Lecture 6a: Solution Strategies for Lumped Parameter Models

Rafiqul Gani
(Plus material from Ian Cameron)

PSE for SPEED
Skyttemosen 6, DK-3450 Allerod, Denmark
rgani2018@gmail.com

www.pseforspeed.com

Overview

- Types of problems encountered
- Dealing with higher order equations
- Types of numerical methods
- Error and stability of solution
- Key techniques for ODEs
  - Runge-Kutta (single step methods)
  - Linear multistep methods
- Solution of DAE systems

Types of Problems

- Ordinary differential equations (ODEs)
- Differential-algebraic equations (DAEs)
- Algebraic equations (AEs)

Example Problems (exercise with MoT)

❖ Ordinary differential equations (ODEs)

Chemical reaction

$$
\begin{aligned}
\dot{y}_1 &= k_1 y_1 + k_2 y_2 y_3 & \quad y_1(0) &= 1 \\
\dot{y}_2 &= k_1 y_1 - k_2 y_2 y_3 - k_3 y_2^2 & \quad y_2(0) &= 0 \\
\dot{y}_3 &= k_3 y_2^2 & \quad y_3(0) &= 0
\end{aligned}
$$

❖ Differential - algebraic equations (DAEs)

Tank dynamics

$$
\begin{aligned}
\dot{z} &= (F_1 - F_2)/A \\
F_1 &= C_V \sqrt{(P_1 - P_2)} \\
F_2 &= C_V \sqrt{(P_2 - P_3)} \\
P_2 &= P_0 + \rho g z
\end{aligned}
$$

❖ Algebraic equations (AEs)

Chemical reaction

steady state

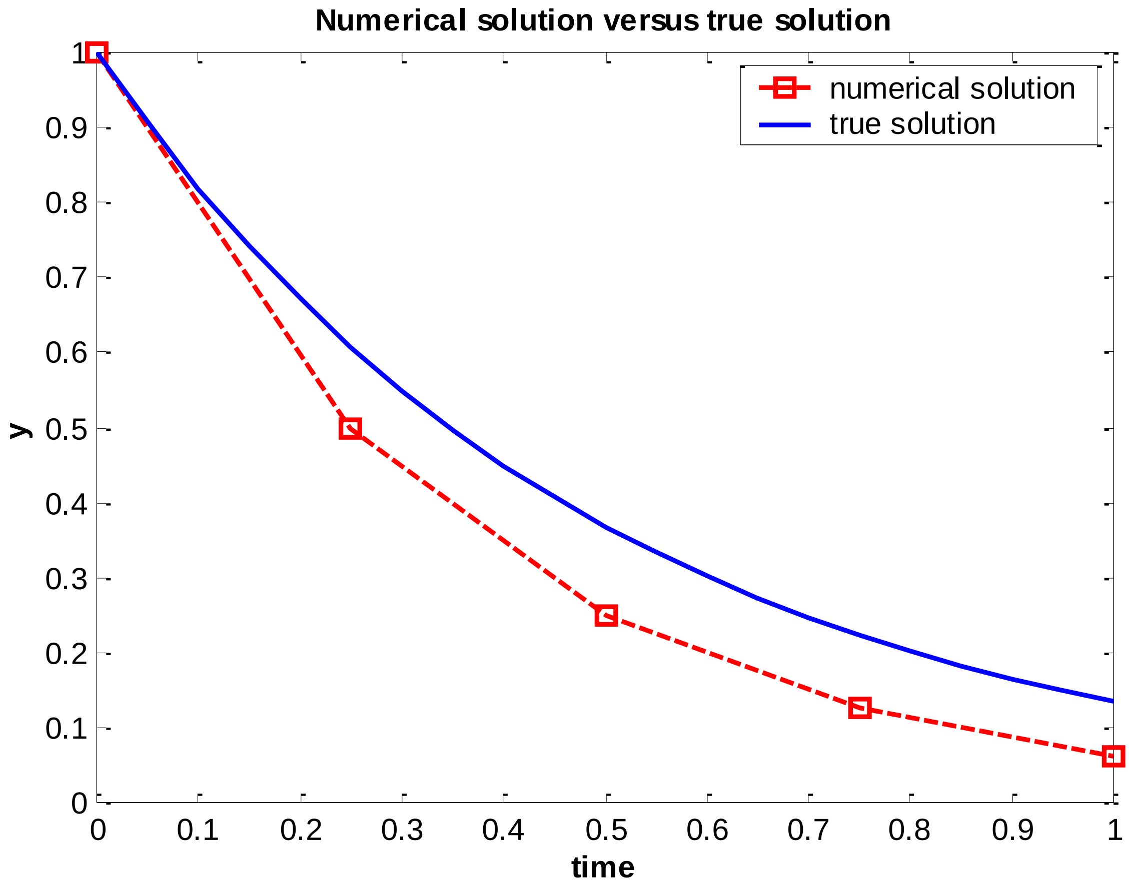
$$
\begin{aligned}
0 &= k_1 y_1 + k_2 y_2 y_3 \\
0 &= k_1 y_1 - k_2 y_2 y_3 - k_3 y_2^2 \\
0 &= k_3 y_2^2
\end{aligned}
$$

Numerical methods - basics

- Method which produces *discrete* solutions for a continuous model
- Essentially involves solving sets of difference equations at each step
- Methods have limited accuracy
- Methods have different characteristics
  - Order (accuracy)
  - Form
  - Stability

Numerical solution – discrete character

ODE $\dot{y} = -2y \qquad y(0) = 1$



Higher-order equations

- Need to convert to first order system
- Done using simple transformation

For $y^{[n]} + f(y^{[n-1]}, \ldots, y^{[2]}, y, t) = 0$

Let $y_i = y^{[i-1]} = \dfrac{d^{i-1}y}{dt^{i-1}}$

Then we obtain :

$$\frac{dy_1}{dt} = y_2$$

$$\frac{dy_2}{dt} = y_3$$

$\ldots$

$$\frac{dy_n}{dt} = f(y_1, y_2, \ldots, y_n, t)$$

Example of transformation

Second order function

$$\tau^2 \frac{d^2 y}{dt^2} + 2\xi\tau \frac{dy}{dt} + y = GI$$

$$y(0) = 0\,; \quad \frac{dy(0)}{dt} = 0$$

Make the transformations

$$y_1 = y \quad \Rightarrow \quad \dot{y}_1 = \frac{dy}{dt} = y_2$$

$$y_2 = \frac{dy}{dt} \quad \Rightarrow \quad \dot{y}_2 = \frac{d^2 y}{dt^2}$$

Substitute and rearrange

$$\tau^2\, \dot{y}_2 + 2\xi\tau\, y_2 + y_1 = GI$$

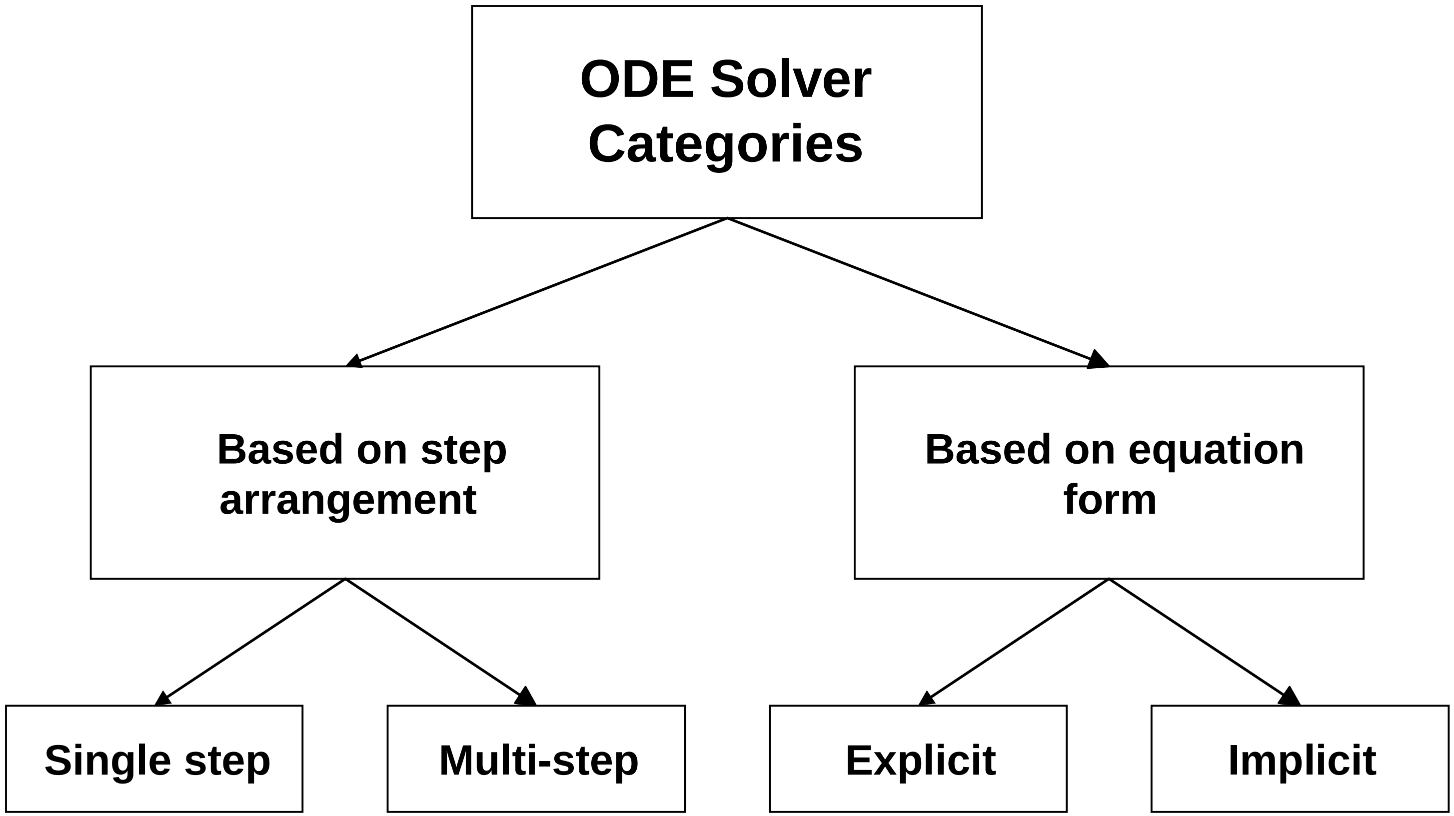
$$y_1(0) = 0\,; \;\; y_2(0) = 0$$

then

$$\dot{y}_1 = y_2\,; \qquad\qquad y_1(0) = 0$$

$$\dot{y}_2 = (GI - y_1)/\tau^2 - 2\xi\, y_2/\tau\,; \qquad y_2(0) = 0$$

Categories of methods



Single step methods

❖ Single step methods

- Euler's method (1768)

$$y_n = y_{n-1} + h_n f(y_{n-1}, t_{n-1})$$

- Runge-Kutta

Note: <u>a</u>, <u>b</u> & <u>c</u> are constants of the method

$$y_n = y_{n-1} + h_n \sum_{i=1}^{s} b_i k_i$$

$$k_i = f(t_{n-1} + c_i h_n, y_{n-1} + \sum_{j=1}^{s} a_{ij} k_j)$$

Multistep Methods

❖Linear multi-step methods

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h \sum_{j=0}^{k} \beta_j f_{n+j}$$

for k = 2 & $\alpha_2 = 1$

$$y_{n+2} = -(\alpha_0 y_n + \alpha_1 y_{n+1}) + h(\beta_0 f_n + \beta_1 f_{n+1} + \beta_2 f_{n+2})$$

Euler explicit: $\alpha_0 = 0$; $\alpha_1 = 1$; $\alpha_2 = 1$; $\beta_0 = 0$; $\beta_1 = 1$: $\beta_2 = 0$

Euler implicit: $\alpha_0 = 0$; $\alpha_1 = 1$; $\alpha_2 = 1$; $\beta_0 = 0$; $\beta_1 = 0$: $\beta_2 = 1$

Explicit - implicit methods

- Euler's method (explicit)

$$y_n = y_{n-1} + h_n f(y_{n-1}, t_{n-1})$$

Explicit Method

Given: h_n; y_{n-1} at t_{n-1}

1. Calculate $f(y_{n-1}, t_{n-1})$
2. Calcualte y_n

- Backward Euler method (implicit)

$$y_n = y_{n-1} + h_n f(y_n, t_n)$$

Implicit Method

Given: h_n; y_{n-1} at t_{n-1}

1. Predict y_n (k)
2. Correct y_n through N-R scheme

Newton-Rahpson solution scheme

$$\left(1 - h\gamma \frac{\partial f}{\partial y}\right)\Delta y^{k+1} = -F(y^k)$$

$$\Delta y^{k+1} = y^{k+1} - y^k \qquad k = 0,1,\ldots$$

$$F = y_n - h_n f(y_n, t_n) - y_{n-1} = 0$$

$\mathbf{\hat{J} = dF/dy}$

Characteristics of numerical methods

- All methods represent the Taylor series to some order

$$y(t+h) = y(t) + h\frac{dy}{dt} + h^2\frac{d^2y}{dt^2} + h^3\frac{d^3y}{dt^3} + \Lambda + h^n\frac{d^ny}{dt^n} + O(h^{n+1})$$

- All methods have a particular accuracy

Euler: $$y_n = y_{n-1} + h_n f(y_{n-1}, t_{n-1})$$ Order 1 accurate

Cause of error

- All methods introduce a truncation error

Euler: $$l_n = h_n^{\ 2}\frac{d^2y}{dt^2} + \Lambda$$ Order 2 error

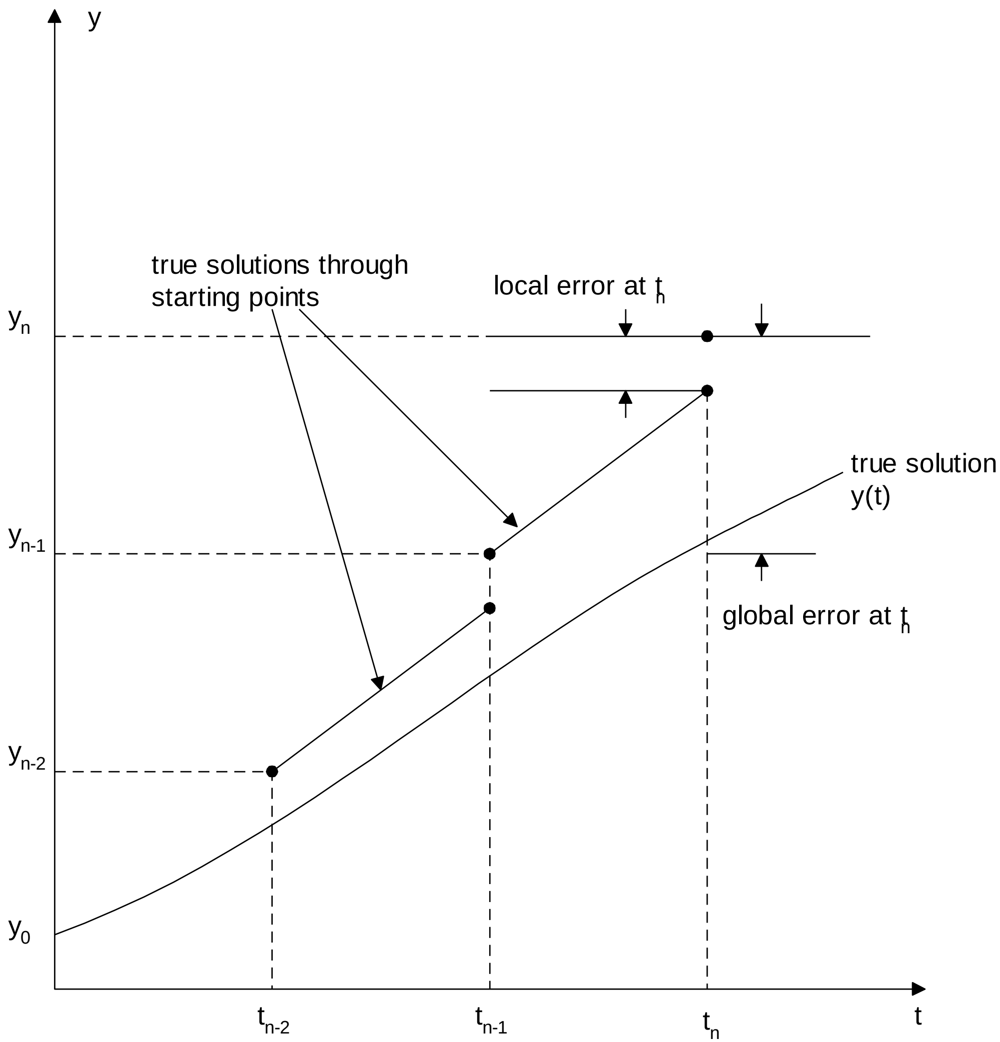
Measure of the size of error

Error and stability of solution

- Causes of error
  - accuracy of the method
  - size of step
  - type of method (explicit, implicit)
  - character of the problem

- Stability
  - Control of error each step

Local and global error



- ❖ Local error is the error introduced per step
- ❖ Global error is the accumulated error

(measures the difference between the numerical solution and the true solution)

True versus numerical solution

- Use a simple problem

$$y' = -Ay \qquad y(0) = y_0$$

- True solution

$$y(t_n) = e^{-Ah} y(t_{n-1}) = S(Ah) y(t_{n-1})$$

Difference between them is the error

- Numerical solution (e.g. Euler's method)

$$y_n = y_{n-1} + h(-Ay_{n-1}) = (1 - Ah) y_{n-1} = K(Ah) y_{n-1}$$

How global error occurs

- Global error $$\varepsilon_n = y(t_n) - y_n$$

- In terms of key operators S(z), K(z)

$$\begin{aligned}
\varepsilon_n &= S(Ah)y(t_{n-1}) - K(Ah)y_{n-1} \\
&= S(Ah)y(t_{n-1}) - K(Ah)y(t_{n-1}) - K(Ah)y_{n-1} + K(Ah)y(t_{n-1}) \\
&= K\varepsilon_{n-1} + (S-K)y(t_{n-1}) \\
\varepsilon_n &= K\varepsilon_{n-1} + Ty(t_{n-1})
\end{aligned}$$

T depends on the method accuracy

K depends on the **problem** AND the **method** used

Controlling error

- Need to ensure $\|K\| \le 1$
- Also control the truncation error $\|Ty\| = 0(h^{p+1})\|y\|$

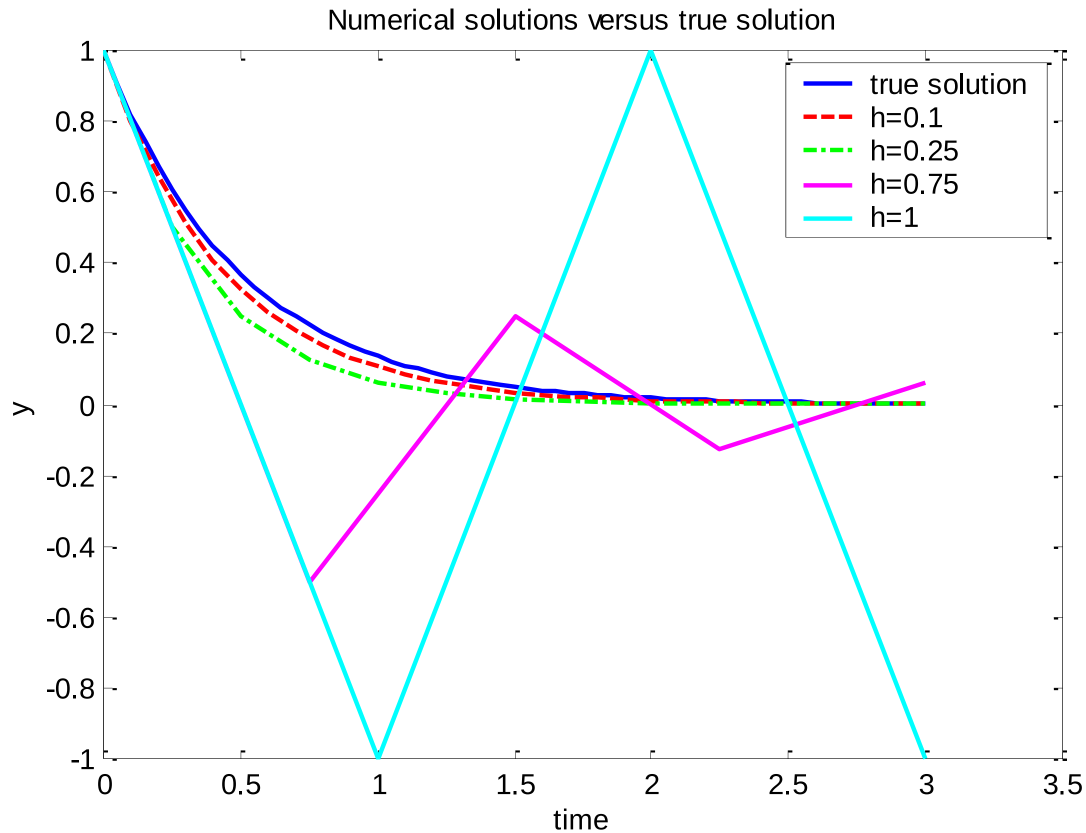
Euler's method for:

$$\dot{y} = -2y \qquad y(0) = 1$$

$$y(t_n) = e^{-2h} y(t_{n-1})$$

$$y(t_n) = (1 - 2h) y(t_{n-1})$$

abs (K) = 0.5 at h = 0.25



Stability Regions for Numerical ODE Solvers

- Each method has a region where error propagation is stable (A-stability region)
- The region is where $|K| < 1$
- K depends on the steplength h, method and problem
  - Euler method gives: K = (1+Ah)
  - Backward Euler gives: K = 1/(1-Ah)
- The key factor of the problem is the eigen-values λ_i
- The region can be plotted on the complex plane
- The region can be defined by a simple test problem

Defining the A-stable region – Euler method

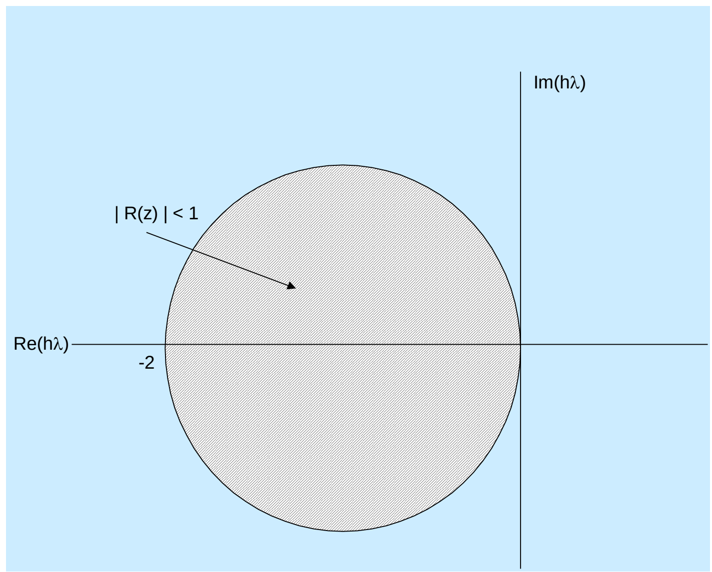
Test problem $\dot{y} = \lambda y$

Euler (explicit) method $y_n = y_{n-1} + h_n f(y_{n-1}, t_{n-1})$

gives $y_n = y_{n-1} + h\lambda\, y_{n-1} = (1 + h\lambda) y_{n-1} = K(h\lambda) y_{n-1}$

K = (1+λh)

$|K| < 1$

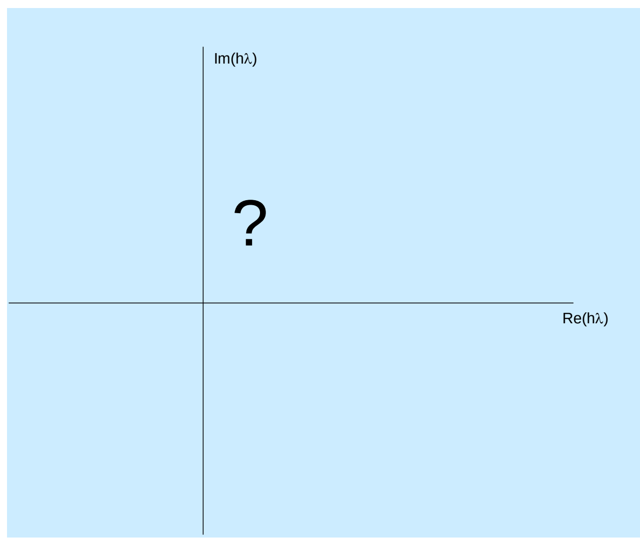


A-stable region – Backward Euler method

Test problem $\dot{y} = \lambda y$

Euler method $y_n = y_{n-1} + h_n f(y_n, t_n)$

gives $y_n = y_{n-1} + h\lambda\, y_n = ??$



A-stable region – Backward Euler method

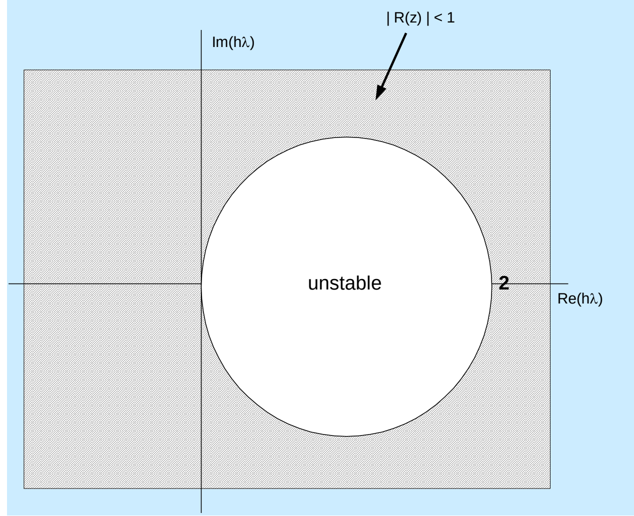
Test problem $$\dot{y} = \lambda y$$

Euler method $$y_n = y_{n-1} + h_n f(y_n, t_n)$$

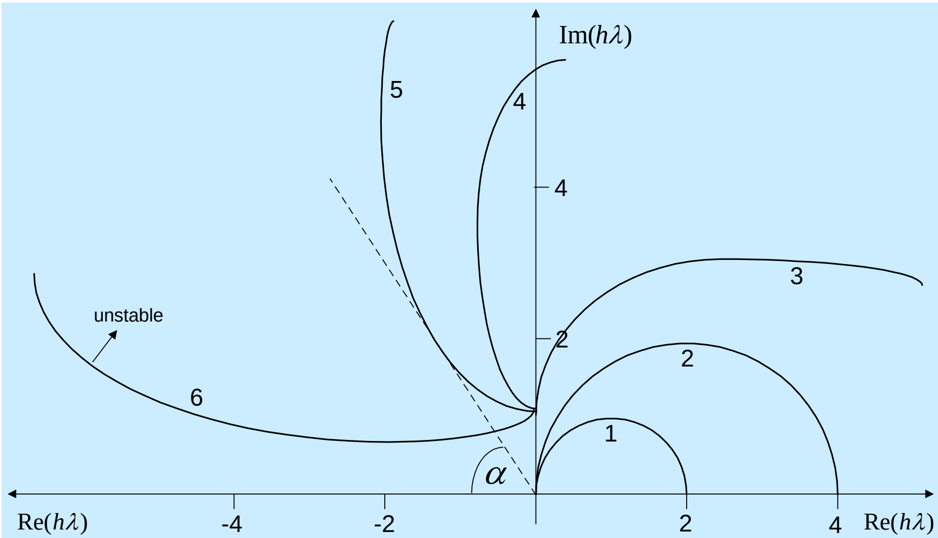
gives $$y_n = y_{n-1} + h\lambda\, y_n \longrightarrow y_n = \frac{1}{(1-h\lambda)} y_{n-1}$$

K = 1/(1-Ah)

$$|K| < 1$$

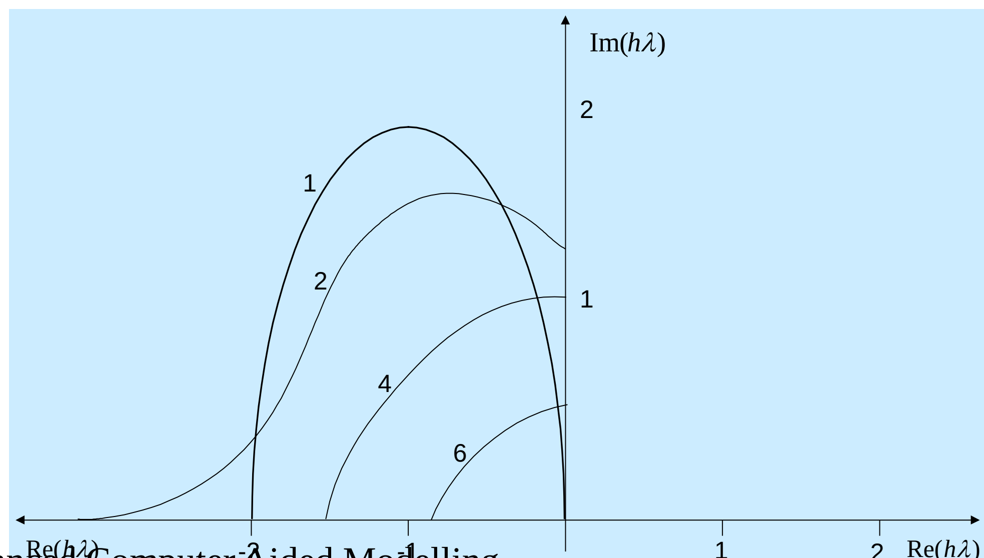


Stability regions for Linear-Multistep methods



BDF methods (backward differentiation formula)

Adams methods (similar to implicit Euler method



Stability summary

- Every problem has an eigen-value spectrum
  - Non-stiff problems have a narrow spectrum
  - Stiff problems have a wide spectrum
- The largest eigen-value determines stability for a numerical method
- The largest step-length depends on the numerical method and the maximum eigen-value of the problem being solved
- Explicit methods always have limited stability regions
- Implicit methods can have very large stability regions

Key numerical methods

- Solvers for AEs
- Euler's method
  - MATLAB code easy to write **(or MoT)**
  - Terrible for stiff problems
  - Low accuracy
- Runge-Kutta methods
  - MATLAB codes ODE23, ODE45 **(see MoT options)**
  - Good for non-stiff problems
  - High accuracy
- Linear multi-step methods
  - Adams methods for non-stiff problems
  - Backward differentiation formulae (BDF) for stiff problems
  - High accuracy
  - Poor on highly oscillatory problems

Solving differential-algebraic systems

$$\frac{dy}{dt} = f(y, z, t)$$

$$0 = g(y, z, t)$$

- Algebraic substitution
- Explicit ODE solvers
- Fully implicit solvers
- Structuring

Method 1: Algebraic substitution

$$\frac{dh}{dt} = (F_1 - F_2)/A$$

$$F_1 = C_{V_1}\sqrt{P_1 - P_2}$$

$$F_2 = C_{V_2}\sqrt{P_2 - P_3}$$

$$P_2 = P_0 + \rho g h$$

Substitute algebraic relations to get

$$\frac{dh}{dt} = \frac{1}{A}\left[C_{V_1}\left(P_1 - P_0 - \rho g h\right)^{\frac{1}{2}} - C_{V_2}\left(P_0 + \rho g h - P_3\right)^{\frac{1}{2}}\right]$$

Method 2: Explicit ODE solver

$$y' = f(y, z, t)$$

$$0 = g(y, z, t)$$

- Know y_n at t_n
- Solve $0=g(y_n,z_n,t_n)$ for z_n
- Evaluate ODEs (using ODE45 etc.)
- Advance step: y_n to y_{n+1}
- Return to step 1 or terminate at t_f

Method 3: Implicit DAE solvers

Backward Euler and other implicit methods, solve:

$$y = h\gamma f(y) + \psi$$

$h =$ current steplength

$\psi =$ known information (eg. y_n)

f(y) = right handside function

Solution of Backward Euler and other implicit methods

$$\left(1 - h\gamma \frac{\partial f}{\partial y}\right)\Delta y^{k+1} = -F(y^k)$$

$$\Delta y^{k+1} = y^{k+1} - y^k \qquad \mathrm{k} = 0,1,\ldots$$

Implicit DAE solvers Part (2)

Now extend to DAE set by adding the algebraic system.

$$\begin{bmatrix} I - h\gamma \dfrac{\partial f}{\partial y} & -h\gamma \dfrac{\partial f}{\partial z} \\ -h\gamma \dfrac{\partial g}{\partial y} & -h\gamma \dfrac{\partial g}{\partial z} \end{bmatrix} \begin{bmatrix} \Delta y^{k+1} \\ \Delta z^{k+1} \end{bmatrix} = \begin{bmatrix} -F(y^k, z^k, t \\ h\gamma g(y^k, z^k, t \end{bmatrix}$$

$$\Delta y^{k+1} = y^{k+1} - y^k$$

$$\Delta z^{k+1} = z^{k+1} - z^k$$

Hence: Iterate both y and z variables <u>simultaneously</u>

Implicit DAE solvers Part (3)

Now extend to DAE set by adding the algebraic system.

$$\begin{bmatrix} I - h\gamma\dfrac{\partial f}{\partial y} & -h\gamma\dfrac{\partial f}{\partial z} \\ -h\gamma\dfrac{\partial g}{\partial y} & -h\gamma\dfrac{\partial g}{\partial z} \end{bmatrix} \begin{bmatrix} \Delta y^{k+1} \\ \Delta z^{k+1} \end{bmatrix} = \begin{bmatrix} -F(y^k, z^k, t \\ h\gamma g(y^k, z^k, t \end{bmatrix}$$

$$\Delta y^{k+1} = y^{k+1} - y^k$$

$$\Delta z^{k+1} = z^{k+1} - z^k$$

Hence: Iterate both y and z variables <u>simultaneously</u>

Method 4: Exploiting structure

Basis:
We analyze the algebraic equation set to try and obtain a sequential calculation of the algebraic variables.

$$z_1 = q_1(y)$$
$$z_2 = q_2(z_1, y)$$
$$\ldots$$
$$z_n = q_n(z_1, \ldots, z_{n-1}, y)$$
$$then$$
$$y' = f(y, z, t)$$

Partitioning and precedence order

Steps:

❖ Establish an output assignment for the algebraic equations

❖ Generate partition(s) and precedence order

How?

❖ Use the incidence matrix

❖ Carry out a digraph analysis

Output assignment

Goal:

Assign to each equation an "output" variable which is calculated knowing all other variables in the equation.

Steps

- Generate an incidence matrix J_{ij}
- Permute rows of J to get a matrix B with diagonal with non-zero elements

$$B = R\ J$$

Output assignment example

$$g_1 = F_1 - C_{V_1}\sqrt{P_1 - P_2}$$

$$g_2 = F_2 - C_{V_2}\sqrt{P_2 - P_3}$$

$$g_3 = P_2 - P_0 - \rho g h$$

Unknowns are P_2, F_1, F_2

$$J = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} = RJ$$

Every row and very column can have only one element with 1 and all other elements must be 0. Only one R-matrix when multiplied with the J-matrix will give the lower-tridiagonal B-matrix. Check the following R-matrix

$$R = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

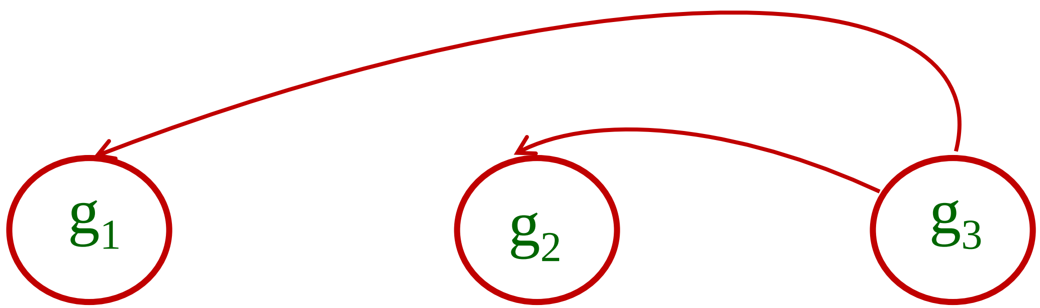
Digraph representation

- Let there be *n* nodes corresponding to each equation
- If the output variable of node *i* is used in node *j*, draw a directed arc between *i* and *j*.

$$g_1 = F_1 - C_{V_1}\sqrt{P_1 - P_2}$$

$$g_2 = F_2 - C_{V_2}\sqrt{P_2 - P_3}$$

$$g_3 = P_2 - P_0 - \rho g h$$

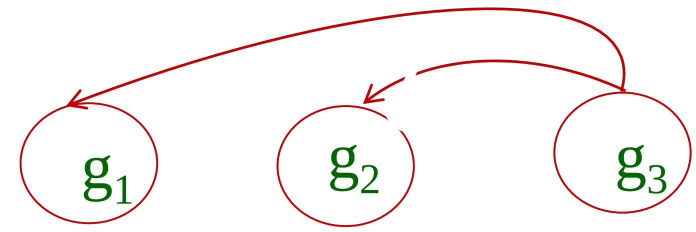


Partitioning and precedence order

- ❖ Start at any node
- ❖ Trace outputs until:
  - no node with output
    - *delete node and add to a list*

    or
  - a former node is revisited (loop made)
    - *merge all loop nodes*
    - *rearrange digraph and continue*
- ❖ Stop when all nodes on list
  - list contains partitions and precedence order

1.



g_3 to g_1, no output

List

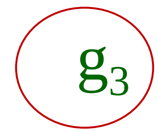
g_1

2.

g_1
g_2

g_3 to g_2, no output

3.



g_3 no output

$$g_1 = F_1 - C_{V_1}\sqrt{P_1 - P_2}$$
$$g_2 = F_2 - C_{V_2}\sqrt{P_2 - P_3}$$
$$g_3 = P_2 - P_0 - \rho g h$$

g_1
g_2
g_3

Hence: 3 partitions & precedence order is g_3, g_2, g_1

Example: Numerical solution

- Solution of the DAE involves:
  - solve for P_2 using g_3
  - solve for F_2 using g_2
  - solve for F_1 using g_1
  - evaluate right hand side of the ODE
  - advance the solution 1 step-length
- All this can be programmed into a simple Matlab.m function file or in ICAS-MoT

$$g_3 = P_2 - P_0 - \rho g z = 0$$
$$g_2 = F_2 - C_{V_2}\sqrt{P_2 - P_3} = 0$$
$$g_1 = F_1 - C_{V_1}\sqrt{P_1 - P_2} = 0$$
$$\frac{dz}{dt} = (F_1 - F_2) / A$$

$P_0 = 2$; $P_1 = 3$; $P_3 = 2$; $z(0) = 0$; $A = 1$; $\rho g = 2$

Modelling exercise 5a-1: Numerical solution

- Solve the reactor model (example 9.8 from book of Fogler)
  - model equations will be supplied in class
- Solve mixer model (from exercise-1)
- Solve ice-cube model (from exercise 2)
- Solve generated models from exercise 3
  - In each case check the eigen-values for ODE/DAE systems

Modelling exercise 5a-2: Numerical solution

- Solve the Williams-Otto plant simulation and optimization problem
  - Use the supplied MoT-file
  - Solve the simulation problem and then the optimization problem