University of Washington
AMATH 301 Spring 2017
Instructor: Dr. King-Fai Li

Lecture Notes
Week 8

Solving coupled ODEs in Matlab

For coupled equations:

$$\frac{dY_1}{dt} = Y_1 - Y_2 + 2 \text{ and } \frac{dY_2}{dt} = -Y_1 + Y_2 + 4t$$

with $Y_1(t=0.4) = -1$, $Y_2(t=0.4) = 0$.

```
F=@(t,Y)[Y(1)-Y(2)+2 ;  -Y(1)+Y(2)+4*t];
t1 = 0.4;
tmax = 2;
[tout,Yout]=ode45(F,[t1  tmax],[-1  0]);
Y1out = Yout(:,1);
Y2out = Yout(:,2);
```

Stability of ODEs — Concept of Stiffness

Consider $\frac{dy}{dt} = f(t,y)$. Suppose there is an error in *y*: $y = y_a + \varepsilon$, where y_a is the approximation of *y* and ε is the corresponding error. Since $\frac{dy_a}{dt} = f(t,y_a)$, the ODE for ε is derived from

$$\frac{d(y_a+\varepsilon)}{dt} = f(t, y_a+\varepsilon) \approx f(t,y_a) + \left.\frac{\partial f}{\partial y}\right|_{t,y_a} \varepsilon$$

$$\frac{d\varepsilon}{dt} = \left.\frac{\partial f}{\partial y}\right|_{t,y_a} \varepsilon \equiv \lambda\varepsilon$$

where $\lambda = \left.\frac{\partial f}{\partial y}\right|_{t,y_a}$.

Example 1: Euler's method

Consider the Euler's method with a mesh size *h*. Then the error grows as

$$\varepsilon_{n+1} = \varepsilon_n + h\lambda\varepsilon_n = (1+h\lambda)\varepsilon_n = (1+h\lambda)^2\varepsilon_{n-1} = \cdots = (1+h\lambda)^n\varepsilon_1$$

Therefore, the error will exponentially grow to ∞ unless $|1+h\lambda|<1$. For the Euler's method to work, thus we require

1. $\lambda < 0$
2. Step-size must satisfy $h < \dfrac{2}{|\lambda|}$

$Q(h\lambda) \equiv 1 + h\lambda$ is called the stability function for the Euler's method and it defines the stiffness of the ODE.

A small step size *h*, and hence a lot more computations, is required for stiff equations.

Example 2: Implicit Euler method

$$\varepsilon_{n+1} = \varepsilon_n + h\lambda\varepsilon_n = \frac{\varepsilon_n}{1-h\lambda} = \frac{\varepsilon_{n-1}}{(1-h\lambda)^2} = \cdots = \frac{\varepsilon_1}{(1-h\lambda)^n}$$

Therefore, as long as $\lambda < 0$, the implicit Euler method is always stable or "stiff". In general implicit methods are stiff.

Stiff solvers in Matlab:

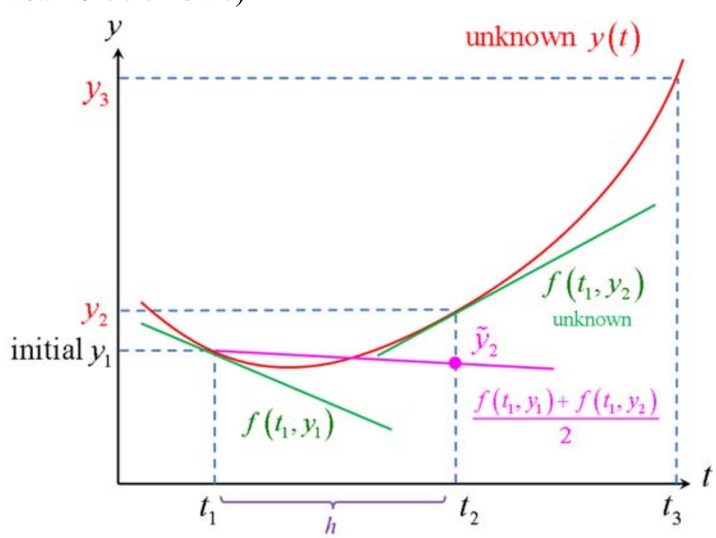
```
ode15s, ode23s, ode23t, ode23tb, ode15i
```

Implicit Trapezoidal Rule (similar but not identical to **`ode23tb`**)

$$y_2 = y_1 + \int_{t_1}^{t_2} f(t, y(t))\,dt$$
$$\approx y_1 + \frac{h}{2}\left[f(t_1, y_1) + f(t_2, y_2)\right]$$

Global errors ~ $O(h^2)$.

Use **`fzero`** or **`fsolve`** to solve for y_2.



Flame Model (From Mathworks.com)

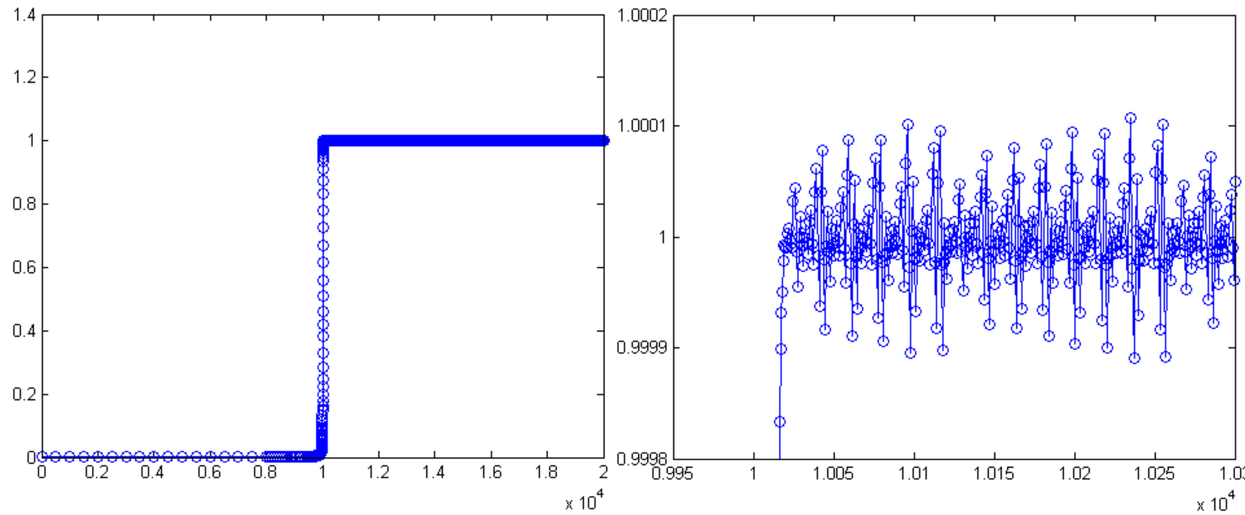
The flame size of a match before and after scratching the matchbox can be described by

$$\frac{dy}{dt} = y^2 - y^3$$

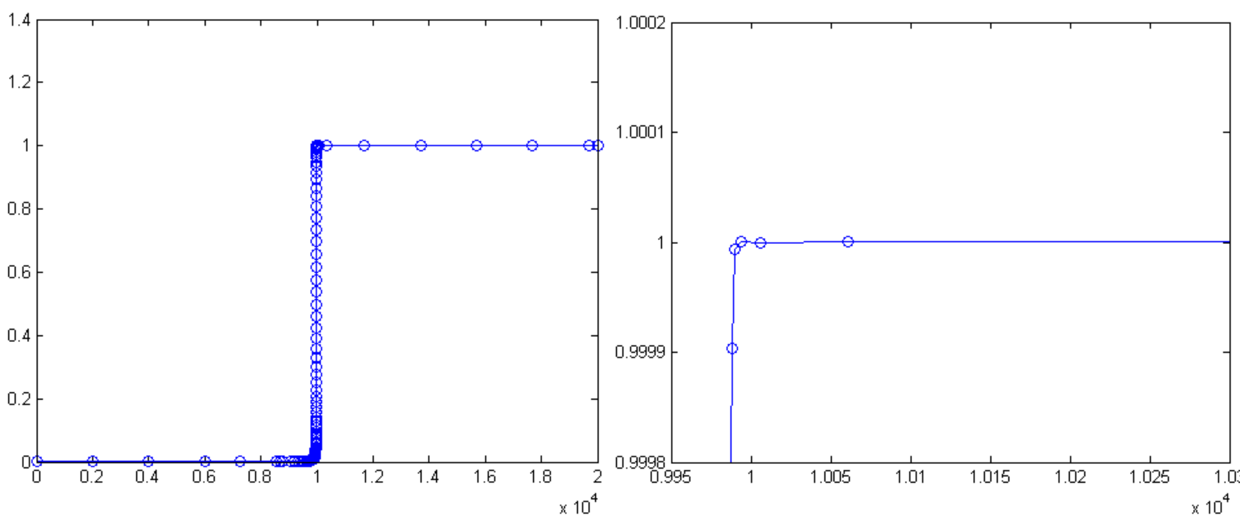
In the following demonstration, **ode45** will march slowly in time after the flame is set up after $t = \frac{1}{y_1}$ because of the stiffness; zooming in to the solution right after $t = \frac{1}{y_1}$ shows some unexpected oscillations. **ode23tb**, on the other hand, produces a smooth solution after $t = \frac{1}{y_1}$, and is much faster to run at an expense of a less accurate solution.

```
% Demonstration of a stiff system
% AMATH301, 2015 Winter Term, U Washington
% Instructor: Dr. King-Fai Li

clear all; close all; clc;
% Flame model described in Mathworks.com
F = @(t,y) y.^2 - y.^3 ;
Y1 = 0.0001;

% Analytic solution
t = linspace(0,2/y1,1000);
a = 1/y1 - 1;
plot(t,1./(1+lambertw(a*exp(a-t))),'-r','LineWidth',1.2);
axis([t(1) t(end) 0 1.2])
hold on

% Plot as ODE45 calculates
ode45(F,[0 2/y1],y1);

figure(2)
plot(t,1./(1+lambertw(a*exp(a-t))),'-r','LineWidth',1.2);
axis([t(1) t(end) 0 1.2])
hold on

% Plot as ODE23tb calculates
ode23tb(F,[0 2/y1],y1);
```

Solution from **ode45**



Solution from **ode23tb**



Stability of Coupled ODE System

An example with 2 variables:

$$\frac{d\mathbf{Y}}{dt} = \mathbf{F}(t, \mathbf{Y}), \text{ where } \mathbf{Y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} \text{ and } \mathbf{F}(t, \mathbf{Y}) = \begin{bmatrix} f_1(t, y_1, y_2) \\ f_2(t, y_1, y_2) \end{bmatrix}$$

$$\frac{d\boldsymbol{\varepsilon}}{dt} = \mathbf{J}\boldsymbol{\varepsilon}, \text{ where } \mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial y_1} & \frac{\partial f_1}{\partial y_2} \\ \frac{\partial f_2}{\partial y_1} & \frac{\partial f_2}{\partial y_2} \end{bmatrix} \text{ is the Jacobian.}$$

For the Euler's method, $\boldsymbol{\varepsilon}_{n+1} = (\mathbf{I} + h\mathbf{J})^n \boldsymbol{\varepsilon}_1$.

Decompose $\mathbf{J}$ into its eigenvalue decomposition: $\mathbf{J} = \mathbf{VSV}^{-1}$, where $\mathbf{S} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$ is a diagonal matrix containing the (complex) eigenvalues. Then

$$\begin{aligned} \boldsymbol{\varepsilon}_{n+1} &= (\mathbf{I} + h\mathbf{J})^n \boldsymbol{\varepsilon}_1 \\ &= (\mathbf{VV}^{-1} + h\mathbf{VSV}^{-1})^n \boldsymbol{\varepsilon}_1 \\ &= \mathbf{V}(\mathbf{I} + h\mathbf{S})^n \mathbf{V}^{-1}\boldsymbol{\varepsilon}_1 \qquad \because (\mathbf{I} + h\mathbf{S}) \text{ is diagonal} \end{aligned}$$

Therefore the Euler's method is stable if all λ satisfies $|1 + h\lambda| < 1$.

Chaotic Systems

$$\frac{d\mathbf{Y}}{dt} = \mathbf{F}(t, \mathbf{Y})$$

$\mathbf{F}(t, \mathbf{Y})$ is nonlinear in t and $\mathbf{Y}$.

Example:

$$\begin{aligned} \frac{dx}{dt} &= \sigma(y - x) \\ \frac{dy}{dt} &= x(\rho - z) - y \\ \frac{dz}{dt} &= xy - \beta z \end{aligned} \qquad \text{with} \qquad \begin{aligned} \sigma &= 10 \\ \beta &= \frac{8}{3} \\ \rho &= 28 \end{aligned} \; .$$

The system is deterministic, but has little predictive power.

1. Deterministic: Given an initial condition, the system has only a unique evolution path.
2. Loss of predictive power: Given a tiny different initial condition, the system has a completely different evolution path. The difference grows exponentially.

#2 persists even if you have a perfect computer with infinite precision and a perfect ODE solver with no error term. So the chaotic behavior is intrinsic and has nothing to do with numerical errors or stability of the ODE solver or stiffness.