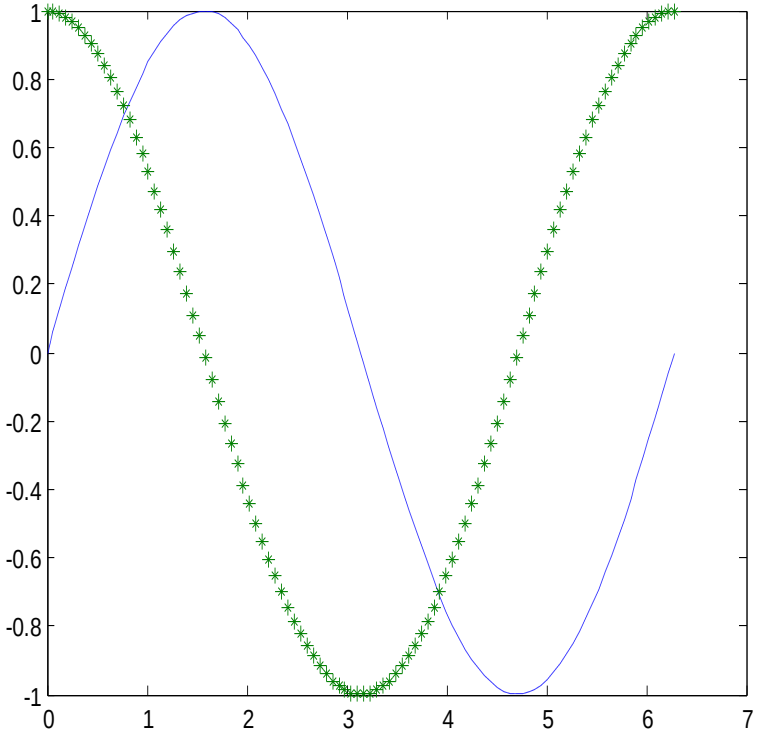



Gambar 5.7. Grafik fungsi trigonometri

Kemudian kita dapat merubah warna, serta tipe titik/garis kurva yang dibuat. Dengan menggunakan properti garis / titik sesuai tabel maka kita tuliskan perintah plotnya:

```
>> plot(x,y1,'--',x,y2,'*');
```

Artinya plot untuk y1 dengan tipe garis putus-putus dan y2 dengan tipe bintang.

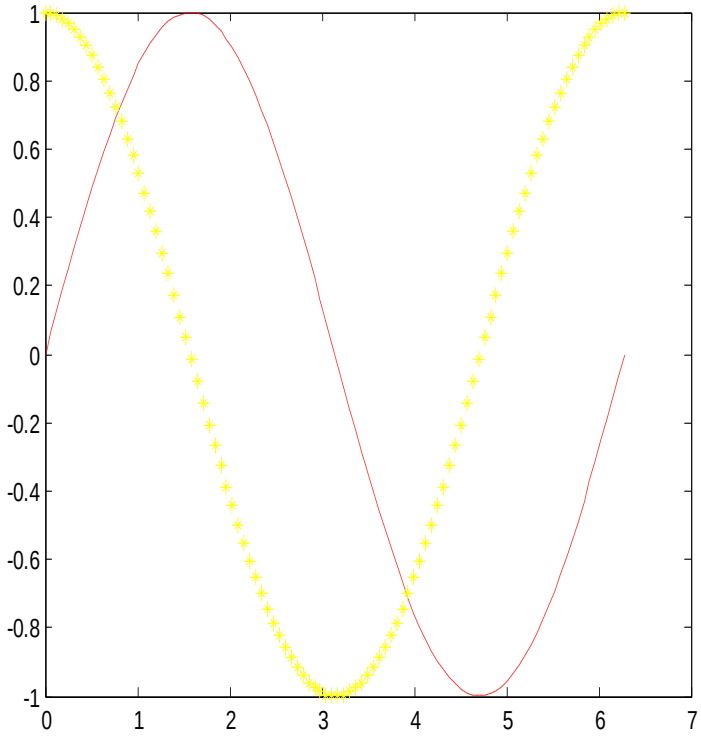


Gambar 5.8. Memanfaatkan properti tabel

disamping itu kita dapat merubah warnanya :

>> plot(x,y1,'--r',x,y2,'*y');

Artinya garis /titik persamaan y 1 dirubah menjadi merah, dan y2 menjadi kuning

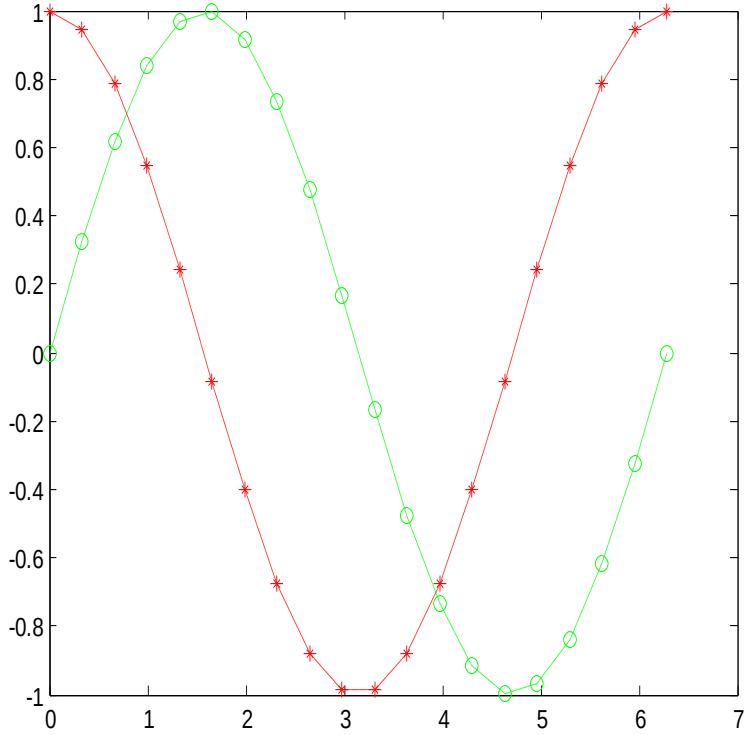



Gambar 5.9. Fungsi properti tabel

Kemudian kita dapat merubah lagi sesuai dengan kebutuhan yang diinginkan. Misalkan nilai interval x dirubah sebagai mana contoh sebelumnya, yaitu diinginkan 20 titik sudut antar 0 sampai dengan $2\pi$.

```
>> x=linspace(0,2*pi,20);

>> y1=sin(x);

>> y2=cos(x);

>> plot(x,y1,'--go',x,y2,'--*r');
```

Artinya persamaan y1, memiliki tipe garis putus-putus dan tiap titiknya berupa huruf o dan persamaan y2 menggunakan penghubung garis putus-putus dan tiap titiknya berupa bintang. Maka didapat grafik sebagai berikut:

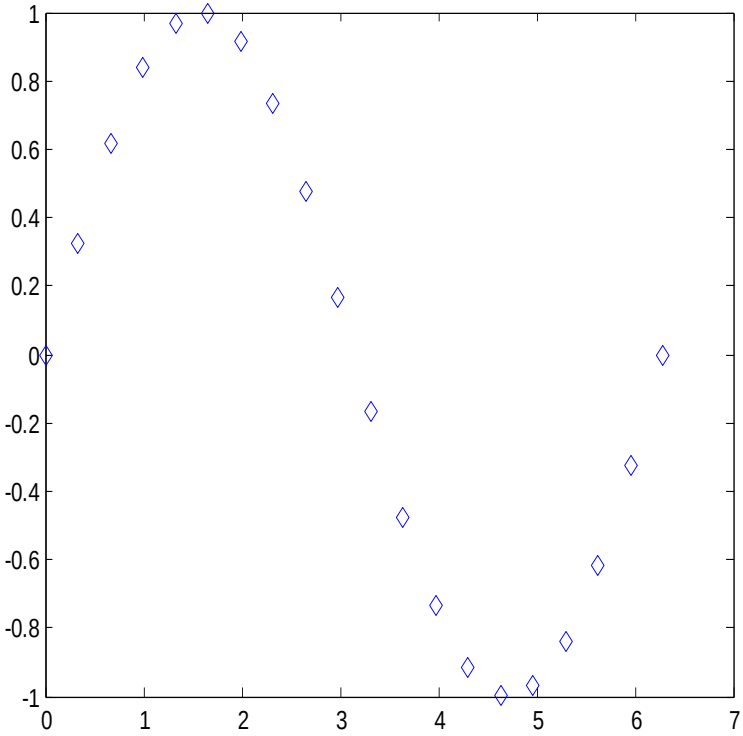



Gambar 5.10. Mengubah properti tabel

Jika kita ingin menampilkan hanya pada titik data yang terukur saja maka dapat dilakukan dengan menggunakan perintah:

```
>> plot(x,y1,'d')
```

Maka dapat kita peroleh grafik y1, dengan titiknya ditunjukkan dengan bentuk permata (diamond).u

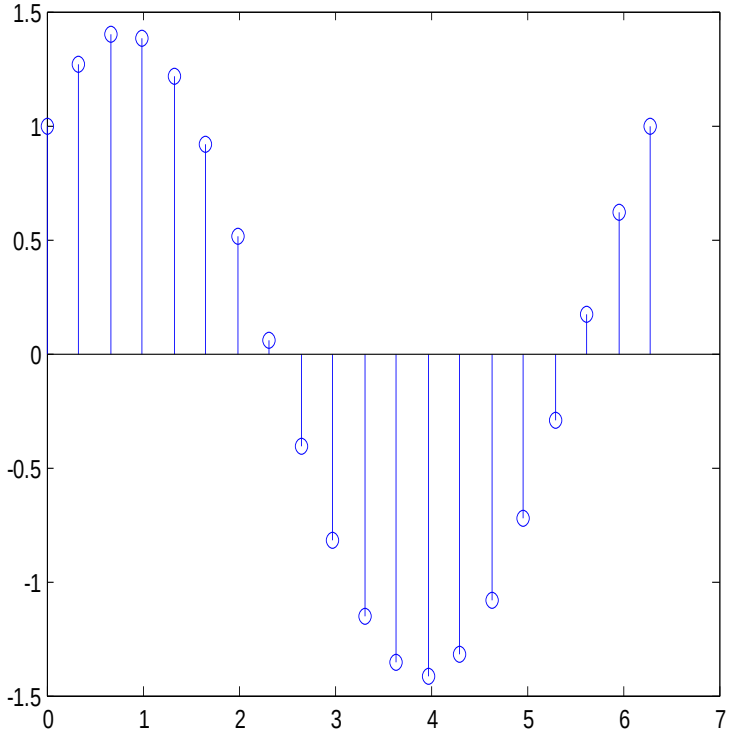



Gambar 5.11. Membuat titik pada grafik

Untuk mendapatkan grafik stem, maka plot stem digunakan untuk membuat plotgrafiknya

```
>> x=linspace(0,2*pi,20);

>> y=sin(x);

>> y2=cos(x);
```

```
>> figure

>> stem(x,y+y2);
```

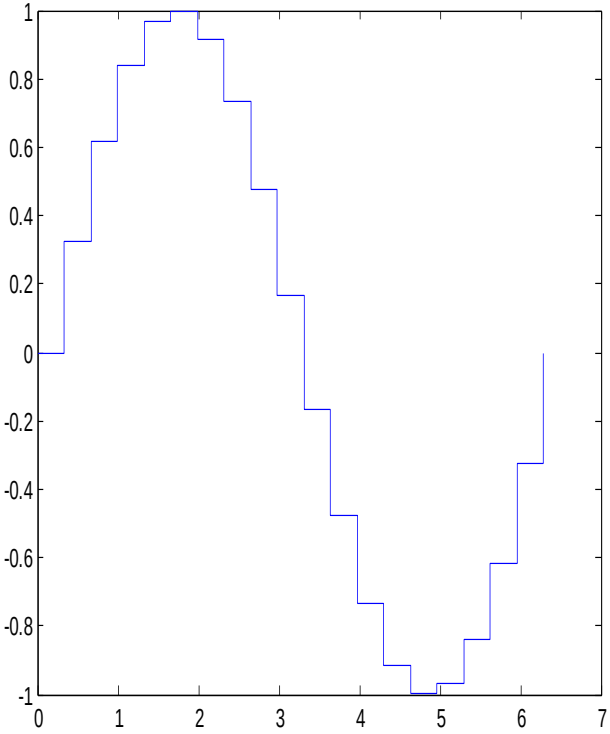



Gambar 5.12. Grafik stem

## 5.2. Membuat plot tangga

Untuk membuat plot tangga dapat dilakukan dengan menggunakan stairs

>> stairs(x,y);




Gambar 5.13. Grafik tangga

## 5.3. Menambahkan legenda grafik

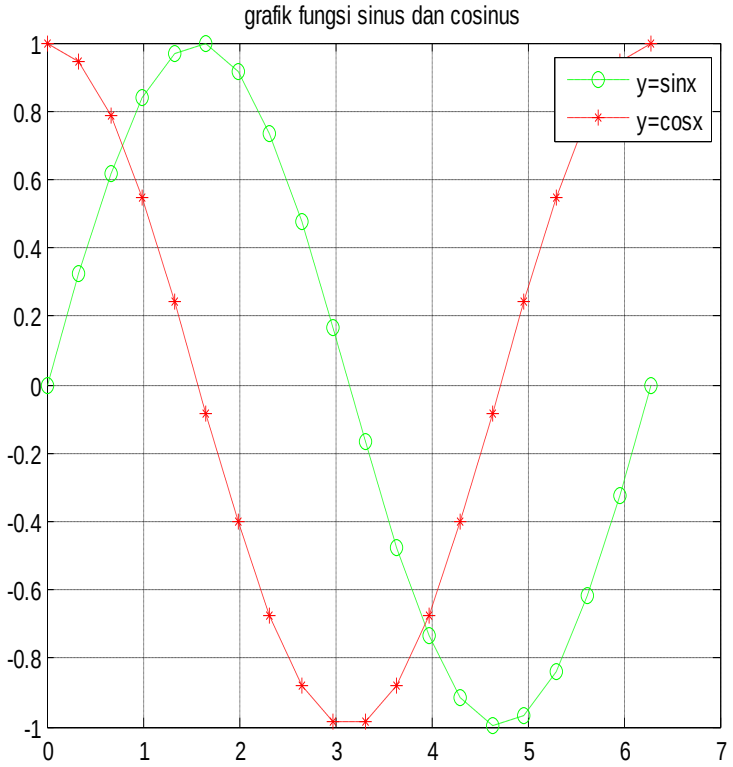
Untuk menambahkan legend pada grafik digunakan perintah :

legend('string');

langkah pertama kita tampilkan grafiknya:

```
>> plot(x,y1,'--go',x,y2,'--*r');
```
Kemudian dimasukkan perintah untuk menampilkan legend dari grafik yang telah dibuat;

```
>> legend('y=sinx','y=cosx')
```
Selanjutnya jika ingin menambah judul grafik, label sumbu x dan sumbu y dapat menggunakan perintah title, xlabel dan ylabel, sebagaimana contoh sebelumnya. Jika ingin menampilkan grid, dapat digunakan perintah grid on, dan jika ingin menghilangkan dengan perintah grid off.




Gambar 5.16. Grafik fungsi sinus dan cosinus

## 5.4. Membuat Subplot

Jika dibutuhkan menggambar beberapa grafik yang berbeda dapat dilakukan dengan menempatkannya pada jendela yang berbeda. Dengan perintah figure maka akan dibuat sebuah jendela baru untuk menggambar sebuah grafik. Namun bila dibutuhkan menampilkan beberapa grafik berbeda dalam satu jendela / figur, dapat digunakan perintah subplot. Susunan grafik tersebut menggunakan aturan indeks matrik. Yaitu berapa baris dan kolom dari grafik yang disusun. Script perintahnya sebagai berikut:

>>subplot(a,b,c)
dimana
a : jumlah baris
b : jumlah kolom
c : nomor urutan grafik yang akan dibuat

Misalnya kita akan membuat grafik persamaan linear, grafik fungsi kuadrat, dan grafik sebuah fungsi polinomial serta grafik fungsi trigonometri hiperbolik. Disini kita akan menggambar empat buah grafik. Misalnya susunan grafik tersebut kita susun dalam dua baris dan dua kolom. Untuk urutan penomoran grafik menggunakan aturan matrik.

Misal pada langkah awal kita definisikan nilai x dan persamaan yang akan dibuat plot grafik. Jika diketahui :

persamaan linear : $y1 = 2x + 3$
persamaan kuadrat : $y2 = 2x^2 – 2$
persamaan polinomial : $y3 = x^3 + 2x^2 + 28x -10$
persamaan hiperbolik : $y4 = \sinh x$
dan akan diplot grafik dengan nilai x = 0 sampai dengan 5.

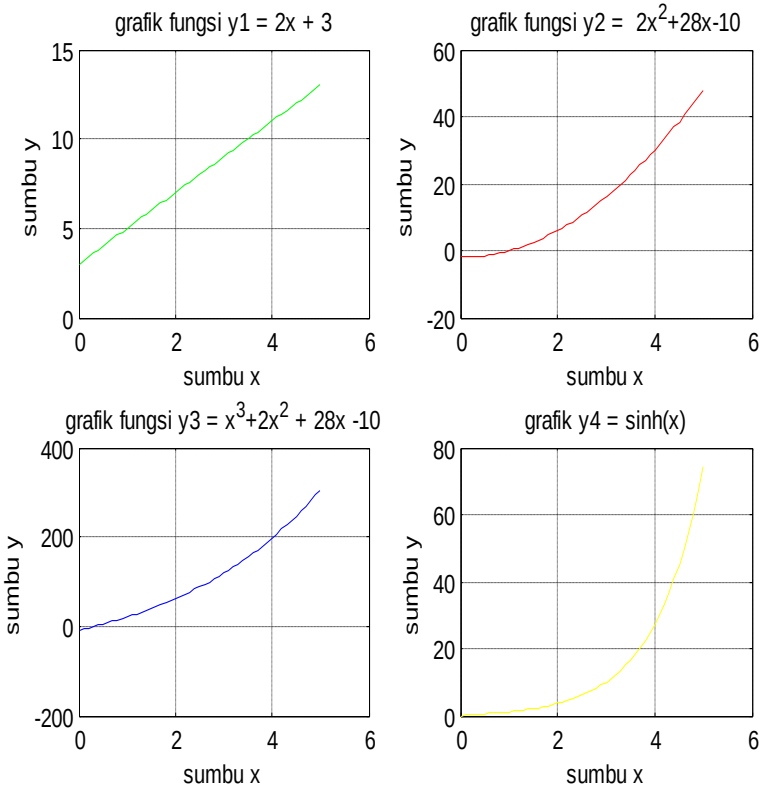
Maka fungsi perintah yang dapat kita tulis:

```
>> x=0:0.1:5;
>> y1=2*x+3;
>> y2=2*(x.^2)-2;
>> y3=x.^3+2*(x.^2)+28*x-10;
>> y4=sinh(x);
```

Membuat subplot untuk grafik nomer 1 (terletak pada baris satu kolom satu)

```
>> subplot(2,2,1)
>> plot(x,y1,'g');
>> title('grafik fungsi y1 = 2x + 3');
>> xlabel('sumbu x');ylabel('sumbu y');
>> grid on
```

```
>> subplot(2,2,2);
>> plot(x,y2,'r');
>> title('grafik fungsi y2 =  2x^2+28x-10');
>> xlabel('sumbu x');ylabel('sumbu y');
>> grid on


>> subplot(2,2,3);
>> plot(x,y3);
>>  title('grafik fungsi y3 = 2x + 3');
>> title('grafik fungsi y3 = x^3+2x^2 + 28x -10');
>> xlabel('sumbu x');ylabel('sumbu y');
>> grid on

>> subplot(2,2,4);
>> plot(x,y4,'y');
>> title('grafik y4 = sinh(x)');
>> xlabel('sumbu x'); ylabel('sumbu y');
>> grin  on
```



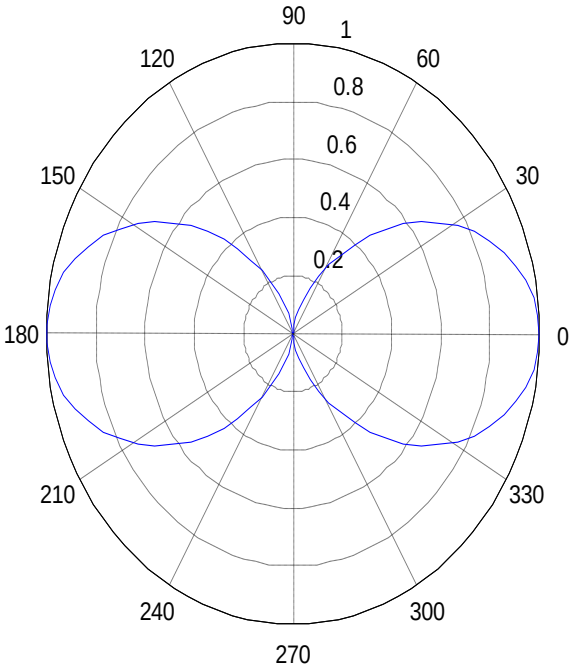

Gambar 5.17. Gambar subplot

## 5.5. Plot koordinat polar

Plot grafik dalam bentuk koordinat polar dapat dilakukan dengan menggunakan perintah polar.

```
>> theta=linspace(0,2*pi,100);

>> y=(cos(theta)).^2;

>> polar(theta,y);
```

Sehingga dapat diperoleh grafik polar



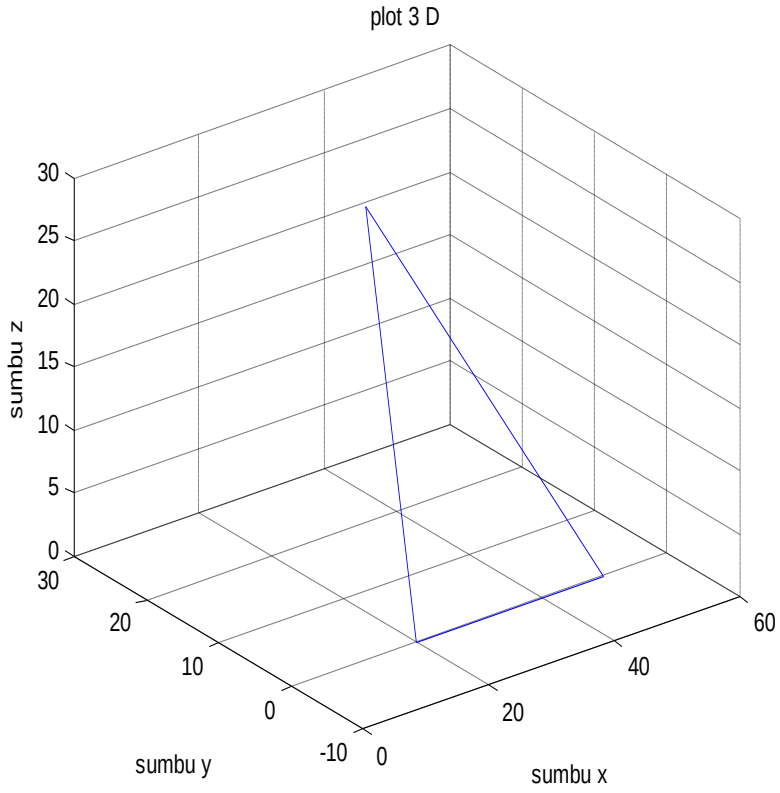

Gambar 5.18. Grafik polar

## 5.6. Plot tiga dimensi

Plot grafik dalam program matlab dapat ditampilkan dalam plot 3 dimensi. Sebagai contoh, kita akan membuat sebuah segita dalam ruang tiga dimensi dengan ordinat masing-masing titik sudut segitiga dalam ordinat x,y, dan z : A(20,0,0), B(50,0,0) dan

C(35,20,25). Maka dengan menggunakan plot3 dapat didefinisikan masing-masing titik ordinatnya dalam bentuk matrik baris:

```
>> x=[20 50 35 20];

>> y=[0 0 20 0];

>> z=[0 0 25 0];

>> plot3(x,y,z);

>> grid on

>> xlabel('sumbu x');ylabel('sumbu y');zlabel('sumbu z');

>> axis([0 60 -10 30 0 30]);
```

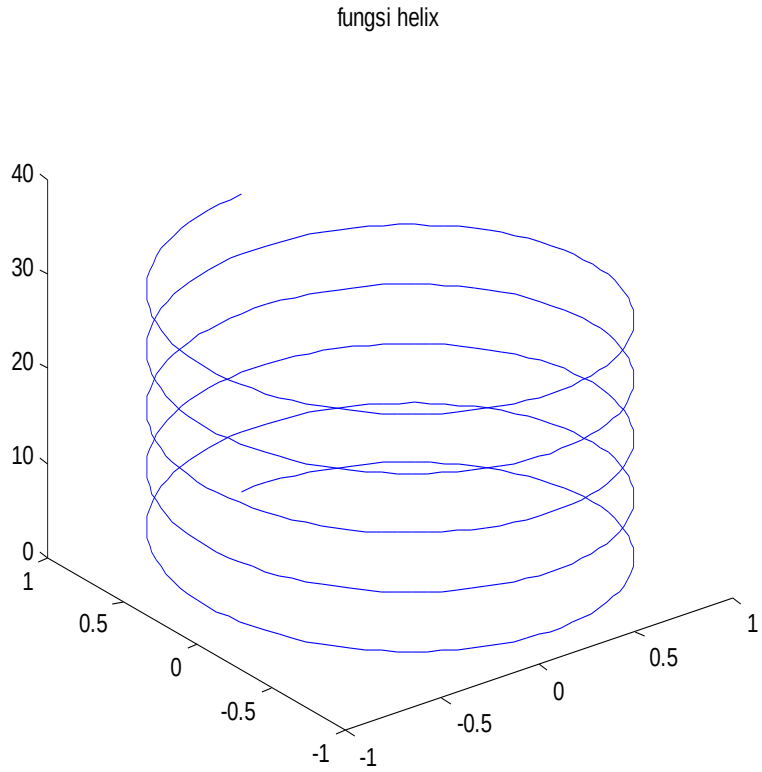



Gambar 5.19. Grafik garis 3D

Untuk contoh yang lain adalah sebagai berikut:

```
>> t=0:pi/50:10*pi;
>> x=sin(t);
>> y=cos(t);
>> z=t;
>> plot3(x,y,z);
>> title('fungsi helix');
```

Maka kita dapatkan plot3 dari fungsi tersebut diatas, yaitu dari hasil fungsi sinus dan cosinus :




Gambar 5.20. Grafik Helix

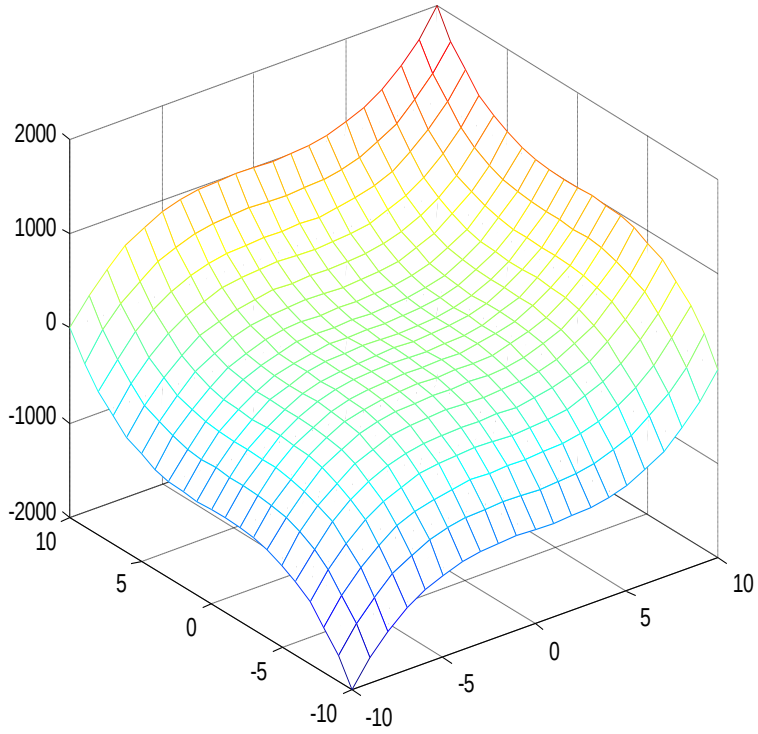
## 5.7. Mesh dan surface plot

Mesh dan surf berfungsi untuk membuat plot permukaan 3 dimensi dari data matrik atau persamaan. Misalnya menggambarkan fungsi $z = x^3 + y^3$ .

Adapun langkah yang dilakukan sebagai berikut:

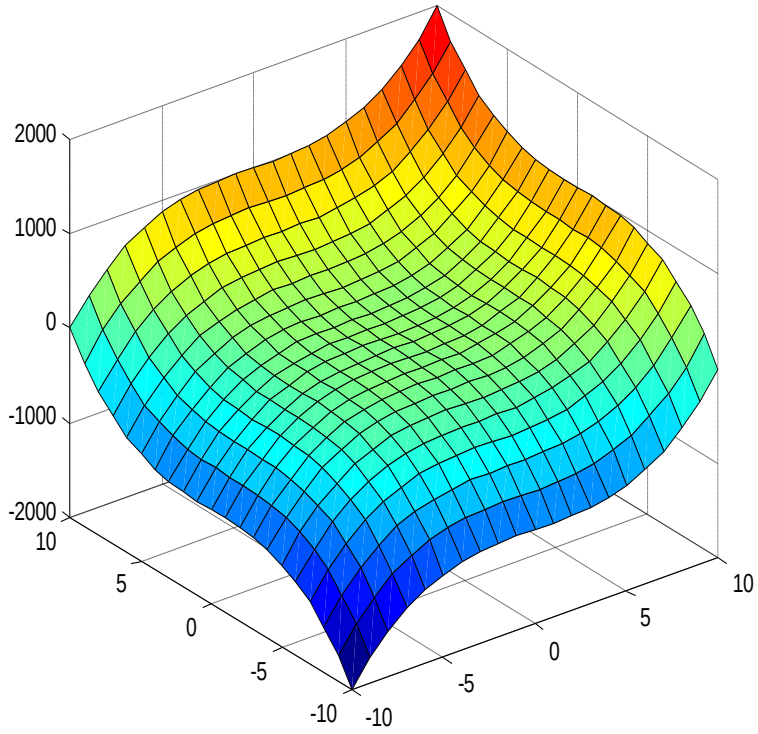
a. Mendefinisikan batas nilai x yang akan diplot
b. Menggunakan command meshgrid untuk mengisi bidan XY dengan jalinan titik
c. Memasukkan persamaan fungsi yang akan diplot
d. Membuat plot yang diinginkan dengan commmand mesh dan surf

```
>> [X,Y]=meshgrid(-10:1:10);
>> Z=X.^3+Y.^3;
>> Z=X.^3+Y.^3;
>> mesh(X,Y,Z);
```




Gambar 5.21. Meshgrid

>> surf9X,Y,Z);

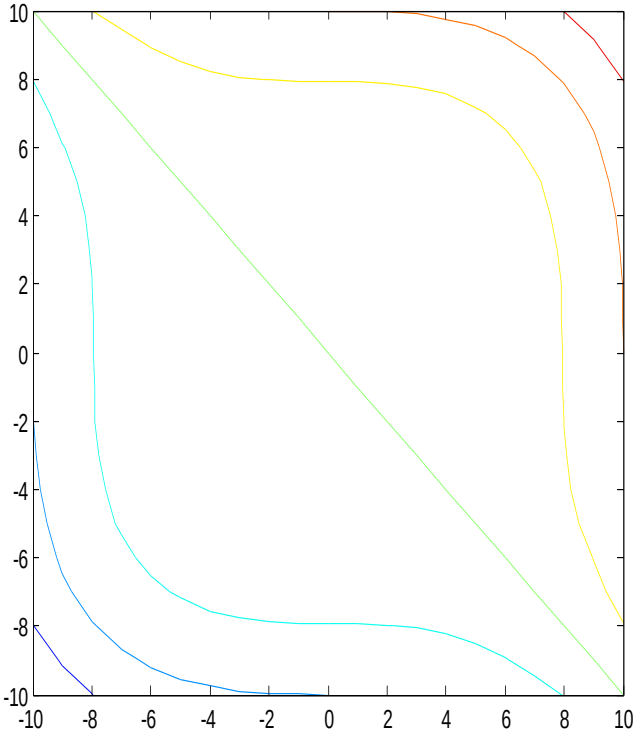



Gambar 5.22 Grafik surf

Plot kontur

Dari plot tiga dimensi yang dibuat, dari persamaan sebelumnya, yaitu : $y = x^3 + y^3$ , disamping dengan membuat plot titik dan permukaan, kita dapat membuat plot kontur.

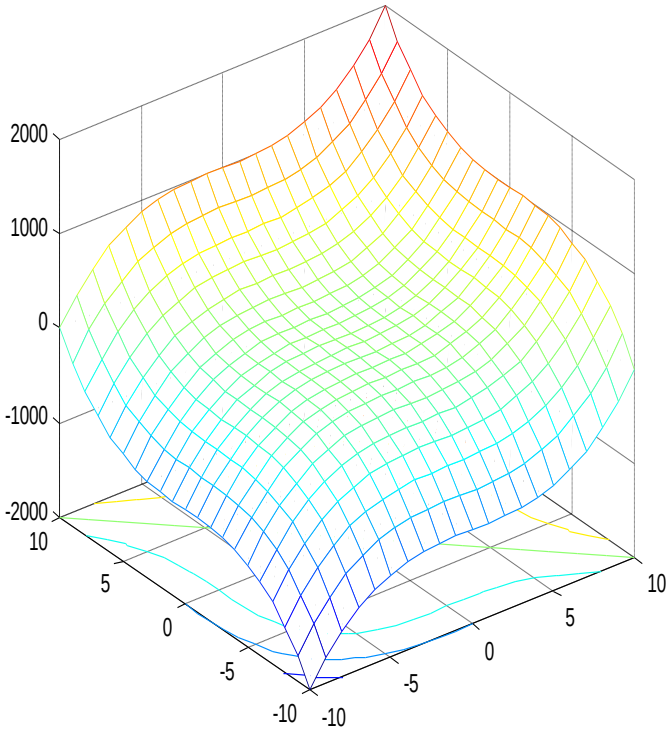
```
>> contour(X,Y,Z);
```




Gambar 5.23. Grafik Kontur

Untuk membandingkan dengan plot permukaan kontur dan perintah mesh maka dapat digunakan perintah meshc

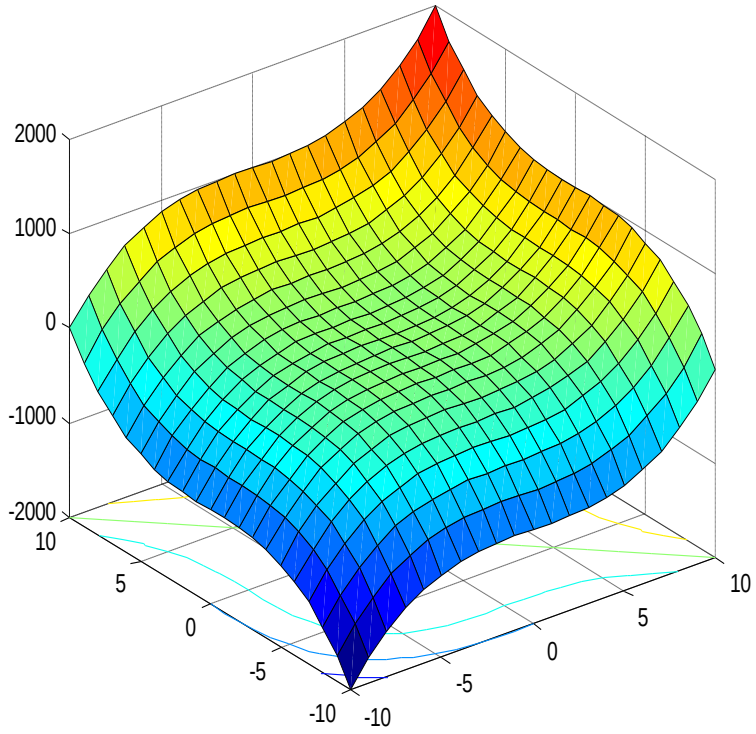
```
>> meshc(X,Y,Z);
```




Gambar 5.24 Meshc

Dengan plot permukaan dan plot kontur

```
>> surfc(X,Y,Z);
```




Gambar 5.25. Surfc

# BAB 6M-FILE

Tujuan instruksional umum

Mahasiswa mampu menggunakan M file editor sebagai tempat untuk membuat dan menuliskan kode-kode pemrograman (coding)

Tujuan instruksional khusus

Setelah mempelajari bab 6 Fungsi M-file, mahasiswa mampu mencapai kompetensi berikut:

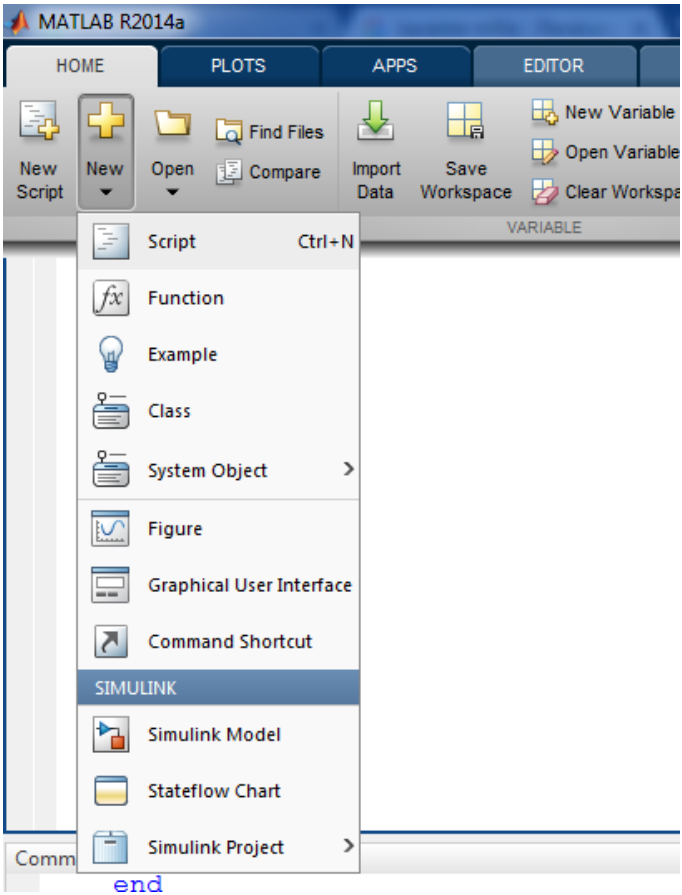
1. mahasiswa mampu membuat M-file sebagai skript program dan menjalankannya dalam command windows matlab
2. mahasiswa mampu membuat M-file sebagai fungsi dan melakukan running program

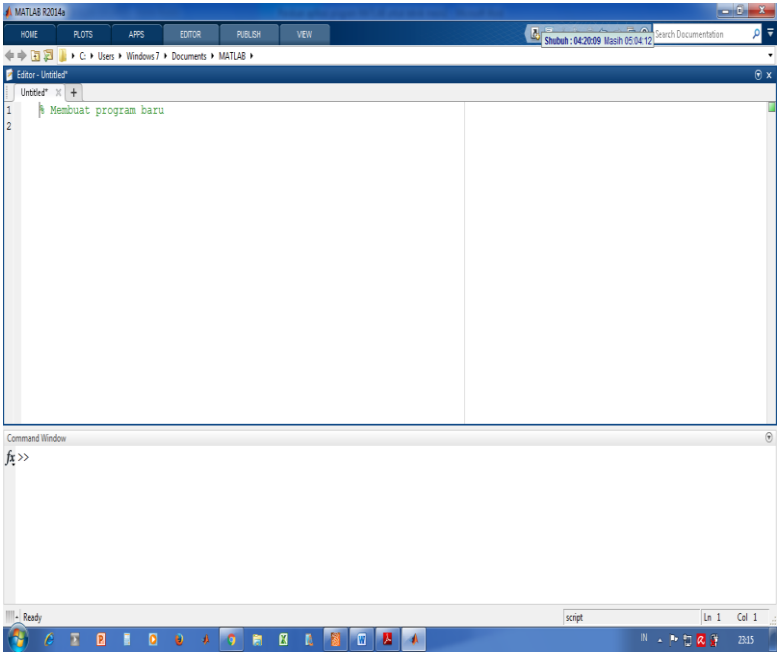
M-File dalam pemrograman matlab merupakan editor. Artinya, berfungsi untuk menuliskan fungsi dan perintah-perintah untuk dijalankan dalam satu sheet. Editor dalam matlab yang disebut sebagai M-File memiliki fungsi yang sama dengan command window. Yang membedakan adalah dengan M-File semua perintah yang telah ditulis tidak langsung dijalankan, tapi menunggu perintah eksekusi/running. Setelah semua perintah selesai diketikkan, bisa langsung dijalankan (di – run) atau disimpan terlebih dahulu, baru kemudian dijalankan. Kelebihan dari M-File ini adalah semua perintah yang ditulis bisa diedit kapan saja tanpa harus menulis ulang. Berbeda dengan penggunaan command window, yang bersifat sekali ditulis maka saat itu pula dieksekusi/dijalankan. Dan kelemahan dari command window adalah kita harus menulis ulang setiap command yang telah

dijalankan jika ingin menjalankan kembali. Hal ini akan menimbulkan masalah jika dibutuhkan scripct yang panjang dan rumit. Maka lebih menguntungkan dan lebih mudah jika dipergunakan editor/ M-file. Program yang telah diketik di M-File akan disimpan dalam bentuk file dengan ekstensi *.m. file ini dapat dijalankan dengan memanggil file tersebut dengan menggunakan command windows. Setiap perintah yang dijalankan di Command windows , akan disimpan di dalam command histroy, namun tidak dapat dijalankan melainkan ditulis kembali di dalam command windows.

## 6.1. Membuat M-File

Untuk membuat M-file, langkah pertama adalah dengan membuka program Matlab kemudian pilih menu Home. Selanjutnya pilih new script atau tekan ctrl-N.


MATLAB R2014a
HOME
PLOTS
APPS
EDITOR
New Script
New
Open
Find Files
Compare
Import Data
Save Workspace
New Variable
Open Variable
Clear Workspa
VARIABLE
Script Ctrl+N
Function
Example
Class
System Object
Figure
Graphical User Interface
Command Shortcut
SIMULINK
Simulink Model
Stateflow Chart
Simulink Project
Comm
end




Gambar 6.1. Dekstop matlab

Sebagai sebuah contoh, kita akan membuat sebuah program sederhana untuk menghitung luas sebuah persegi panjang.

Misal diketahui sebuah persegi panjang memiliki panjang 10 cm dan lebar 5 cm. Kita diminta menghitung berapakah luasnya, kemudian menghitung keliling dari segi empat tersebut, serta terakhir kita diminta menghitung panjang diagonal yang membagi bangun persegi empat tersebut menjadi dua.

Sebelum dikerjakan, perlu diketahui bahwa M-file dalam matlab bisa dikerjakan dengan editor teks lain yang bukan dari program bawaan matlab. Misalnya kita bisa menuliskan M-file dalam program editor Notepad, wordpad, dll. Yang harus diperhatikan adalah pada saat menyimpan file menggunakan eksistensi *.m.

Selanjutnya kita kerjakan soal segi empat diatas:

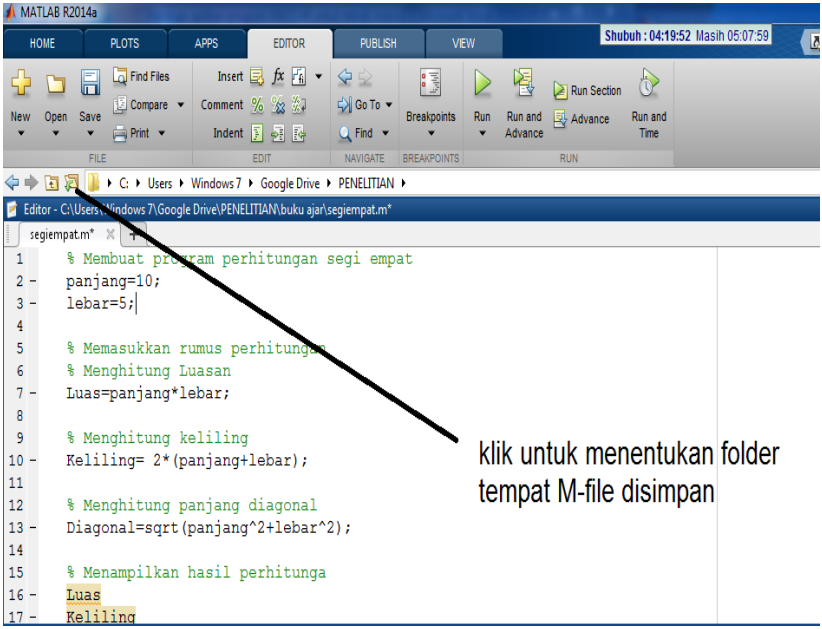
Langkah awal bukalah atau buatlah sebuat M-file baru, sebagaimana dengan contoh sebelumnya. Setelah tampil jendela M-file, selanjutnya kita membuat variabel yang dibutuhkan . variabel yang dibuat adalah panjang, lebar, luas , keliling dan diagonal. Berikut contoh penulisan skrip programnya.

```
% Membuat program perhitungan segi empat
panjang=10;
lebar=5;

% Memasukkan rumus perhitungan
% Menghitung Luasan
Luas=panjang*lebar;

% Menghitung keliling
Keliling= 2*(panjang+lebar);

% Menghitung panjang diagonal
Diagonal=sqrt(panjang^2+lebar^2);

% Menampilkan hasil perhitunga
Luas
Keliling
```

Langkah selanjutnya adalah menyimpan file tersebut. Dengan cara memililih menu editor kemudian mengeklik logo save (gambar disket). Setelah klik logo save, makakita diminta meletakan file tersebut pada sebuah folder atau dokumen. Untuk memudahkan pencarian, gunakanlah nama file yang mudah dikenali dan mewakili isi dari file tersebut, serta tempat menyimpan yang mudah diingat dan dicari.

Untuk memudahkan M-file tersebut kita simpan dengan nama segiempat. Sebagai catatan dalam memberi nama, sebaiknya nama file yang diberikan tidak menggunakan spasi.

Langkah berikutnya adalah membuat atau melakukan pengaturan direktori. Direktori yang aktif disesuaikan dengan folder tempat menyimpan M-file, dengan cara menekan logo folder kemudian kita memilih folder sesuai tempat penyimpanan file. Pengaturan direktori




Gambar 6.2. Menentukan direktori

Setelah pengaturan direktori maka kita bisa memanggil M-file yang telah disimpan pada command window.

Kita panggil dengan nama file

>> segiempat

Maka akan tampil

Luas =

50

Keliling =

30

Diagonal =

11.1803

## 6.2. Membuat fungsi

Jika menginginkan sebuat script dijalankan secara berulang kali, tanpa mengubah algoritma yang dipakai, namun dengan inputan yang berbeda-beda, maka bisa digunakan fungsi. Membuat fungsi dengan menggunakan script

>> function(output)=nama_fungsi(input);

Dari penulisan fungsi diatas, setidaknya ada 3 hal yang harus didefinisikan, yaitu:

a. Input, merupakan inputan yang akan diisi oleh pengguna
b. Output, merupakan hasil perhitungan/eksekusi dari program yang dibuat
c. Nama_fungsi merupakan nama fungsi yang dibuat, setidaknya menyatakan identitas atau menjelaskan apa yang dibutuhkan /dibuat program.

Contoh sebuah program sederhana yang menggunakan fungsi, yaitu rumus bangun datar

```
% Membuat program perhitungan segi empat
dengan menggunakan fungsi

function[Luas,Keliling,Diagonal]=segiempa
t(panjang,lebar);

% Memasukkan rumus perhitungan
% Menghitung Luasan
Luas=panjang*lebar;

% Menghitung keliling
Keliling= 2*(panjang+lebar);

% Menghitung panjang diagonal
Diagonal=sqrt(panjang^2+lebar^2);

% Menampilkan hasil perhitungan
Luas
Keliling
Diagonal
```

Kemudian script diatas disimpan, misalnya dengan nama segi_empat. Maka kita dapat memanggil fungsi perhitungan segiempat serta memasukkan variabel yang diinginkan. Misalkan diketahui panjang = 3 m dan lebar = 5 m, keliling, luas dan panjang diagonalnya dapat langsung diketahui dengan memanggil program segi_tiga.

```
>>
[luas,keliling,diagonal]=segi_empat(5,3)
 Maka akan didapat hasil eksekusi program
luas =
```

```
    15
keliling =
    16
diagonal =
    5.8310
```

Jika panjanga dan lebarnya adalah 10 m dan 5 m maka tinggal mengubah variabel input panjang dan lebar dalam fungsi tersebut.

## 6.3. Menggunakan display dan input

Untuk program yang membutuhkan inputan data untuk dijalankan serta menampilkan hasil perhitungan, dapat menggunakan perintah input (untuk masukan input data) dan display (untuk menampilkan hasil eksekusi program). Misal dari program sederhana tentang menghitung segiempat diatas dapat juga dengan memanfaatkan perintah input.

```
% Membuat program perhitungan segi
empatMemasukkan rumus perhitungan

% Menampilkan judul
clc;
disp('Program Menghitung Segiempat');
disp('****************************');

% Meminta input data
p=input('panjang =');
l=input('lebar=');

% Menghitung Luas
Luas=p*l;
```

```
% Menghitung keliling
Keliling= 2*(p+l);

% Menghitung panjang diagonal
Diagonal=sqrt(p^2+l^2);

% Menampilkan hasil perhitungan
Luas
Keliling
Diagonal
Maka kita jalankan programnya

Program Menghitung Segiempat
****************************
panjang =30
lebar=15
Luas =
   450
Keliling =
    90
Diagonal =
   33.5410
```

## 6.4. Statement

Sebagai bahasa program, penggunaan statement sebagai mana dalam bahasa pemrograman yang lain dapat digunakan untuk melakukan kontrol terhadap sebuah statement.

Statement if..else..end

Statement ini memiliki sifat sebagai berikut:

```
If kondisi
        Perintah dijalankan jika kondisi terpenuhi
end
```

statement if else end

if kondisi

perintah dijalankan jika kondisi terpenuhi

else

dijalankan jika kondisi tidak terpenuhi

end

statement if..elseif..else..end

if kondisi

perintah dijalankanjika kondisi terpenuhi

elseif kondisi 2

dijalankan jika kondisi 2 terpenuhi

... dst

else

Dijalankan jika kondisi manapun tidak dipenuhi

End

Statement for ...end

For ..end digunakan dalam perhitungan loop (berulang).

for variabel

Perintah untuk dijalankan

End

Contoh

```
for i=0:3;
    x=2*i
    x
end
```

maka hasilnya

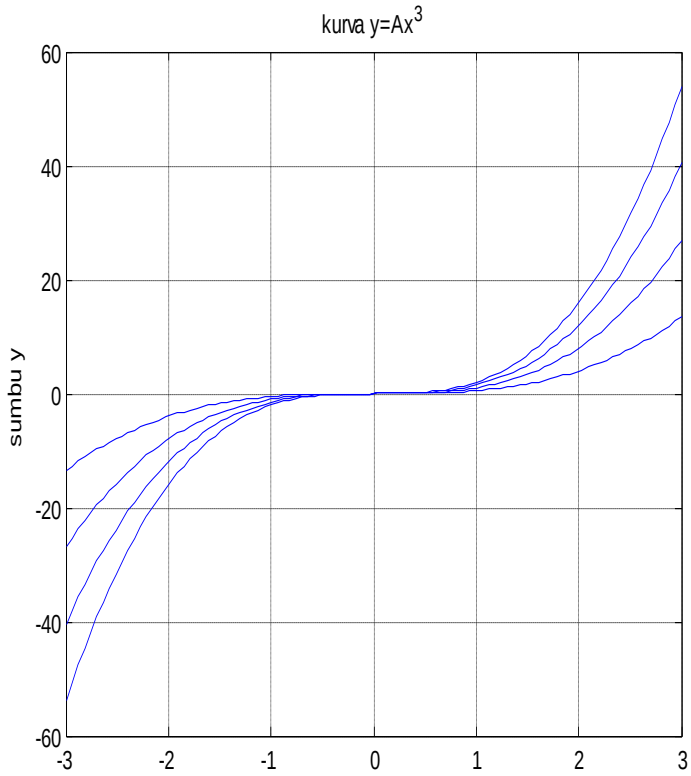
```
x=
   0
x =
   2
x =
   4
x =
```

6

Untuk contoh lain, misalkan untuk membuat plot grafik dari persamaan $y= Ax^3$, dimana A memiliki nilai antara 0.5, 1, 1.5, 2. Maka dapat dibuat programnya dengan menggunakan statement for end.

```
x=linspace(-3,3,100);
A=0.5:0.5:2;
for i=1:length(A)
    y=A(i)*x.^3;
    plot(x,y);
    hold on;
    grid on
    title('kurva y=Ax^3')
    xlabel('sumbu x');
    ylabel('sumbu y');
end
```

maka akan didapat plot grafik dari fungsi y=Ax^3 sebagai berikut:



Gambar 6.3. Penggunaan for end

### 6.4.1. Pemakaian statement for end untuk operasi penjumlahan matrik

Dalam operasi penjumlahan matrik, dapat digunakan statement for end dalam menyelesaikan penjumlahan maupun pengurangan dan perkalian matrik. Walaupun dengan instant kita dapat mencari hasil operasi matrik, namun di sini kita akan belajar bagaimana pemakaian for end dalam membantu memudahkan dan menyederhanakan sebuah algoritma.

Dalam penjumlahan matrik kita dapat menggunakan command prompt:

Misal kita memiliki matrik A dan matrik B

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

>> B=[4 2; 1 7; 4 3];
```

Maka matrik A dan B adalah

```
A =

    2    3

    4    3

    2    5


>> B

B =

    4    2

    1    7

    4    3
```

Jika kita mencari hasil penjumlahan dari matrik A dan matrik B maka dengan mudah kita dapat langsung mendapatkannya.

```
>> A+B

ans =
```

6 5

5 10

6 8

Namun berikutya kita akan mencoba menggunakan statement for end untuk mendapatkan penjumlahan matrik A dan B. Dengan menggunakan M – file kita akan mencoba menyusun algoritma sederhana untuk mendapatkan penjumlahan matrik A dan B.

Langkah awal kita buat sebuah skrip untuk mendapatkan penjumlahan matrik A dan B. Dengan menggunakan M-File kita menuliskan penjumlahan matrik dengan mendefinisikannya sesuai dengan indeknya.

```
clear all
 clc

 A=[2 3;4 3;2 5]; % mendefinisikan matrik
A

 B=[4 2;1 7;4 3]; % mendefisinikan matrik
B

% ---proses penjumlahan matrik----
 C(1,1)=A(1,1)+B(1,1);
 C(1,2)=A(1,2)+B(1,2);
 C(2,1)=A(2,1)+B(2,1);
 C(2,2)=A(2,2)+B(2,2);
 C(3,1)=A(3,1)+B(3,1);
 C(3,2)=A(3,2)+B(3,2);

% ---menampilkan matrik A, B dan C----
```

```
A
B
C
```

Jika skript tersebut dijalankan maka akan kita dapatkan

A =

2 3

4 3

2 5

B =

4 2

1 7

4 3

C =

6 5

5 10

6 8

Maka kita dapatkan hasilnya sama persis dengan pengerjaan awal dengan menggunakan penjumlahan matrik A dan B secara langsung. Artinya algoritma yang kita buat adalah sesuai atau benar (tidak ada kesalahan). Dalam script tersebut kita telah menjumlahkan matrik A dan B sesuai dengan indeksnya. Artinya untuk elemen $A_{1x1}$ kita jumlahkan dengan elemen matrik $B_{1x1}$.

Demikian juga untuk elemen matrik $A_{1x2}$ dijumlahkan dengan elemen matrik $B_{1x2.}$. demikian seterusnya sampai dengan elemen matrik $A_{3x2}$ dan $B_{3x2}$

Selanjutnya kita menggunakan statement for end untuk menjumlahkan matrik A dan B di atas. Jika diketahui Matrik $A_{ixj}$ artinya Matrik A dengan indek baris i dan indeks kolom j. Demikian juga untuk matrik $B_{ixj}$ memiliki indeks matrik baris i dan kolom j. Syarat penjumlahan matrik adalah jika kedua matrik yang dijumlahkan memiliki ukuran baris dan kolom yang sama. Matrik A dan B memiliki jumlah baris dan kolom yang sama, maka matrik A dan B memenuhi syarat untuk penjumlahan matrik. Indeks matrik A dan B berturut-turut adalah indeks baris i = 3 dan indeks kolom j = 2. Dalam script penjumlahan matrik kita dapat menyusunnya sebagai berikut:

Dari script tersebut kita mendapati bahwa indek kolom j baik matrik A dan B pada penjumlahan baris 1, indeks kolom j berubah dari kolom j=1 menjadi j=2. Demikian juga untuk penjumlahan elemen baris kedua, indeks kolom j=1 bergerak menjadi j=2. Demikian juga untuk baris ke tiga. Dengan memanfaat indeks j, maka kita dapat menuliskan pergerakan j dengan mendefinisikan j=1:2, artinya j bergerak mulai dari nilai j=1 sampai dengan j=2. Demikian juga dengan penjumlahan elemen baris kedua dan baris ketiga. Maka kita dapat menuliskan script algoritma dengan for end untuk nilai pergerakan indeks j sebagai berikut:

```
clear all
 clc

 A=[2 3;4 3;2 5]; % mendefinisikan matrik
A
```

```
 B=[4 2;1 7;4 3]; % mendefisinikan matrik
B

% ---proses penjumlahan matrik----
for j=1:2
        C(1,j)=A(1,j)+B(1,j);
end

for j=1:2
        C(2,j)=A(2,j)+B(2,j);
end

for j=1:2
        C(3,j)=A(3,j)+B(3,j);
end

% ---menampilkan matrik A, C dan D----
 A
 B
 C
```

Kemudian akan kita jalankan dengan run /menekan F-9 . caranya adalah dengan kita blok semua scriptnya (select all) selanjutnya klik kanan dan tekan evaluate selection /F9. Maka kita periksa hasilnya adalah sama dengan sebelumnya. Artinya program /algoritma yang dibuat tidak ada kesalahan.

Selanjutnya kita dapat memodifikasi lagi skript algoritmanya dengan menggunakan / memanfaatkan indeks matriknya. Kita dapat melihat bahwa indeks baris berubah/bergerak dari 1 sampai dengan 3. Jika sebelumnya kita hanya menggunakan indek kolom j maka sekarang kita akan memasukkan indeks baris i. Kita pisahkan masing-masing operasi baris dengan menambahkan pada indeks baris i. Program dapat dituliskan sebagai berikut:

```
clear all
 clc

 A=[2 3;4 3;2 5]; % mendefinisikan matrik
A

 B=[4 2;1 7;4 3]; % mendefisinikan matrik
B

% ---proses penjumlahan matrik----
    i=1;
for j=1:2
                C(i,j)=A(i,j)+B(i,j);
end

    i=2;
for j=1:2
                C(i,j)=A(i,j)+B(i,j);
end

    i=3;
for j=1:2
                C(i,j)=A(i,j)+B(i,j);
end

% ---menampilkan matrik A, C dan D----
 A
 B
 C
```

Kemudian hasilnya kita jalankan (run). Maka kita dapatkan hasil yang sama dengan algoritma sebelumnya. Artinya algoritma yang kita buat tidak ada kesalahan sama sekali.

Selanjutnya kita akan memodifikasi lagi algoritma yang telah dibuat supaya menjadi lebih sederhana. kita lihat bahwa indek baris i bergerak dari 1 sampai dengan 3 (kedua matrik memiliki 3 baris). Maka algoritmanya dapat kita tulis sebagai berikut:

```
clear all
clc

A=[2 3;4 3;2 5]; % mendefinisikan matrik
A

B=[4 2;1 7;4 3]; % mendefisinikan matrik
B

% ---proses penjumlahan matrik----
for i=1:3
for j=1:2
        C(i,j)=A(i,j)+B(i,j);
end
end

% ---menampilkan matrik A, C dan D----
 A
 B
 C
```

Kemudian kita jalankan. Maka kita dapatkan hasil running program memiliki nilai yang sama persis, artinya program/ algoritma yang kita buat tidak mengandung kesalahan. Dengan algoritma ini program kita menjadi lebih simple daripada sebelumnya. Dan kelebihan lain adalah jika kita memiliki penjumlahan matrik yang lain, dengan indeks matrik yang berbeda maka kita dapat dengan mudah menjalankannya hanya dengan merubah nilai indeks matrik i dan j. Contohnya adalah jika kita

menjumlahkan matrik X dan Y dengan hasil penjumlahan matrik Z maka :

```
clear all
 clc

 X=[3 8 5 4 ; 6 4 5 7;3 4 4 5 ]; %
inisialisasi matrik X

 Y=[9 5 3 3; 7 3 2 1; 3 1 2 3]; %
inisialisasi matrik Y

% ---proses penjumlahan matrik----
for i=1: 3
for j=1:4
                Z(i,j)=X(i,j)+Y(i,j);
end
end

% ---menampilkan matrik X, Y dan Z----
 X
 Y
 Z
```

Jika kita jalankan maka akan kita dapatkan

X =

```
   3   8   5   4

   6   4   5   7

   3   4   4   5
```

Y =

9 5 3 3

7 3 2 1

3 1 2 3

Z =

12 13 8 7

13 7 7 8

6 5 6 8

Demikian seterusnya sehingga kita dapat mendapatkan hasil penjumlahan matrik berapapun indeks baris maupun kolomnya.

### 6.4.2. Statement for end untuk operasi perkalian matrik

Untuk menyelesaikan perkalian matrik, for end dapat membantu penyusunan algoritma yang lebih sederhana dan mudah. Aturan perkalian matrik tidak sama dengan aturan penjumlahan matrik. Penjumlahan matrik merupakan operasi elemen per elemen, artinya elemen matrik dioperasikan/dijumlahkan dengan indeks yang sama dari matrik penjumlahnya. Perkalian matrik adalah operasi perkalian baris dengan kolom dari matrik pengalinya. Sebagai contoh, misalkan kita memiliki dua buah matrik, A dan B, maka hasil perkalian dari A.B adalah perkalian baris matrik A dengan Kolom matrik B. Berikut aksioma perkalian matrik:

$X_{3x3}=A_{3x2}. B_{2x3}$

Jika diketahui $X_{3x3}$ adalah matrik dengan ordo 3x3 merupakan hasil perkalian dari matrik $A_{3x2}$ dan matrik $B_{2x3}$

Jika diketahui matrik A dan matrik B

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

>> B=[5 3 1; 5 7 3];
```

Sehingga Matrik A dan B

```
A =

    2    4

    3    5

    6    8

>> B

B =

    5    3    1

    5    7    3
```

Maka X sebagai hasil perkalian A.B

```
>> X=A*B

X =

   30   34   14

   40   44   18
```

```
  70   74   30
```

Hasil Perkalian X adalah dari operasi perkalian baris dan kolom dari matrik A dan B.

Untuk perkalian matrik dengan menggunakan statemen for end, maka dapat kita lakukan sebagaimana dengan operasi penjumlahan matrik sebelumnya. Dengan membuka M –File, maka kita buat script algoritma perkalian matrik A dan B. (Suparno, 2011)

```
clear all
clc
 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B

% ---proses perkalian matrik----

E(1,1)=A(1,1)*B(1,1)+A(1,2)*B(2,1)+A(1,3)
*B(3,1);

E(1,2)=A(1,1)*B(1,2)+A(1,2)*B(2,2)+A(1,3)
*B(3,2);

E(2,1)=A(2,1)*B(1,1)+A(2,2)*B(2,1)+A(2,3)
*B(3,1);

E(2,2)=A(2,1)*B(1,2)+A(2,2)*B(2,2)+A(2,3)
*B(3,2);

% ---menampilkan matrik A, B dan E----
 A
 B
 E
```

Untuk melakukan pengecekan ketepatan hitungan maka jika di run hasilnya harus sama dengan perkalian matrik A dan matrik B

A =

```
   2   5   8
   4   2   5
```

B =

```
   2   5
   3   6
   4   2
```

E =

```
  51  56
  34  42
```

Jika elemen matrik (1,1) dihitung tiga kali

```
clear all
clc

 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B

% ---proses perkalian matrik----
% ---E(1,1) dihitung 3 kali
 E(1,1)=A(1,1)*B(1,1);
 E(1,1)=E(1,1)+A(1,2)*B(2,1);
 E(1,1)=E(1,1)+A(1,3)*B(3,1);
```

```
% ---E(1,2); E(2,1); dan E(2,2) masih
seperti semula

E(1,2)=A(1,1)*B(1,2)+A(1,2)*B(2,2)+A(1,3)
*B(3,2);

E(2,1)=A(2,1)*B(1,1)+A(2,2)*B(2,1)+A(2,3)
*B(3,1);

E(2,2)=A(2,1)*B(1,2)+A(2,2)*B(2,2)+A(2,3)
*B(3,2);

% ---menampilkan matrik A, B dan E----
 A
 B
 E
```

Jika dijalankan masih tetap hasilnya sama

```
clear all
 clc

 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B

% ---proses perkalian matrik----
% ---E(1,1) dihitung 3 kali
 E(1,1)=0;
 E(1,1)=E(1,1)+A(1,1)*B(1,1);
 E(1,1)=E(1,1)+A(1,2)*B(2,1);
 E(1,1)=E(1,1)+A(1,3)*B(3,1);
```

```
% ---E(1,2); E(2,1); dan E(2,2) masih
seperti semula

E(1,2)=A(1,1)*B(1,2)+A(1,2)*B(2,2)+A(1,3)
*B(3,2);

E(2,1)=A(2,1)*B(1,1)+A(2,2)*B(2,1)+A(2,3)
*B(3,1);

E(2,2)=A(2,1)*B(1,2)+A(2,2)*B(2,2)+A(2,3)
*B(3,2);

% ---menampilkan matrik A, B dan E----
 A
 B
 E
```

Selanjutnya kita akan menggunakan statemen for end untuk elemen perkalian (1,1)

```
clear all
 clc

 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B

% ---proses perkalian matrik----
 E(1,1)=0;
for k=1:3 % k bergerak dari 1 sampai 3
           E(1,1)=E(1,1)+A(1,k)*B(k,1);
end
```

```
% ---E(1,2); E(2,1); dan E(2,2) masih
seperti semula

E(1,2)=A(1,1)*B(1,2)+A(1,2)*B(2,2)+A(1,3)
*B(3,2);

E(2,1)=A(2,1)*B(1,1)+A(2,2)*B(2,1)+A(2,3)
*B(3,1);

E(2,2)=A(2,1)*B(1,2)+A(2,2)*B(2,2)+A(2,3)
*B(3,2);

% ---menampilkan matrik A, B dan E----
 A
 B
 E
```

Jika di run hasilnya benar sesauai dengan sebelumnya maka algoritma yang dibuat bisa diterima.

Kemudian dibuat masing-masing elemen baik E(1,2), E(2,1) dan E(2,2) disusun algoritma yang serupa. Sehingga kita dapatkan :

```
clear all
 clc

 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B
% ---proses perkalian matrik----
 E(1,1)=0;
for k=1:3
          E(1,1)=E(1,1)+A(1,k)*B(k,1);
end
```

```
 E(1,2)=0;
for k=1:3
            E(1,2)=E(1,2)+A(1,k)*B(k,2);
end

 E(2,1)=0;
for k=1:3
          E(2,1)=E(2,1)+A(2,k)*B(k,1);
end

 E(2,2)=0;
for k=1:3
          E(2,2)=E(2,2)+A(2,k)*B(k,2);
end

% ---menampilkan matrik A, B dan E----
 A
 B
 E
```

Kemudian di jalankan. Ternyata hasilnya sama, artinya tidak ada kesalahan dalam penulisan algoritmanya. Berikutnya dicoba modifikasi lagi dari for end yang telah dibuat untuk elemen E(1,1)

```
clear all
 clc

 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B
```

```
% ---proses perkalian matrik----
for i=1:2 % i bergerak dari 1 sampai 2
for j=1:2 % j bergerak dari 1 sampai 2
            E(i,j)=0;
end
end

for k=1:3
        E(1,1)=E(1,1)+A(1,k)*B(k,1);
end

for k=1:3
        E(1,2)=E(1,2)+A(1,k)*B(k,2);
end

for k=1:3
        E(2,1)=E(2,1)+A(2,k)*B(k,1);
end

for k=1:3
        E(2,2)=E(2,2)+A(2,k)*B(k,2);
end

% menampilkan matrik A, B dan E
 A
 B
 E
```

Ketika dijalankan program di atas ternyata hasilnya sama. Jadi algoritma yang dibuat tidak ada kesalahan.Kemudian dilakukan lagi modifikasi algoritma for end

```
clear all
 clc
```

```
 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B

% ---proses perkalian matrik----
for i=1:2                          % i
bergerak dari 1 sampai 2
for j=1:2                 % j bergerak dari
1 sampai 2
                  E(i,j)=0;
end
end

 j=1;
for k=1:3
        E(1,j)=E(1,j)+A(1,k)*B(k,j);
end

 j=2;
for k=1:3
        E(1,j)=E(1,j)+A(1,k)*B(k,j);
end

for k=1:3
        E(2,1)=E(2,1)+A(2,k)*B(k,1);
end

for k=1:3
        E(2,2)=E(2,2)+A(2,k)*B(k,2);
end

% ---menampilkan matrik A, B dan E----
 A
 B
```

```
 E
```

Dari modifikasi ini ketika dijalankan , hasilnya tetap sama. Artinya algoritma yang disusun adalah benar. Berikutnya karena j bergerak dari 1 ke 2 maka dapat dimodifikasi.

```
clear all
 clc

 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B

% ---proses perkalian matrik----
for i=1:2                          % i
bergerak dari 1 sampai 2
for j=1:2                  % j bergerak
dari 1 sampai 2
                E(i,j)=0;
end
end

for j=1:2
for k=1:3

E(1,j)=E(1,j)+A(1,k)*B(k,j);
end
end

for k=1:3
        E(2,1)=E(2,1)+A(2,k)*B(k,1);
end
```

```
for k=1:3
        E(2,2)=E(2,2)+A(2,k)*B(k,2);
end

% ---menampilkan matrik A, B dan E----
 A
 B
 E
```

Setelah dijalankan ternyata hasilnya tetap sama.

Berikutnya kita lihat untuk elemen E(2,1) dan E(2,2), karena sama maka kita dapat menggunakan algroritma for end yang sama.

```
clear all
 clc

 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B

% ---proses perkalian matrik----
for i=1:2 % i bergerak dari 1 sampai 2
for j=1:2 % j bergerak dari 1 sampai 2
            E(i,j)=0;
end
end

for j=1:2
for k=1:3
            E(1,j)=E(1,j)+A(1,k)*B(k,j);
end
```

```
end

for j=1:2
for k=1:3

E(2,j)=E(2,j)+A(2,k)*B(k,j);
end
end

% ---menampilkan matrik A, B dan E----
 A
 B
 E
```

Jika dijalankan , didapatkan hasil yang sama. Berikutnya dapat dimodifikasi lagi :

```
clear all
 clc

 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B

% ---proses perkalian matrik----
for i=1:2 % i bergerak dari 1 sampai 2
for j=1:2 % j bergerak dari 1 sampai 2
                E(i,j)=0;
end
end

 i=1;
for j=1:2
for k=1:3
```

```
E(i,j)=E(i,j)+A(i,k)*B(k,j);
end
end

 i=2;
for j=1:2
for k=1:3

E(i,j)=E(i,j)+A(i,k)*B(k,j);
end
end

% ---menampilkan matrik A, B dan E----
 A
 B
 E
```

Ketika dijalankan,masih didapatkan hasil yang sama. Kemudian algoritma dapat dimodifikasi lagi dengan algoritma berikut:

```
clear all
clc
 A = [2 5 8; 4 2 5]; % inisialisasi
matrik A
 B = [2 5; 3 6 ;4 2]; % inisialisasi
matrik B

% ---proses perkalian matrik----
for i=1:2
for j=1:2
                E(i,j)=0;
end
```

```
end

for i=1:2
for j=1:2
for k=1:3

E(i,j)=E(i,j)+A(i,k)*B(k,j);
end
end
end

% ---menampilkan matrik A, B dan E----
 A
 B
 E
```

Ketika dijalankan kita dapatkan hasil yang sama. Namun dengan algoritma yang terakhir ini algoritma menjadi lebih sederhana yang simpel.

### 6.4.3. Membuat fungsi

Pada algoritma penjumlahan, sdh didapatkan algoritma untuk mencari penyelesaian dari penjumlahan matrik. Dari pelajaran sebelumnya kita dapatkan algoritma penyelesaiannya:

```
clear all
 clc

 A=[4 3 8 6; 5 1 2 3; 6 7 9 1;8 3 4 5]; % inisialisasi matrik A
 C=[2 6 7 2; 9 1 3 8; 5 8 4 7;3  9 4 5]; % inisialisasi matrik B

% ---proses penjumlahan matrik----
```

```
for i=1:4
for j=1:4
            D(i,j)=A(i,j)+C(i,j);
end
end

% ---menampilkan matrik A, C dan D----
 A
 C
 D
```

Selanjutnya kita akan menggunakan fungsi, sehingga dapat menjadikan program lebih baik lagi. Kita akan menambahkan pembuatan variabel dimensi (untuk menyimpan/mendefinisikan ukuran matrik). Sebagai berikut:

```
clear all
 clc

 A=[4 3 8; 5  2 3;  7 9 1]; %
inisialisasi matrik A
 C=[2 6 7 ; 9  3 8;  8 4 7]; %
inisialisasi matrik B

% ---proses penjumlahan matrik----
 dim=size(A);
 n=dim(1);
 m=dim(2);
for i=1:n
for j=1:m
                    D(i,j)=A(i,j)+C(i,j);
end
end

% ---menampilkan matrik A, C dan D----
```

```
 A
 C
 D
```

Berikutnya kita membuat sebuah fungsi :

```
function D=jumlah(A,C)

dim=size(A);
 n=dim(1);
 m=dim(2);
for i=1:n
for j=1:m
                        D(i,j)=A(i,j)+C(i,j);
end
end
```

Fungsi tersebut kita simpan dengan nama jumlah.Selanjutnya fungsi jumlah tersebut kita panggil dalam skrip penjumlahan matriksebelumnya.

```
clear all
 clc

 A=[3 8 5; 6 4 7]; % inisialisasi matrik
A
 C=[9 5 3; 7 2 1]; % inisialisasi matrik
B

% ---proses penjumlahan matrik----
 D=jumlah(A,C)

% ---menampilkan matrik A, C dan D----

 A
```

```
C
D
```

Maka berikutnya kita sudah memiliki algoritma penjumlahan matrik. Hasil coding yang dibuat menjadi lebih sederhana.  dan kita pun dapat menginputkan matrik dengan ukuran yang lain, tanpa perlu merubah indeks i dan j.

# BAB 7 PERSAMAAN DIFERENSIAL

Matlab dapat melakukan operasi kalkulus melalui ekspresi yang telah disediakan, seperti pada perintah berikut

```
syms x

f=sin(x^2)
```

yang menghasilkan output sebagai berikut

```
f =

sin(x^2)
```

ekspresi tersebut dapat didifferensiasikan ketika menggunakan perintah ***diff***

```
diff(f,x)

ans =

2*x*cos(x^2)
```

Teknik yang sama juga dapat diimplementasikan untuk mengoperasikan dua atau lebih variable, seperti pada contoh berikut

```
syms x y

q=x^2*y^3*exp(x)

q =
```

x^2*y^3*exp(x)

pretty(q)

 2  3

 x  y  exp(x)

diff(q,y)

ans =

3*x^2*y^2*exp(x)

Dengan demikian MATLAB dapat menghitung derivatif parsial semudah derivatif biasa. Salah satu penggunaan kemampuan ini untuk menguji apakah suatu fungsi tertentu adalah solusi dari suatu persamaan diferensial tertentu.Misalnya, ketika ingin memeriksa apakah fungsi u (t) = $e^{at}$adalah solusi dari ODE

$$\frac{du}{dt} - au = 0$$

Didefinisikan

syms a t

u=exp(a*t)

u =

exp(a*t)

kemudian dapat dihitung sisi kiri persamaan diferensial, dan melihat apakah dia sama dengan sisi kanan (nol):

diff(u,t)-a*u

ans =

0

Dengan demikian fungsi yang diberikan adalah sebuah solusi. Apakah fungsi v (t) = pada solusi yang lain? Dapat diperiksa sebagai berikut**:**

**v=a*t**

**v =**

**a*t**

**diff(v,t)-a*v**

**ans =**

**- t*a^2 + a**

Karena hasilnya tidak nol, fungsinya bukan solusi.Tidaklah sulit untuk memeriksa apakah suatu fungsi dari beberapa variabel adalah solusi dari PDE. Sebagai contoh w (x, y) = sin (πx) + sin (πy) solusi dari persamaan diferensial.

$$\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = 0?$$

Seperti pada contoh sebelumnya pertanyaan ini dapat dijawab dengan mendefinisikan fungsinya dan menggantinya menjadi persamaan diferensial.

```
syms x y

w=sin(pi*x)+sin(pi*y)

w =

sin(pi*x) + sin(pi*y)

diff(w,x,2)+diff(w,y,2)

ans =

- pi^2*sin(pi*x) - pi^2*sin(pi*y)

simplify(ans)

ans =

-pi^2*(sin(pi*x) + sin(pi*y))
```

Karena hasilnya tidak nol, fungsi w bukan merupakan solusi dari PDE.

Contoh di atas menunjukkan bagaimana menghitung turunan yang lebih tinggi dari sebuah ekspresi. Misalnya, fungsi tersebut derivatif kelima dari *w* yang berhubungan dengan x

```
diff(w,x,5)

ans =

pi^5*cos(pi*x)
```

Untuk menghitung turunan parsial campuran, harus mengulangi perintah *diff*. Berikut adalah parsial campuranturunan w (x, y) = $x^2 + xy^2$ yang berhubungan dengan x dan y:

```
syms x y

w=x^2*exp(y)+x*y^2

w =

x^2*exp(y) + x*y^2

diff(diff(w,x),y)

ans =

2*y + 2*x*exp(y)
```

Sebagai ganti fungsi diatas, dapat digunakan juga fungsi berdasarkan ekspresi berikut:

```
clear

syms a x

f=@(x)exp(a*x)

f =

@(x)exp(a*x)
```

# DAFTAR ISI

# DAFTAR PUSTAKA

**Ferreira, A.J.M. 2009**`. MATLAB Codes for Finite Element Analysis. Porto, Portugal : Springer, 2009`.

**Suparno, Supriyanto. 2011.***Komputasi untuk Sains dan Teknik Menggunakan Matlab.* Jakarta : Departemen Fisika-FMIPA Universitas Indonesia, 2011.

**—. 2011.***Komputasi untuk Sains dan Teknik Menggunakan Matlab.* Jakarta : Departemen Fisika FMIPA Universitas Indonesia, 2011.

**Teknologi Industri Pertanian Fakultas Pertanian. 2014.***Petunjuk Praktikum Dasar Pemrograman.* Bangkalan, Madura : Universitas Trunojoyo, 2014.

**Widiarsono, Teguh. 2005.***Tutorial Praktis Belajar MATLAB.* Jakarta : s.n., 2005.

# Kata Pengantar

Alhamdulillah, puji syukur kehadirat Allah SWT, atas limpahan rahmat dan hidayahNya, penulis dapat memulai menyusun panduan aplikasi Matlab khususnya untuk aplikasi teknik mesin. Mata kuliah algoritma pemrograman untuk teknik mesin memang memiliki porsi yang sedikit dalam struktur kurikulum. Hal ini disebabkan salah satunya adalah begitu luasnya kajian dalam bidang teknik mesin, atau lebih dikenal dengan istilah mechanical engineering. Namun demikian, kehadiran mata kuliah algoritma pemrograman, tetap diperlukan, mengingat untuk dapat menyelesaikan permasalahan teknik dalam mekanika, diperlukan suatu piranti yang mumpuni dan akurat. Kehadiran komputer sebagai alat hitung tidak terpisahkan dalam perkembangan pengetahuan dan teknologi. Pemrograman Matlab, atau dikenal dengan Matrik Laboratory, sudah dikenal memiliki kemampuan dalam perhitungan matematik yang tidak diragukan dan luas dipergunakan dalam penelitian maupun industri. Untuk dapat menjembatani mahasiswa dalam penguasaan program Matlab untuk aplikasi dalam teknik mesin, dibutuhkan metoda dan panduan yang memudahkan untuk diaplikasikan dalam menyelesaikan masalah dan melakukan perhitungan. Kehadiran modul ajar ini diharapkan dapat memberikan panduan sederhana dan mudah dipahami serta mudah diaplikasikan dengan konsep-konsep dasar penyelesaian persamaan matematis. Modul ini memberikan peletakan dasar pemahaman dalam penguasaan program matlab. Untuk dapat menguasai lebih jauh dalam aplikasi program untuk masalah yang lebih kompleks, dibutuhkan

kajian yang lebih mendalam dengan referesensi yang spesifik dan relevan. Pada akhirnya penulis berharap modul yang akan disusun ini mampu memberikan pengantar dasar penguasaan dan pemahaman bagi mahasiswa, terutama mereka yang memulai dari nol untuk pembelajaran algoritma dan pemrograman.

Sidoarjo,
30 September
2017

Penulis

Edi
Widodo

Bab 1 Pendahuluan Pengenalan Matlab

Tujuan instruksional umum

Mahasiswa dapat memahami dan mengenal karakter pemrograman Matlab

Tujuan instruksional khusus

Setelah mempelajari bab 1, diharapkan mahasiswa dapat mencapai kompetensi berikut:

1. Memahami cakupan algoritma pemrograman, meliputi pengertian, contoh aplikasi, tujuan dari mempelajari algoritma dan memberi contoh sederhana dari sebuah algoritma
2. Mampu menjelaskan Pemrograman Matlab sebagai sebuah aplikasi untuk membuat algoritma.
3. Mampu melakukan instalasi salah satu bahasa pemrograman
4. Mampu mengoperasikan matlab.
5. Mampu mengaplikasikan bahasa matlab untuk menyelesaikan permasalahan dalam bidang ilmu teknik mesin

Bab 2. Operasi Dasar

Tujuan instruksional umum

Mahasiswa mampu memahami perintah dasar mengoperasikan Matlab

Tujuan instruksional khusus

Setelah mempelajari bab 2 Operasi dasar pemrograman matlab, mahasiswa mampu mencapai kompetensi berikut:

1. Mahasiswa mampu menggunakan command windows Matlab
2. Mahasiswa mampu mengoperasikan matlab sebagai kalkulator sederhanan untuk menyelesaikan persamaan matematika,
3. Mahasiswa mampu menciptkan variabel untuk menyelesaikan operasi perhitungan
4. Mahasiswa mampu menyelesaikan fungsi matematika

Bab 3 Pengenalan Awal

Tujuan instruksional umum

Mahasiswa mampu mengoperasikan Matlab untuk menyelesaikan vektor , fungsi dan polinomial

Tujuan instruksional khusus

Setelah mempelajari bab 3, mahasiswa mampu mencapai kompetensi berikut:

1. Mahasiswa mampu membuat fungsivektor
2. Mahasiswa mampu mendefinisikan fungsi polinomial dan menghitungnya

Bab 4 Matrik

Tujuan instruksional umum

Mahasiswa mampu mengoperasikan Matlab untuk menyelesaikan matrik dan manipulasi matrik

Tujuan instruksional khusus

Setelah mempelajari bab 3 Matrik, mahasiswa mampu mencapai kompetensi berikut:

1. Mahasiswa mampu membuat fungsi skalar, vektor dan matrik
2. Mahasiswa mampu mendefinisikan ukuran matrik, mengenali dan membuat matrik khusus
3. Mahasiswa mampu memanipulasi indeks matriks, membuat deret dan membentuk ulang matrik
4. Mahasiswa mampu melakukan penjumlahan , pengurangan dan perkalian matrik
5. Mahasiswa mampu menyelesaikan persamaan linear dengan menggunakan matriks
6. Mahasiswa mampu melakukan transposisi, operasi elemen per elemen, dan menerapkan operasi matrik untuk membuat sebuah program algoritma sederhana

Bab 5 Membuat Plot grafik 2D dan 3D

Tujuan instruksional umum

Mahasiswa mampu mengoperasikan Matlab membuat plot grafik

Tujuan instruksional khusus

Setelah mempelajari bab 5 Membuat Grafik 2D dan 3D, mahasiswa mampu mencapai kompetensi berikut:

1. Mahasiswa mampu membuat plot grafik 2 dimensi untuk sebuah persamaan
2. Mahasiswa mampu membuat grafik 3 dimensi meliputi plot garis, plot permukaan, dan plot kontur

Bab 6 M-file

Tujuan instruksional umum

Mahasiswa mampu menggunakan M file editor sebagai tempat untuk membuat dan menuliskan kode-kode pemrograman (coding)

Tujuan instruksional khusus

Setelah mempelajari bab 6 Fungsi M-file, mahasiswa mampu mencapai kompetensi berikut:

1. Mahasiswa mampu membuat M-file sebagai skript program dan menjalankannya dalam command windows matlab
2. Mahasiswa mampu membuat M-file sebagai fungsi dan melakukan running program

Bab 7 Persamaan Differensial

Tujuan instruksional umum

Mahasiswa mampu mengaplikasikan matlab untuk menyelesaikan persamaan differensial

Tujuan instruksional khusus

Setelah mempelajari bab persamaan differensial, mahasiswa mampu mencapai kompetensi berikut:

1. Mahasiswa mampu membuat dan menyelesaikan persamaan differensial dengan menggunakan matlab
2. Mahasiswa mampu membuat plot dari fungsi differesial