Numerical Solution of Differential-Algebraic Equations

Linda Petzold
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455

Numerical Solution of Differential-Algebraic Equations

by
Linda Petzold
Department of Computer Science
University of Minnesota
Minneapolis, MN 55455

In this lecture we will review and explain some of the basic concepts and results on the numerical solution of differential-algebraic equation (DAE) systems, with emphasis on the solution of mechanical systems. Recent developments and ongoing work in DAE software will be discussed.

Outline

- Introduction to DAEs
- Mathematical structure
- Numerical methods
- Challenges for mechanical system simulation
- The DASSL family

General DAE System

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

- Mathematical structure is more complex than standard-form ODE $y' = f(t, y)$
- $\frac{\partial F}{\partial y'}$ may be singular, and in this case it is not equivalent to ODE
- Simple and natural formulation for modeling many physical systems
- Requires special consideration for formulating problem and choosing and implementing numerical methods

Multibody system

$$\begin{aligned} M(q)q'' &= f(q, q', t) + G^T(q)\lambda \\ 0 &= g(q) \end{aligned}$$

where $\frac{\partial g}{\partial q} = G$

Example: Pendulum in Cartesian coordinates

$$\begin{aligned} x' &= u \\ y' &= v \\ u' &= \lambda x \\ v' &= \lambda y - g \\ x^2 + y^2 - L^2 &= 0 \end{aligned}$$

Numerical ODE Methods

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

- Approximate y, y' by numerical ODE method
- Some methods may also make use of derivatives of F

Implicit Euler method

$$0 = F\left(t_n, y_n, \frac{y_n - y_{n-1}}{\Delta t}\right)$$

Gear, (1971)

Solve by Newton iteration

$$y_n^{(k+1)} = y_n^{(k)} - \underbrace{\left(\frac{1}{\Delta t}\frac{\partial F}{\partial y'} + \frac{\partial F}{\partial y}\right)^{-1}}_{\textit{iteration matrix}} F\left(t_n, y_n^{(k)}, \frac{y_n^{(k)} - y_{n-1}}{\Delta t}\right)$$

Many advantages for direct formulation:
simple, preserves sparsity, satisfies constraints

But, it doesn't always work ...

Index and Simple DAEs

Index is a measure of the degree of singularity of the DAE system

Index One

$$y = g(t)$$

Index Two

$$y_1 = g(t)$$
$$y_2 = y_1'$$

Index Three

$$y_1 = g(t)$$
$$y_2 = y_1'$$
$$y_3 = y_2'$$

- Solutions are completely determined by right hand side
- Initial values must be consistent with right hand side
- Solution may be less continuous than input
- Higher index DAE may have hidden algebraic constraints

Index of Nonlinear DAE Systems

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

global index: number of differentiations needed to solve for y' uniquely in terms of y and t

$$\begin{aligned} F(t, y, y') &= 0 \\ \frac{dF}{dt}(t, y, y', y'') &= 0 \\ &\vdots \\ \frac{d^m F}{dt^m}(t, y, y', \ldots, y^{(m+1)}) &= 0 \end{aligned}$$

- For semi-explicit systems, it is often possible to differentiate only the constraints

Example-Index for Pendulum

$$\begin{aligned}
x' &= u \\
y' &= v \\
u' &= \lambda x \\
v' &= \lambda y - g \\
x^2 + y^2 - L^2 &= 0
\end{aligned}$$

- Differentiate constraint once
  $xx' + yy' = 0$

  Substitute for x', y'
  $xu + yv = 0$
- Differentiate constraint again
  $xu' + yv' + u^2 + v^2 = 0$

  Substitute for u', v'
  $x^2\lambda + y^2\lambda - yg + u^2 + v^2 = 0$

  Simplify using the original constraint
  $L^2\lambda - yg + u^2 + v^2 = 0$
- Differentiate constraint again and solve for λ'
  $\lambda' = \frac{1}{L^2}(vg - 2\lambda ux - 2v(\lambda y - g))$

Thus, the index is three.

Nonlinear DAE Systems and Derivative Array

Derivative Array

$$
\begin{aligned}
F(t, y, y') &= 0 \\
\frac{d}{dt} F(t, y, y', y'') &= 0 \\
&\vdots \\
\frac{d^m F}{dt^m}(t, y, y', \ldots, y^{(m+1)}) &= 0
\end{aligned}
$$

Existence and Uniqueness

Rheinboldt and Rabier, (1990) use a variant of this idea to prove existence and uniqueness, for index-one systems if $F \in C^2$ and $\frac{\partial F}{\partial y'}$ constant rank, and also for higher-index systems, given more continuity and certain matrices constant rank.

General DAE Solvers

ODE-methods can be very effective for certain DAE systems, but they are not applicable to general higher-index systems.

- Construct general DAE solvers using the derivative array via automatic differentiation
- Develop methods and theory for large classes of DAEs arising commonly in applications

Structural Forms

Semi-explicit index-1

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

$$\frac{\partial g}{\partial y} \quad \text{nonsingular}$$

Hessenberg index-2

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

$$\frac{\partial g}{\partial x}\frac{\partial f}{\partial y} \quad \text{nonsingular}$$

Hessenberg index-3

$$\begin{aligned} x' &= F(x, y, z) \\ y' &= G(x, y) \\ 0 &= H(y) \end{aligned}$$

$$\frac{\partial H}{\partial y}\frac{\partial G}{\partial x}\frac{\partial F}{\partial z} \quad \text{nonsingular}$$

Numerical Methods

What can happen?

Consider simple index-3 system

$$\begin{aligned} y_1 &= g(t) \\ y_2 &= y_1' \\ y_3 &= y_2' \end{aligned}$$

Implicit Euler method

$$\begin{aligned} y_{1,n+1} &= g(t_{n+1}) \\ y_{2,n+1} &= \frac{y_{1,n+1} - y_{1,n}}{h_{n+1}} \cong g'(t_{n+1}) \\ y_{3,n+1} = \frac{y_{2,n+1} - y_{2,n}}{h_{n+1}} &= \frac{\left(\frac{y_{1,n+1} - y_{1,n}}{h_{n+1}}\right) - \left(\frac{y_{1,n} - y_{1,n-1}}{h_n}\right)}{h_{n+1}} \end{aligned}$$

OK for constant stepsizes, but blows up as $h_{n+1} \to 0$

Work has focused on:

- Order conditions for linear multistep and Runge-Kutta for index-one and higher-index Hessenberg systems
- Stability, including effect of problem formulation on numerical stability

Order Results for Linear Multistep Methods

Semi-explicit index-1

$$\begin{aligned} x' &= f(x,y,t) \\ 0 &= g(x,y,t) \end{aligned} \qquad \frac{\partial g}{\partial y} \quad \text{nonsingular}$$

Linear multistep method

$$\begin{aligned} \sum_{j=0}^{k} a_j x_{n-j} &= h \sum_{j=0}^{k} b_j f(x_{n-j}, y_{n-j}, t_{n-j}) \\ 0 &= g(x_n, y_n, t_n) \end{aligned}$$

Converges with same order, stability as for ODE

Fully-implicit index-1

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

multistep methods converge if they satisfy the strict stability condition (excludes methods such as trapezoid). The methods must satisfy an *additional set of order conditions* to attain order > 2. The additional order conditions are satisfied by BDF.

März & Griepentrog, (1986)
Lötstedt & Petzold, (1986)

- This convergence result for BDF underlies the index-one BDF DAE codes LSODI (*Hindmarsh & Painter (1981)*) and DASSL (*Petzold (1982)*) and DASRT (*Petzold (1984)*), DASPK (*Brown, Hindmarsh, Petzold (1991)*)

Order Results for BDF

Index-2 Variable-stepsize BDF$(k)(k < 7)$ converges globally to $O(h^k)$ if starting values are sufficiently accurate

Lötstedt & Petzold, (1984)
Brenan & Engquist, (1984)
Gear, Gupta, Leimkuhler, (1985)

Index-3 Fixed-stepsize BDF $(k)(k < 7)$ converges globally to $O(h^k)$ *after k+1 steps* if starting values are sufficiently accurate

Lötstedt & Petzold, (1984)
Brenan & Engquist, (1984)

- Convergence follows an initial boundary layer of nonconvergence. Convergence in a distributional sense has been shown by *Campbell, (1989)*.
- Higher-index systems via BDF studied by *Clark (1987), Gear & Keiper, (1987)*
- Many practical problems for higher-index systems: stability restrictions, ill-conditioning, determining initial conditions, stepsize control, etc.

Runge-Kutta Methods

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

M-stage implicit Runge-Kutta

$$F(t_{n-1} + c_i h, y_{n-1} + h \sum_{j=1}^{M} a_{ij} Y_j', Y_i') = 0 \qquad i = 1, 2, \ldots, M$$

$$y_n = y_{n-1} + h \sum_{i=1}^{M} b_i Y_i'$$

Methods must satisfy a strict stability condition.

Semi-explicit index-1 If $a_{M_j} = b_j$, or if we force constraints to be satisfied after the last stage, order and convergence same as ODEs. Otherwise, there is an additional set of order conditions

Petzold, (1986)
Hairer, Lubich, Roche, (1987)

Higher-index For higher-index systems, the method must satisfy additional order conditions. (Related to B-convergence theory for stiff ODEs.)

Brenan & Petzold, (1988)
Hairer,Lubich, Roche, (1989)
Kværno (1988)

Half-explicit methods for nonstiff systems

Hairer, Lubich, Roche, (1989)

Runge-Kutta Codes

- Radau 5 (3-stage Radau IA order 5), semi-explicit index 1,2,3.

  Hairer, Lubich, Roche, (1989)

- LIMEX extrapolation for semi-implicit index 1

  Deuflhard, Hairer, Zugck, (1985)

- MEXX extrapolation based on half-explicit midpoint for Hessenberg index-2 mechanical systems, also includes stiff option

  Lubich, (1991)

Computational Challenges for Mechanical Systems

$$
\begin{aligned}
M(q)q'' &= f(t,q,q') + G^T(q)\lambda \\
0 &= g(q)
\end{aligned}
$$

- Problem formulation and numerical stability
- Efficiency/parallelization
- High-frequency oscillations
- Rank-deficient constraints
- Discontinuities
- Delays

The DASSL Family

DASSL *(Petzold, (1982))*

$$\begin{aligned} F(t, y, y') &= 0 \\ y(t_0) &= y_0 \end{aligned}$$

- Backward differentiation formulas (BDF)

$$F(t_{n+1}, y_{n+1}, \frac{1}{h} \sum_{i=0}^{k} \alpha_i y_{n+1-i}) = 0$$

- Variable-stepsize, variable-order (1-5) fixed-leading coefficient
- Modified Newton iteration for solving linear system at each time step (forms and factorizes Jacobian only when necessary), user-supplied or finite-difference Jacobian
- LINPACK dense and banded linear solvers
- Must be started with a consistent set of initial conditions (New initialization software for index-1 will be released soon, *Brown, Hindmarsh and Petzold, 1995, higher-index under development*)
- Cannot solve higher-index DAEs without modification
- Available on internet via netlib:

```
mail netlib@ornl.gov
Subject: send ddassl from ode
```

Solving Higher Index Systems Directly

- For index > 2, extensive modification or reformulation to lower index is necessary
- For index-2 Hessenberg DAE, BDF converges but error estimates in DASSL are not designed to handle this class of problems without modification

$$\begin{aligned} x' &= f(t,x) - B(t,x)y \\ 0 &= g(t,x) \end{aligned}$$

  modify the error test so that it does not include errors in y

- Newton convergence test may be modified so that it computes errors in hy
- Matrix is ill-conditioned. May be helpful to scale the constraints, i.e. solve $h^{-1}g(t,x) = 0$

Extensions to DASSL

Root-finding DASRT

- Stop at root of user-prescribed function $g(t, y)$
- Useful for problems with discontinuities
  - Use root-finder to locate discontinuity
  - Restart after discontinuity with new function
- Available on internet via netlib

DASPK

Large-scale systems of DAEs

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

$$y(t_0) = y_0$$

(Brown, Hindmarsh & Petzold, (1992))

- Uses BDF methods of DASSL for time-stepping
- Nonlinear system at each step solved by inexact Newton
- Linear system at each Newton iteration solved by preconditioned GMRES.
- Matrix-vector product is approximated by finite-difference

$$Av \approx \frac{F(t, y + \sigma v, y' + \frac{\alpha}{h}\sigma v) - F(t, y, y')}{\sigma}$$

where $A = \frac{\partial F}{\partial y} + \frac{\alpha}{h}\frac{\partial F}{\partial y}$

- Two versions of parallel DASPK *(Maier and Petzold, (1993))*

DAE Sensitivity Analysis

- DAE system

$$F(t, y, y', p) = 0, \quad y(0) = y_0(p)$$

- Sensitivity analysis computes dy/dp_i for each parameter p_i
- Useful in parameter estimation, optimization, model simplification, experimental design, and process sensitivity
- New solution approach is efficient and easy to use
  - Sensitivity system solved simultaneously with the original DAE system, with little added cost
  - Block-diagonal approximation of the system Jacobian is efficient and achieves 2-step quadratic convergence
  - Directional difference approximation saves storage, requires no added user information
- Three codes have been written and are available
  - DASSLSO, DASPKSO
  - SENSD

Design Optimization and Optimal Control for DAE Systems

$$
\begin{array}{rl}
\text{find} & u(t) \text{ and } x(t) \text{ for } t_0 \le t \le t_f \\
\text{to minimize} & J = \int_{t_0}^{t_f} L(x(t), u(t), t)dt + V(x(t_f)) \\
\text{subject to} & x(t_0) \text{ is given} \\
 & f(t, x(t), x'(t), u(t)) = 0 \\
 & g(t, x(t), u(t)) \ge 0
\end{array}
$$

- Bolza-type objective function
- Differential-algebraic equations (DAEs)
- Constraints
- Includes parameter estimation

Software package DASOPT (under development, *Petzold, Rosen, Park, Gill, Murray, Saunders*)

- Large-scale nonlinear inequality-constrained optimization (SQP)
- Solution and sensitivity analysis for large-scale differential-algebraic (DAE) systems
- Optimization and control of the DAE system
- Parallel methods and software