MA2151
Simulasi dan Komputasi Matematika

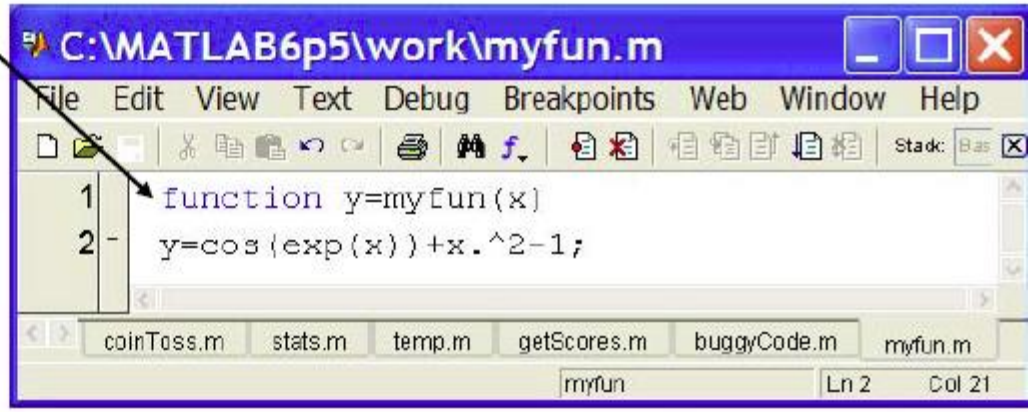
Mencari Akar dari Fungsi

Banyak masalah nyata yang dimodelkan menjadi f(x) = 0, sehingga dicari solusi x yang memenuhi. Gunakan **fzero** untuk mendapatkan akar untuk fungsi sembarang.

Definisikan suatu fungsi terpisah misal myfun.m

```
» x=fzero('myfun',1)
» x=fzero(@myfun,1)
```

Di sini 1 menyatakan banyaknya akar yang dicari



Mencari nilai maksimum atau minimum

fminbnd : meminimumkan fungsi atas satu selang tertutup.
>> x = fminbnd('myfun',-1,2);
Fungsi myfun memerlukan input skalar berasal dari interval [-1,2] dan output skalar juga.
fminsearch : mencari minimum tanpa batas interval
>> x = fminsearch('myfun',.5);
Cari nilai minimum lokal dari fungsi berawal dari x = 0,5.

Penulisan fungsi yang sederhana dapat langsung:
>> x = fminsearch(@(x) (cos(exp(x))+x*2-1), -1);
>> x = fzero(@(x) (cos(exp(x))+x*2-1), 1);

Integrasi Numerik

Metode quadratur Simpson (inputnya fungsi)

```
>> q = quad('myfun', 0, 10);
```

q adalah hasil dari integral fungsi dari 1 sampai 10.

```
>> q2 = quad(@(x) sin(x)*x, 0, pi);
```

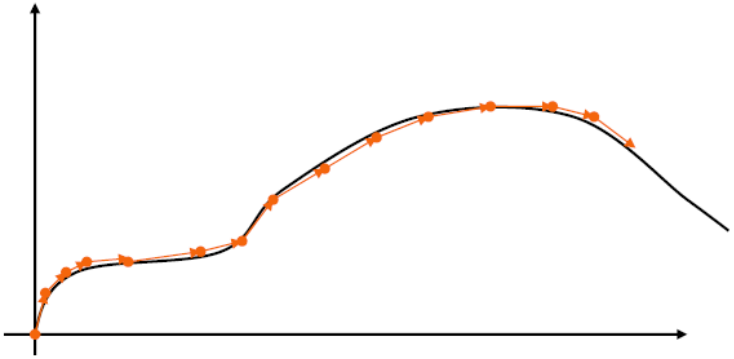
Aturan Trapesium (inputnya vektor)

```
>> x = 0:0.01:pi;
>> z = trapz(x, sin(x));
```

z adalah hasil integral sin(x) dari 0 ke pi

Menyelesaikan persamaan diferensial

Diberikan satu persamaan diferensial, solusi dapat dicari menggunakan integrasi:



Evaluasi turunan di satu titik dan aproksimasi dengan garis lurus.
Galat akan terakumulasi.
Variabel waktu dapat menurunkan jumlah iterasi.
Menggunakan ODE solver yang tepat dapat menghemat waktu tapi hasilnya lebih akurat.

Solver	Problem Type	Order of Accuracy	When to Use
`ode45`	Nonstiff	Medium	Most of the time. This should be the first solver you try.
`ode23`	Nonstiff	Low	For problems with crude error tolerances or for solving moderately stiff problems.
`ode113`	Nonstiff	Low to high	For problems with stringent error tolerances or for solving computationally intensive problems.
`ode15s`	Stiff	Low to medium	If `ode45` is slow because the problem is stiff.
`ode23s`	Stiff	Low	If using crude error tolerances to solve stiff systems and the mass matrix is constant.
`ode23t`	Moderately Stiff	Low	For moderately stiff problems if you need a solution without numerical damping.
`ode23tb`	Stiff	Low	If using crude error tolerances to solve stiff systems.

Untuk menggunakan command dengan option dan variabel waktu yang standard:

>> [t, y] = ode45('myODE', [1,10], [1 ; 0])

Input:

-'myODE : Nama fungsi ODE yang memiliki input (t,y) dan output dy/dt.

-[1, 10] : Interval langkah waktu berupa vektor

-[1; 0] : vektor kolom berisi nilai awal untuk ODE.

Output:

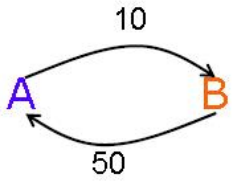
-t : titik-titik diskrit waktu

-y : solusi dari ODE

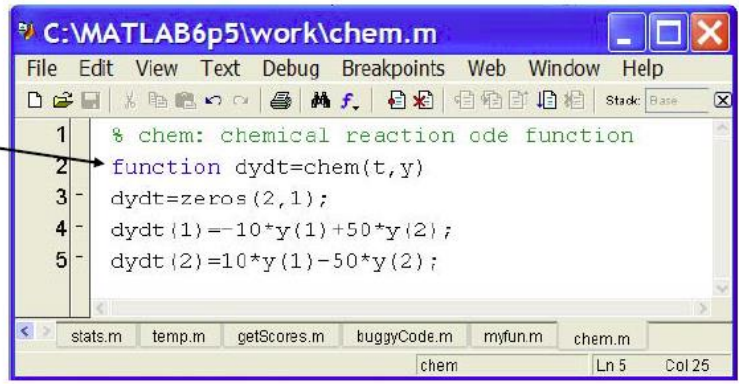
Contoh: Reaksi kimia

$$\frac{dA}{dt} = -10A + 50B$$

$$\frac{dB}{dt} = 10A - 50B$$



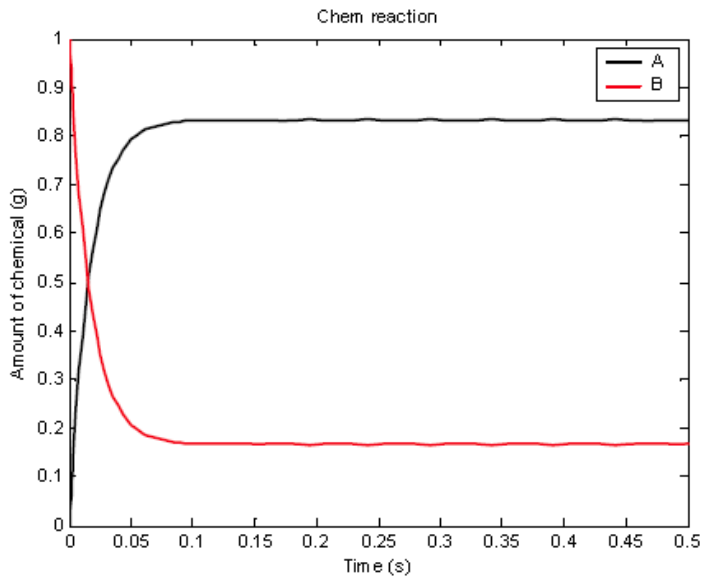
- ODE file:
  - y has [A;B]
  - dydt has [dA/dt;dB/dt]



```
» [t,y]=ode45('chem',[0 0.5],[0 1]);
```

Saat awal A(0)=0, B(0)=1, hanya ada B

```
» plot(t,y(:,1),'k','LineWidth',1.5);
» hold on;
» plot(t,y(:,2),'r','LineWidth',1.5);
» legend('A','B');
» xlabel('Time (s)');
» ylabel('Amount of chemical (g)');
» title('Chem reaction');
```



- Must make into a system of first-order equations to use ODE solvers
- Nonlinear is OK!
- Pendulum example:

$$\ddot{\theta} + \frac{g}{L} sin(\theta) = 0$$

$$\ddot{\theta} = -\frac{g}{L} sin(\theta)$$

$$let \quad \dot{\theta} = \gamma$$

$$\dot{\gamma} = -\frac{g}{L} sin(\theta)$$

$$\vec{x} = \begin{bmatrix} \theta \\ \gamma \end{bmatrix}$$

$$\frac{d\vec{x}}{dt} = \begin{bmatrix} \dot{\theta} \\ \dot{\gamma} \end{bmatrix}$$

C:\MATLAB6p5\work\pendulum.m

File Edit View Text Debug Breakpoints Web Window Help

```
1   % pendulum
2   function dxdt = pendulum(t,x)
3   L = 1;
4   theta = x(1);
5   gamma = x(2);
6
7   dtheta = gamma;
8   dgamma = -(9.8/L)*sin(theta);
9
10  dxdt = zeros(2,1);
11
12  dxdt(1)=dtheta;
13  dxdt(2)=dgamma;
```

temp.m getScores.m buggyCode.m myfun.m chem.m pendulum.m

pendulum Ln 13 Col 5

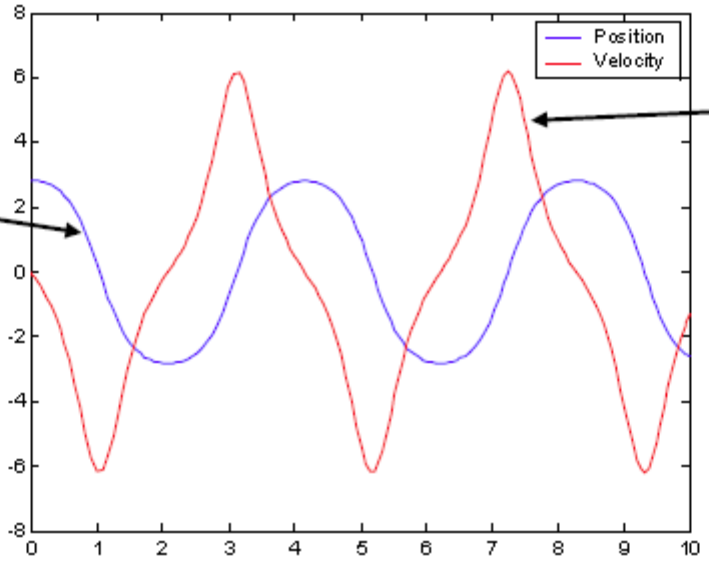
- We can solve for the position and velocity of the pendulum:

```
» [t,x]=ode45('pendulum',[0 10],[0.9*pi 0]);
```

  - assume pendulum is almost horizontal

```
» plot(t,x(:,1));
» hold on;
» plot(t,x(:,2),'r');
» legend('Position','Velocity');
```

Position in terms of angle (rad)

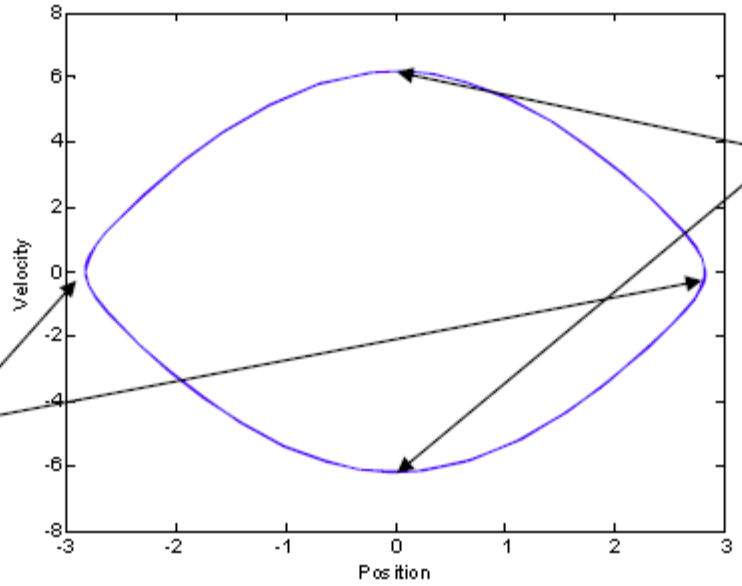


Velocity (m/s)

- Or we can plot in the phase plane:

```
» plot(x(:,1),x(:,2));
» xlabel('Position');
» yLabel('Velocity');
```

- The phase plane is just a plot of one variable versus the other:



Velocity is greatest when theta=0

Velocity=0 when theta is the greatest

- MATLAB's ODE solvers use a variable timestep
- Sometimes a fixed timestep is desirable

```
» [t,y]=ode45('chem',[0:0.001:0.5],[0 1]);
```

  - Specify the timestep by giving a vector of times
  - The function value will be returned at the specified points
  - Fixed timestep is usually slower because function values are interpolated to give values at the desired timepoints

- You can customize the error tolerances using odeset

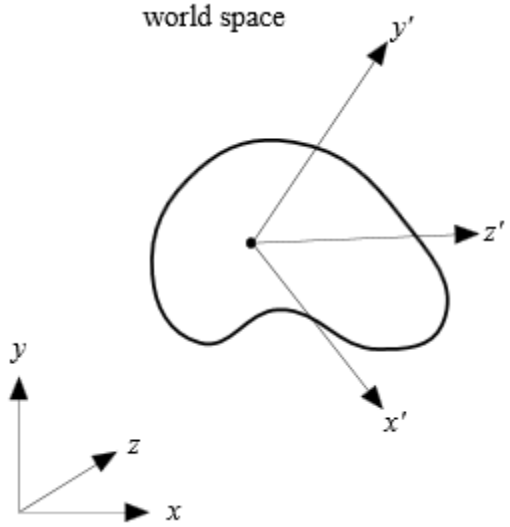
```
» options=odeset('RelTol',1e-6,'AbsTol',1e-10);
» [t,y]=ode45('chem',[0 0.5],[0 1],options);
```

  - This guarantees that the error at each step is less than `RelTol` times the value at that step, and less than `AbsTol`
  - Decreasing error tolerance can considerably slow the solver
  - See doc odeset for a list of options you can customize

Menyelesaikan Sistem Persamaan Diferensial

Pergerakan benda padat (rigid body) tanpa gaya luar
Dalam fisika, Rigid Body adalah idealisasi dari benda padat yang faktor deformasi diabaikan. Dengan kata lain, jarak antara dua poin yang diberikan dari rigid body tetap konstan dalam waktu tanpa gaya luar. Meskipun demikian suatu objek tidak dapat secara fisik ada karena relativitas, obyek biasanya dapat diasumsikan seperti ini jika mereka tidak bergerak mendekati kecepatan cahaya.

Posisi rigid body ditentukan oleh posisi titik pusat massanya dan attitude (total ada 6 parameter).



Menyelesaikan Sistem Persamaan Diferensial

Contoh: Pergerakan benda padat tanpa gaya luar
x = y(1), y = y(2), z = y(3)

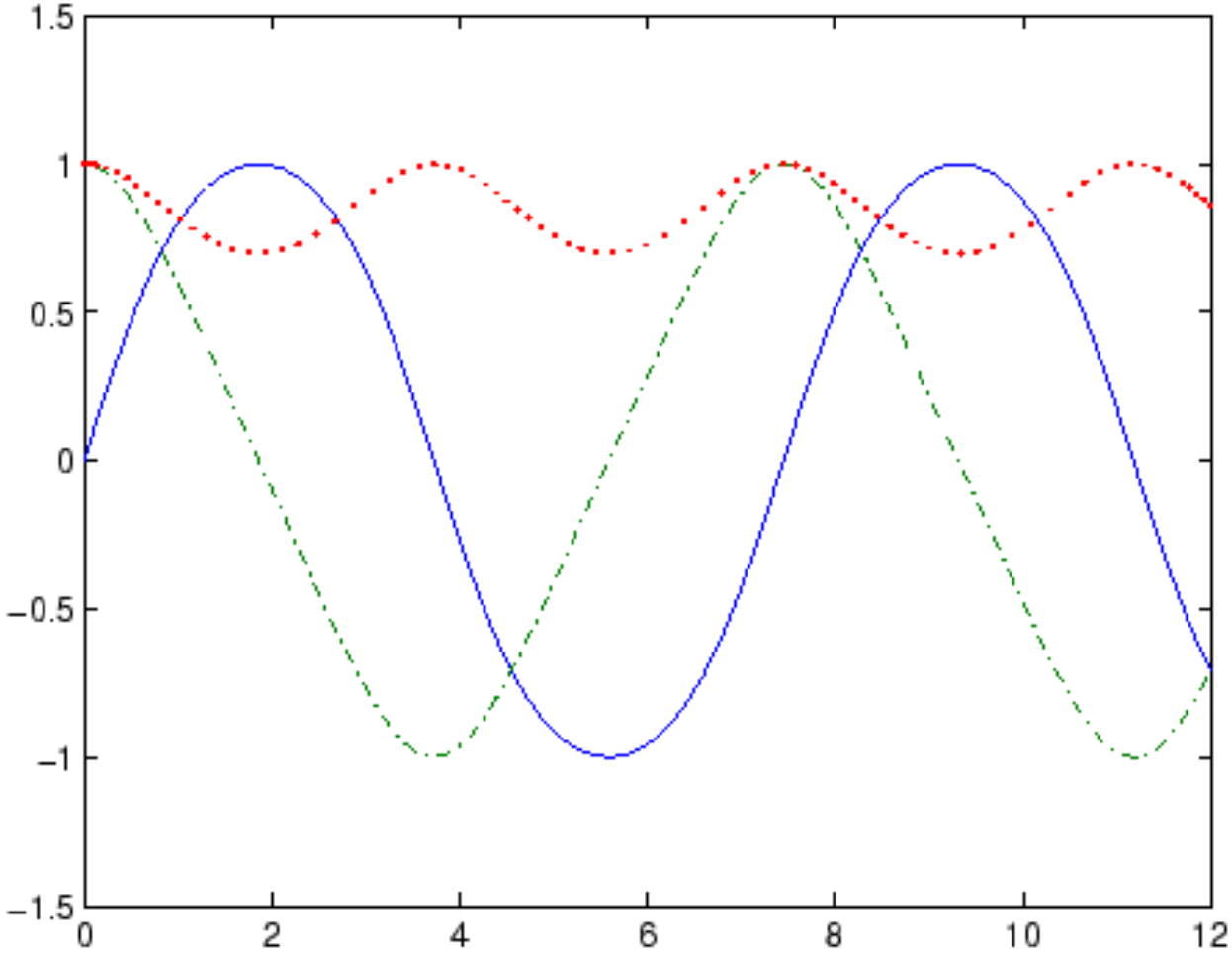
```
function dy = rigid(t,y)
dy = zeros(3,1); % a column vector
dy(1) = y(2) * y(3);
dy(2) = -y(1) * y(3);
dy(3) = -0.51 * y(1) * y(2);
```

dengan 0≤ t ≤ 12 dengan nilai awal saat t=0 **y** = [0 1 1].

```
options = odeset('RelTol',1e-4,'AbsTol',[1e-4 1e-4 1e-5]);
[T,Y] = ode45(@rigid,[0 12],[0 1 1],options);
```

Plot solusi Y terhadap T :

```
plot(T,Y(:,1),'-',T,Y(:,2),'-.',T,Y(:,3),'.')
```


1.5
1
0.5
0
−0.5
−1
−1.5
0
2
4
6
8
10
12

Terima kasih....