

Lecture 2

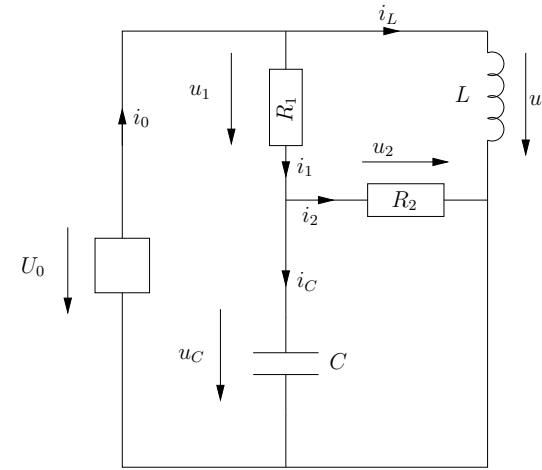
Simulation of differential-algebraic equations

Erik Frisk

erik.frisk@liu.se

Department of Electrical Engineering
Linköping University

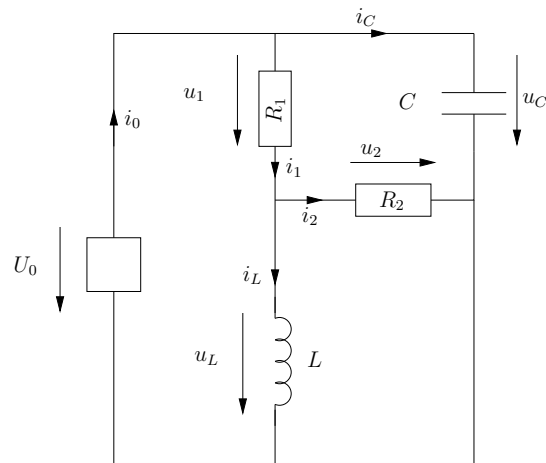
November 08, 2017



$$\begin{aligned} e_1 : u_0 &= f(t) \\ e_2 : u_1 &= R_1 i_1 \\ e_3 : u_2 &= R_2 i_2 \\ e_4 : i_C &= C \frac{du_C}{dt} \\ e_5 : u_L &= L \frac{di_L}{dt} \\ e_6 : i_0 &= i_1 + i_L \\ e_7 : i_1 &= i_2 + i_C \\ e_8 : u_0 &= u_1 + u_C \\ e_9 : u_L &= u_1 + u_2 \\ e_{10} : u_C &= u_2 \end{aligned}$$

$$x_1 = (u_C, i_L), x_2 = (u_2, i_2, u_0, u_1, u_L, i_1, i_C, i_0)$$

Simple circuit model, index > 1



$$\begin{aligned} e_1 : u_0 &= f(t) \\ e_2 : u_1 &= R_1 i_1 \\ e_3 : u_2 &= R_2 i_2 \\ e_4 : i_C &= C \frac{du_C}{dt} \\ e_5 : u_L &= L \frac{di_L}{dt} \\ e_6 : i_0 &= i_1 + i_C \\ e_7 : i_1 &= i_2 + i_L \\ e_8 : u_0 &= u_1 + u_L \\ e_9 : u_C &= u_1 + u_2 \\ e_{10} : u_L &= u_2 \end{aligned}$$

$$x_1 = (u_C, i_L), x_2 = (u_2, i_2, u_0, u_1, u_L, i_1, i_C, i_0)$$

First case in computational form

$$\begin{aligned} e_1 : u_0 &= f(t) \\ e_2 : u_1 &= R_1 i_1 \\ e_3 : u_2 &= R_2 i_2 \\ e_4 : i_C &= C \frac{du_C}{dt} \\ e_5 : u_L &= L \frac{di_L}{dt} \\ e_6 : i_0 &= i_1 + i_L \\ e_7 : i_1 &= i_2 + i_C \\ e_8 : u_0 &= u_1 + u_C \\ e_9 : u_L &= u_1 + u_2 \\ e_{10} : u_C &= u_2 \end{aligned}$$

⇒

$$\begin{aligned} e_4 : \frac{du_C}{dt} &= \frac{1}{C} i_C \\ e_5 : \frac{di_L}{dt} &= \frac{1}{L} u_L \\ e_{10} : u_2 &:= u_C \\ e_3 : i_2 &:= \frac{1}{R_2} u_2 \\ e_1 : u_0 &:= f(t) \\ e_8 : u_1 &:= u_0 - u_C \\ e_9 : u_L &:= u_1 + u_2 \\ e_2 : i_1 &:= \frac{1}{R_1} u_1 \\ e_7 : i_C &:= i_1 - i_2 \\ e_6 : i_0 &:= i_1 + i_L \end{aligned}$$

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

Definition

The minimum number of times the DAE has to be differentiated with respect to t to be able to determine \dot{y} as a function of t och y is called the (differential-) index of the DAE.

- index might be solution dependent, uniform index
- There are several types of index, the above is called differential index.
- Perturbation index
- variants of the above (see paper)

Anyhow: index is a measure how far from an ODE the DAE is.

Simulation of DAE with index > 1

- Starts with a linear DAE with constant coefficients, this is sufficient to illustrate the main reasons why high index problems are difficult

$$A\dot{y}(t) + By(t) = g(t), \text{ linear, constant coefficients}$$

$$A(t)\dot{y}(t) + B(t)y(t) = g(t), \text{ linear, time-varying coefficients}$$

$$F(\dot{y}, y, t) = 0, \text{ general DAE}$$

Consider the index-3 problem

$$x(t) = g(t)$$

$$\dot{x} = y$$

$$\dot{y} = z$$

which has the solution $x(t) = g(t)$, $y(t) = \dot{g}(t)$, and $z(t) = \ddot{g}(t)$.

- *Simulation of high index DAE:s, key problems*
- *Simulation of index 1 DAE:s*
 - *State-space method*
 - *ϵ -embedding*
 - *BDF*
- *Index reduction*
 - *Index reduction by differentiation*
 - *Drift stabilization*

Backward Euler, fix step length

A backward Euler on the problem gives the equations

$$x_n = g_n$$

$$y_n = \frac{x_n - x_{n-1}}{h}$$

$$z_n = \frac{y_n - y_{n-1}}{h}$$

Substitutions give

$$x_n = g_n$$

$$y_n = \frac{g_n - g_{n-1}}{h} = \dot{g}_n + \mathcal{O}(h)$$

$$z_n = \frac{g_n - 2g_{n-1} + g_{n-2}}{h^2} = \ddot{g}_n + \mathcal{O}(h)$$

This looks great!

$$x(t_n) - x_n = 0, \quad y(t_n) - y_n = \mathcal{O}(h), \quad z(t_n) - z_n = \mathcal{O}(h)$$

Fix step length

Theorem

If a k step BDF ($k < 7$) is applied to

$$A\dot{y}(t) + By(t) = g(t)$$

the solution will be $O(h^k)$ after $\max(m-1)k + 1$ steps.

This would indicate that an algorithm with fixed step length would work fine also for high index.

Unfortunately, this breaks down for variable step length integrators.

Constant DAE, variable step length, cont.

One result for variable step length

Theorem

If a k step BDF ($k < 7$) is applied on

$$A\dot{y}(t) + By(t) = g(t)$$

and the ratio between successive step lengths are bounded, then the solution will be $O(h_{\max}^q)$ where $q = \min(k, k - m + 2)$.

- This gives that an index 6 problem could be solved by a 6-step BDF, or?
- Which precondition is here questionable, and why?
- One step backwards Euler with fixed step length is a recommended approach for linear high-index problems with **constant** coefficients.

Constant DAE, variable step length

Now assume variable step length, i.e., the algorithm becomes

$$\begin{aligned}x_n &= g_n \\y_n &= \frac{x_n - x_{n-1}}{h_n} \\z_n &= \frac{y_n - y_{n-1}}{h_n}\end{aligned}$$

Now, the error in z_n , i.e.,

$$z_n - g_n'' = \dots = \frac{1}{2} \left(\frac{h_{n-1}}{h_n} - 1 \right) \ddot{g}_n + O(h)$$

This means that the error will diverge(!) with decreasing h_n , $O(h_n^{-1})$ (index > 2). The error in y will be $O(1)$ (index 2).

One of the exercises is to show ... in the expression.

Hint: Taylor expansion around $t = t_n$.

Linear, non-constant DAE:s

For a DAE

$$A(t)\dot{y}(t) + B(t)y(t) = g(t)$$

you can define local index at time t and global index via a transformation to a canonical form by $y(t) = H(t)z(t)$ and $G(t)$ to

$$G(t)A(t)(H'(t)z(t) + H(t)\dot{z}(t)) + G(t)B(t)H(t)z(t) = G(t)g(t)$$

such that

$$G(t)A(t)H(t) = \begin{bmatrix} I & 0 \\ 0 & E \end{bmatrix}, \quad G(t)A(t)H'(t) + G(t)B(t)H(t) = \begin{bmatrix} C(t) & 0 \\ 0 & I \end{bmatrix}$$

where E is nilpotent, i.e., the same canonical form as previously shown for linear, constant, DAE:s

$$\begin{aligned}\dot{z}_1 + C(t)z_1 &= G_1(t)g(t) \\ E\dot{z}_2 + z_2 &= G_2(t)g(t)\end{aligned}$$

- If we can transform the DAE, all is well. Then the time variable problem is no more difficult than the time invariant.
- Problem: Finding the transformation matrices is not easy
- What happens if you "go with it" anyway with a fixed step length BDF for a time variable system?

From Gear, Petzold:

If the local index is two, we may have a stability problem depending on how fast the matrices change with time. If the local index is larger than 2, we almost always have a stability problem.

Note: Stability problem, not an accuracy problem

Simplify to

$$(E + hI)Q_n z_n = EQ_n z_{n-1} + hq_n$$

Now, solve for z_n according to

$$z_n = Q_n^{-1}(E + hI)^{-1}EQ_n z_{n-1} + hQ_n^{-1}(E + hI)^{-1}q_n = S_n z_{n-1} + u_n$$

The real solution satisfies

$$z(t_n) = S_n z(t_{n-1}) + u_n - \frac{h^2}{2} S_n z''(\xi)$$

With $e_n = z_n - z(t_n)$ we get the recursion

$$e_n = S_n e_{n-1} + \frac{h^2}{2} S_n z''(\xi)$$

Now we have a (recursive) expression for the simulation error, time to analyze!

A DAE with local index m can be transformed into

$$(P_n E Q_n) z' + (P_n Q_n) z = P_n q_n$$

where P_n and Q_n are time variable transformation matrices and the constant matrix E is in the form ($m = 3$)

$$E = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad E^m = 0$$

Apply BDF on the DAE

$$(P_n E Q_n)(z_n - z_{n-1}) + h(P_n Q_n)z_n = hP_n q_n$$

which then can be written as

$$(P_n E Q_n + hP_n Q_n)z_n = P_n E Q_n z_{n-1} + hP_n q_n$$

Expansion of the recursion gives

$$e_n = S_n e_{n-1} + \frac{h^2}{2} S_n z''(\xi)$$

where we have the explicit expression for the fault

$$e_n = \frac{h^2}{2} \sum_{i=1}^n \left(\prod_{j=i}^n S_j \right) z''_i + \prod_{j=0}^n S_j e_0$$

Three problems

- z'' not bounded
- $e_0 \neq 0$
- S_j not bounded

If	$\dot{x} = f(x, y)$	If	$F(t, y, \dot{y}) = 0$
	$0 = g(x, y)$	has index ν then	
has index ν then		$\dot{y} = u$	
	$\dot{x} = f(x, \dot{u})$	$0 = F(t, y, u)$	
	$0 = g(x, \dot{u})$	index $\nu + 1$.	

index $\nu - 1$.

Rule of thumb: The semi-explicit case behaves as the general but with a higher index (and vice versa)

Simulation of semi-explicit index 1 DAE:s

- State-space method
- ϵ -embedding
- BDF (DASSL/DASPK/DASKR)
- DASSL (with variants) is a commonly used DAE solver. Can be downloaded online.

Is described in detail in the nice book "Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations" by K.E. Brenan, S.L. Campbell and L.R. Petzold. The book is in our library and I have uploaded the chapter on DASSL on the course web page for the interested.

I will show the basic principles of DASSL at the end of this lecture.

Outline

- Simulation of high index DAE:s, key problems
- Simulation of index 1 DAE:s
 - State-space method
 - ϵ -embedding
 - BDF
- Index reduction
 - Index reduction by differentiation
 - Drift stabilization

State-space method

$$\begin{aligned}y' &= f(y, z) \\ 0 &= g(y, z), \quad g_z \text{ invertible}\end{aligned}$$

Implicit function theorem then gives that there exists (locally) a function $G(y)$ such that

$$z = G(y)$$

Substitution into the first equations gives the ODE

$$y' = f(y, G(y))$$

which can be solved using your method of choice, no new theory. You can even use an explicit solver if you like.

Lose the structure of the problem which might lead to unnecessary numerical difficulties.

Outline

- Simulation of high index DAE:s, key problems
- Simulation of index 1 DAE:s
 - State-space method
 - ϵ -embedding
 - BDF
- Index reduction
 - Index reduction by differentiation
 - Drift stabilization

ϵ -embedding, forts.

Now, let $\epsilon \rightarrow 0$ and RK becomes

$$\begin{aligned}(Y_{ni} - y_n)_{i=1,\dots,s} &= hA(f(Y_{nj}, Z_{nj}))_{j=1,\dots,s} \\ 0 &= g(Y_{nj}, Z_{nj})_{j=1,\dots,s} \\ y_{n+1} &= y_n + hb^T(f(Y_{nj}, Z_{nj}))_{j=1,\dots,s} \\ 0 &= g(y_{n+1}, z_{n+1})\end{aligned}$$

- for a stiffly accurate solver it holds that $z_{n+1} = Z_{ns}$ and the last rewrite is not necessary.
- The solution is identical to the state-space method.
- Methods pretty similar but has some pros and cons respectively
- State-space form does not require an implicit solver
- ϵ -embedding technique possible to generalize to systems not in semi-explicit form (see Hairer-Wanner)

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

ϵ -embedding

Write down, for example, a Runge-Kutta for the ODE

$$y' = f(y, z), \quad \epsilon z' = g(y, z), \quad g_z \text{ invertible}$$

You then get

$$\begin{aligned}(Y_{ni} - y_n)_{i=1,\dots,s} &= hA(f(Y_{nj}, Z_{nj}))_{j=1,\dots,s} \\ \epsilon(Z_{ni} - z_n)_{i=1,\dots,s} &= hA(g(Y_{nj}, Z_{nj}))_{j=1,\dots,s} \\ y_{n+1} &= y_n + hb^T(f(Y_{nj}, Z_{nj}))_{j=1,\dots,s} \\ \epsilon z_{n+1} &= \epsilon z_n + hb^T(g(Y_{nj}, Z_{nj}))_{j=1,\dots,s}\end{aligned}$$

Assume an implicit method which gives that the A matrix is invertible

$$h(g(Y_{nj}, Z_{nj}))_{j=1,\dots,s} = \epsilon A^{-1}(Z_{ni} - z_n)_{i=1,\dots,s}$$

Order of solver: RK for semi-explicit index 1

- If the method is stiffly accurate, the solution is equivalent to $y' = f(y, G(y))$ and a p order method gives

$$y_n - y(t_n) = \mathcal{O}(h^p), \quad z_n - z(t_n) = \mathcal{O}(h^p)$$

under a Lipschitz assumption on G .

- If the solver is not stiffly accurate, you may lose order on the z -component.
- really the same phenomenon, *order reduction*, as for stiff ODE:s
- Example on page 269 in Ascher-Petzold.

- Stiffly accurate is sufficient for semi-explicit index 1
- Does not apply for higher index
- Stiff decay DIRK (Diagonally Implicit Runge Kutta) methods gets serious order reduction for semi-explicit DAE:s of index 2

A s step BDF can be directly applied to the general problem

$$F(t, y', y) = 0$$

without modification with respect to the ODE case.

- Popular method. "BDF is so beautiful that it is hard to imagine something else could be better", Petzold, 1988.
- DASSL/DDASRT/DASPK/DASKR, ...
- Last time I checked OpenModelica used ddasrt (Dubbel precision, dassl with root solver), a predecessor to DASKR

- *Simulation of high index DAE:s, key problems*
- *Simulation of index 1 DAE:s*
 - *State-space method*
 - *ϵ -embedding*
 - *BDF*
- *Index reduction*
 - *Index reduction by differentiation*
 - *Drift stabilization*

- DASSL (and successors) is perhaps the most used DAE solver
- DASSL is designed to solve DAE:s with index 0 and 1 in the general form

$$F(t, y', y) = 0$$

- BDF of order 1 to 5. No order reduction
- Variable step length by an extension of fix step length BDF
- Will spend half next lecture to describe the basics

- *Simulation of high index DAE:s, key problems*
- *Simulation of index 1 DAE:s*
 - *State-space method*
 - *ϵ -embedding*
 - *BDF*
- *Index reduction*
 - *Index reduction by differentiation*
 - *Drift stabilization*

Index reduction by differentiation

Solving high index problems is difficult. For a DAE with index k

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

we can derive an ODE by differentiating the equations k times

$$\begin{aligned} F(y, \dot{y}) &= 0 \\ \frac{d}{dt} F(y, \dot{y}) &= 0 \\ &\vdots \\ \frac{d^k}{dt^k} F(y, \dot{y}) &= 0 \end{aligned}$$

This DAE has exactly the same solution set as the original DAE. One major problem: it is overdetermined!

Find the underlying ODE and simulate that one?

There is a plethora of methods to reduce index.

- Index reduction through differentiation
- Change of variables; think pendulum in polar coordinates. But which coordinate change? Differential-geometry.
- The basic state-space form from control theory

$$\dot{x} = f(x, u)$$

$$y = h(x, u)$$

- dummy-derivatives, will come back to this next time where automatic methods suitable for large scale models (Modelica) is discussed.
- ...

Index reduction by differentiation

Consider the index 1 DAE

$$\begin{aligned} \dot{x}_1 &= f(x_1, x_2) \\ 0 &= g(x_1, x_2), \quad g_{x_2}(x_1, x_2) \text{ invertible} \end{aligned}$$

It is direct to differentiate the second equation and derive the ODE

$$\begin{aligned} \dot{x}_1 &= f(x_1, x_2) \\ \dot{x}_2 &= -g_{x_2}(x_1, x_2)^{-1} g_{x_1}(x_1, x_2) f(x_1, x_2) \end{aligned}$$

What will happen with the solution?

Index reduction through differentiation

We have done this before, an index 3 example is

$$\begin{aligned} e_1 : \dot{x} &= u, & e_3 : \dot{y} &= v, & e_5 : 0 &= x^2 + y^2 - 1 \\ e_2 : \dot{u} &= -x\lambda, & e_4 : \dot{v} &= -1 - y\lambda \end{aligned}$$

Differentiate the algebraic equation 2 times and we have an index 1 DAE.

- Solution sets to the two
- Initial conditions a difficult problem
- Underlying ODE (UODE) and the original DAE
- Invariants, which are maintained?
- Requires projections or other more or less advanced techniques to fulfill the original algebraic constraints.

Baumgarte stabilization

The first (1976) method to stabilize drift. The principle is simple, instead of using the second derivative of the algebraic constraint

$$\ddot{g} = 0$$

in the solver you use

$$\ddot{g} + \alpha \dot{g} + \beta g = 0$$

where α and β are chosen such that the zeros of the polynomial

$$s^2 + \alpha s + \beta$$

lies in the left half plane.

Simple to generalize. Can be tricky to choose parameters α and β with respect to stiffness and other numerical properties.

Drift

- Illustrate drift
- Explain why it appears

Theorem

If we apply a method of order p we will (in the example from the last slide)

$$\|x^2 + y^2 - 1\| \leq h^p (At_n + Bt_n^2)$$

What can you do about this drift?

- Baumgarte stabilization
- Projection based methods
- Use another index reduction technique

Stabilization through projection

Show the basic principle on a semi-explicit DAE with index 2. Not easy to generalize for higher index, see Hairer-Wanner for further discussions.

$$\begin{aligned} e_1 y' &= f(y, z) \\ e_2 0 &= g(y) \end{aligned}$$

Differentiate once

$$0 = g_y(y) f(y, z)$$

By solving an index 1 DAE (e_1, e'_2) we will not necessarily fulfill $g(y_n) = 0$ at each step, even if we start in a consistent starting point.

Principle

- 1 Start in a point y_{n-1}, z_{n-1} .
- 2 Take a step to \tilde{y}_n, \tilde{z}_n with any method.
- 3 Project! One projection that has been suggested is defined by

$$\min_{y_n} \|\tilde{y}_n - y_n\|, \quad g(y_n) = 0$$

This is a non-linear optimization's problem with constraints.
There are many other ways to project to the surface \mathcal{M} .

$$y' = f(y), \quad \varphi(y) = 0$$

- Invariants from conservation laws, index reduction
- Difference compared to DAE, over determined
- $\varphi(y) = 0$ is called a first integral if $\varphi(y)f(y) \equiv 0$ in the neighborhood of the solution.
- Linear first integrals is fulfilled for most methods of integration
- Quadratic first integrals is fulfilled by, e.g., symplectic Runge-Kutta
- More complex invariants are normally not fulfilled.

Lecture 2

Simulation of differential-algebraic equations

Erik Frisk
erik.frisk@liu.se

Department of Electrical Engineering
Linköping University

November 08, 2017