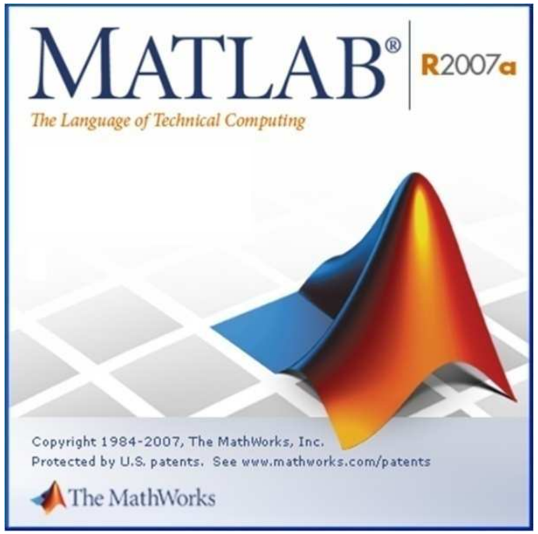


MATLAB for Chemical Engineering

Dr. M. Subramanian

Associate Professor
Department of Chemical Engineering
Sri Sivasubramaniya Nadar College of Engineering
OMR, Chennai – 603 110
msubbu.in[AT]gmail.com

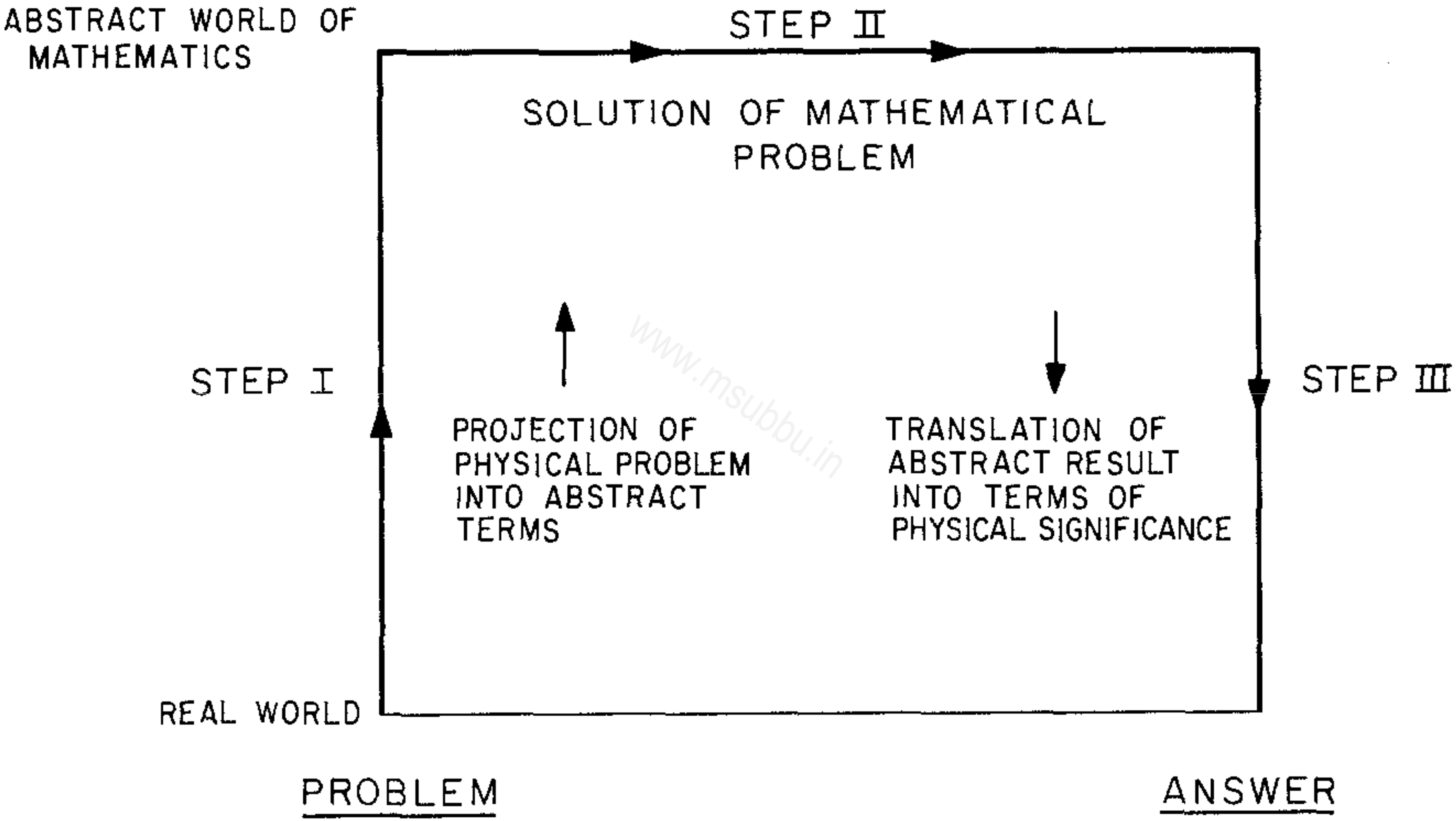
16th March 2012

Contents

Part I

Mathematical Models in Chemical Engineering

Modeling & Simulation



Types of Mathematical Models

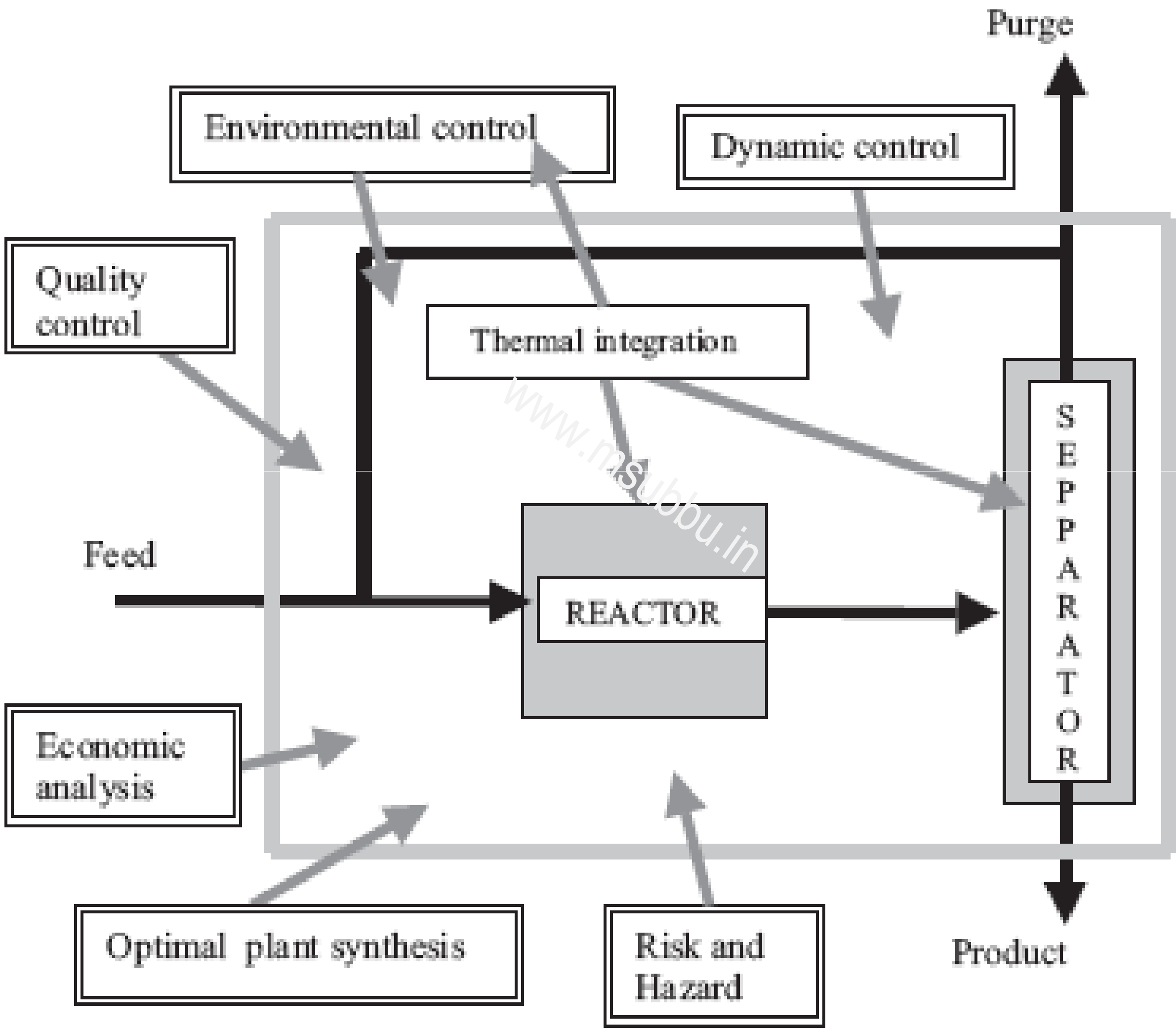
- Mathematical models are of two types:
  - based on physical theory
  - empirical

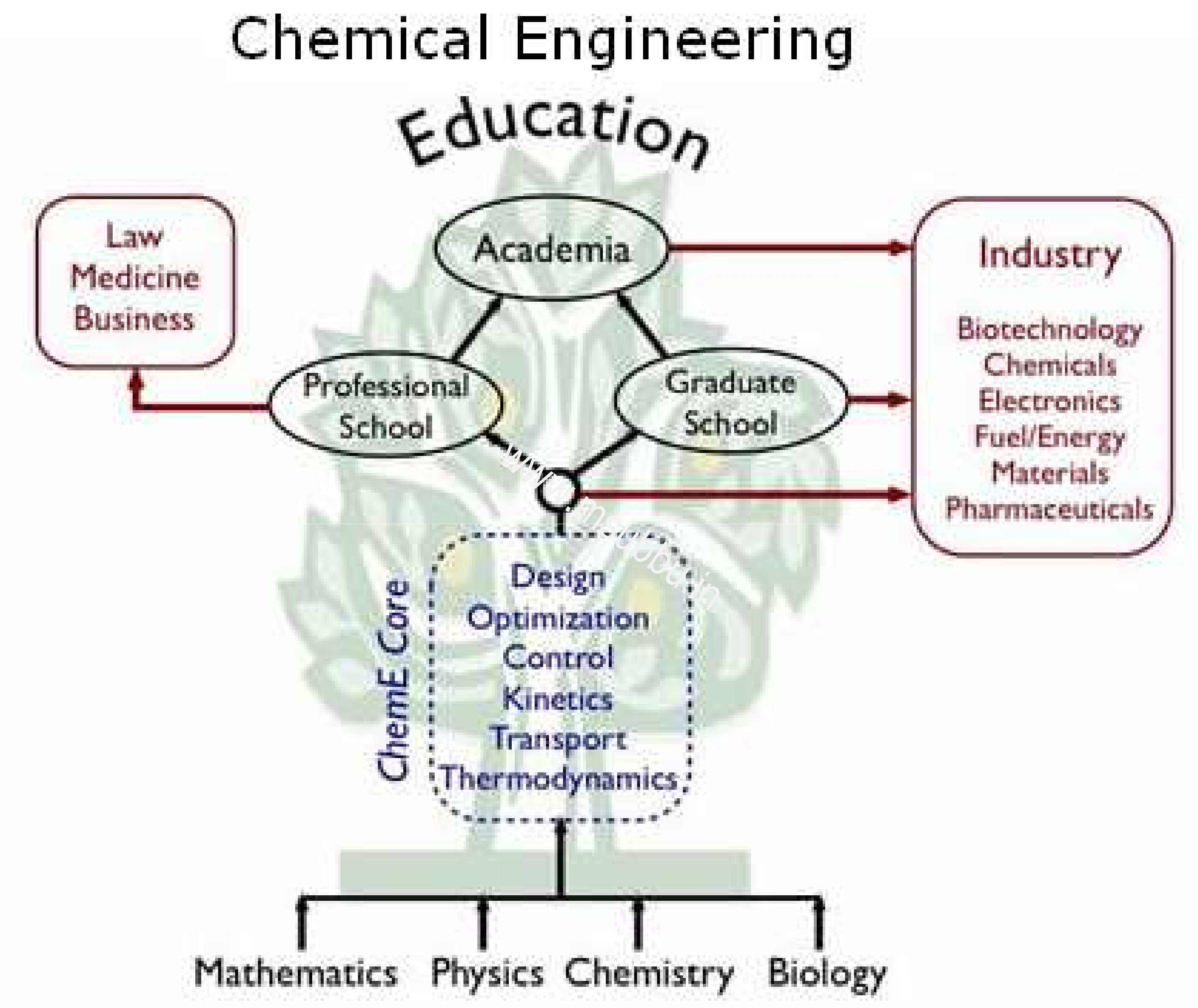
- Models based on physical theory can be further divided into the following categories:
  - Linear / non-linear
  - Steady / unsteady
  - Lumped / distributed parameter system
  - Continuous / discrete variables

Forms of Mathematical Models

- Algebraic equations
  - Linear
  - nonlinear
- Integral equations
- Differential equations
  - Ordinary differential
  - Partial differential
- Difference equations

Chemical Industry Problems




Chemical Engineering
Education
Law
Medicine
Business
Academia
Industry
Biotechnology
Chemicals
Electronics
Fuel/Energy
Materials
Pharmaceuticals
Professional School
Graduate School
ChemE Core
Design
Optimization
Control
Kinetics
Transport
Thermodynamics
Mathematics
Physics
Chemistry
Biology

Basis of Mathematical Models in Chemical Engineering

- Laws of physics, chemistry, such as conservation of mass, energy, and momentum
- Equation of state
- Equilibrium relationships
- Rate laws

Mathematical Problems in Chemical Engineering

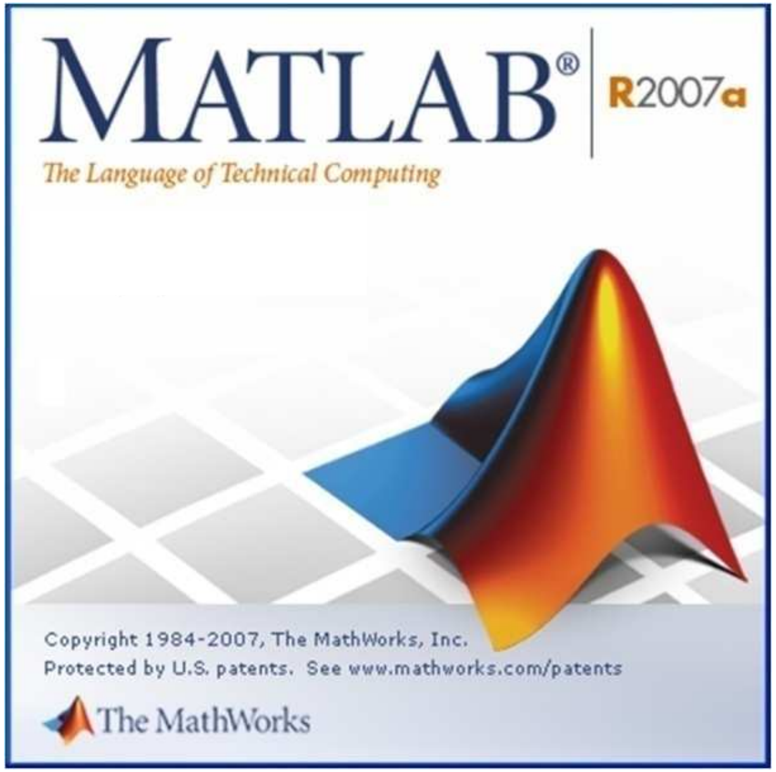
- Linear algebraic equations
- Non-linear equations
- Curve fitting – polynomial, non-linear
- Interpolation
- Integration
- Differential equations
- Partial differential equations

Part II

Getting Started with MATLAB

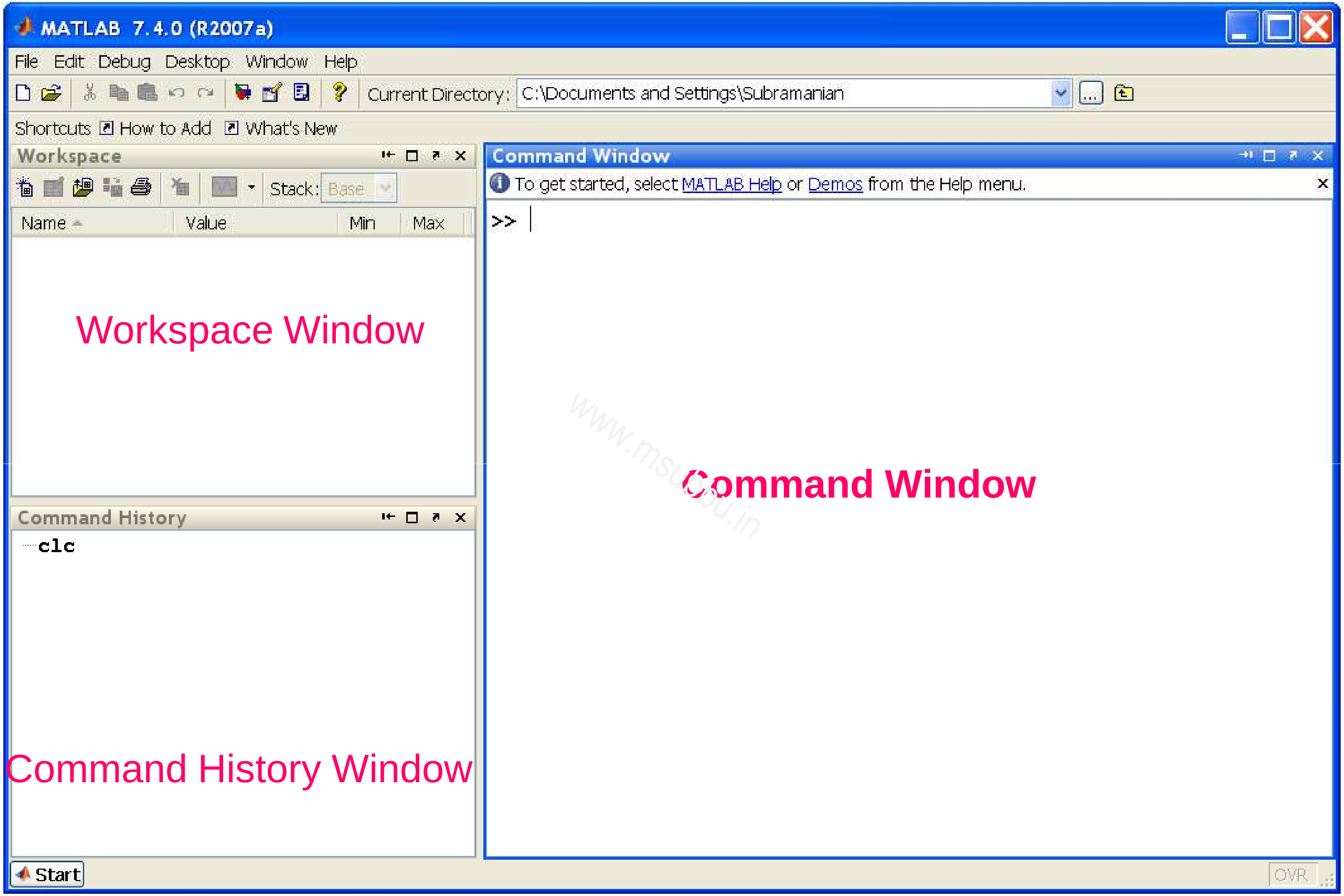
About MATLAB

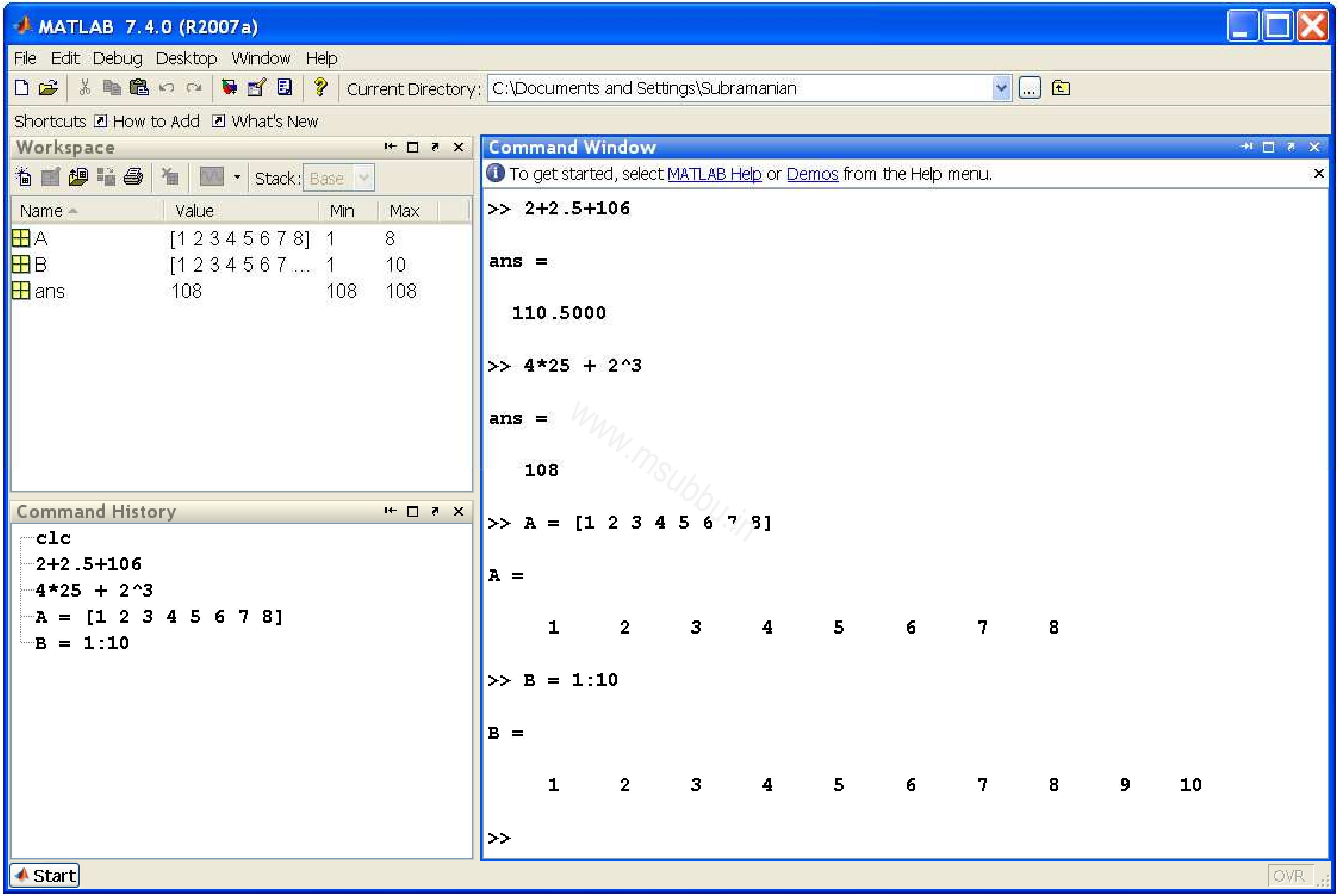
- MATLAB has become a standard among academic and industrial users
- Developed by Math Works Inc.
- http://www.mathworks.com
- MATLAB - acronym for MATrix LABoratory
- Matrices and arrays - the heart of MATLAB
- Offers programming features - similar to other languages



Capabilities of MATLAB

- Provides extensive numerical resources
- Contains lot of reliable, accurate mathematical subprograms
- The subprograms provide solutions to broad range of mathematical problems including:
  - matrix algebra, complex arithmetic, differential equations, nonlinear systems, and many special functions
- Provides a variety of graphic output displays:
  - linear, log-log, semilog, polar, bar chart, and contour plots
  - 2-D and 3-D views
- Provides GUI tool: Simulink® – block diagram representation, simulation


MATLAB 7.4.0 (R2007a)
File Edit Debug Desktop Window Help
Current Directory: C:\Documents and Settings\Subramanian
Shortcuts How to Add What's New
Workspace
Stack: Base
Name Value Min Max
Workspace Window
Command History
clc
Command History Window
Command Window
To get started, select MATLAB Help or Demos from the Help menu.
>>
Command Window
Start
OVR


MATLAB 7.4.0 (R2007a)
File Edit Debug Desktop Window Help
Current Directory: C:\Documents and Settings\Subramanian
Shortcuts How to Add What's New
Workspace
Stack: Base
Name Value Min Max
A [1 2 3 4 5 6 7 8] 1 8
B [1 2 3 4 5 6 7 ... 1 10
ans 108 108 108
Command History
clc
2+2.5+106
4*25 + 2^3
A = [1 2 3 4 5 6 7 8]
B = 1:10
Command Window
To get started, select MATLAB Help or Demos from the Help menu.
>> 2+2.5+106
ans =
110.5000
>> 4*25 + 2^3
ans =
108
>> A = [1 2 3 4 5 6 7 8]
A =
1 2 3 4 5 6 7 8
>> B = 1:10
B =
1 2 3 4 5 6 7 8 9 10
>>
Start
OVR

MATLAB Variables

```
» D = 2

D =
     2

» v = 3

v =
     3
» rho = 1000;
» mu = 0.001;
» Re = D*v*rho/mu

Re =

     6000000
»
```

ans	Default variable name used for resuts
pi	Value of π
inf	Stands for infinity (e.g., 1/0)
NaN	Stands for Not-a-Number (e.g., 0/0)
i, j	$i = j = \sqrt{-1}$

```
» c1 = 2+3i

c1 =

   2.0000 + 3.0000i
```

Mathematical Functions

```
» x=sqrt(2)/2

x =

    0.7071

» y=sin(x)

y =

    0.6496

»
```

Trigonometric functions	sin, cos, tan, sin, acos, atan, sinh, cosh, tanh, asinh, acosh, atanh, csc, sec, cot, acsc, …
Exponential functions	exp, log, log10, sqrt
Complex functions	abs, angle, imag, real, conj
Rounding and Remainder functions	floor, ceil, round, mod, rem, sign

Array Operations

```
» x = 1:10;
» y = sin(x)
y =
  Columns 1 through 7
    0.8415  0.9093  0.1411 -0.7568 -0.9589  -0.2794    0.6570
  Columns 8 through 10
    0.9894    0.4121   -0.5440

» y(3)

ans =
    0.1411

» y(1:5)

ans =
    0.8415    0.9093    0.1411   -0.7568   -0.9589
```

Array Manipulation

```
» A = [1 2; 3 4; 5 6]

A =

     1     2
     3     4
     5     6

» B = [1 2 3; 4 5 6];

» A'

ans =

     1     3     5
     2     4     6
```

```
» A+B
??? Error using ==> plus
Matrix dimensions must agree.

»

» A*B

ans =

     9    12    15
    19    26    33
    29    40    51
»
```

Element by Element Operation

```
» clear
» a = [1 2; 3 4];
» b = [1 1; 2 2];
»
» a.*b

ans =


     1      2
     6      8


»
```

```
» a./b

ans =
    1.0000    2.0000
    1.5000    2.0000

» a/b
Warning: Matrix is singular to
working precision.

ans =
   -Inf   Inf
   -Inf   Inf
»
```

Matrix Operations

```
» A = [1 2; 3 4];
» B = [1 1; 2 2];

» [A B]

ans =

     1     2     1     1
     3     4     2     2

» ans-1

ans =

     0     1     0     0
     2     3     1     1
```

```
» C = [A B]
C =

     1     2     1     1
     3     4     2     2

» C(1,:)

ans =
     1     2     1     1

» C(:,2:end)

ans =

     2     1     1
     4     2     2
```

Matrix Functions

```
» A

A =
      1      2
      3      4

» det(A)

ans =
      -2
» inv(A)

ans =
   -2.0000    1.0000
    1.5000   -0.5000
»
```

```
» [a b] = eig(A)

a =

   -0.8246   -0.4160
    0.5658   -0.9094


b =

   -0.3723         0
         0    5.3723

»
```

Related: **`diag, triu, tril, rank, size`**

Using Script M-files

```
» frictionfactor
Dia in meter = .1
Velocity in m/s = 2
Re =
       200000

    0.0037
»
```

Editor - K:\CurrentWorks\MATLAB\mfiles\frictionfactor.m

```matlab
% frictionfactor.m
% Done on 16-Dec-2001
% Re-Done on 28-Jan-2011
D = input('Dia in meter = ');
v = input('Velocity in m/s = ');
rho = 1000; % Density of water in kg/m3
mu = 0.001; % Viscoisity of water in kg/m.s

Re = D*v*rho/mu

f = 0.079*Re^(-0.25);

disp(f);
```

Control Flow Statements

```
for <index=start:end>
…
end
```

```
if <expression>
…
else
…
end
```

```
while <condition>
…
end
```

```
switch expression

    case option1
    …

    case option2
    …
    .
    case optionN
    …

    otherwise

end
```

Part III

Capabilities of MATLAB in Solving Typical Chemical Engineering Problems

Matrix Problems

To calculate the quantities of each of the three acids required for making 100 kg blended acid:

Component	Compositions of acids			Composition of blended acid
	Spent acid **X**	Aqueous HNO_3 **Y**	Aqueous H_2SO_4 **Z**	
H_2SO_4	44.4	0	98	60
HNO_3	11.3	90	0	32
H_2O	44.3	10	2	8

H_2SO_4 balance:

44.4 X + 0 Y + 98 Z = 60

HNO_3 balance:

11.3 X + 90 Y + 0 Z = 32

H_2O balance:

44.3 X + 10 Y + 2 Z = 8

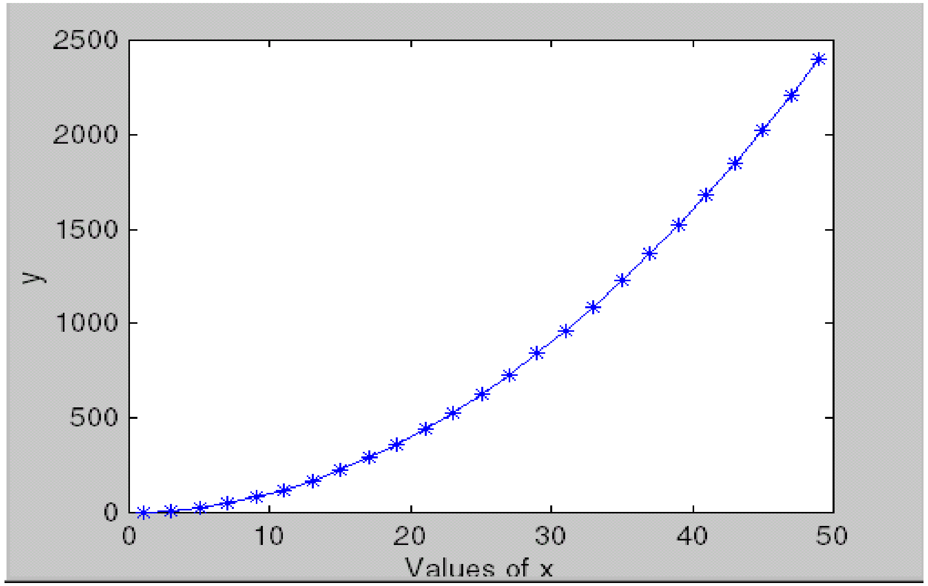
```
» A = [44.4  0    98
11.3   90   0
44.3   10   2]

» B = [60
32
8]
```

```
» x = inv(A)*B

x =
    0.0764
    0.3460
    0.5776
```

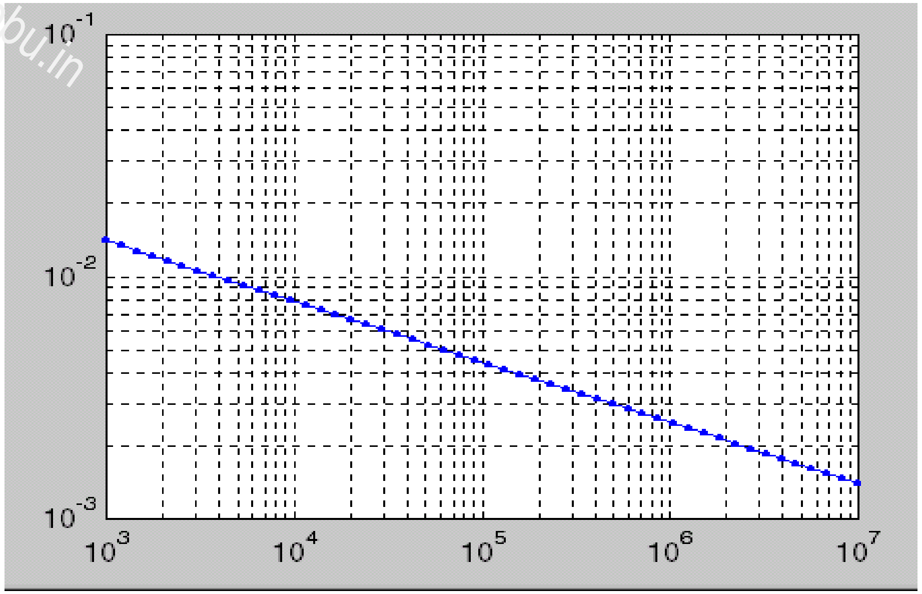
Similar Problems: Stage by stage calculations of absorber, extractor, with linear equilibrium relationships, under isothermal operation…

Plots

```
» x = 1:2:50;
» y = x.^2;
» plot(x,y,'*-')
» xlabel('Values of x')
» ylabel('y')
»
```




```
» P = logspace(3,7);
» Q = 0.079*P.^(-0.25);
» loglog(P,Q, '.-')
» grid
```



Non-linear Equation

$$\left(P + \frac{a}{V^2}\right)(V - b) = RT$$

$$a = \frac{27}{64}\frac{R^2T_c^2}{P_c}$$

$$b = \frac{RT_c}{8P_c}$$

```
% pvt_calculation.m
global R T P Tc Pc
T = input('Temperature in K ')
P = input('Pressure in Bar ')
R = 0.08314;           % Bar.m3/kmol.K
Tc = 190.6;   % Tc of Methane K
Pc = 45.99;   % Pc of Methane bar
v_ig = R*T/P
v_vander = fzero('vander',v_ig)
%-------------------------------
```

```
% vander.m
function v2 = vander(vol);
global R T P Tc Pc
 a = 27*(R^2)*(Tc^2)/(64*Pc);
 b = R*Tc/(8*Pc);
 v2 = R*T - (P + a/vol^2)*(vol-b);
%-----------------------------------
```

```
» pvt_calculation

Temperature in K 350
Pressure in Bar 30

v_ig =
    0.9700

v_vander =
    0.9347
»
```

Non-linear Equations

van Laar equations:

$$\ln \gamma_1 = A'_{12}\left(1 + \frac{A'_{12}x_1}{A'_{21}x_2}\right)^{-2}$$

$$\ln \gamma_2 = A'_{21}\left(1 + \frac{A'_{21}x_2}{A'_{12}x_1}\right)^{-2}$$

Relates γ_i with x_i. Estimating the parameters (A_{12}', A_{21}') based on γ_i , x_i data, involves solving the nonlinear algebraic equations.

Solution to Non-linear Equations

```
function Eqn = vanLaarEqns(A, x1, g1, g2)
% to solve for vanLaar parameters A1, A2
x2 = 1-x1;
Eqn(1) = log(g1) - A(1)*(1+ (A(1)*x1/(A(2)*x2)))^(-2);
Eqn(2) = log(g2) - A(2)*(1+ (A(2)*x2/(A(1)*x1)))^(-2);
% end
```

```
» x_1 = 0.561; g_1 = 1.4167; g_2 = 1.4646; Azero = [2 2];

» Asolved = fsolve(@vanLaarEqns, Azero, [], x_1, g_1, g_2)
Optimization terminated: first-order optimality is less than
options.TolFun.

Asolved =
    1.2015    1.7911
»
```

Non-linear Equations in ChE

- Terminal settling velocity
- Pressure drop vs. velocity for flow through pipelines, Piping network calculations, pressure drop / velocity calculations in packed and fluidized beds
- PVT relations – nonlinear algebraic
- Internal Rate of Return (IRR) calculations

Interpolations

V = ZRT/P $Z = f(T_r, P_r)$ $T_r = T/T_c$, and $P_r = P/P_c$

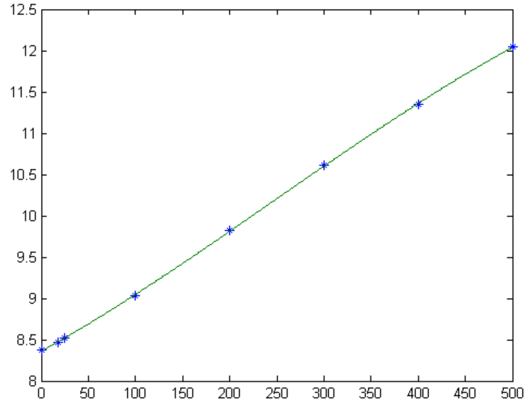
Zº									
Tr ↓	Pr →	0.01	0.05	0.10	0.20	0.40	0.60	0.80	1.00
0.30		0.0029	0.0145	0.0290	0.0579	0.1158	0.1737	0.2315	0.2892
0.35		0.0026	0.0130	0.0261	0.0522	0.1043	0.1564	0.2084	0.2604
0.40		0.0024	0.0119	0.0239	0.0477	0.0953	0.1429	0.1904	0.2379
0.45		0.0022	0.0110	0.0221	0.0442	0.0882	0.1322	0.1762	0.2200
0.50		0.0021	0.0103	0.0207	0.0413	0.0825	0.1236	0.1647	0.2056
0.55		0.9804	0.0098	0.0195	0.0390	0.0778	0.1166	0.1553	0.1939
0.60		0.9849	0.0093	0.0186	0.0371	0.0741	0.1109	0.1476	0.1842
0.65		0.9881	0.9377	0.0178	0.0356	0.0710	0.1063	0.1415	0.1765
0.70		0.9904	0.9504	0.8958	0.0344	0.0687	0.1027	0.1366	0.1703
0.75		0.9922	0.9598	0.9165	0.0336	0.0670	0.1001	0.1330	0.1656
0.80		0.9935	0.9669	0.9319	0.8539	0.0661	0.0985	0.1307	0.1626


```
» interp2(Pr_data,Tr_data,Z0_data,0.5,0.48)
ans =
    0.1059
```

Polynomial Fitting

$C_p = aT^3 + bT^2 + cT + d$

```
>> T=[0,18,25,100,200,300,400,500];
>> Cp=[8.371, 8.472, 8.514, 9.035, 9.824, 10.606, 11.347, 12.045];
>> P=polyfit(T,Cp,3)

P =
   -0.0000    0.0000    0.0063    8.3590

>> T_range = [0:500];
>> Cp_fit = P(1).*T_range.^3+P(2).*T_range.^2+P(3).*T_range+P(4);
>> plot(T,Cp,'*',T_range,Cp_fit)
```

Nonlinear Curve Fitting

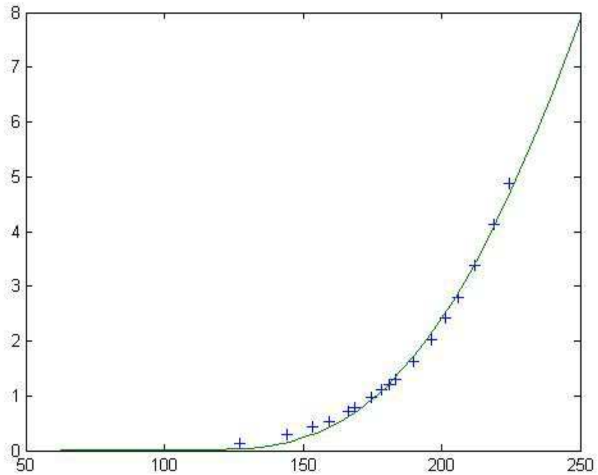
ln P = A - B/(T+C)

```
% AntoineFit.m
function AntoineFit
% T in oC; P in kPa
% ln P = A - B/(T+C)

T = [127.371  144.129  153.240  159.318  166.330  168.757  174.720  ...
   178.420  181.160  183.359  189.673  196.222  201.605  206.080  ...
212.190  218.896  224.570];

P = [0.139    0.293    0.424    0.538    0.706    0.774    0.964 ...
1.101    1.213   1.309    1.627    2.024    2.414    2.784   ...
3.369    4.128    4.882];

ABC0 = [5 500 -50];  % Starting guess
```

```
% AntoineFit.m contd..
ABCfit = lsqcurvefit(@EqnForm,ABC0,T,P)

Tfit = [50:5:250];
Pfit = exp(ABCfit(1) - ABCfit(2)./(Tfit + ABCfit(3)));

plot(T,P,'+',Tfit,Pfit,'-')

function Psat = EqnForm(ABC,Tdata)
Psat = exp(ABC(1) - ABC(2)./(Tdata + ABC(3)));
% ------------------------------
```



```
» AntoineFit

ABCfit =

    5.7313   748.0393   -45.8242
```

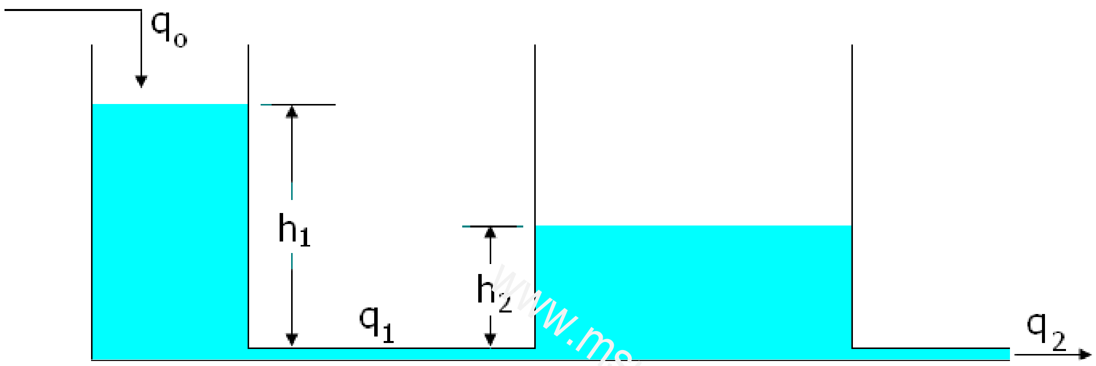


Nonlinear Curve Fitting in ChE

- Langmuir-Hinshelwood kinetics
- Parameters of Redlich-Kister expansion for property changes of mixing
- Activity coefficient-composition models

Ordinary Differential Equations

To plot the variation in tank levels for two interacting tanks



From mass balance, and using Bernoulli equations, we get:

$$\frac{dh_1}{dt} = \beta_o - \beta_1\sqrt{h_1 - h_2}$$

$$\frac{dh_2}{dt} = \beta_2\sqrt{h_1 - h_2} - \beta_3\sqrt{h_2}$$

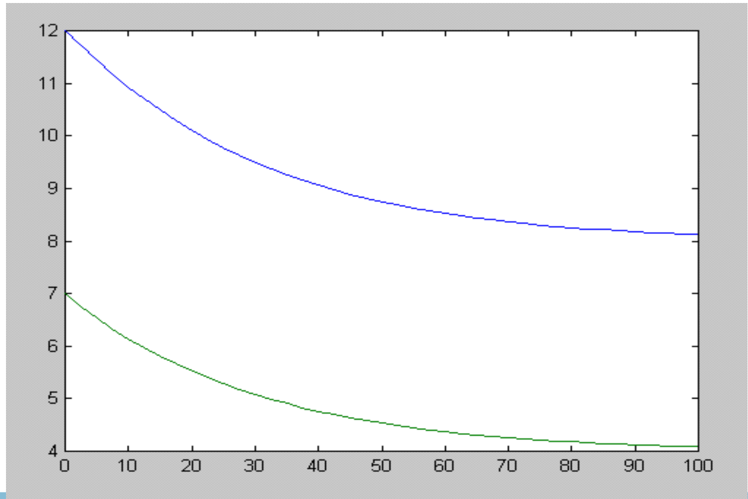
Where $\beta_o = \frac{q_o}{A_1}$; $\beta_1 = \frac{A_{p1}}{A_1}\sqrt{2g}$; $\beta_2 = \frac{A_{p1}}{A_2}\sqrt{2g}$; and $\beta_3 = \frac{A_{p2}}{A_2}\sqrt{2g}$

Solving ODEs

```
%twotnk.m
function hdot = twotnk(t,h)
% To solve
% dh1/dt = 1 - 0.5*sqrt(h1-h2)
% dh2/dt = 0.25*sqrt(h1-h2) - 0.25*sqrt(h2)
hdot = zeros(2,1);  % a column vector
hdot(1) = 1- 0.5*sqrt(h(1)-h(2));
hdot(2) = 0.25*sqrt(h(1) - h(2)) - 0.25*sqrt(h(2));
```



```
» [t, h] = ode45(@twotnk, [0 100], [12 7]');
» plot(t, h)
```



ODE Problems in ChE

- Reaction Engineering
  - Concentration vs time (dC/dt),

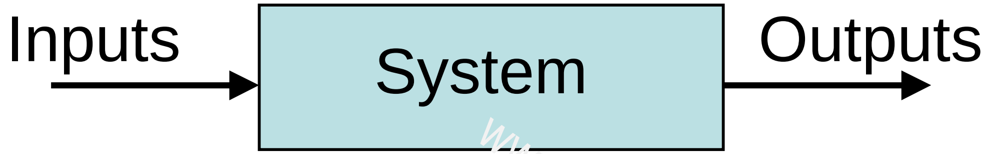
- Heat Transfer
  - Temperature vs time(dT/dt), Temperature vs distance (dT/dx)

Part IV

Getting Started with Simulink

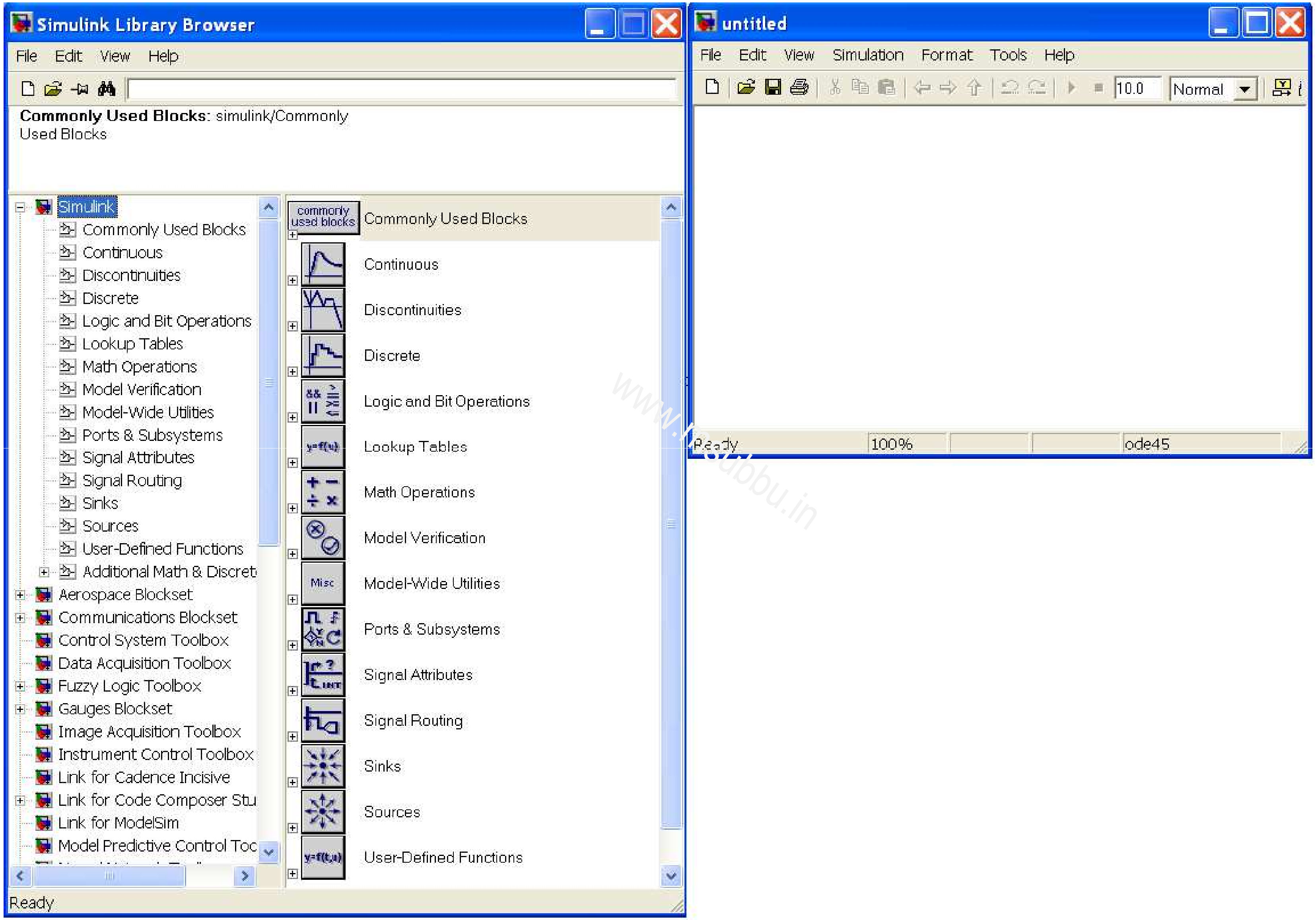
Simulink

- **SIMULINK** is an extension to **MATLAB** which uses a icon-driven interface for the construction of a block diagram representation of a process.

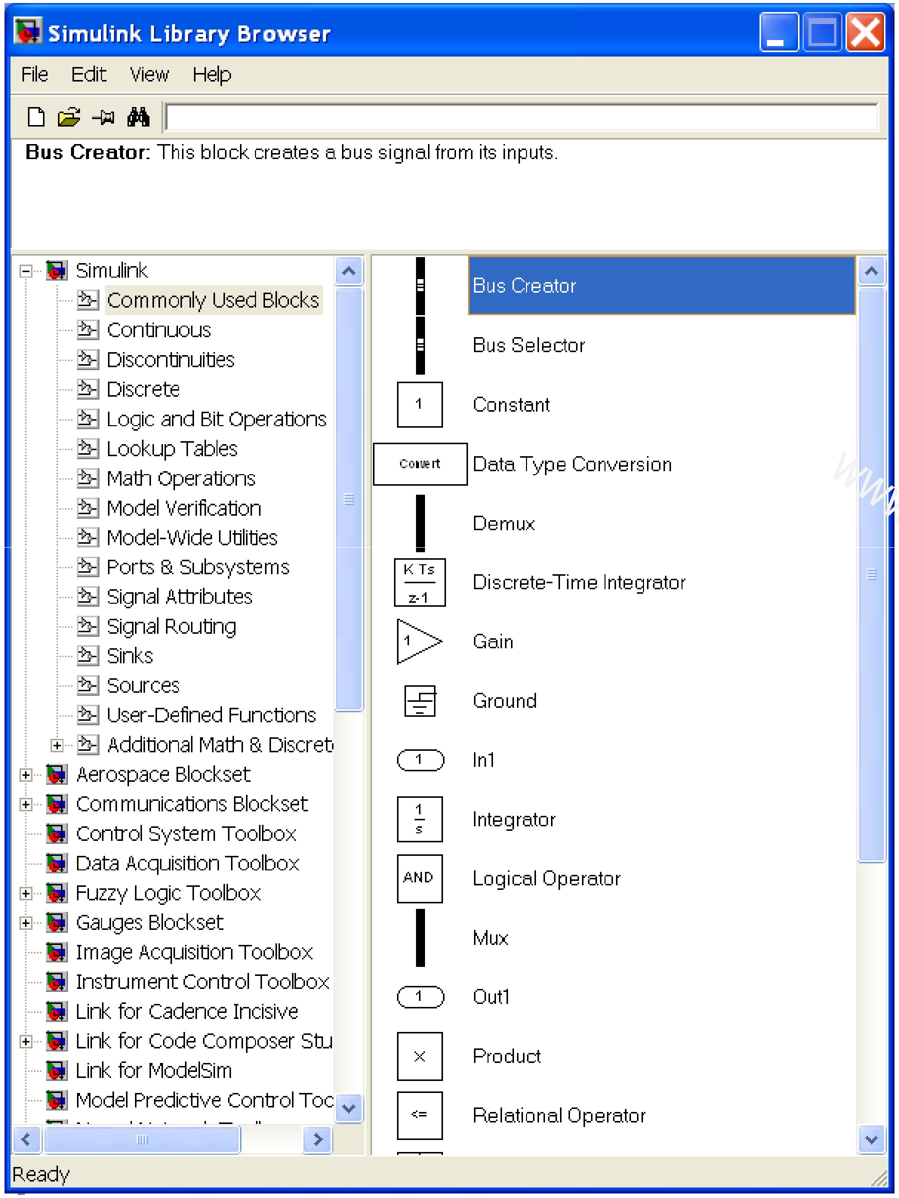


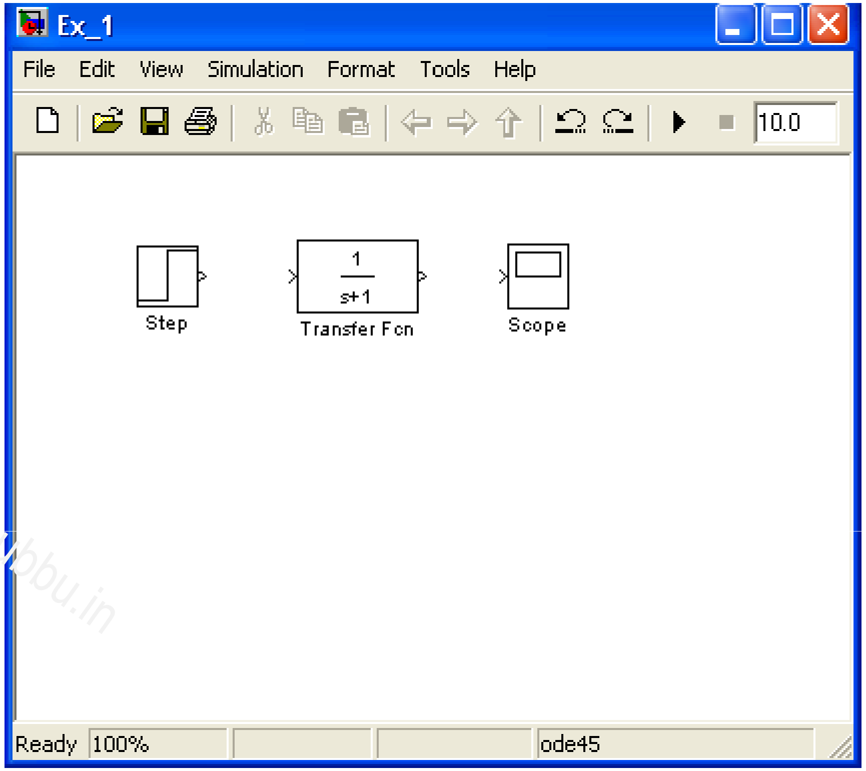
- **Steps involved in Simulink Solution**: 
  - Creating the Block Diagram
  - Specifying the Block Parameters
  - Setting up the Solver
  - Running the Simulation

New Simulink File

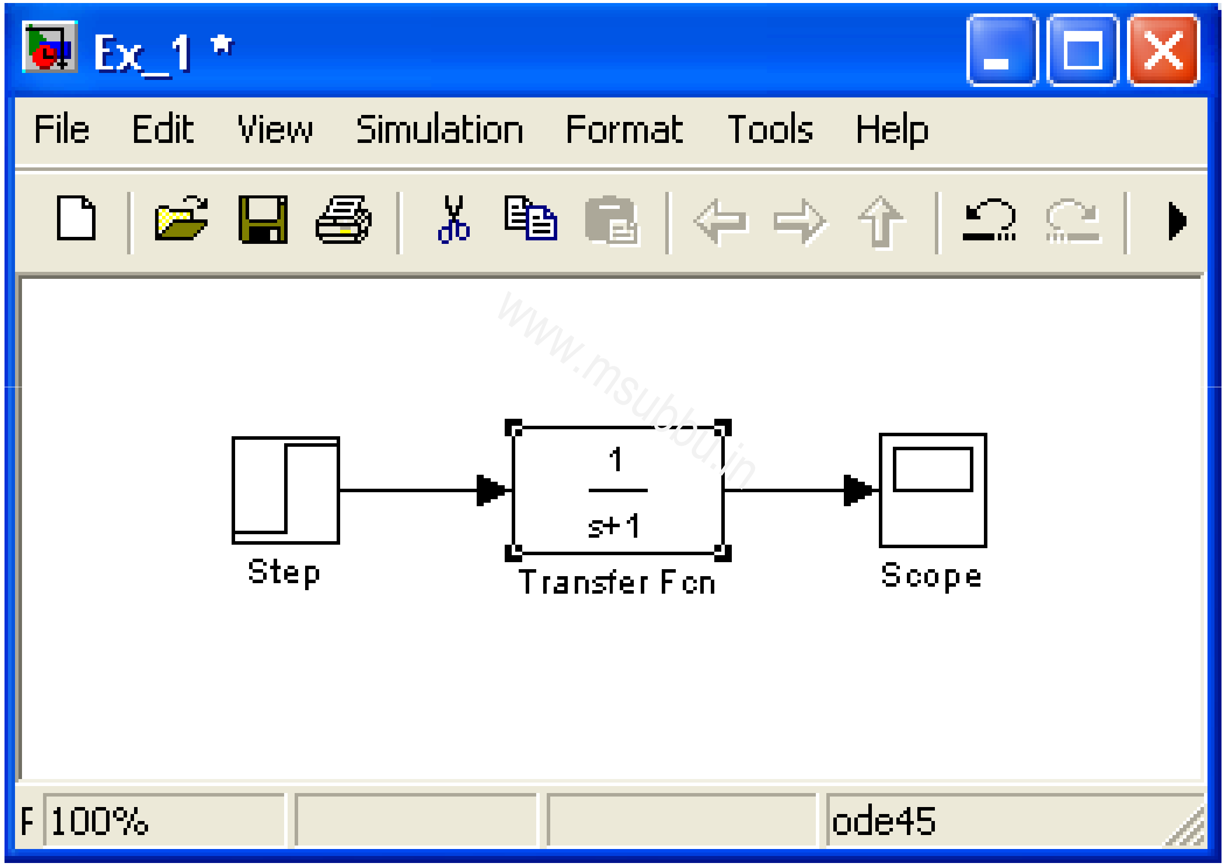


Arranging the Blocks

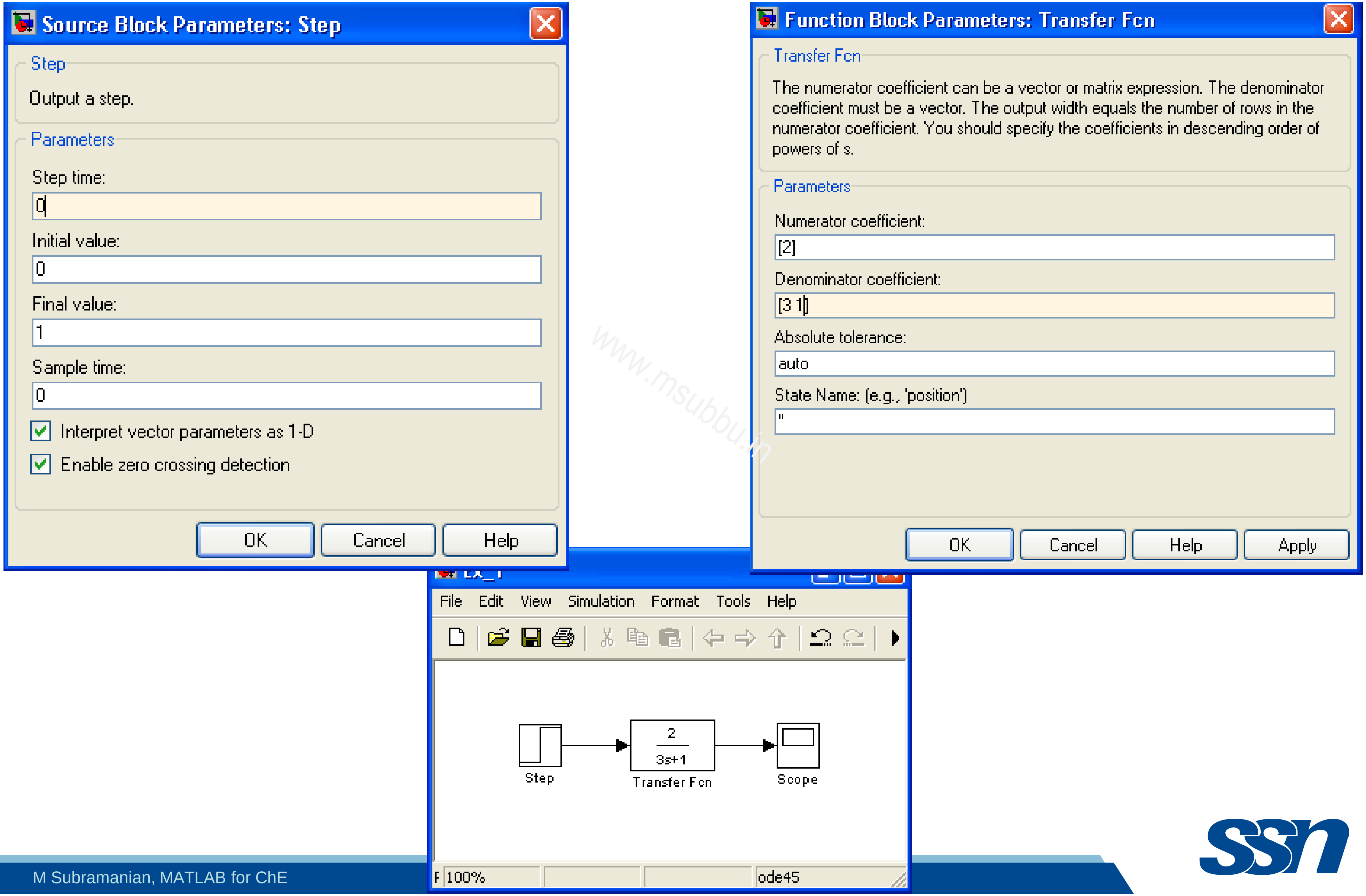




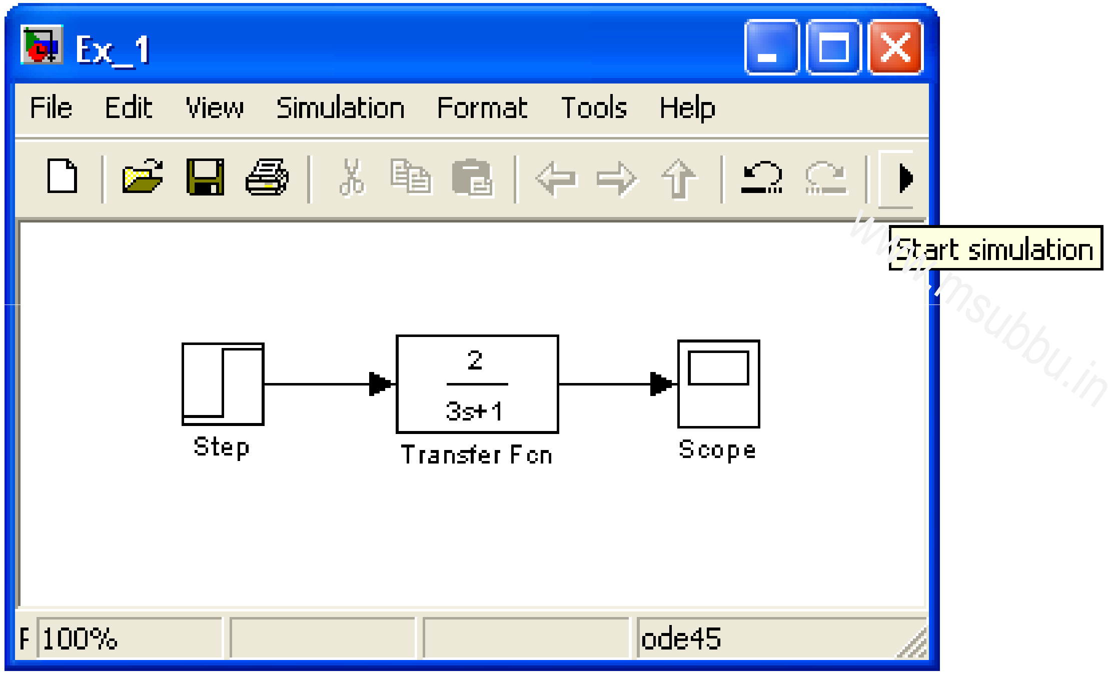
Connecting the Blocks

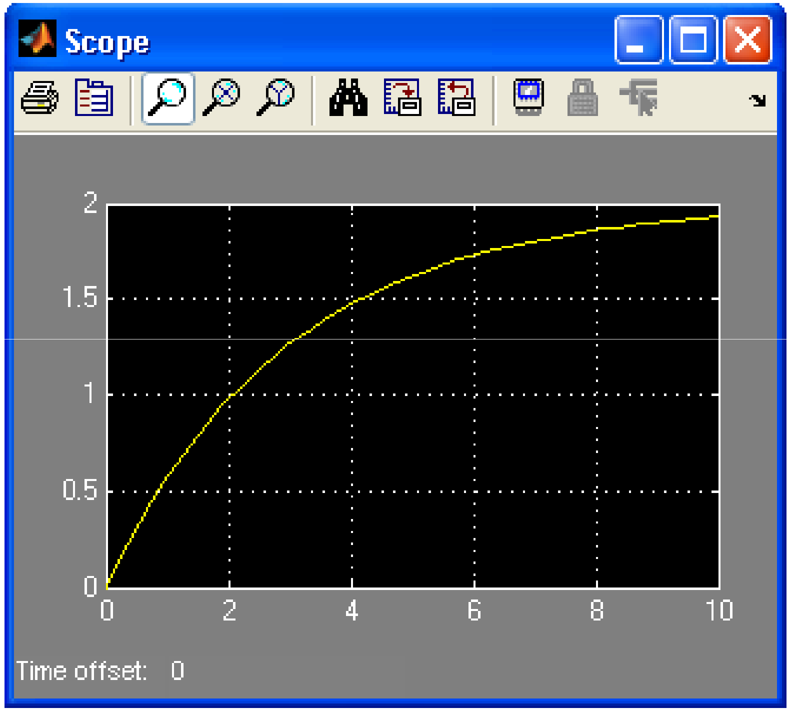


Specifying Block Parameters

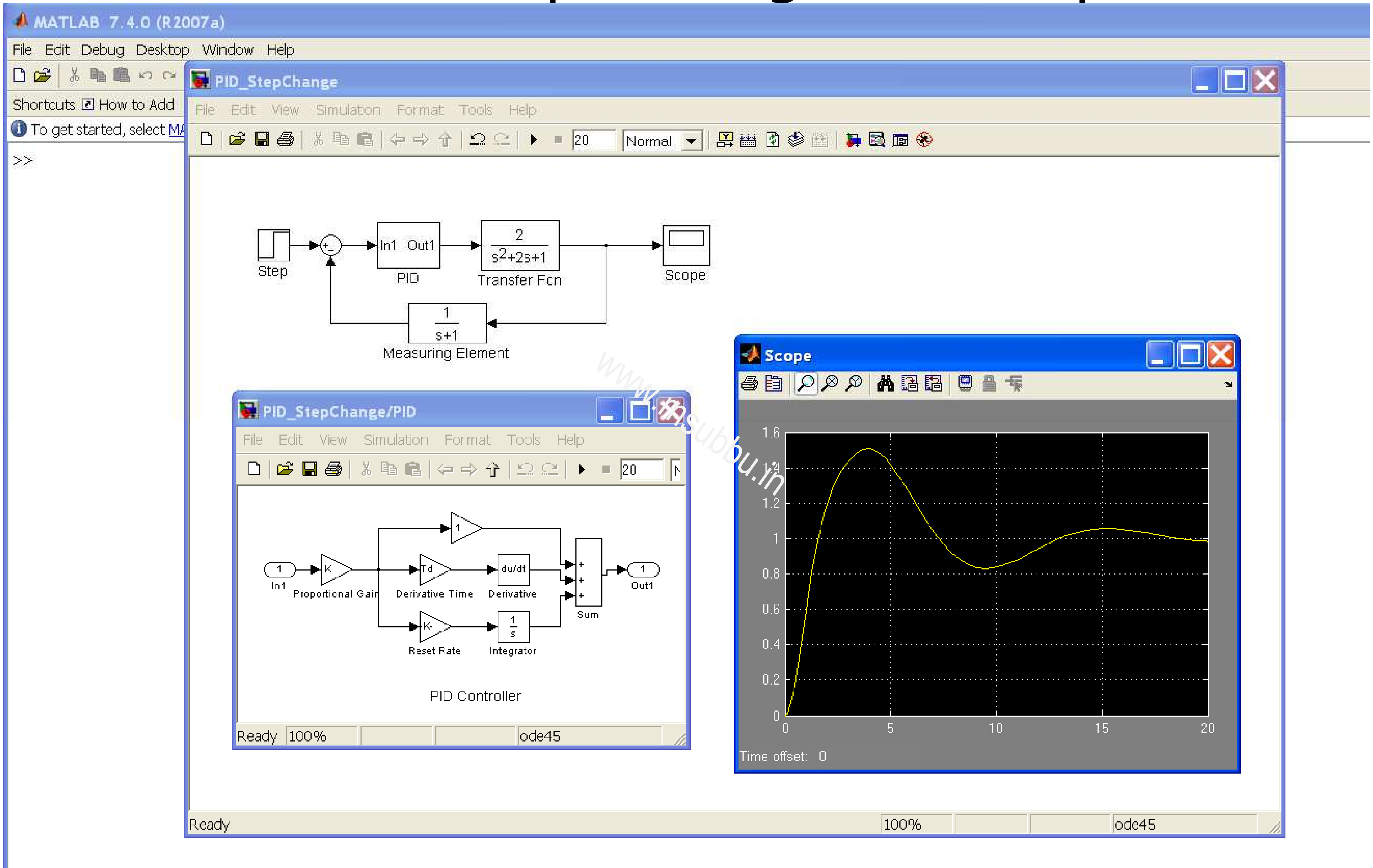


Running the Simulation

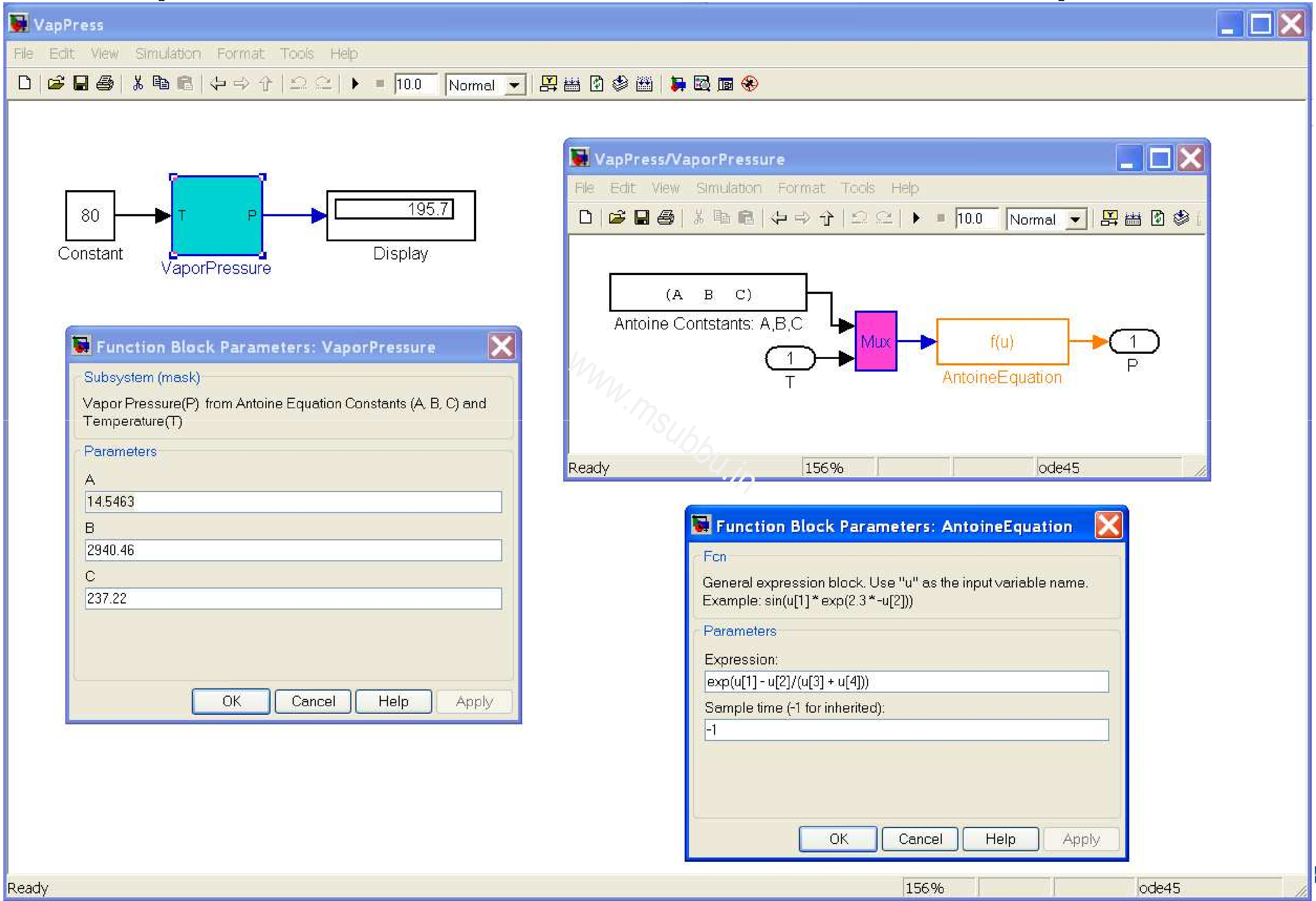




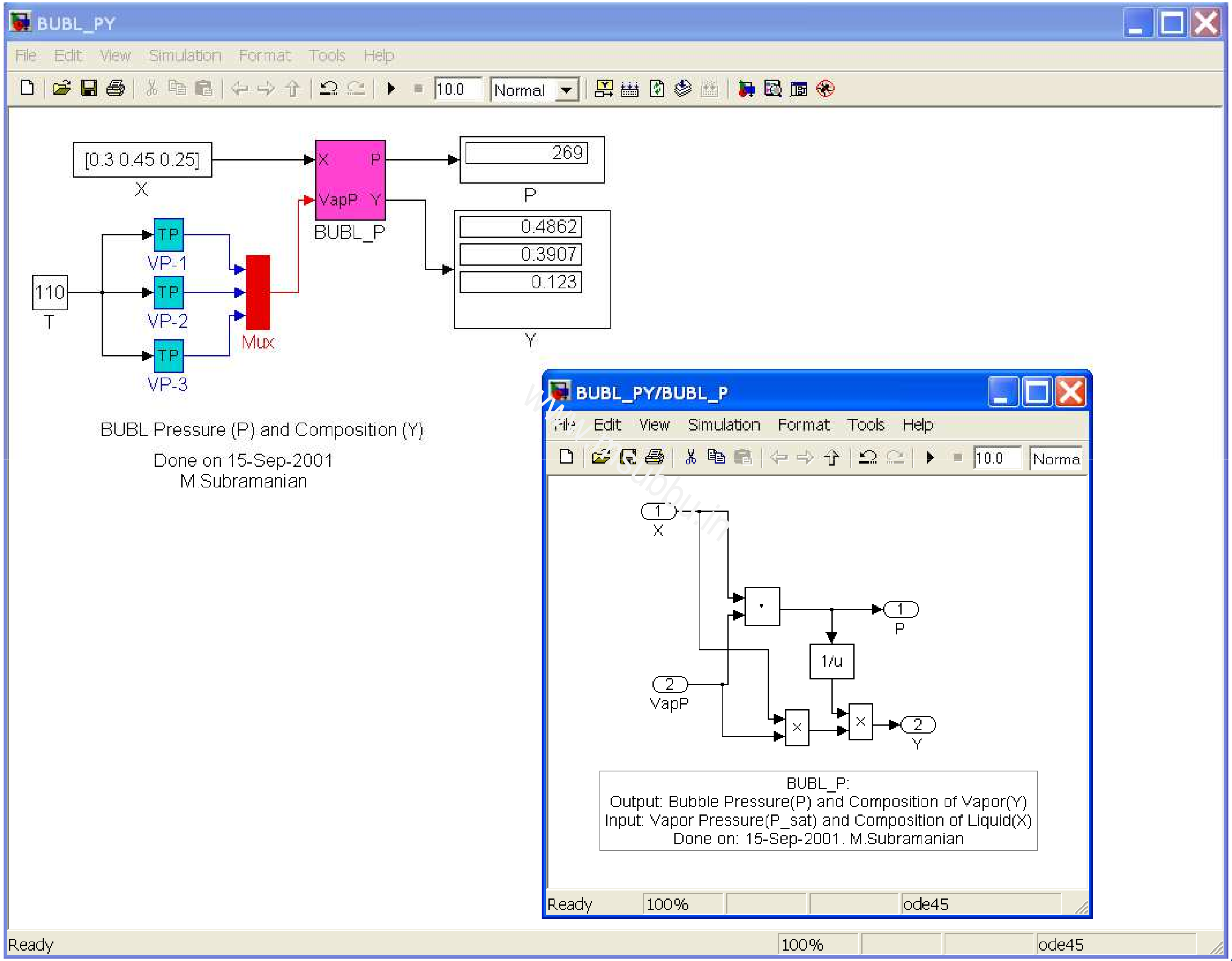
PID with Step Change in Set-point



Vapor Pressure from Antoine Equation



Bubble Pressure and Composition



Thank You!