Numerical methods for dynamical systems

Julien Alexandre dit Sandretto

Department U2IS
ENSTA Paris
INF6561 2022-2023

ENSTA

IP PARIS

Contents

IVP

Recall our starting point is the IVP of ODE defined by

$$\dot{\mathbf{y}} = f(t, \mathbf{y}) \quad \text{with} \quad \mathbf{y}(0) = \mathbf{y}_0 \ , \tag{1}$$

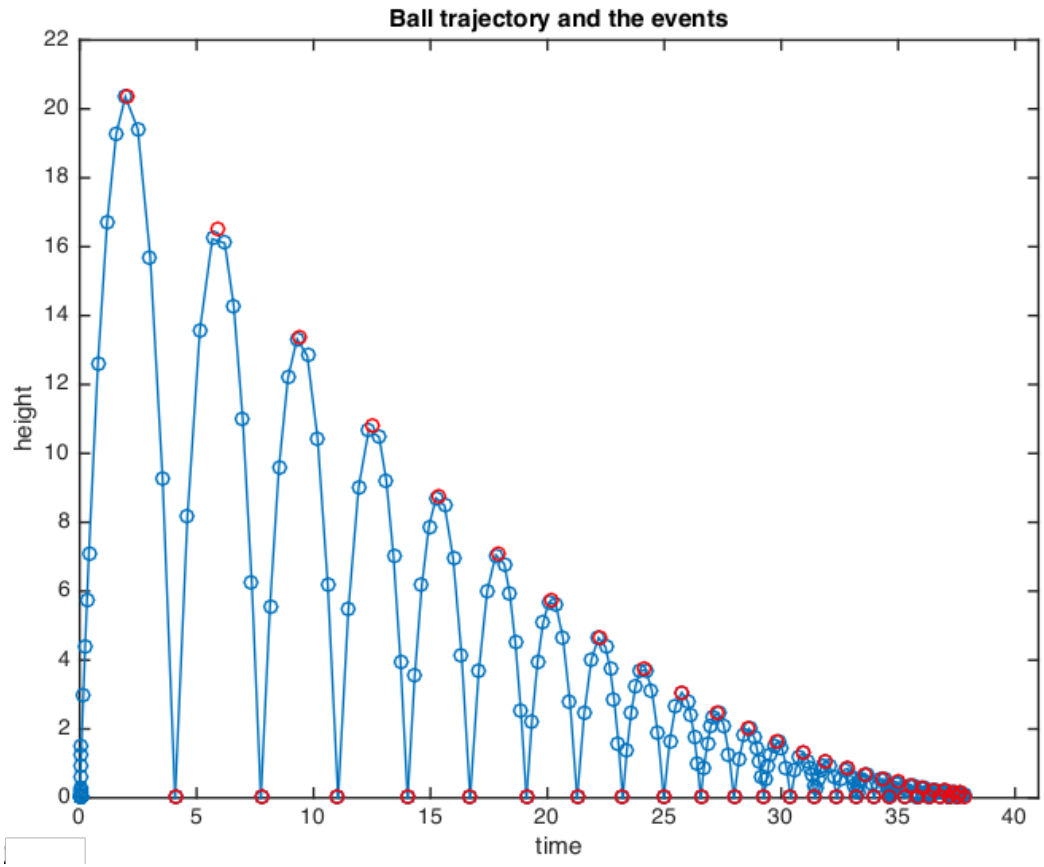
for which we want the solution $\mathbf{y}(t; \mathbf{y}_0)$ given by numerical integration methods i.e. a sequence of pairs $(t_i, \mathbf{y}_i)$ such that

$$\mathbf{y}_i \approx \mathbf{y}(t_i; \mathbf{y}_0) \ .$$

Why do we consider discontinuities?

Need to model

- non-smooth behaviors, *e.g.*, solid body in contact with each other
- interaction between computer and physics, *e.g.*, control-command systems
- constraints on the system, *e.g.*, robotic arm with limited space



Simulation with discontinuous systems

There are two kinds of events:

- **time event:** only depending on time as sampling
- **state event:** depending on a particular value of the solution of ODE or DAE.

To handle these events we need to adapt the simulation algorithm.

- Time events are known before the simulation starting. Hence we can use the step-size control to handle this.
- State event should be detect and handle on the fly. New algorithms are needed.

IVP with discontinuities

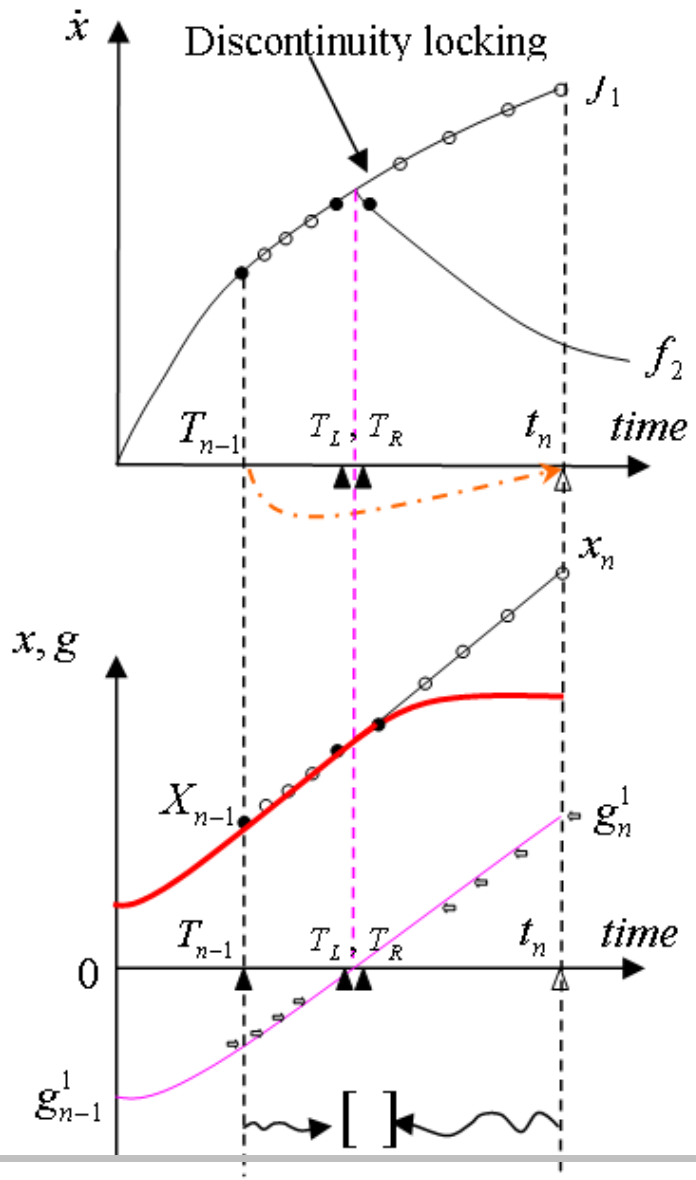
An IVP for ODE with discontinuities is defined by

$$\dot{\mathbf{y}} = \begin{cases} f_1(t, \mathbf{y}) & \text{if } g(t, \mathbf{y}) \geqslant 0 \\ f_2(t, \mathbf{y}) & \text{otherwise} \end{cases} \quad \text{with} \quad \mathbf{y}(0) = \mathbf{y}_0 \ , \tag{2}$$

for which we want the solution $\mathbf{y}(t; \mathbf{y}_0)$ given by numerical integration methods i.e. a sequence of pairs $(t_i, \mathbf{y}_i)$ such that

$$\mathbf{y}_i \approx \mathbf{y}(t_i; \mathbf{y}_0) \ .$$

Example: zero-crossing detection



A simple example

$$\dot{\mathbf{y}} = \begin{cases} f_1(t, \mathbf{y}) & \text{if } g(\mathbf{y}) \geqslant 0 \\ f_2(t, \mathbf{y}) & \text{otherwise} \end{cases} .$$

Legend

- Minor step state x
- Major step in X
- Search process
- Zc value pair
- First trial step from Tn-1 to tn
- Integration results

Zero-crossing event detection

Main steps

- **Detection** of zero-crossing event
  Is one of the zero-crossing changed its sign between $[t_n, t_n + h_n]$?
- **Localization**: if detection is true
  Bracket the most recent zero-crossing time using bisection method.
- **Pass through** the zero-crossing event in two steps:
  - Set the next major output to the left bound of the bracket time.
  - Reset the solver with the state estimate at the right bound of bracket time.

Ingredients for zero-crossing events – 1

Detection of the event.
We check that

$$g(t_n, \mathbf{y}_n) \cdot g(t_{n+1}, \mathbf{y}_{n+1}) < 0$$

We observe is there is a sign changement of the zero-crossing function g.
Remark this is a not robust method (is the sign changes twice for example)

Zero-crossing event detection

Main steps

- **Detection** of zero-crossing event
  Is one of the zero-crossing changed its sign between $[t_n, t_n + h_n]$?
- **Localization**: if detection is true
  Bracket the most recent zero-crossing time using bisection method.
- **Pass through** the zero-crossing event in two steps:
  - Set the next major output to the left bound of the bracket time.
  - Reset the solver with the state estimate at the right bound of bracket time.

Ingredients for zero-crossing events – 2

Continuous extension (method dependent) to easily estimate state.
For example, `ode23` uses Hermite interpolation

$$p(t) = (2\tau^3 - 3\tau^2 + 1)\mathbf{y}_n + (\tau^3 - 2\tau^2 + \tau)(t_2 - t_1)f(\mathbf{y}_n) + (-2\tau^3 + 3\tau^2)\mathbf{y}_{n+1} + (\tau^3 - \tau^2)(t_2 - t_1)f(\mathbf{y}_{n+1})$$

with $\tau = \frac{t - t_n}{h_n}$

Zero-crossing event detection

Main steps

- **Detection** of zero-crossing event
  Is one of the zero-crossing changed its sign between $[t_n, t_n + h_n]$?
- **Localization**: if detection is true
  Bracket the most recent zero-crossing time using bisection method.
- **Pass through** the zero-crossing event in two steps:
  - Set the next major output to the left bound of the bracket time.
  - Reset the solver with the state estimate at the right bound of bracket time.

Ingredients for zero-crossing events – 2

The solve the equation

$$g(t, p(t)) = 0$$

instead of $g(t, y(t)) = 0$
Note: as this equation is 1D then algorithm as bisection or Brent's method can be used instead of Newton's iteration.

Zero-crossing event detection

Main steps

- **Detection** of zero-crossing event
  Is one of the zero-crossing changed its sign between $[t_n, t_n + h_n]$?
- **Localization**: if detection is true
  Bracket the most recent zero-crossing time using bisection method.
- **Pass through** the zero-crossing event in two steps:
  - Set the next major output to the left bound of the bracket time.
  - Reset the solver with the state estimate at the right bound of bracket time.

Ingredients for zero-crossing events – 3

Enclosing the time of event produce a time interval $[t^-, t^+]$ for which we have

- the left limit of the solution $\mathbf{y}(t^-)$
- an approximation of the right limit of the solution $\mathbf{y}(t^+)$ which is used as initial condition for the second dynamics

Simulation algorithm

Data: f_1 the dynamic, f_2 the dynamic, g the zero-crossing function, $\mathbf{y}_0$ initial condition, t_0 starting time, t_{end} end time, h integration step-size, tol

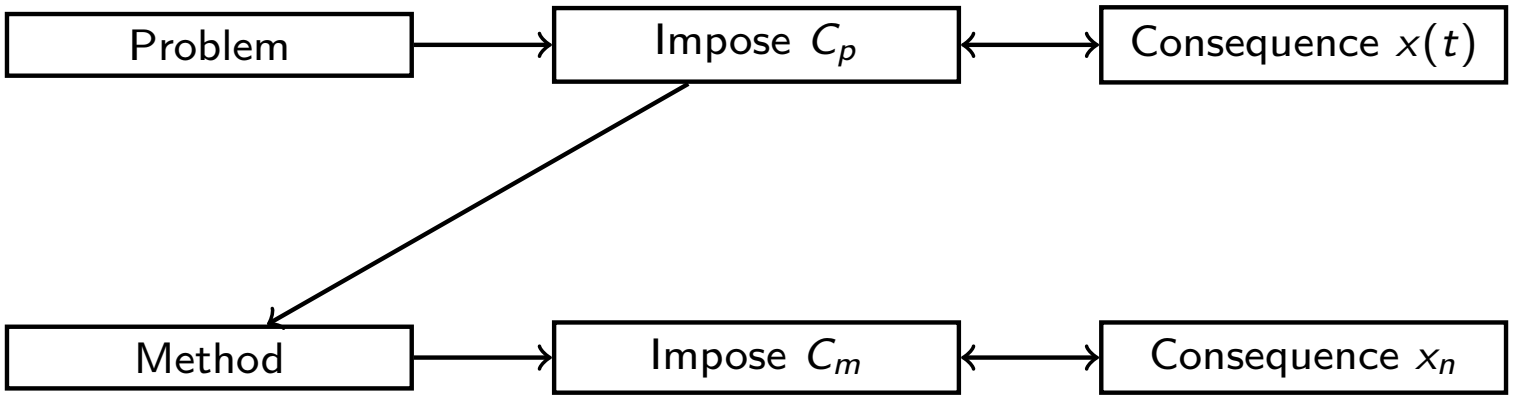
$t \leftarrow t_0$ $\mathbf{y} \leftarrow \mathbf{y}_0$ $f \leftarrow f_1$ **while** $t < t_{\text{end}}$ **do**
  Print(t, $\mathbf{y}$) $y_1 \leftarrow$ Euler(f,t,$\mathbf{y}$,h) $y_2 \leftarrow$ Heun(f,t,$\mathbf{y}$,h) **if** ComputeError($\mathbf{y}_1, \mathbf{y}_2$) is smaller than tol **then**
    if $g(\mathbf{y}) \cdot g(\mathbf{y}_1) < 0$ **then**
      Compute $p(t)$ from $\mathbf{y}$, $f(\mathbf{y})$, $\mathbf{y}_1$ and $f(\mathbf{y}_1)$ $[t^-, t^+] = \text{FindZero}\ (g(p(t)))$ Print $(t + t^-, p(t^-))$
      $f \leftarrow f_2$ $\mathbf{y} \leftarrow p(t^+)$ $t \leftarrow t + t^+$
    end
    $\mathbf{y} \leftarrow \mathbf{y}_1$ $t \leftarrow t + h$ $h \leftarrow \text{ComputeNewH}\ (h, \mathbf{y}_1, \mathbf{y}_2)$
  end
  $h \leftarrow h/2$
end

Remark

One-step methods are more robust than multi-step in case of discontinuities (starting problem)

Stability properties: a graphical view

Note: there are several kinds of stability.



From a generic point of view we have:

- Impose a certain conditions C_p on IVP which force the exact solution $x(t)$ to exhibit a certain stability
- Apply a numerical method on IVP
- **Question:** what conditions must be imposed on the method such that the approximate solution $(x_n)_{n\in\mathbb{N}}$ has the same stability property?

Total stability of IVP

Consider, a perturbed IVP

$$\dot{\mathbf{y}} = f(t, \mathbf{y}) + \delta(t) \qquad \text{with} \quad \mathbf{y}(0) = \mathbf{y}_0 + \delta_0 \quad \text{and} \quad t \in [0, b]$$

$(\delta(t), \delta_0)$ denotes the perturbations

Definition: totally stable IVP

From

- $(\delta(t), \delta_0)$ and $(\delta^*(t), \delta_0^*)$ two perturbations
- $\mathbf{y}(t)$ and $\mathbf{y}^*(t)$ the associated solutions

if

$$\forall t \in [0, b], \forall \varepsilon > 0, \exists K > 0,$$
$$\| \delta(t) - \delta^*(t) \| \leq \varepsilon \wedge \| \delta_0 - \delta_0^* \| \leq \varepsilon \Rightarrow \| \mathbf{y}(t) - \mathbf{y}^*(t) \| \leq K\varepsilon$$

then IVP is **totally stable**.

Zero stability of numerical methods

We consider the application of numerical method on a perturbed IVP so we have a perturbed numerical scheme

Definition: zero-stability

From

- δ_n and δ_n^* two discrete-time perturbation
- $\mathbf{y}_n$ and $\mathbf{y}_n^*$ the associated numerical solution

if

$$\forall n \in [0, N], \forall \varepsilon > 0, \exists K > 0, \forall h \in (0, h_0]$$
$$\| \delta_n - \delta_n^* \| \leq \varepsilon \Rightarrow \| \mathbf{y}_n - \mathbf{y}_n^* \| \leq K\varepsilon$$

then the method is **zero-stable**

In a different point of view, we want to solve $\dot{y} = 0$ with $y(0) = y_0$ and so numerical method should produce as a solution $y(t) = y_0$. (It is obvious for RK methods)

Zero stability for multi-step methods

First and second characteristic polynomials for linear multi-step methods are

$$\rho(z) = \sum_{i=0}^{k} \alpha_i z^i \qquad \text{and} \qquad \sigma(z) = \sum_{i=0}^{k} \beta_i z^i$$

Root condition

A linear multi-step method satisfies the **root condition** is the roots of the first characteristic polynomial ρ have modulus less than or equal to one and those of modulus one are simple.

Theorem

A multi-step method is zero stable is it satisfies the root condition.

Theorem

No zero-stable linear k-step method can have order exceeding $k + 1$

Consistency of numerical methods

We denote by $\Phi_f\left(t_n, \mathbf{y}_n; h\right)$ a Runge-Kutta method such that

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\Phi_f\left(t_n, \mathbf{y}_n; h\right)$$

If Φ_f is such that

$$\lim_{h\to 0} \Phi_f\left(t_n, \mathbf{y}_n; h\right) = f(t_n, \mathbf{y}_n) \ .$$

then the Runge-Kutta method is **consistent** to the IVP.
As a consequence, the truncation error is such that:

$$\lim_{h\to 0} \mathbf{y}(t_{n+1}) - \mathbf{y}_n - h\Phi_f\left(t_n, \mathbf{y}_n; h\right) = 0$$

Consistency for s-stage RK methods

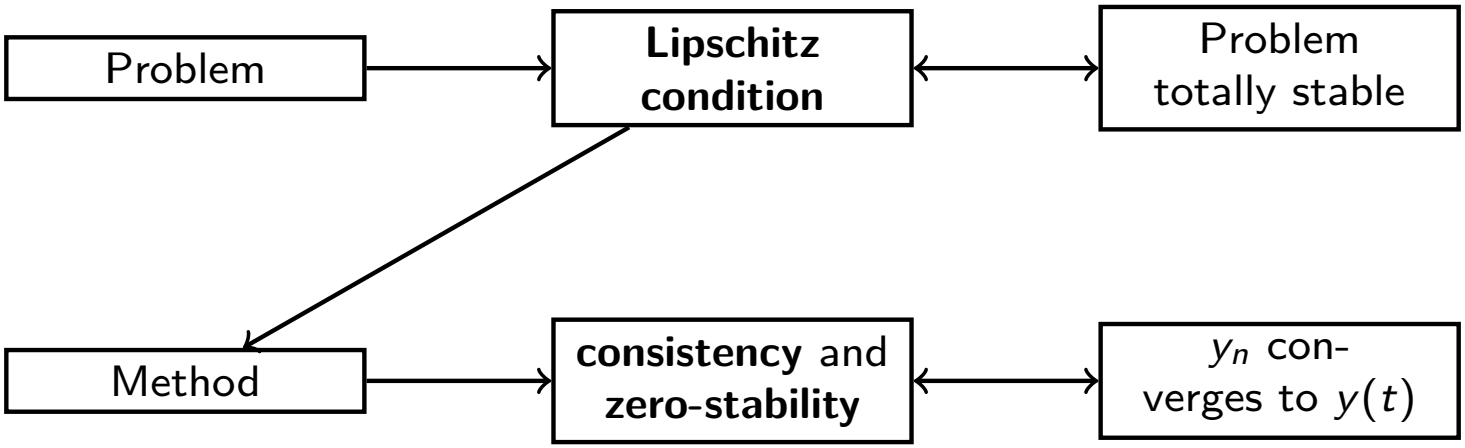
A necessary and sufficient condition is that

$$\sum_{i=1}^{s} b_i = 1$$

Convergence of numerical methods

A Runge-Kutta method is said **convergent** if

$$\lim_{h \to 0} \mathbf{y}_n = \mathbf{y}(t_n)$$



Linear stability

We consider the IVP:

$$\dot{y} = \lambda y \quad \text{with} \quad \lambda \in \mathbb{C}, \Re(\lambda) < 0$$

Applying a RK method, we get

$$y_{n+1} = R(\hat{h}) y_n \quad \text{with} \quad \hat{h} = \lambda h$$

$R(\hat{h})$ is called the *stability function* of the method.

Stability function of RK methods

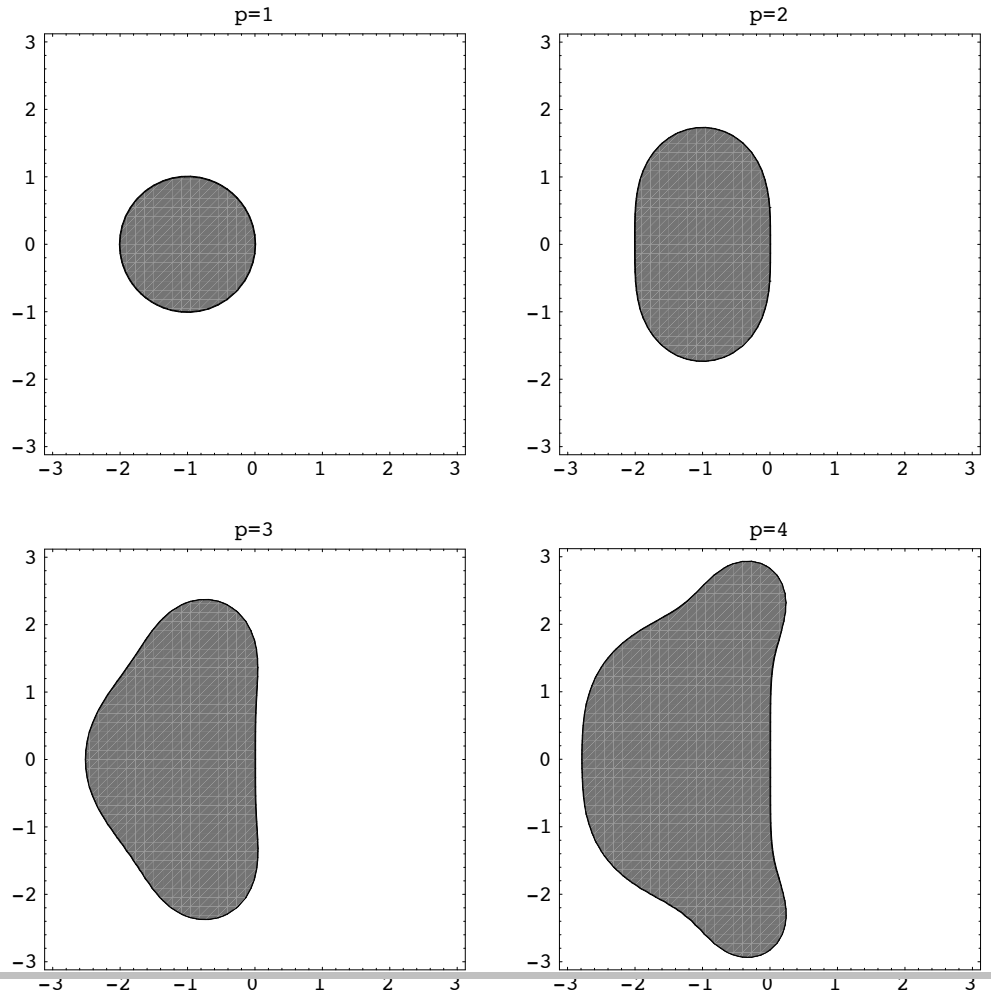
$$R(\hat{h}) = \frac{\det\left(I - \hat{h}A + \hat{h}\mathbb{1}b^t\right)}{\det\left(I - \hat{h}A\right)}$$

So, $\lim_{n\to\infty} x_n = 0$ when $|R(\hat{h})| < 1$

Linear stability of ERK – 1

The stability function for s-stage ($s = 1, 2, 3, 4 \Rightarrow p = s$) ERK is reduced to a polynomial function:

$$R(\hat{h}) = 1 + \hat{h} + \frac{1}{2!}\hat{h}^2 + \cdots + \frac{1}{s!}\hat{h}^s$$



Linear stability of ERK – 1

The stability function for s-stage ($s > 4 \Rightarrow p < s$) ERK is reduced to a polynomial function:

$$R(\hat{h}) = 1 + \hat{h} + \frac{1}{2!}\hat{h}^2 + \cdots + \frac{1}{p!}\hat{h}^p + \sum_{q=p+1}^{s} \gamma_q \hat{h}^q$$

with γ_q depending only on the coefficients of the ERK methods.
For example,

- for RKF45 ($s = 5$ and $p = 4$)

$$R(\hat{h}) = 1 + \hat{h} + \frac{1}{2!}\hat{h}^2 + \frac{1}{6}\hat{h}^3 + \frac{1}{24}\hat{h}^4 + \frac{1}{104}\hat{h}^5$$

- DOPIR54 ($s = 6$ and $p = 5$)

$$R(\hat{h}) = 1 + \hat{h} + \frac{1}{2!}\hat{h}^2 + \frac{1}{6}\hat{h}^3 + \frac{1}{24}\hat{h}^4 + \frac{1}{120}\hat{h}^5 + \frac{1}{600}\hat{h}^6$$

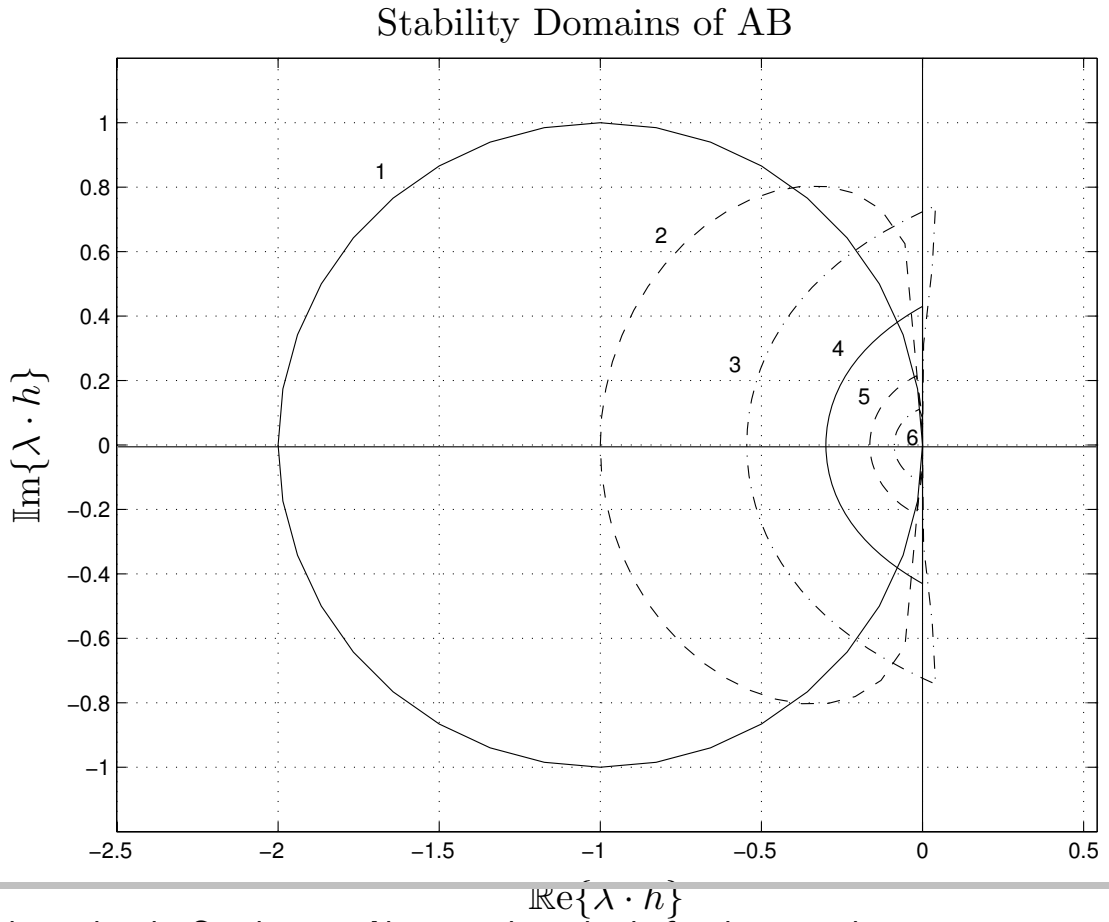
Linear stability of Adams-Bashworth methods

We consider the scalar linear IVP

$$\dot{y} = \lambda y \quad \text{with} \quad \lambda \in \mathbb{C}, \Re(\lambda) < 0$$

For linear problem, the **stability polynomial** of a multi-step method is

$$\pi(r, \hat{h}) = \rho(r) - \hat{h}\sigma(r) \quad \text{with} \quad \hat{h} = \lambda h$$



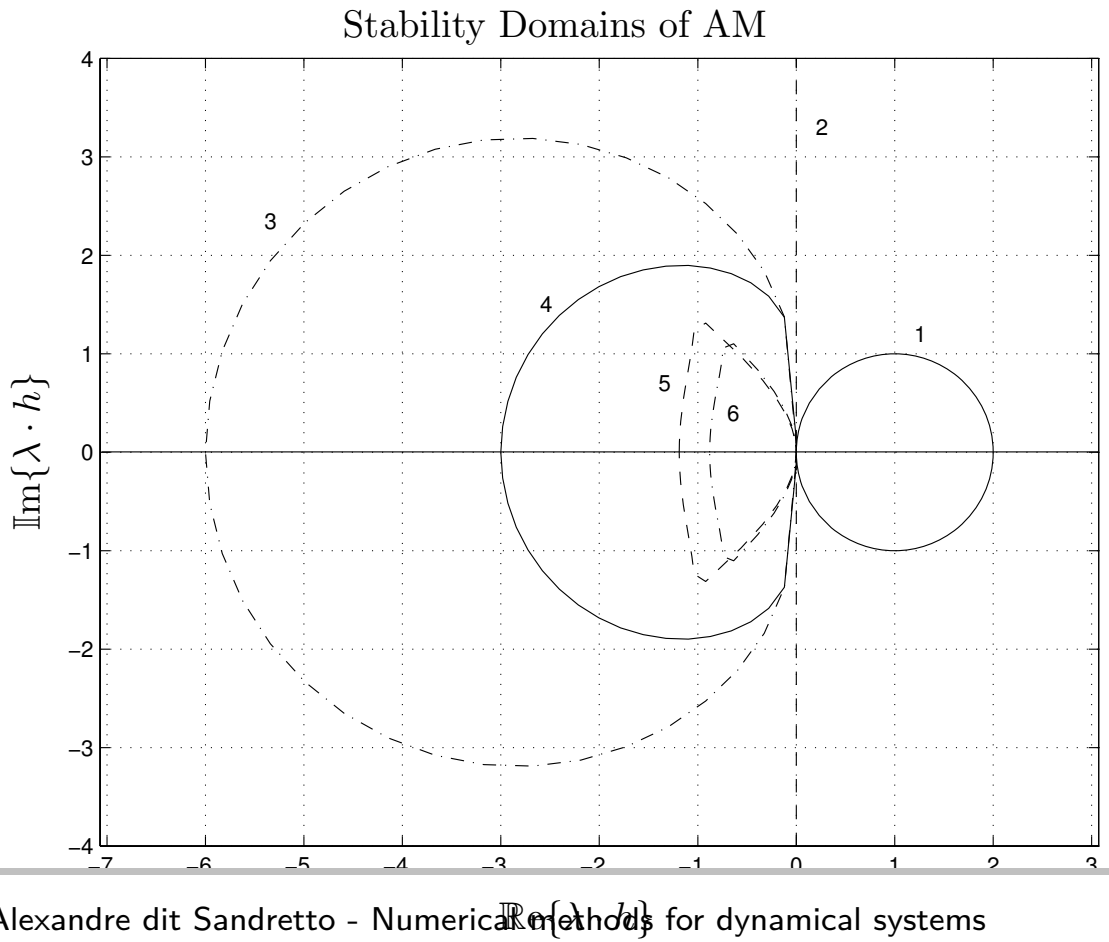
Linear stability of Adams-Moulton methods

We consider the scalar linear IVP

$$\dot{y} = \lambda y \quad \text{with} \quad \lambda \in \mathbb{C}, \Re(\lambda) < 0$$

For linear problem, the **stability polynomial** of a multi-step method is

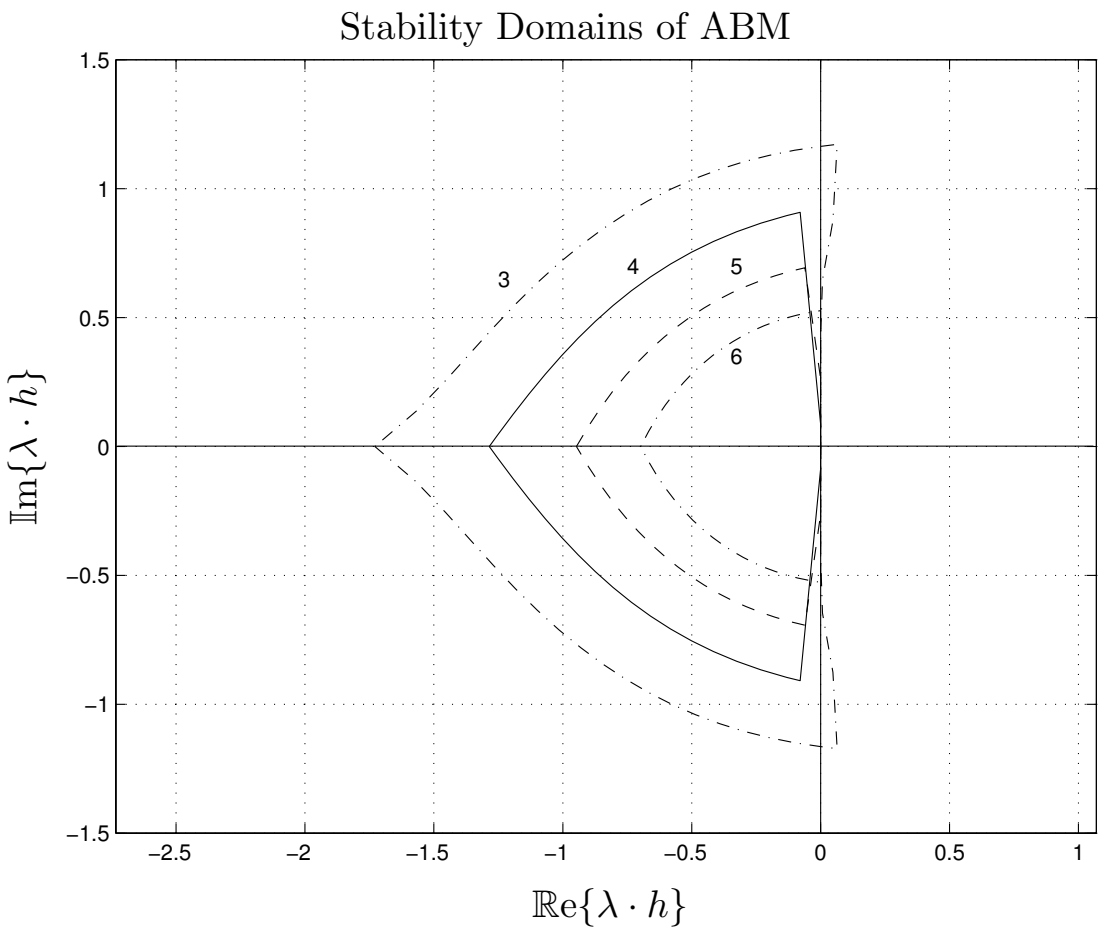
$$\pi(r, \hat{h}) = \rho(r) - \hat{h}\sigma(r) \quad \text{with} \quad \hat{h} = \lambda h$$



Linear stability of Adams-Bashworth-Moulton methods

We consider the IVP:

$$\dot{x} = \lambda x \quad \text{with} \quad \lambda \in \mathbb{C}, \Re(\lambda) < 0$$



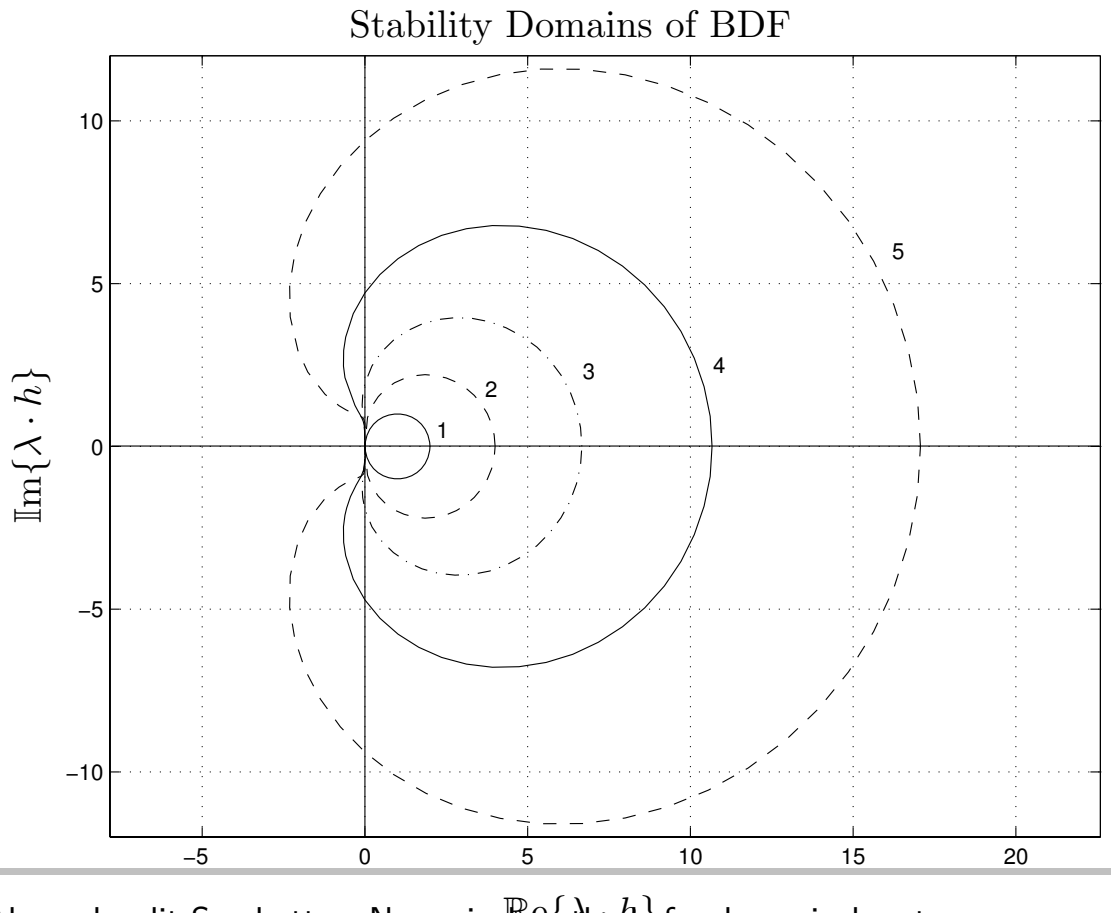
Linear stability of BDF

We consider the scalar linear IVP

$$\dot{y} = \lambda y \quad \text{with} \quad \lambda \in \mathbb{C}, \Re(\lambda) < 0$$

For linear problem, the **stability polynomial** of a multi-step method is

$$\pi(r, \hat{h}) = \rho(r) - \hat{h}\sigma(r) \quad \text{with} \quad \hat{h} = \lambda h$$



Stiff versus non-stiff problems

Problem 1

$$\begin{pmatrix}\dot{y}_1\\ \dot{y}_2\end{pmatrix} = \begin{pmatrix}-2 & 1\\ 1 & -2\end{pmatrix}\begin{pmatrix}y_1\\ y_2\end{pmatrix} + \begin{pmatrix}2\sin(t)\\ 2(\cos(t) - sin(t))\end{pmatrix}$$

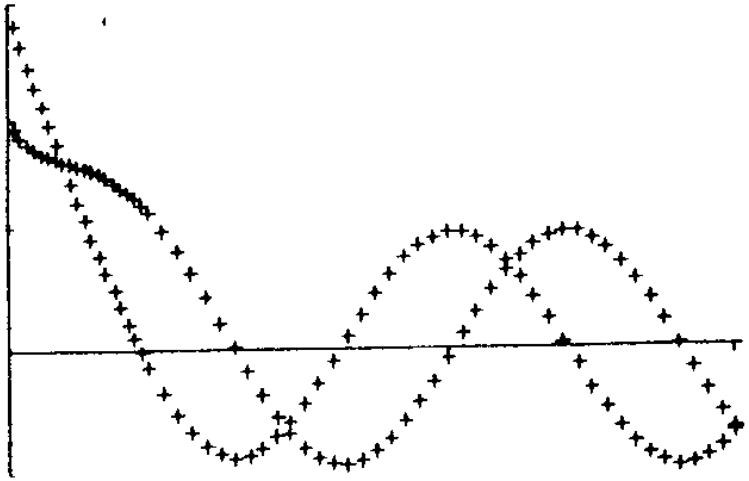
Problem 2

$$\begin{pmatrix}\dot{y}_1\\ \dot{y}_2\end{pmatrix} = \begin{pmatrix}-2 & 1\\ 998 & -999\end{pmatrix}\begin{pmatrix}y_1\\ y_2\end{pmatrix} + \begin{pmatrix}2\sin(t)\\ 999(\cos(t) - sin(t))\end{pmatrix}$$

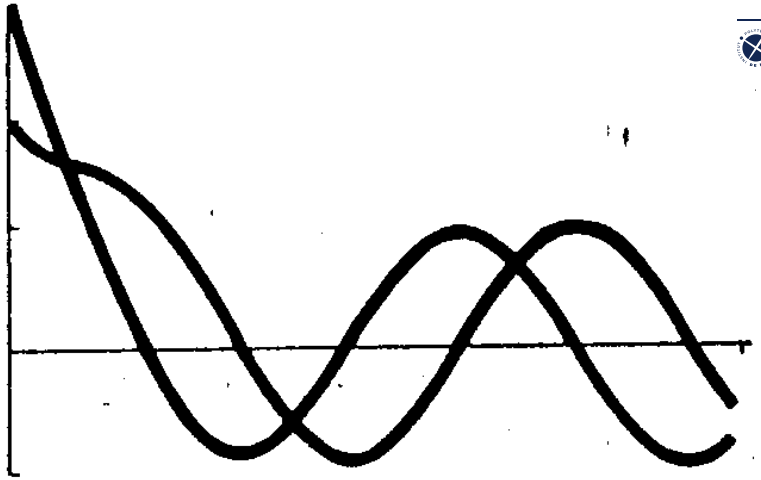
Both have the same exact solution:

$$\begin{pmatrix}y_1(t)\\ y_2(t)\end{pmatrix} = 2\exp(-t)\begin{pmatrix}1\\ 1\end{pmatrix} + \begin{pmatrix}\sin(t)\\ \cos(t)\end{pmatrix} \quad \text{with initial values} \begin{pmatrix}y_1(0)\\ y_2(0)\end{pmatrix} = \begin{pmatrix}2\\ 3\end{pmatrix}$$

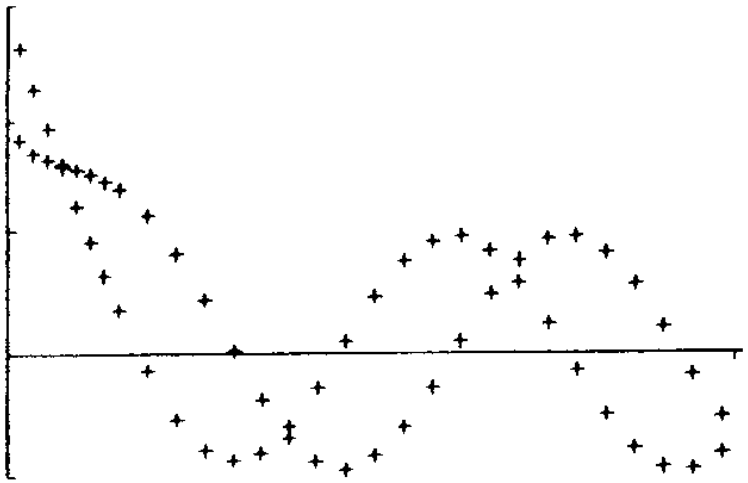
Simulation results



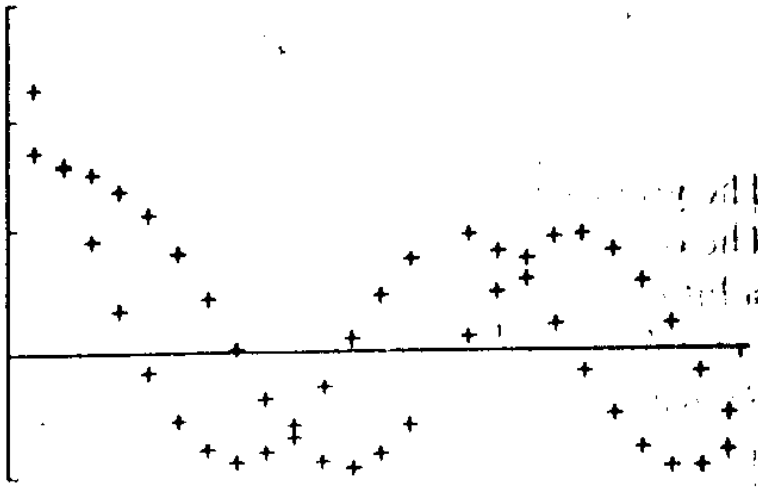
(b) Problem 1, RKF45; $N = 60$.



(c) Problem 2, RKF45; $N = 3373$.



(d) Problem 1, 2-stage Gauss; $N = 29$.



(e) Problem 2, 2-stage Gauss; $N = 24$.

Stiff linear ODE: a definition

We consider linear constant coefficients IVP of the form:

$$\dot{\mathbf{y}} = A\mathbf{y} + \phi(t)$$

assuming that all eigenvalues λ are such that $\Re(\lambda) < 0$
We denote by

- $| \Re(\overline{\lambda}) | = \max_{1 \leq i \leq n} | \Re(\lambda_i) |$
- $| \Re(\underline{\lambda}) | = \min_{1 \leq i \leq n} | \Re(\lambda_i) |$
- the **stiffness ratio** is defined by $| \Re(\overline{\lambda}) | / | \Re(\underline{\lambda}) |$

Stiffness definition - 1 (Lambert)

A linear constant coefficients system is stiff iff all eigenvalues are such that $\Re(\lambda) < 0$ and the stiffness ratio is large.

Others stiffness definitions

Definition 2 (Lambert)

Stiffness occurs when stability requirements, rather than those of accuracy, constrain the step size.

Definition 3 (Lambert)

Stiffness occurs when some components of the solution decay much more quickly than others.

Global definition (Lambert)

If a numerical method with a finite region of absolute stability, applied to a system with any initial values, if forced to use in a certain interval of integration a step size which is excessively small in relation to the smoothness of the exact solution in that interval, then the system is said to be **stiff** in that interval.

Linear stability definition for stiff systems - 1

A-stability

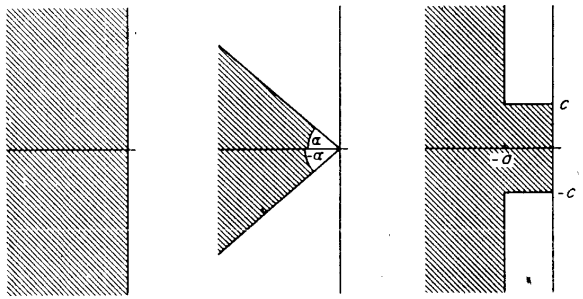
A method is **A-stable** if $\mathcal{R}_s \supseteq \{\hat{h} : \Re(\hat{h}) < 0\}$

$A(\alpha)$-stability

A method is $A(\alpha)$-**stable**, $\alpha \in]0, \pi/2[$, if $\mathcal{R}_s \supseteq \{\hat{h} : -\alpha < \pi - \arg(\hat{h}) < \alpha\}$

Stiffly stability

A method is **stiffly stable** if $\mathcal{R}_S \supseteq \mathcal{R}_1 \cup \mathcal{R}_2$ such that $\mathcal{R}_1 = \{\hat{h} : \Re(\hat{h}) < -a\}$ and $\mathcal{R}_2 = \{\hat{h} : -a \leq \Re(\hat{h}) \leq 0, -c \leq \Im(\hat{h}) \leq c\}$ with a and c two positive real numbers.



Linear stability definition for stiff systems - 2

L-stability

A one step method is *L*-**stable** if

- it is *A*-stable
- and when applied to stable scalar test equations $\dot{y} = \lambda y$ it yields

$$y_{n+1} = \Re(h\lambda)x_n \quad \text{where} \quad |\Re(h\lambda)| \to 0 \text{ as } \Re(h\lambda) \to -\infty$$

Relation between the stability definitions

$$L\text{-stability} \Rightarrow A\text{-stability} \Rightarrow \text{stiffly stability} \Rightarrow A(\alpha)\text{-stability}$$

Numerical methods for linear stiff problems

Runge-Kutta methods

Method	Order	Linear stability prop.
Gauss	$2s$	A-stability
Radau IA, IIA	$2s-1$	L-stability
Lobatto IIIA, IIIB	$2s-2$	A-stability
Lobatto IIIC	$2s-2$	L-stability

Theorems (Dahlquist barrier)

- Explicit RK cannot be A-stability or stiffly stability or $A(\alpha)$-stability!
- Explicit linear multi-step method cannot be A-stable
- The order of an A-stable linear multi-step method cannot exceed 2
- The second order A stable multi-step method with the smallest error constant (C_3) is the Trapezoidal rule.

For the particular case of BDF

- BF1 and BDF2 are L-stable
- other BDF(3-4-5-6) are $A(\alpha)$-stable
- BF6 has a very narrow stability area, it is not used in practice

Definition of Differential Algebraic Equations (DAE)

We consider a differential system of equation

$$F(\dot{x}, x, t) = \begin{pmatrix} F_1(\dot{x}(t), x(t), t) \\ F_2(\dot{x}(t), x(t), t) \\ \vdots \\ F_n(\dot{x}(t), x(t), t) \end{pmatrix} = 0$$

with $\dot{x}(t), x(t) \in \mathbb{R}^n$.
This system is a **DAE** if the Jacobian matrix

$$\frac{\partial F}{\partial \dot{x}} \quad \text{is } \textbf{singular}$$

Example of DAE

The following system is a DAE

$$\begin{aligned} x_1 - \dot{x}_1 + 1 &= 0 \\ \dot{x}_1 x_2 + 2 &= 0 \end{aligned} \quad \Rightarrow \quad F(\dot{x}, x, t) = \begin{pmatrix} x_1 - \dot{x}_1 + 1 \\ \dot{x}_1 x_2 + 2 \end{pmatrix} \quad \text{with} \quad x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

The Jacobian of F w.r.t. $\dot{x}$ is

$$\frac{\partial F}{\partial \dot{x}} = \begin{pmatrix} \frac{\partial F_1}{\partial \dot{x}_1} & \frac{\partial F_1}{\partial \dot{x}_2} \\ \frac{\partial F_2}{\partial \dot{x}_1} & \frac{\partial F_2}{\partial \dot{x}_2} \end{pmatrix} = \begin{pmatrix} -1 & 0 \\ x_2 & 0 \end{pmatrix} \quad \Rightarrow \quad \det\left(\frac{\partial F}{\partial \dot{x}}\right) = 0$$

Note in this example $\dot{x}_2$ is not explicitly defined.

Example of DAE continued

Solving DAE is a hard challenge either symbolically or numerically.
Special DAE forms are usually considered: linear, Hessenberg form, etc.

Example, we rewrite the previous system

- solving for $\dot{x}_1$ the equation $x_1 - \dot{x}_1 + 1 = 0 \Rightarrow \dot{x}_1 = x_1 + 1$
- Substitute $\dot{x}_1$ in $\dot{x}_1 x_2 + 2 = 0$ we get

$$\dot{x}_1 = x_1 + 1 \quad \texttt{Ordinary differential equation}$$
$$(x_1 + 1)\, x_2 + 2 = 0 \quad \texttt{Algebraic equation}$$

Note: this form of DAE is used in many engineering applications.

- mechanical engineering, process engineering, electrical engineering, etc.
- **Usually:** dynamics of the process + laws of conservation

Engineering examples of DAE - Chemical reaction

An isothermal continuous flow stirred-tank reactor[1] (CSTR) with elementary reaction

$$A \rightleftharpoons B \rightarrow C$$

assuming

- reactant A with a in-flow rate F_a and concentration C_{A_0}
- Reversible reaction $A \rightleftharpoons B$ is much faster that $B \rightarrow C$, i.e., $k_1 \gg k_2$

$$\dot{V} = F_a - F$$

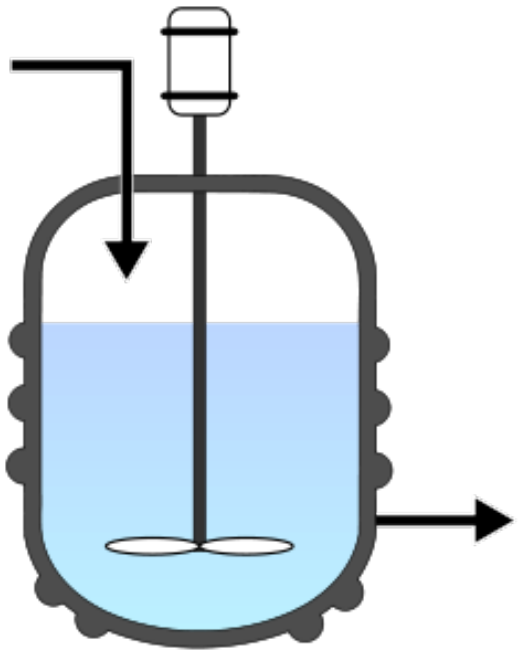
$$\dot{C}_A = \frac{F_a}{V}\left(C_{A_0} - C_A\right) - R_1$$

$$\dot{C}_B = -\frac{F_a}{V} C_B + R_1 - R_2$$

$$\dot{C}_C = -\frac{F_a}{C_C} + R_2$$

$$0 = C_A - \frac{C_B}{K_{eq}}$$

$$0 = R_2 - k_2 C_B$$



[1] Control of Nonlinear DAE Systems with Applications to Chemical Processes

Engineering examples of DAE - Chemical reaction

- R_1 and R_2 rates of reactions
- F output flow
- C_A, C_B and C_C are concentrations of A, B and C.

Let

$$x = (V, C_A, C_B, C_C)$$
$$z = (R_1, R_2)$$

we get

$$\dot{x} = f(x, z)$$
$$0 = g(x, z)$$

Engineering examples of DAE - Mechanical system

Pendulum

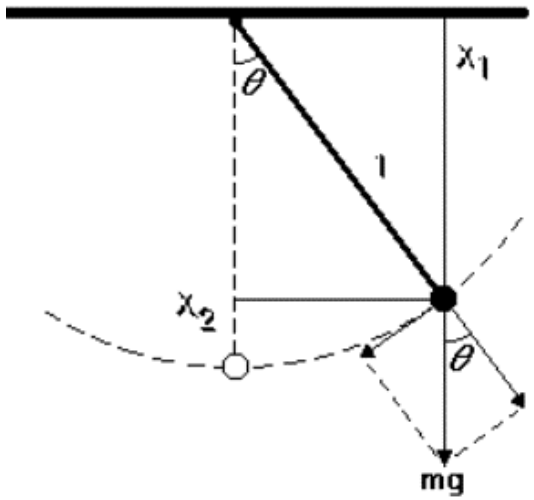
- second Newton's law

$$m\ddot{x} = -\frac{F}{\ell}x$$
$$m\ddot{y} = mg\frac{F}{\ell}y$$

- Mechanical energy conservation

$$x^2 + y^2 = \ell^2$$

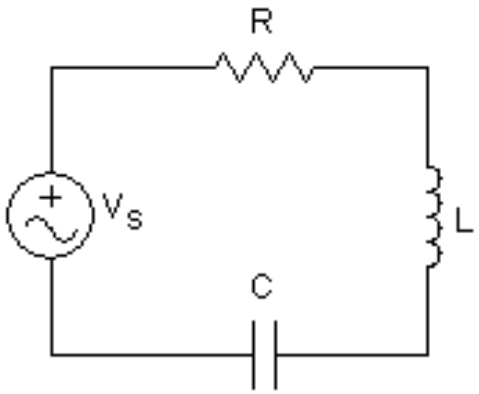
$$\dot{x}_1 = x_3$$
$$\dot{x}_2 = x_4$$
$$\dot{x}_3 = -\frac{F}{\ell}x_1$$
$$\dot{x}_4 = g\frac{F}{\ell}x_2$$
$$0 = x_1^2 + x_2^2 - \ell^2$$



Engineering examples of DAE - Electrical system

Ohm's law

$$C\dot{V}_C = i_C, \quad L\dot{V} = i_L, V_R = Ri_R$$



Kirchoff's voltage and current laws

- Conservation of current

$$i_E = i_R, \quad i_R = i_C, \quad i_C = i_L$$

- Conservation of energy

$$V_R + V_L + V_C + V_E = 0$$

And we get

$$\dot{V}_C = \frac{1}{C} i_L$$
$$\dot{V}_L = \frac{1}{L} i_L$$
$$0 = V_R + Ri_E$$
$$0 = V_E + V_R + V_C + V_L$$
$$0 = i_L - i_E$$

Engineering examples of DAE - Electrical system

Let

$$x = (V_C, V_L, V_R, i_L, i_E)$$

we have

$$\dot{x} = \begin{pmatrix} \frac{1}{C} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{L} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} x$$

$$0 = \begin{pmatrix} 0 & 0 & 1 & 0 & R \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} V_E$$

which is of the form

$$\dot{x} = Ax$$
$$0 = Bx + Dz$$

Method of Lines for PDE

Consider the linear PDE (diffusion equations)

$$\frac{\partial u}{\partial t}(x,t) = D\frac{\partial^2 u}{\partial x^2}(x,t) \quad \text{with} \quad \begin{cases} u(x = x_0, t) = u_b \\ \dfrac{\partial u}{\partial x}(x = x_f, t) = 0 \end{cases}$$

and D a constant.
Using method of lines, we have with an equally spaced grid for x (finite difference)

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} + \mathcal{O}\left(\Delta x^2\right)$$

Hence, we get

$$\frac{du_i}{dt} = D\frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} \quad \text{for} \quad i = 1, 2, \ldots, M$$

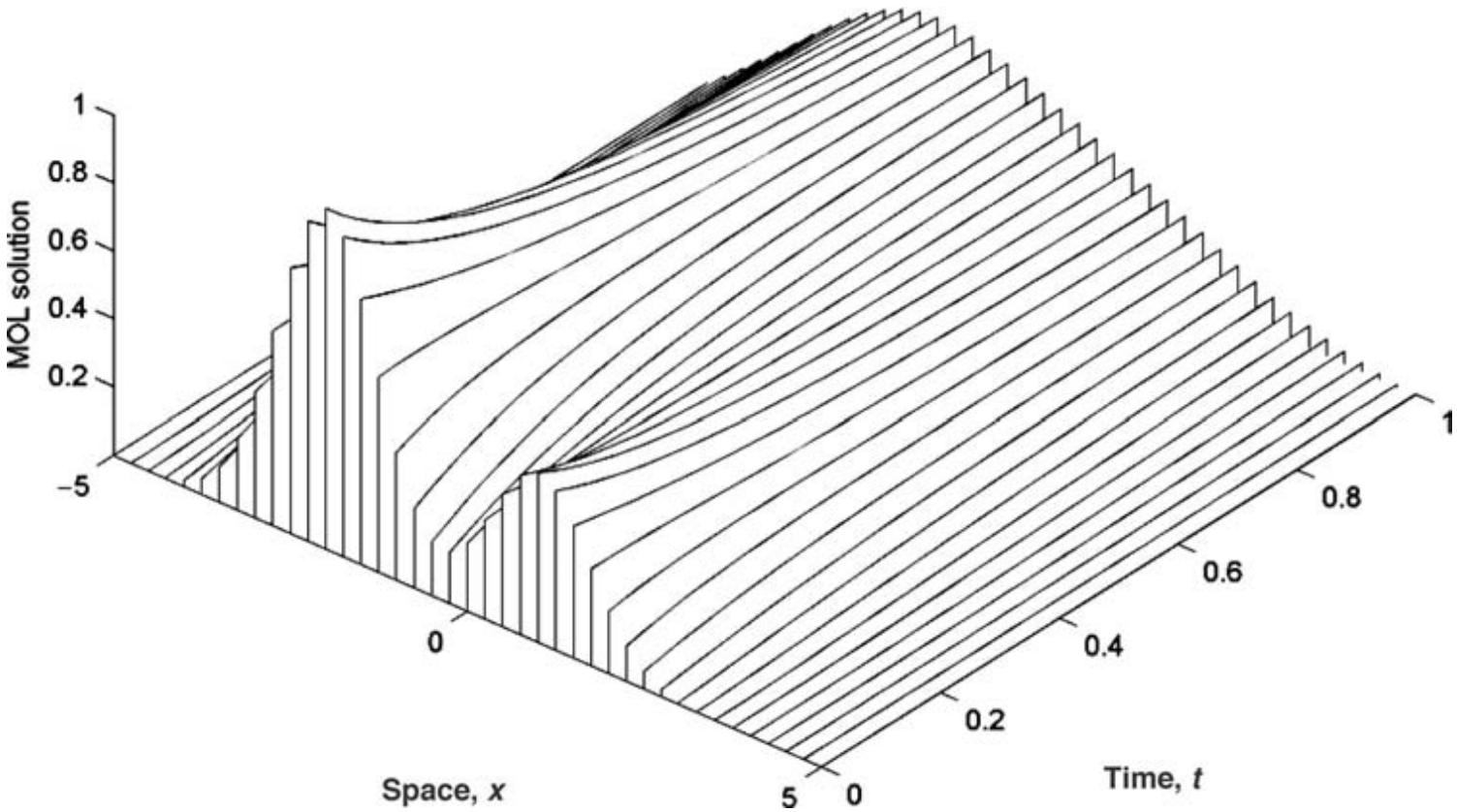
Method of Lines for PDE

In other words, we get the system

$$
\begin{aligned}
u_1 &= u_b \\
\frac{du_2}{dt} &= D\frac{u_3 - 2u_2 + u_b}{\Delta x^2} \\
\frac{du_3}{dt} &= D\frac{u_4 - 2u_3 + u_2}{\Delta x^2} \\
&\vdots \\
\frac{du_M}{dt} &= D\frac{u_{M+1} - 2u_M + u_{M-1}}{\Delta x^2} \\
u_{M+1} &= u_M
\end{aligned}
$$

Note u_{M+1} is outside of the grid so we add an extra constraints.
Hence we get a **DAE**

Method of Lines for PDE



Classification of DAE

- **Nonlinear DAE** if it is of the form

$$F(\dot{x}, x, t) = 0$$

  and it is nonlinear w.r.t. any one of $\dot{x}$, x, or t
- **Linear DAE** if it is of the form

$$A(t)\dot{x} + B(t)x = c(t)$$

  If $A(t) \equiv A$ and $B(t) \equiv B$ then the DAE is **time-invariant**
- **Semi-explicit DAE** it is of the form

$$\dot{x} = f(t, x, z)$$
$$0 = g(t, x, z)$$

  z is the **algebraic variable** and x is a **differential/state variable**
- **Fully implicit DAE** it is of the form

$$F(\dot{x}, x, t) = 0$$

Classification of DAE - cont

Note any DAE can be written in a semi-explicit form.

Conversion of fully implicit form

$$F(\dot{x}, x, t) = 0 \quad \overset{\dot{x}=z}{\Leftrightarrow} \begin{cases} \dot{x} = z \\ 0 = F(z, x, t) \end{cases}$$

Remark this transformation does not make the solution more easier to get
But useful in case of linear DAE, see next.

Classification of DAE - cont

Consider a linear time-invariant DAE

$$A\dot{x} + Bx + b(t) = 0$$

assuming that $\lambda A + B$ (**matrix pencil**) is not singular for some scalar λ.
Then it exists non-singular matrices G and H of size $n \times n$ such that:

$$GAH = \begin{pmatrix} I_m & 0 \\ 0 & N \end{pmatrix} \quad \text{and} \quad GBH = \begin{pmatrix} J & 0 \\ 0 & I_{n-m} \end{pmatrix}$$

- I_m is the identity matrix of size $m \times m$ ($m \leq n$)
- I_{n-m} is the identity matrix of size $(n-m) \times (n-m)$
- N is a **nilpotent matrix**, i.e., $\exists p \in \mathbb{N}^+, N^p = 0$
- $J \in \mathbb{R}^{m \times m}$

Classification of DAE - cont

Hence

$$
\begin{aligned}
A\dot{x} + Bx + b(t) = 0 &\Leftrightarrow (GAH)(H^{-1})\dot{x} + (GBH)(H^{-1})x + Gb(t) = 0 \\
&\Leftrightarrow \begin{pmatrix} I_m & 0 \\ 0 & N \end{pmatrix} H^{-1}\dot{x} + \begin{pmatrix} J & 0 \\ 0 & I_{n-m} \end{pmatrix} H^{-1}x + Gb(t) = 0 \\
&\Leftrightarrow \quad \text{with} \quad w(t) = H^{-1}x \\
&\begin{pmatrix} I_m & 0 \\ 0 & N \end{pmatrix} \dot{w} + \begin{pmatrix} J & 0 \\ 0 & I_{n-m} \end{pmatrix} w + Gb(t) = 0
\end{aligned}
$$

Let $w = (w_1, w_2)^T$ with $w_1 \in \mathbb{R}^m$ and $w_2 \in \mathbb{R}^{n-m}$, $b = (b_1, b_2)^T$ we get

$$
\begin{aligned}
\dot{w}_1 + Jw_1 + b_1(t) = 0 \\
Nw_1 + w_2 + b_2(t) = 0
\end{aligned}
$$

From Nilpotency property, we get

$$
\begin{aligned}
\dot{w}_1 &= -Jw_1 - b_1(t) \\
0 &= -(N^p)^{-1}w_2 - (N^p)^{-1}b_2(t)
\end{aligned}
$$

Index of DAE

Remark

There are several definitions of an index.
Each measure a different aspect of the DAE.

- **Differential index** (δ) measure the degree of singularity.
- **Perturbation index** (π) measure the influence of numerical approximation.
- etc.

Definition of differential index

The index of a DAE system $F(\dot{x}, x, t) = 0$ is the minimum number of times certain equations in the DAE must be differentiated w.r.t. t, in order to transform the problem into an ODE.

Remark: (differential) index can be seen as a measure of the distance between the DAE and the corresponding ODE.
Remark: mathematical properties are lost with differentiation!

DAE and index

Definition of index

The differential index k of a sufficiently smooth DAE is the smallest k such that:

$$
\begin{aligned}
F(\dot{x}, x, t) &= 0 \\
\frac{\partial F}{\partial t}(\dot{x}, x, t) &= 0 \\
&\vdots \\
\frac{\partial^k F}{\partial t^k}(\dot{x}, x, t) &= 0
\end{aligned}
$$

uniquely determines $\dot{\mathbf{x}}$ as a continuous function of $(\mathbf{x}, t)$.

Differential index and DAE – example

Let

$$\dot{x}_1 = x_1 + 1$$
$$(x_1 + 1)\, x_2 + 2 = 0$$

with x_2 the algebraic variable.
Differentiation of g w.r.t. t,

$$\frac{d}{dt} g(x_1, x_2) = 0 \quad \Rightarrow \quad \dot{x}_1 x_2 + (x_1 + 1)\dot{x}_2 = 0 \quad \Rightarrow \quad \dot{x}_2 = -\frac{\dot{x}_1 x_2}{x1_1} = -x_2$$

Only one differentiation is needed to define $\dot{x}_2$, this DAE is **index** 1
Other examples,

- CSTR is index 2
- Pendulum is index 3

There are **higher index** DAEs (index > 1)
Index reduction is used to go from higher index to lower index DAE (cf Khalil Ghorbal's lecture)

DAE family and differential index

Index 0

ODE system $\dot{x} = f\left(t, x(t)\right)$

Index 1

Algebraic equation $y = q(t)$

Index 1

DAE in Hessenberg form of index 1

$$\dot{x} = f\left(t, x, y\right)$$

$$0 = g\left(x, y\right) \quad \text{with} \quad \frac{\partial g}{\partial y} \quad \text{is non-singular}$$

Examples of differential index - cont.

Index 2

DAE in Hessenberg form of index 2

$$\dot{x} = f(t, x, y)$$
$$0 = g(t, x) \quad \text{with} \quad \frac{\partial g}{\partial x}\frac{\partial f}{\partial y} \quad \text{is non-singular}$$

Index 3

DAE in Hessenberg form of index 3

$$\dot{x} = f(t, x, y, z)$$
$$\dot{y} = g(t, x, y)$$
$$0 = h(t, y) \quad \text{with} \quad \frac{\partial h}{\partial y}\frac{\partial g}{\partial x}\frac{\partial f}{\partial z} \quad \text{is non-singular}$$

e.g., mechanical systems

Perturbation index

The DAE has the perturbation index k along a solution x if k is the smallest integer such that,
for all functions $x(t)$ having the defect

$$f(\dot{x}_\delta, x_\delta, t) = \delta(t)$$

there exists an estimate

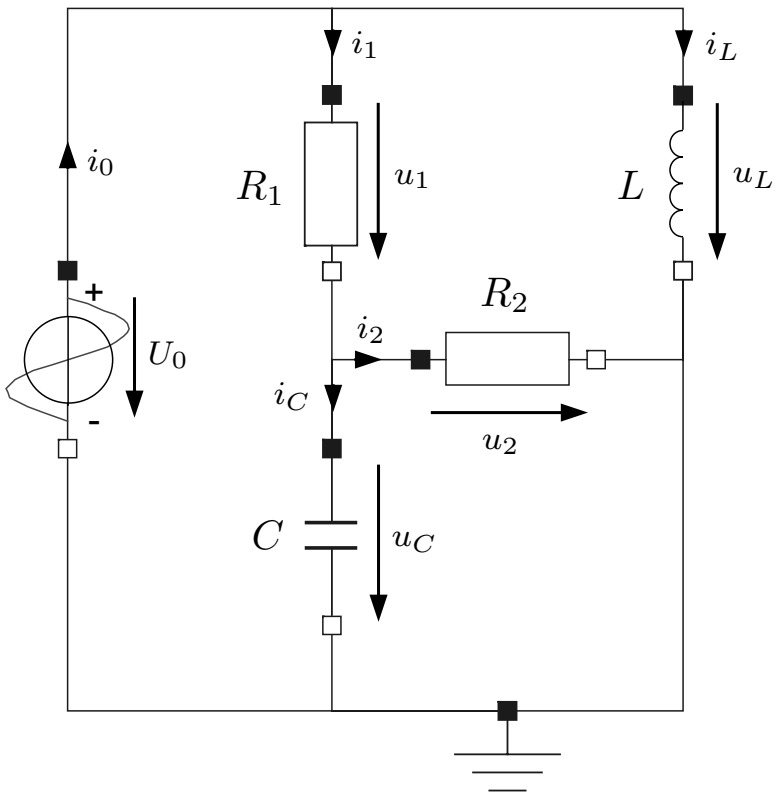
$$\| x(t) - x_\delta(t) \| \leq C \left(\| x(t_0) - x_\delta(t_0) \| + \max_t \| \delta(t) \| + max_t \| \delta'(t) \| + \cdots + \max_t \| \delta^{(k-1)}(t) \| \right)$$

for a constant $C > 0$, if δ is small enough.

Property:

$$\delta \leq \pi \leq \delta + 1$$

RLC circuit



$$u_0 = f(t) \quad (3)$$
$$u_1 = R_1 i_1 \quad (4)$$
$$u_2 = R_2 i_2 \quad (5)$$
$$u_L = L\frac{di_L}{dt} \quad (6)$$
$$i_C = C\frac{du_C}{dt} \quad (7)$$
$$u_0 = u_1 + u_C \quad (8)$$
$$u_L = u_1 + u_2 \quad (9)$$
$$u_C = u_2 \quad (10)$$
$$i_0 = i_1 + i_L \quad (11)$$
$$i_1 = i_2 + i_C \quad (12)$$

We want to compute a state-space form of this RLC circuit.

Structure incidence matrix

	u_0	i_0	u_1	i_1	u_2	i_2	u_L	$\frac{di_L}{dt}$	$\frac{du_C}{dt}$	i_C
Eq. (3)	1	0	0	0	0	0	0	0	0	0
Eq. (4)	0	0	1	1	0	0	0	0	0	0
Eq. (5)	0	0	0	0	1	1	0	0	0	0
Eq. (6)	0	0	0	0	0	0	1	1	0	0
Eq. (7)	0	0	0	0	0	0	0	0	1	1
Eq. (8)	1	0	1	0	0	0	0	0	0	0
Eq. (9)	0	0	1	0	1	0	1	0	0	0
Eq. (10)	0	0	0	0	1	0	0	0	0	0
Eq. (11)	0	1	0	1	0	0	0	0	0	0
Eq. (12)	0	0	0	1	0	1	0	0	0	1

Structure incidence matrix

Relation between equations (rows) and unknowns (columns)

- if the i-th equation contains the j-th variable then the matrix coefficient (i, j) contains 1 and 0 otherwise.

Structure incidence matrix - cont.

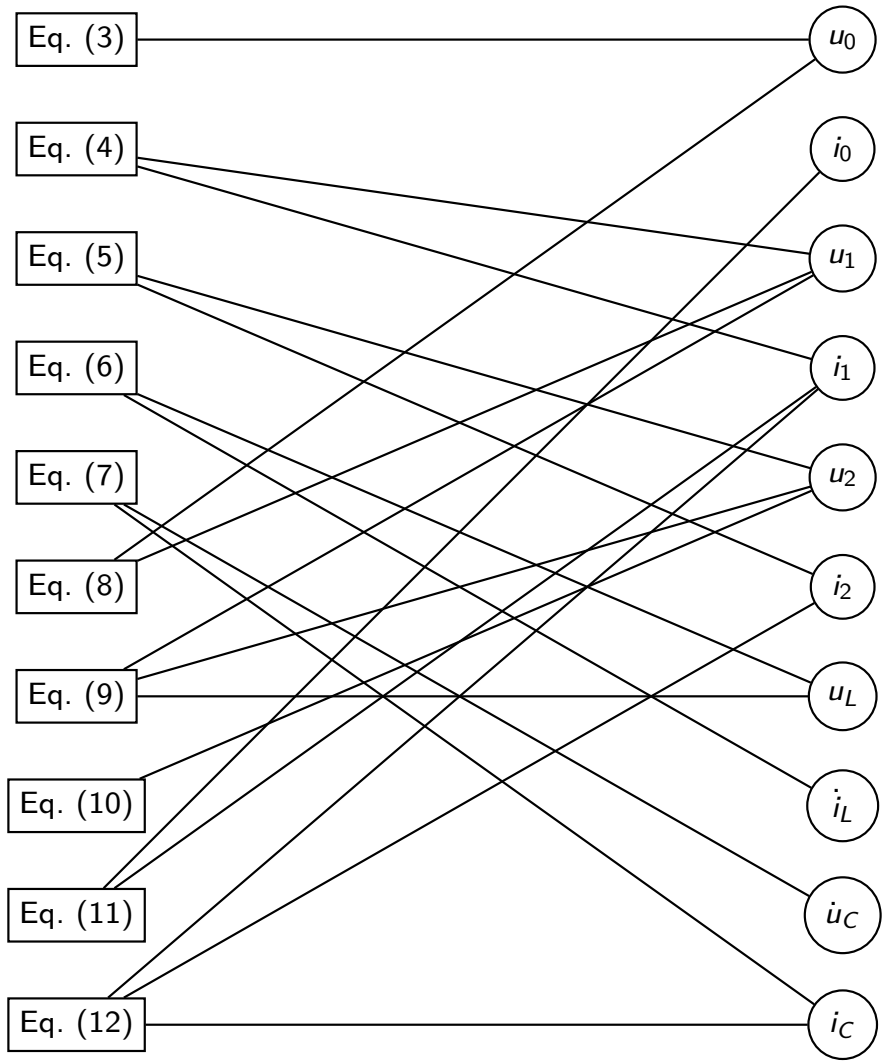
By default all equations are implicit (or **acausal**)
Two rules to choose the set of variables to solve

- if an equations contains only a single unknown then we need that variable to solve it (i.e., this equation is **causal**, e.g., Eq. (3))
- If an unknown only appears in one equation, that equation must use to solve it. E.g., Eq. (11) i_0 only appears in that equation.

Apply iteratively these rules:

- if a row only contains one 1, that equation needs to be solved for that variable so eliminate both row and column
- if a column only contains one 1, that variable needs to be solved for that equation so eliminate both row and column

Structure digraph



Remark the number of equations must always equal to the number of variables

Structure digraph - cont.

Building: There is a link between a node of equations and a node of variable is this variable appears in that equation.
Finding which variable needs to be solved from which equations, is based on a graph coloring algorithm (Tarjan)

- When a variable is selected to be solved from an equation the link between them is colored in red.
- When a variable is known or when the equation in which it occurs is being used to solve an other variable, the link is colored in blue
- A causal equation has exactly one red link connected to it
- An acausal equation has block or blue connected edges
- A known variable has exactly one red input edge
- An unknown variable has only black or blue input edges
- No equation or variable has more than one red edges

Structure digraph - cont.

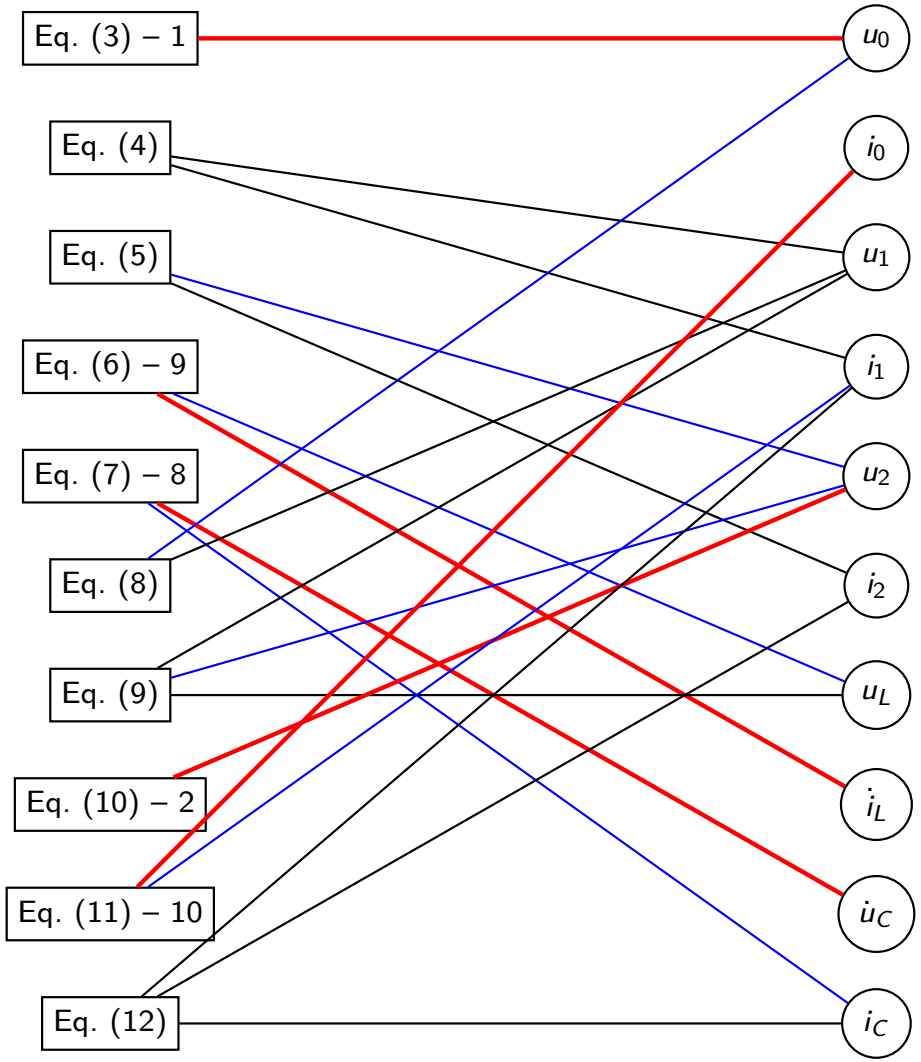
Rules to find variables and equations

- For all acausal equations, if an equation has only one black line attached to it, color that line red, follow it to the variable it points at, and color all other connections ending in that variable in blue. Renumber the equation using the lowest free number starting from 1.
- For all unknown variables, if a variable has only one black line attached to it, color that line red, follow it back to the equation it points at, and color all other connections emanating from that equation in blue. Renumber the equation using the highest free number starting from n, where n is the number of equations.

These rules are applied recursively.

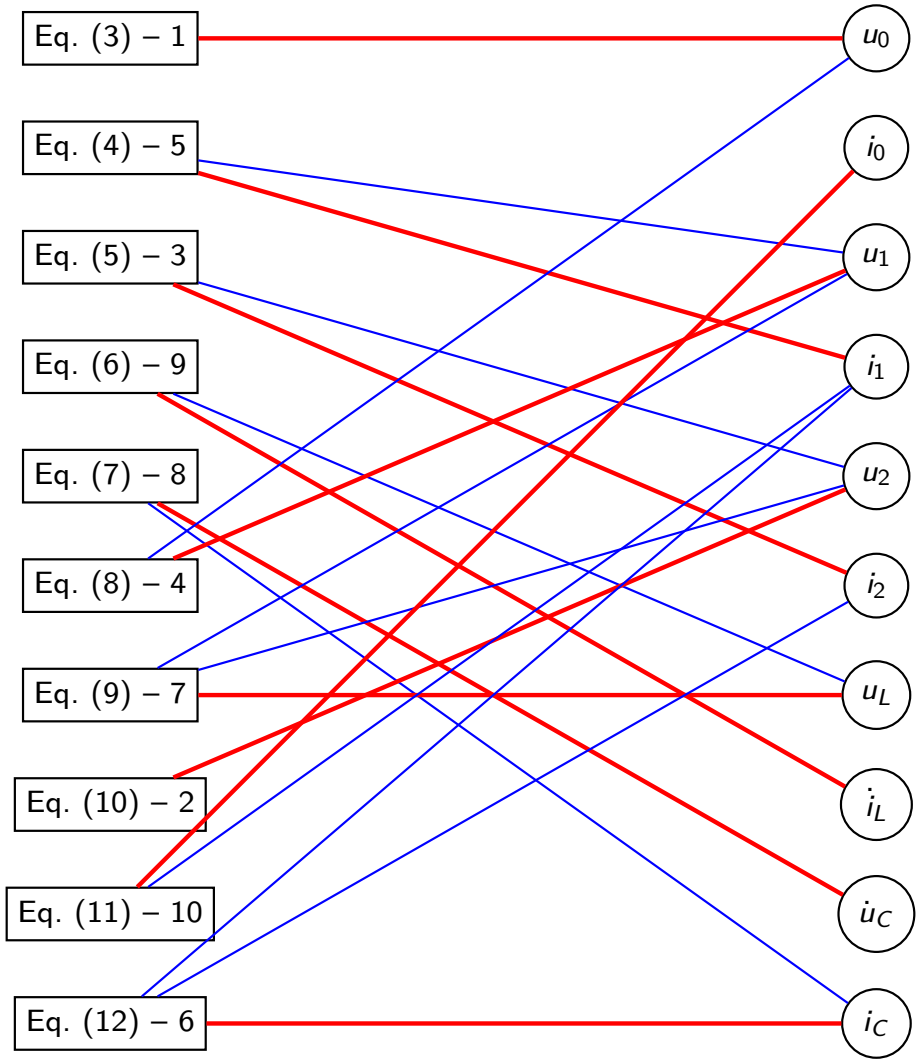
Structure digraph

After one iteration of the algorithm.



Structure digraph

At the end of the algorithm



Structure digraph

At the end of the algorithm and the system of equations is written as

$$u_0 = f(t) \tag{13}$$
$$u_2 = u_C \tag{14}$$
$$i_2 = u_2/R_2 \tag{15}$$
$$u_1 = u_0 - u_C \tag{16}$$
$$i_1 = u_1\ R1 \tag{17}$$
$$i_C = i_1 - i_2 \tag{18}$$
$$u_L = u_1 + u_2 \tag{19}$$
$$\frac{du_C}{dt} = i_C/C \tag{20}$$
$$\frac{di_L}{dt} = u_L/L \tag{21}$$
$$i_0 = i_1 + i_L \tag{22}$$

Note these equations are causal and in order to be evaluated.

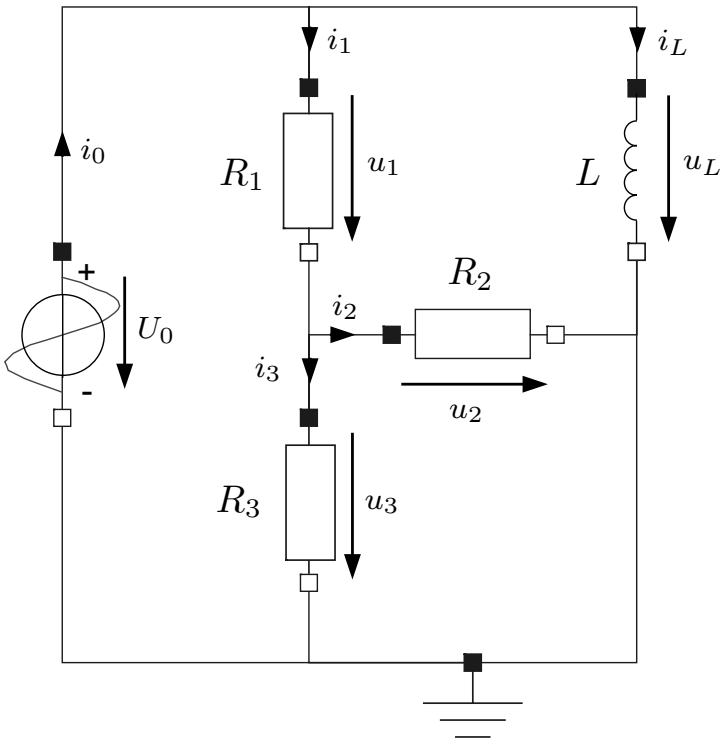
Structure incidence matrix and Tarjan algorithm

	u_0	u_2	i_2	u_1	i_1	i_C	u_L	$\frac{di_L}{dt}$	$\frac{du_C}{dt}$	i_0
Eq. (13)	1	0	0	0	0	0	0	0	0	0
Eq. (14)	0	1	0	0	0	0	0	0	0	0
Eq. (15)	0	1	1	0	0	0	0	0	0	0
Eq. (16)	1	0	0	1	0	0	0	0	0	0
Eq. (17)	0	0	0	1	1	0	0	0	0	0
Eq. (18)	0	0	1	0	1	1	0	0	0	0
Eq. (19)	0	1	0	1	0	0	1	0	0	0
Eq. (20)	0	0	0	0	0	1	0	1	0	0
Eq. (21)	0	0	0	0	0	0	1	0	1	0
Eq. (22)	0	0	0	0	1	0	0	0	0	1

Note 1 the matrix is lower triangular (Tarjan $\Leftrightarrow$ matrix permutation)
Note 2 Tarjan algorithm has a linear complexity in the number of equations. Also used in **Pantelides algorithm**

Algebraic loops

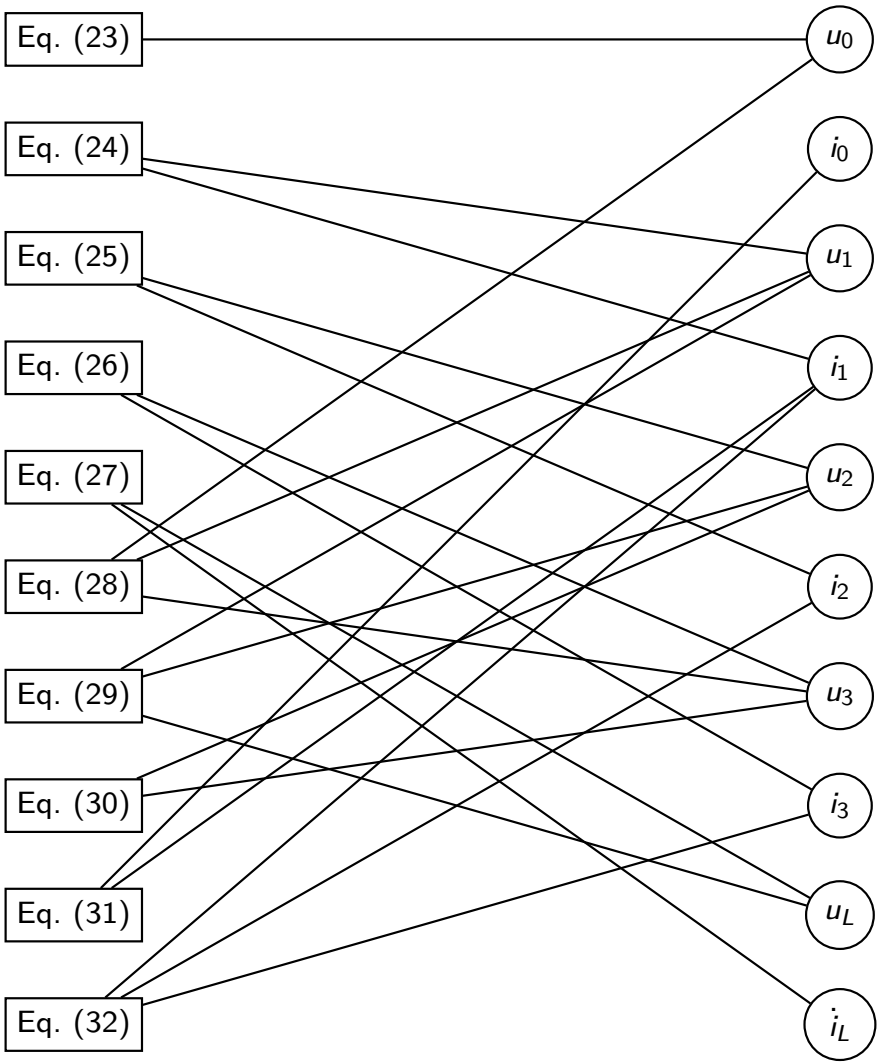
A tiny modification of the RLC circuit



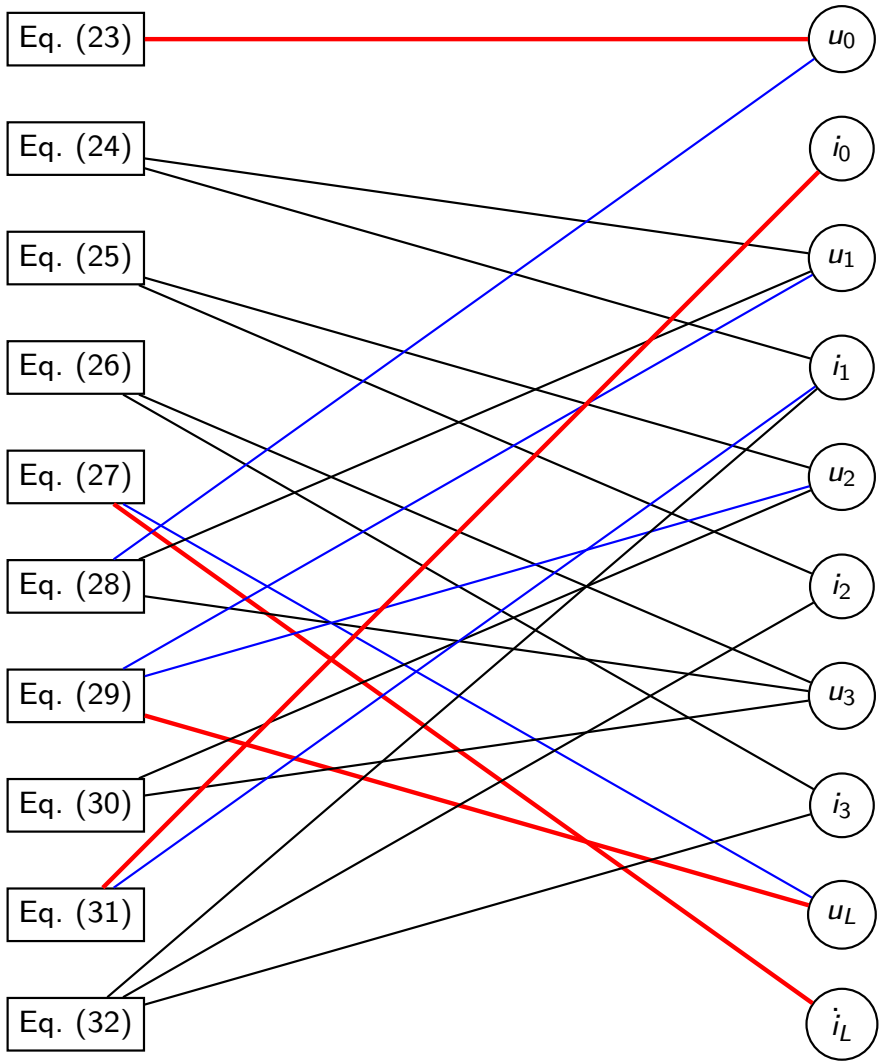
$$u_0 = f(t) \quad (23)$$
$$u_1 = R_1 i_1 \quad (24)$$
$$u_2 = R_2 i_2 \quad (25)$$
$$u_3 = R_3 i_3 \quad (26)$$
$$u_L = L\frac{di_L}{dt} \quad (27)$$
$$u_0 = u_1 + u_3 \quad (28)$$
$$u_L = u_1 + u_2 \quad (29)$$
$$u_3 = u_2 \quad (30)$$
$$i_0 = i_1 + i_L \quad (31)$$
$$i_1 = i_2 + i_3 \quad (32)$$

Note the capacitor is replaced by a resistor.

Algebraic loop - structure digraph



Algebraic loop - structure digraph - Tarjan



Remark after 2 iterations the Tarjan algorithm cannot progress any more

Algebraic loop - structure digraph - Tarjan

$$u_0 = f(t) \tag{33}$$
$$u_1 - R_1 i_1 = 0 \tag{34}$$
$$u_2 - R_2 i_2 = 0 \tag{35}$$
$$u_3 - R_3 i_3 = 0 \tag{36}$$
$$u_1 + u_3 = u_0 \tag{37}$$
$$u_2 - u_3 = 0 \tag{38}$$
$$i_1 - i_2 - i_3 = 0 \tag{39}$$
$$u_L = u_1 + u_2 \tag{40}$$
$$\frac{di_L}{dt} = u_L / L \tag{41}$$
$$i_0 = i_1 + i_L \tag{42}$$

Note The last six equations form an algebraic loop and cannot be sorted then they must be solved all together.

Algebraic loop - structure digraph - Tarjan - cont

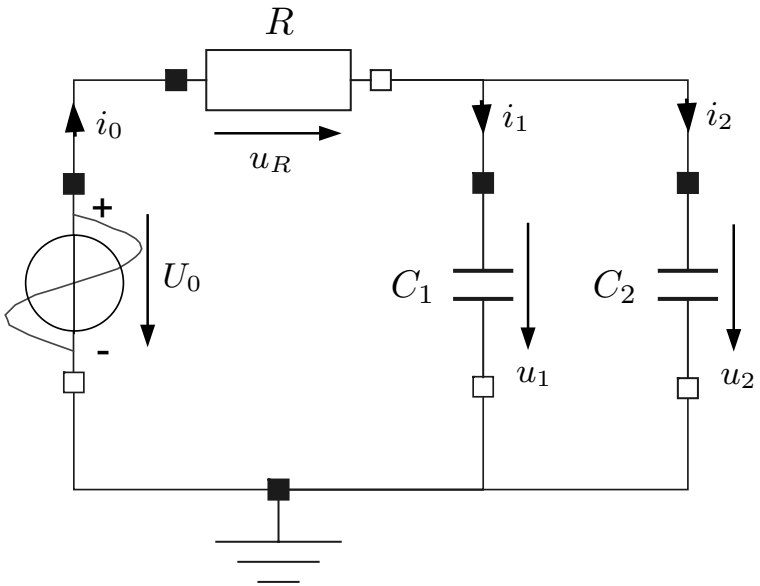
$$
\mathbf{S} = \begin{array}{c} \\ Eq.(7.6a) \\ \\ Eq.(7.6b) \\ Eq.(7.6c) \\ Eq.(7.6d) \\ Eq.(7.6e) \\ Eq.(7.6f) \\ Eq.(7.6g) \\ \\ Eq.(7.6h) \\ \\ Eq.(7.6i) \\ \\ Eq.(7.6j) \end{array}
\begin{array}{c}
\begin{array}{ccccccccccccc}
u_0 & & u_1 & i_1 & u_2 & i_2 & u_3 & i_3 & & u_L & & \frac{di_L}{dt} & & i_0 \\
1 & | & 0 & 0 & 0 & 0 & 0 & 0 & & 0 & & 0 & & 0 \\
- & + & - & - & - & - & - & - & . & & & & & \\
0 & | & 1 & 1 & 0 & 0 & 0 & 0 & | & 0 & & 0 & & 0 \\
0 & | & 0 & 0 & 1 & 1 & 0 & 0 & | & 0 & & 0 & & 0 \\
0 & | & 0 & 0 & 0 & 0 & 1 & 1 & | & 0 & & 0 & & 0 \\
1 & | & 1 & 0 & 0 & 0 & 1 & 0 & | & 0 & & 0 & & 0 \\
0 & | & 0 & 0 & 1 & 0 & 1 & 0 & | & 0 & & 0 & & 0 \\
0 & | & 0 & 1 & 0 & 1 & 0 & 1 & | & 0 & & 0 & & 0 \\
 & . & - & - & - & - & - & - & + & - & . & & & \\
0 & & 1 & 0 & 1 & 0 & 0 & 0 & | & 1 & | & 0 & & 0 \\
 & & & & & & & & . & - & + & - & . & \\
0 & & 0 & 0 & 0 & 1 & 0 & 0 & & 1 & | & 1 & | & 0 \\
 & & & & & & & & & & . & - & + & - \\
0 & & 0 & 1 & 0 & 0 & 0 & 0 & & 0 & & 0 & | & 1
\end{array}
\end{array}
\tag{7.7}
$$

Algebraic loops deserve special treatment:

- in case of linear system: Gauss elimination
- otherwise: Newton algorithm

Algebraic loops are very frequent in multi-body dynamics.

Structural singularity elimination



$$u_0 = f(t) \quad (43)$$

$$u_R = Ri_0 \quad (44)$$

$$i_1 = C_1 \frac{du_1}{dt} \quad (45)$$

$$u_2 = C_2 \frac{du_2}{dt} \quad (46)$$

$$u_0 = u_R + u_1 \quad (47)$$

$$u_2 = u_1 \quad (48)$$

$$i_0 = i_1 + i_2 \quad (49)$$

If the state variables are u_1 and u_2 then Eq. (48) is a **constraint** (a variable as only blue edges in the structure digraph).
Pantelides algorithm can can be used to handle this situation

Pantelides and structural singularity elimination

If $u_2 = u_1$ is true for all t then

$$\frac{du_2}{dt} = \frac{du_1}{dt} \quad \text{for all t} \tag{50}$$

Idea use symbolic differentiation to compute Eq. (50) and replace the constraint by its derivative. Hence,

$$u_0 = f(t) \tag{51}$$
$$u_R = Ri_0 \tag{52}$$
$$i_1 = C_1 \frac{du_1}{dt} \tag{53}$$
$$u_2 = C_2 \frac{du_2}{dt} \tag{54}$$
$$u_0 = u_R + u_1 \tag{55}$$
$$\frac{du_2}{dt} = \frac{du_1}{dt} \tag{56}$$
$$i_0 = i_1 + i_2 \tag{57}$$

Using Tarjan algorithm we get an algebraic loop but we know how to deal with.

Pantelides and structural singularity elimination

Structurally singular systems are also known as **higher index problems**.

- an index-0 contains neither algebraic loop nor structural singularities
- index 1 contains algebraic loops but no structural singularities

Pantelides is a **symbolic index reduction** algorithm. One application reduces the index by 1.

Issues of index reduction

Issues

- **Consistent initial conditions** finding initial value for differential and algebraic variables may be very difficult.
  For
  $$F(\dot{x}, x, t) = 0$$
  x_0 is a consistent initial value, if there exists a smooth solution that fulfills $x(0) = x_0$ and this solution is defined for all t.
  E.g., semi-explicit DAE with only $x(0) = x_0$ what about the algebraic variable?
- **Drift off effect** when applying index reduction the solution of the lower index DAE may not be of the original index.

In consequence, tools/methods to solve DAE should

- provide automatic index reduction
- be able to find consistent initial values

e.g., Dymola/Modelica

Example of consistent initial value

Let

$$\dot{u} = -0.5(u + v) + q_1(t)$$
$$0 = 0.5(u - v) - q_2(t)$$

If $u(0)$ is given we can determine $v(0) = u(0) - 2q_2(0)$ and so $\dot{u}(0)$.
Set $u = y_1 + y_2$ and $v = y_1 - y_2$ we get

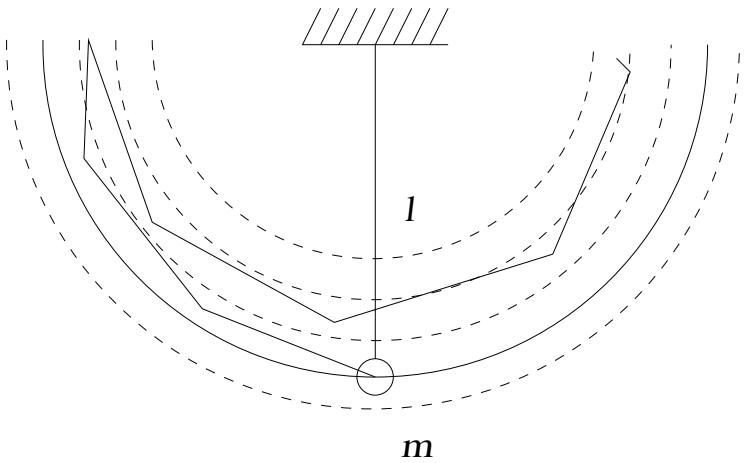
$$\dot{y}_1 + \dot{y}_2 = -y_1 + q_1(t)$$
$$0 = y_2 - q_2(t)$$

For consistency we must have $y_2(0) = q_2(0)$ but we can choose $y_1(0)$ arbitrarily but we cannot determine $\dot{y}_1(0)$ without using $\dot{y}_2(0) = \dot{q}_2(0)$.

Example of drift off effect

Going from index 3 pendulum to index 2 by differentiating the constraint $x_1^2 + x_2^2 - \ell^2 = 0$ leads to

$$\dot{x}_1 = x_3 \tag{58}$$
$$\dot{x}_2 = x_4 \tag{59}$$
$$\dot{x}_3 = -\frac{F}{\ell} x_1 \tag{60}$$
$$\dot{x}_4 = g\frac{F}{\ell} x_2 \tag{61}$$
$$0 = x_1 x_3 + x_2 x_4 \tag{62}$$



Comments:

- solid line curve is the result of index 3 pendulum problem
- Constraint (62) says the velocity should orthogonal to the position. Index reduction increase the space of solution with dashed line curves

A small theory of DAE

For ODE, we have a theorem applying on a large class of problem proving the existence and unicity of the solution
No such theorem exists for DAE
Instead we have some theorems of **solvability** of different kinds of DAE

- Linear constant coefficient DAE
- Linear time varying coefficient DAE
- Non-linear DAE

Solvability of DAE

Definition

Let $\mathcal{I}$ be an open sub-interval of $\mathbb{R}$, Ω a connected open subset of $\mathbb{R}^{2m+1}$, and F a differentiable function from Ω to $\mathbb{R}^m$. Then the DAE $F(\dot{x}, x, t) = 0$ is *solvable* on $\mathcal{I}$ in Ω if there is an r-dimensional family of solutions $\phi(t, c)$ defined on a connected open set $\mathcal{I} \times \tilde{\Omega}$, $\tilde{\Omega} \subset \mathbb{R}^r$, such that

1. $\phi(t, c)$ is defined on all of $\mathcal{I}$ for each $c \in \tilde{\Omega}$
2. $(\phi'(t, c), \phi(t, c), t) \in \Omega$ for $(t, c) \in \mathcal{I} \times \tilde{\Omega}$
3. If $\psi(t)$ is any other solution with $(\psi'(t, c), \psi(t, c), t) \in \Omega$ then $\psi(t) = \phi(t, c)$ for some $c \in \tilde{\Omega}$
4. The graph of ϕ as a function of (t, c) is an $r + 1$-dimensional manifold.

Solvability of linear constant constant DAE

Let

$$A\dot{x} + Bx = f$$

And consider the **matrix pencil** $\lambda A + B$
A matrix pencil is regular if $\det(\lambda A + B)$ is not identically zero as a function of λ.

Theorem

The linear constant coefficient DAE is solvable if and only if $\lambda A + B$ is regular pencil.

Note: the degree of nilpotency of the matrix N used in the decomposition is also the index number of the DAE.

Conclusion

DAE are a generalisation of ODE but

- there is no general theorem to prove existence of the solution of DAE
- differentiation used to index reduction can introduce singularities
- the class of numerical methods used to solve DAE is rather small compare to ODE.

IVP for DAE

We will consider DAE in Hessenberg form of index 1

$$\dot{\mathbf{y}} = f\left(t, \mathbf{y}, \mathbf{z}\right)$$
$$0 = g\left(\mathbf{y}, \mathbf{z}\right) \quad \text{with} \quad \frac{\partial g}{\partial \mathbf{z}} \quad \text{is non-singular}$$
$$\text{with} \quad \mathbf{z}(0) = \mathbf{z}_0 \quad \text{and} \quad \mathbf{y}(0) = \mathbf{y}_0$$

and sometimes, DAE of the following form can be considered

$$\mathbf{M}\dot{\mathbf{y}}(t) = f(\mathbf{y}(t))$$

M is known as the *Mass Matrix*

Relation between DAE and stiff ODE

Singularly perturbed ODE systems are of the form

$$\dot{\mathbf{y}} = f(t, \mathbf{y}, \mathbf{z}) \tag{63}$$
$$\varepsilon \dot{\mathbf{z}} = g(t, \mathbf{z}, \mathbf{y}) \tag{64}$$

When $\varepsilon = 0$ then we get a DAE but Eq. (63) is usually stiff.
DAE can be seen as infinitely stiff.

Consequence

not all numerical method to solve ODE can be used to solve DAE!
we want *A*-stable methods (event *L*-stable) but stiffly stable is enough (as for BDF)

State-space method to solve DAE index 1

$$\dot{\mathbf{y}} = f\left(t, \mathbf{y}, \mathbf{z}\right)$$

$$0 = g\left(\mathbf{y}, \mathbf{z}\right) \quad \text{with} \quad \frac{\partial g}{\partial \mathbf{z}} \quad \text{is non-singular}$$

$$\text{with} \quad \mathbf{z}(0) = \mathbf{z}_0 \quad \text{and} \quad \mathbf{y}(0) = \mathbf{y}_0$$

By *Implicit function theorem* there exists (at leat locally) a function $G(\mathbf{y})$ such that

$$\mathbf{z} = G(\mathbf{y})$$

By substitution we can have

$$\dot{\mathbf{y}} = f(t, \mathbf{y}, G(y))$$

which can be solved by any method for IVP ODE **but**

- you lose the structure of the problem
- G is not so simple to get

ε-embedding approach – Runge-Kutta case

$$\dot{\mathbf{y}} = f(t, \mathbf{y}, \mathbf{z})$$

$$\varepsilon \dot{\mathbf{z}} = g(\mathbf{y}, \mathbf{z}) \quad \text{with} \quad \frac{\partial g}{\partial \mathbf{z}} \quad \text{is non-singular}$$

$$\text{with} \quad \mathbf{z}(0) = \mathbf{z}_0 \quad \text{and} \quad \mathbf{y}(0) = \mathbf{y}_0$$

Applying a Runge-Kutta method,

$$\mathbf{Y}_{\text{ni}} = \mathbf{y}_n + h \sum_{j=1}^{s} a_{ij} f(\mathbf{Y}_{\text{nj}}, \mathbf{Z}_{\text{nj}})$$

$$\varepsilon \mathbf{Z}_{\text{ni}} = \varepsilon \mathbf{z}_n + h \sum_{j=1}^{s} a_{ij} g(\mathbf{Y}_{\text{nj}}, \mathbf{Z}_{\text{nj}})$$

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h \sum_{i=1}^{s} b_i f(\mathbf{Y}_i, \mathbf{Z}_i)$$

$$\varepsilon \mathbf{z}_{n+1} = \varepsilon \mathbf{z}_n + h \sum_{i=1}^{s} b_i g(\mathbf{Y}_i, \mathbf{Z}_i)$$

ε-embedding approach - Runge-Kutta case – cont'

Applying a Runge-Kutta method,

$$\mathbf{Y}_{\mathrm{ni}} = \mathbf{y}_n + h \sum_{j=1}^{s} a_{ij} f(\mathbf{Y}_{\mathrm{nj}}, \mathbf{Z}_{\mathrm{nj}})$$

$$\varepsilon \mathbf{Z}_{\mathrm{ni}} = \varepsilon \mathbf{z}_n + h \sum_{j=1}^{s} a_{ij} g(\mathbf{Y}_{\mathrm{nj}}, \mathbf{Z}_{\mathrm{nj}})$$

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h \sum_{i=1}^{s} b_i f(\mathbf{Y}_i, \mathbf{Z}_i)$$

$$\varepsilon \mathbf{z}_{n+1} = \varepsilon \mathbf{z}_n + h \sum_{i=1}^{s} b_i g(\mathbf{Y}_i, \mathbf{Z}_i)$$

assuming the matrix $\mathbf{A}$ of coefficients a_{ij} is non singular,

$$h g(\mathbf{Y}_{\mathrm{ni}}, \mathbf{Z}_{\mathrm{ni}}) = \varepsilon \sum_{j=1}^{s} \omega_{ij} \left(\mathbf{Y}_{\mathrm{nj}} - \mathbf{z}_n\right) \quad \text{with} \quad \omega_{ij} = (a_{ij})^{-1}$$

ε-embedding approach - Runge-Kutta case – cont'

From

$$hg(\mathbf{Y}_{\text{ni}}, \mathbf{Z}_{\text{ni}}) = \varepsilon \sum_{j=1}^{s} \omega_{ij} \left(\mathbf{Y}_{\text{nj}} - \mathbf{z}_n\right) \quad \text{with} \quad \omega_{ij} = (a_{ij})^{-1}$$

we get,

$$\begin{aligned}
\mathbf{Y}_{\text{ni}} &= \mathbf{y}_n + h \sum_{j=1}^{s} a_{ij} f(\mathbf{Y}_{\text{nj}}, \mathbf{Z}_{\text{nj}}) \\
0 &= g\left(\mathbf{Y}_{\text{ni}}, \mathbf{Z}_{\text{ni}}\right) \\
\mathbf{y}_{n+1} &= \mathbf{y}_n + h \sum_{i=1}^{s} b_i f(\mathbf{Y}_i, \mathbf{Z}_i) \\
\mathbf{z}_{n+1} &= \left(1 - \sum_{i,j=1}^{s} b_i \omega_{ij}\right) \mathbf{z}_n + \sum_{i,j=1}^{s} b_i \omega_{ij} \mathbf{Z}_{\text{nj}} \quad \text{independence wrt } \varepsilon
\end{aligned}$$

Remark: this approach can lead to numerical divergence as the solution may not respect the constraint $g(y, z) = 0$

ε-embedding approach/State-space method

Approximating state-space method can be reached by the formula

$$\mathbf{Y}_{\mathrm{ni}} = \mathbf{y}_n + h \sum_{j=1}^{s} a_{ij} f(\mathbf{Y}_{\mathrm{nj}}, \mathbf{Z}_{\mathrm{nj}})$$

$$0 = g\left(\mathbf{Y}_{\mathrm{ni}}, \mathbf{Z}_{\mathrm{ni}}\right)$$

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h \sum_{i=1}^{s} b_i f(\mathbf{Y}_i, \mathbf{Z}_i)$$

$$0 = g(\mathbf{y}_{n+1}, \mathbf{z}_{n+1})$$

Remarks

- For stiffly accurate methods (see next slide) ε-embedding method and state-space method are identical
- ε-embedding method can be generalized to other classes of DAE index 1 (mass matrix form or implicit form)

Solving DAE with Runge-Kutta methods

A Runge-Kutta is defined by its Butcher tableau

$$\begin{array}{c|ccc} c_1 & a_{11} & a_{12} & \cdots & a_{1s} \\ \vdots & \vdots & \vdots & & \vdots \\ c_s & a_{s1} & a_{s2} & \cdots & a_{ss} \\ \hline & b_1 & b_2 & \cdots & b_s \\ & b'_1 & b'_2 & \cdots & b'_s \end{array} \quad \text{(optional)}$$

Remark

For DAE, we only consider fully implicit Runge-Kutta methods which are L-stable, with A non-singular and with $b_j = a_{sj}$ $(j = 1, 2, \ldots, s)$.
The most used method are **Backward Euler's method** and **Radau IIA order 5**.

Remark:

- the last condition $b_j = a_{sj}$ is good as the last step of RK method is not applied on algebraic variable.
- Stiffly accurate is sufficient for semi-explicit index 1 but not for higher index

Multi-step methods

Recall: single-step methods solve IVP using one value $\mathbf{y}_n$ and some values of f.
A multi-step method approximate solution $\mathbf{y}_{n+1}$ of IVP using k previous values of the solution $\mathbf{y}_n$, $\mathbf{y}_{n-1}$, $\ldots$, $\mathbf{y}_{n-k-1}$.
Different methods implement this approach

- Adams-Bashworth method (explicit)
- Adams-Moulton method (implicit)
- Backward Difference Method (implicit)

The general form of such method is

$$\sum_{j=0}^{k} \alpha_j \mathbf{y}_{n+j} = h \sum_{j=0}^{k} \beta_j f(t_{n+j}, \mathbf{y}_{n+j}) \ .$$

with α_j and β_j some constants and $\alpha_k = 1$ and $|\alpha_0| + |\beta_0| \neq 0$

Solving DAE with multi-step methods

We consider

$$\dot{\mathbf{y}} = f(t, \mathbf{y}, \mathbf{z})$$
$$0 = g(\mathbf{y}, \mathbf{z}) \quad \text{with} \quad \frac{\partial g}{\partial \mathbf{z}} \quad \text{is non-singular}$$
$$\text{with} \quad \mathbf{z}(0) = \mathbf{z}_0 \quad \text{and} \quad \mathbf{y}(0) = \mathbf{y}_0$$

by using ε-embedding method.

$$\dot{\mathbf{y}} = f(t, \mathbf{y}, \mathbf{z})$$
$$\varepsilon\dot{\mathbf{z}} = g(\mathbf{y}, \mathbf{z}) \quad \text{with} \quad \frac{\partial g}{\partial \mathbf{z}} \quad \text{is non-singular}$$
$$\text{with} \quad \mathbf{z}(0) = \mathbf{z}_0 \quad \text{and} \quad \mathbf{y}(0) = \mathbf{y}_0$$

Applying, multi-step method, we get

$$\sum_{i=0}^{k} \alpha_i \mathbf{y}_{n+i} = h \sum_{i=0}^{k} \beta_i f(\mathbf{y}_{n+i}, \mathbf{z}_{n+i})$$
$$\varepsilon \sum_{i=0}^{k} \alpha_i \mathbf{z}_{n+i} = h \sum_{i=0}^{k} \beta_i g(\mathbf{y}_{n+i}, \mathbf{z}_{n+i})$$

ε-embedding method – multi-step case - cont'

Applying, multi-step method, we get

$$\sum_{i=0}^{k} \alpha_i \mathbf{y}_{n+i} = h \sum_{i=0}^{k} \beta_i f(\mathbf{y}_{n+i}, \mathbf{z}_{n+i})$$
$$\varepsilon \sum_{i=0}^{k} \alpha_i \mathbf{z}_{n+i} = h \sum_{i=0}^{k} \beta_i g(\mathbf{y}_{n+i}, \mathbf{z}_{n+i})$$

and setting $\varepsilon = 0$ we get

$$\sum_{i=0}^{k} \alpha_i \mathbf{y}_{n+i} = h \sum_{i=0}^{k} \beta_i f(\mathbf{y}_{n+i}, \mathbf{z}_{n+i})$$
$$0 = h \sum_{i=0}^{k} \beta_i g(\mathbf{y}_{n+i}, \mathbf{z}_{n+i})$$

A state-space method can be applied by using

$$\sum_{i=0}^{k} \alpha_i \mathbf{y}_{n+i} = h \sum_{i=0}^{k} \beta_i f(\mathbf{y}_{n+i}, \mathbf{z}_{n+i})$$
$$0 = g(\mathbf{y}_{n+k}, \mathbf{z}_{n+k})$$

Solving DAE index 1 with BDF

For BDF one has

$$\frac{1}{h\beta_0}\sum_{i=0}^{k}\alpha_i \mathbf{y}_{n+i} = f(\mathbf{y}_{n+k}, \mathbf{z}_{n+k})$$

$$0 = g(\mathbf{y}_{n+k}, \mathbf{z}_{n+k})$$

Remarks

- we still need stiffly accurate method so BDF has to be considered
- Can be applied on DAE index 2 also

Convergence

m-step BDF with $m < 6$ converge; *i.e.*,

$$\mathbf{y}(t_i) - \mathbf{y}_i \leq \mathcal{O}(h^m) \quad \text{and} \quad \mathbf{z}(t_i) - \mathbf{z}_i \leq \mathcal{O}(h^m)$$

for **consistent initial values**.