MATLAB: Nonlinear ODEs

1. Background
2. In-class exercise

MATLAB: Nonlinear ODEs

Background

ODE Solution Functions

$$\frac{dy_1}{dx} = f_1(x, y_1, \ldots, y_m) \quad y_1(x_0) = y_{10}$$
$$\vdots$$
$$\frac{dy_m}{dx} = f_m(x, y_1, \ldots, y_m) \quad y_m(x_0) = y_{m0}$$

$$\Rightarrow \quad \frac{d\mathbf{y}}{dx} = \mathbf{f}(x, \mathbf{y}) \quad \mathbf{y}(x_0) = \mathbf{y}_0$$

Solver	Problems	Method
ode45	Nonstiff ODEs	Runge-Kutta
ode23	Nonstiff ODEs	Runge-Kutta
ode113	Nonstiff ODEs	Adams-Bashforth
ode15s	Stiff ODEs	Numerical differentiation
ode23s	Stiff ODEs	Rosenbrock
ode23t	Moderately stiff ODEs	Trapezoidal
ode23tb	Stiff ODEs	Trapezoidal & numerical differentiation
ode15i	Implicit ODEs	Numerical differentiation

ODE Solution Functions

- Solution of ODE systems with Matlab
- ode23, ode45, ode113, ode15s, ode23s, ode23t, ode23tb
  - » Matlab functions for solving initial value problems for ordinary differential equations
  - » Syntax: [x,y] = solver(odefun,xspan,y0,options)
    - – solver: one of the Matlab ODE solvers
    - – odefun: name of function that evaluates the RHS of $d\mathbf{y}/dx=f(x,\mathbf{y})$
    - – xspan: vector specifying the integration interval [x0,xf]
    - – y0: vector of initial conditions
    - – options: solver options (optional)
- ode15i
  - » Matlab function for solving fully implicit differential equations
  - » See ‘help ode15i’ for details

van der Pol Equation

- van der Pol equation as an ODE system:

$$y'' - \mu(1-y^2)y' + y = 0 \quad \rightarrow \quad \frac{d}{dt}\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} y_2 \\ \mu(1-y_1{}^2)y_2 - y_1 \end{pmatrix}$$

- Built-in Matlab functions for the van der Pol ODEs:

  μ=1 vdp1(t,y)

  μ=1000 vdp1000(t,y)

  >> type vdp1

- Use Matlab ODE solvers to find the solution:

  >> [t,y]=ode45(@vdp1,[0 20],[2 0]); plot(t,y(:,1),'-o')

  >> [t,y]=ode15s(@vdp1000,[0 3000],[2 0]); plot(t,y(:,1),'-o')

van der Pol Equation

- Value of μ effects the stiffness of the system
- Small μ, nonstiff, ode45 more efficient:

```
>> tic, for i=1:10; [t,y]=ode45(@vdp1,[0 20],[2 0]); end; toc
>> tic, for i=1:10; [t,y]=ode15s(@vdp1,[0 20],[2 0]); end; toc
```

- large μ, stiff, stiff solver required:

```
>> [t,y]=ode45(@vdp1000,[0 3000],[2 0])
```

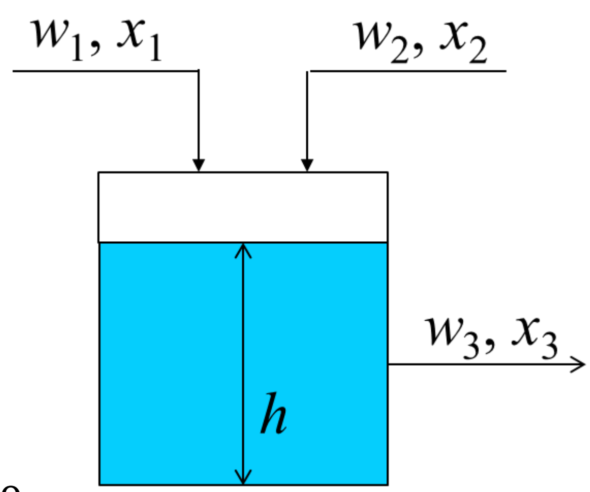
Too slow, abort with ‘Ctrl-C’

Binary Mixing Tank

- Mass balances:

$$\frac{dh}{dt} = \frac{w_1 + w_2 - C_v\sqrt{h}}{\rho A} \qquad h(0) = h_0$$

$$\frac{dx_3}{dt} = \frac{w_1(x_1 - x_3) + w_2(x_2 - x_3)}{\rho A h} \qquad x_3(0) = x_{30}$$



- Parameters: w_1 = 10, x_1 = 0.1, w_2 = 2, x_2 = 0.9, C_v = 5; ρA = 10
- Use MATLAB to numerically solve the two coupled ODEs for $h(t)$ and $x_3(t)$

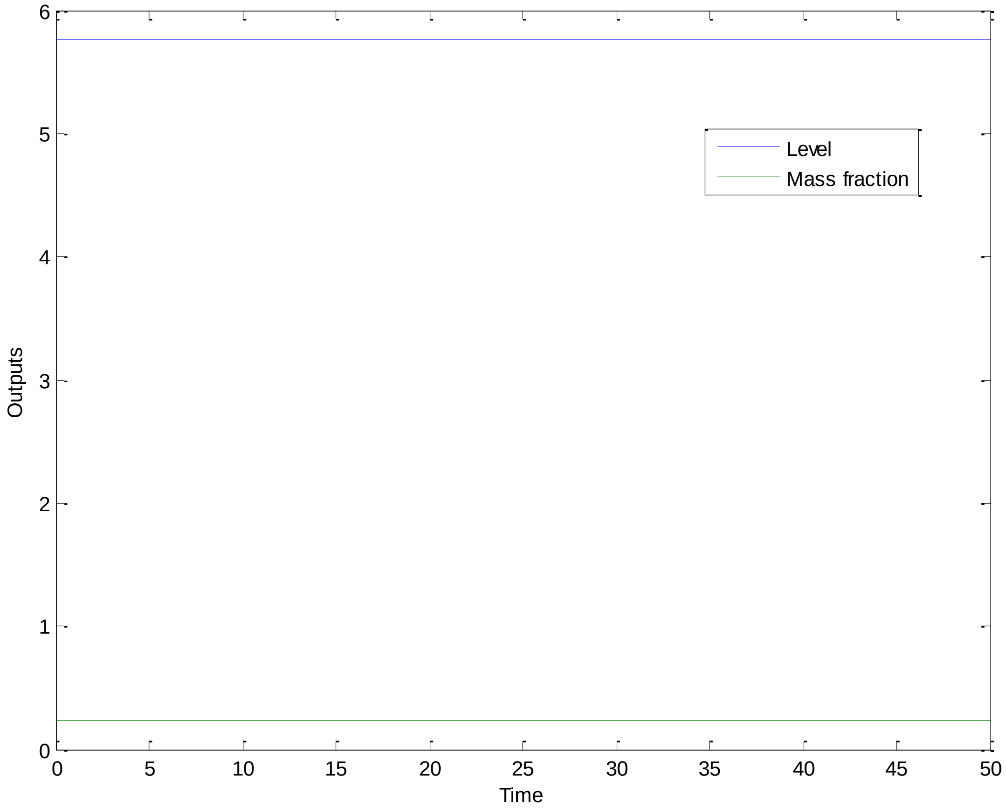
binary_mixing.m

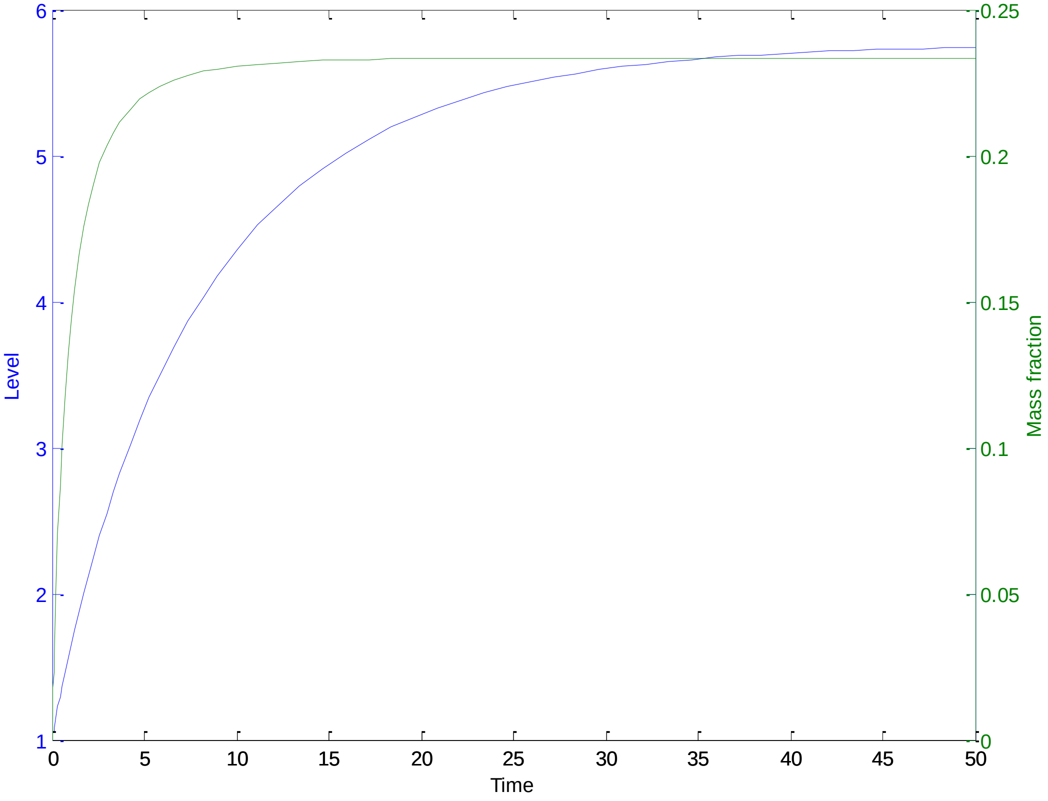
```
function f = binary_mixing(x)
w1 = 10;
x1 = 0.1;
w2 = 2;
x2 = 0.9;
rhoa = 10;
cv = 5;
h = x(1);
x3 = x(2);
f(1) = (w1+w2-cv*sqrt(h))/rhoa;
f(2) = (w1*(x1-x3)+w2*(x2-x3))/(rhoa*h);
```

Binary Mixing Tank

```
>> xss = fsolve(@binary_mixing,[10 0.5],[])
xss =
   5.7600    0.2333
>> df = @(t,x) binary_mixing(x);
>> [t,x]=ode45(df,[0 50],xss,[]);
>> plot(t,x)
>> [t,x]=ode45(df,[0 50],[1 0],[]);
>> ax=plotyy(t,x(:,1),t,x(:,2));
>> xlabel('Time')
>> ylabel(ax(1),'Concentration [kmol/m^{3}]');
>> ylabel(ax(1),'Level');
>> ylabel(ax(2),'Mass fraction');
```

Binary Mixing Tank





MATLAB: Nonlinear ODEs

In-class Exercise

Continuous Stirred Tank Chemical Reactor

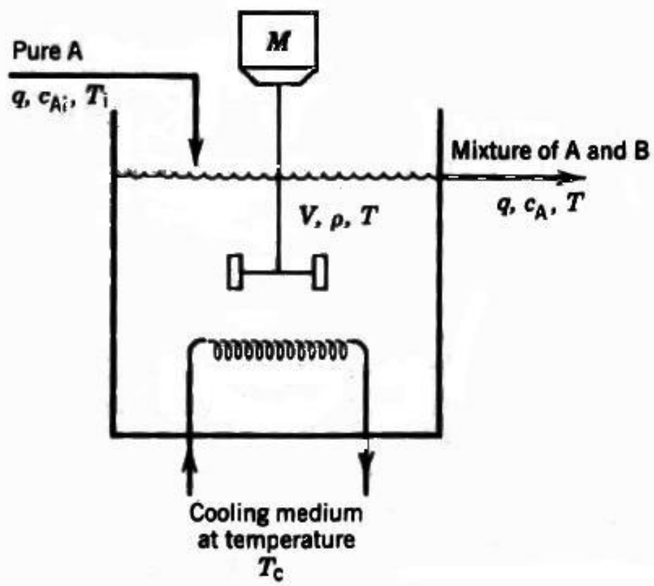


Figure 2.3. A nonisothermal continuous stirred-tank reactor.

- Reaction: A → B
- Assumptions
  - Pure A in feed
  - Perfect mixing
  - Negligible heat losses
  - Constant properties (ρ, C_p, ΔH, U)
  - Constant cooling jacket temperature

$$\frac{dC_A}{dt} = \frac{q}{V}(C_{Af} - C_A) - k_0 \exp(-E/RT)C_A$$

$$\frac{dT}{dt} = \frac{q}{V}(T_f - T) + \frac{(-\Delta H)k_0 \exp(-E/RT)C_A}{\rho C_p} + \frac{UA}{\rho C_p V}(T_j - T)$$