scholarpedia.org

Differential-algebraic equations - Scholarpedia

Instructions for Authors

23-29 minutes

- Dr. Stephen L. Campbell, North Carolina State University, Raleigh, NC, USA.
- Vu Hoang Linh, Faculty of Mathematics, Mechanics and Informatics, Vietnam National University, Hanoi, Vietnam
- Linda R. Petzold, Department of Mechanical Engineering, and Department of Computer Science, University of California Santa Barbara, CA

A **differential-algebraic equation** (**DAE**) is an equation involving an unknown function and its derivatives. A (first order) DAE in its most general form is given by

$$F(t,x,x')=0, t_0 \le t \le t_f, \quad (1)$$

where $x=x(t)$, the unknown function, and $F=F(t,u,v)$ have N components, denoted by x_i and F_i, $i=1,2,\ldots,N$, respectively. Every DAE can be written as a first order DAE. The term DAE is usually reserved for the case when the highest derivative x' cannot be solved for in terms of the other terms t,x, when (1) is viewed as an algebraic relationship between three variables t,x,x' . The Jacobian $\partial F/\partial v$ along a particular solution of the DAE may be singular. Systems of equations like (1) are also called implicit systems, generalized systems, or descriptor systems. The DAE may be an initial value problem where x is specified at the initial time, $x(t_0)=x_0$, or a boundary value problem, where the solution is subject to N two-point boundary conditions $g(x(t_0),x(t_f))=0$.

The method of solution of a DAE will depend on its structure. A special but important class of DAEs of the form (1) is the semi-explicit DAE or ordinary differential equation (ODE) with constraints

$$y'0==f(t,y,z)g(t,y,z),(2)$$

which appear frequently in applications. Here x=(y,z) and g(t,y,z)=0 are the explicit constraints.

Where do DAEs arise?

DAEs in either the general form (1) or the special form (2) arise in the mathematical modeling of a wide variety of problems from engineering and science such as in multibody and flexible body mechanics, electrical circuit design, optimal control, incompressible fluids, molecular dynamics, chemical kinetics (quasi steady state and partial equilibrium approximations), and chemical process control.

Example. A simple example of a DAE arises from modeling the motion of a pendulum in Cartesian coordinates.

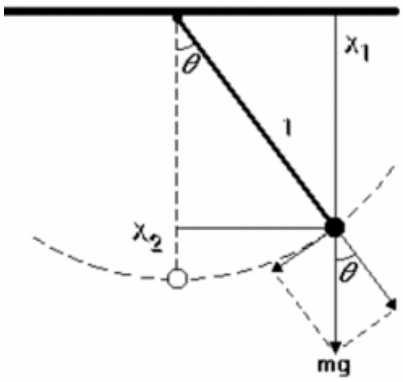

Figure 1: Pendulum

Suppose the pendulum has length 1 and let the coordinates of the tiny ball of mass 1 at the end of the rod be (x1,x2) . Newton's equations of motion give

$$x''1x''2==-\lambda x1-\lambda x2-g,(3)$$

where g is the force of gravity and λ is a Lagrange multiplier. The λxi terms represent the force which holds the solution onto

the constraint

$$x_1^2+x_2^2=1,$$

which expresses the condition that the rod has fixed length 1 . After rewriting the two second order equations as four first order ODEs, a DAE system of the form (2) with four differential and one algebraic equations results:

$$\dot{x}_1=x_3,\quad \dot{x}_2=x_4,\quad \dot{x}_3=-\lambda x_1,\quad \dot{x}_4=-\lambda x_2-g,\quad 0=x_1^2+x_2^2-1. \tag{4}$$

In this very simple case of a multibody mechanical system, the change of variables $x_1=\sin\theta, x_2=\cos\theta$ followed by some algebra gives the well-known ODE for a pendulum $\ddot{\theta}=-g\sin\theta$. However, such a simple elimination procedure is usually impossible in more general situations.

Additional examples of real-life DAE systems, including multibody mechanical systems, an electrical circuit, and a prescribed path control problem can be found in Brenan et al. (1996). It should be noted that the constraint in mechanics, e.g. in the pendulum example, is physical, while the constraint in other problems such as a prescribed path problem is not physical but rather part of the performance specifications.

Why are they important?

DAEs are a generalization of an ordinary differential equations (ODEs)

$$\dot{x}=f(t,x), \tag{5}$$

for which there is a very rich literature for both mathematical theory and numerical solution. While the standard-form ODE can be written as a DAE, the more general DAE form admits problems that can be quite different from a standard-form ODE. The class of DAEs includes problems exhibiting fundamental mathematical properties that are different from those of ODEs, and also pose additional challenges for their numerical solution. On the other hand, implicit DAE models are formulated in a more natural way than explicit ones, as the

above examples demonstrate. It is easier to derive a complex DAE model so it is highly desirable to be able to work with the DAE model if possible.

To get an idea of the difference between DAEs and ODEs, consider the very simple example

$$\tag{6}
\begin{array}{rcl}
y^' & = & z \\
0 & = & y - q(t),
\end{array}$$

where a sufficiently smooth function q is given. Clearly, the only solution is y=q(t), z=q′(t) , and no initial or boundary conditions are needed. That is, if an arbitrary initial condition is imposed, it may well be inconsistent with the DAE. Furthermore, it can be seen that the solution depends on the derivative of the inhomogeneous part (or the derivative of the input, if function q plays the role of input) which cannot happen in the ODE case. Another difference is that even if consistent initial values are given, the existence and uniqueness theory is more complicated and involves additional technical assumptions besides just sufficient smoothness as in the ODE case. The fact that in the first equation of this example, one needs to differentiate y , which implies differentiation of the input function q , in order to find z , makes a key difference. For a standard-form ODE, the solution is always more continuous than the input. In other words, a DAE may involve both integrations and differentiations.

Index and mathematical structure

Index

Index is a notion used in the theory of DAEs for measuring the distance from a DAE to its related ODE. The index is a nonnegative integer that provides useful information about the mathematical structure and potential complications in the

analysis and the numerical solution of the DAE. In general, the higher the index of a DAE, the more difficulties one can expect for its numerical solution. There are different index definitions: Kronecker index (for linear constant coefficient DAEs), differentiation index (Brenan et al. 1996), perturbation index (Hairer et al. 1996), tractability index (Griepentrog et al. 1986), geometric index (Rabier et al. 2002), and strangeness index (Kunkel et al. 2006). On simple problems they are identical. On more complicated nonlinear and fully implicit systems they can be different. In fact, the index can become a local concept with different values in different regions. The index may even be undefined at so-called singular points, which typically exhibit impasse phenomena (Rabier et al. 2002, Riaza 2008).

Since a DAE involves a mixture of differentiations and integrations, one may hope that differentiating the constraints (in a semi-explicit DAE system) and substituting as needed from the differential equations, repeatedly if necessary, will yield an explicit ODE system for all unknowns. The solutions of the DAE are those solutions of this ODE which are in a subset called the solution manifold. The number of repetitions needed for this transformation is called the differential index of the DAE. Thus, ODEs have index 0 . Consider some simple examples.

Example. Let q(t) be a given, smooth function, and consider the following problems for x(t) .

- The scalar equation

  x(t)=q(t)(7)

  is a (trivial) index-1 DAE, since it takes one differentiation to obtain the ODE x′=q′(t) .

- For the system

  x1x2==q(t)x′1,(8)

  one differentiates the first equation to get x2=x′1=q′(t) and then x′2=x′′1=q′′(t) . The index is 2 since two differentiations are needed.

Note that whereas m initial or boundary value conditions must be given to specify the solution of a first order ODE of size m , for the simple DAEs in the above example the solution is completely determined by the right hand side and there is only one initial condition that is consistent. General DAE systems usually include also some ODE subsystems. Thus, a DAE system will in general have *l degrees of freedom*, where l is anywhere between 0 and m . However, in general it may be difficult, or at least not immediately obvious, to determine which l pieces of information are needed to determine the solution. Initial or boundary condition which are specified for the DAE must be consistent. In other words, they must satisfy the constraints and possibly even the differentiated constraints of the system. For example, an initial condition on the index-1 system (7) (which is needed if one writes it as an ODE) must satisfy $x1(0)=q(0)$. For the index-2 system (8), the situation is somewhat more complicated. Not only must any solution satisfy the obvious constraint $x1(t)=q(t)$, there is also a hidden constraint $x2(t)=q'(t)$, so the only consistent initial conditions are $x1(0)=q(0)$, $x2(0)=q'(0)$. This is an important difference between index-1 and *higher-index* (index greater than 1) DAEs. Higher-index DAEs include some hidden constraints.

Consider again the semi-explicit DAE (2). The index is one if $\partial g/\partial z$ is nonsingular because in that case, one differentiation of the algebraic equation yields z' . For the semi-explicit index-1 DAE one can distinguish between differential variables y whose derivative appears in the equations and algebraic variables z whose derivative does not explicitly appear. It is also worth noting that the algebraic variables may be less smooth than the differential variables by one derivative, e.g. the algebraic variables may be non-differentiable.

In the general case (1), each component of the solution x may be a mix of differential and algebraic components, which makes the qualitative analysis as well as the numerical solution of such high-index problems much harder and riskier. The semi-explicit form is decoupled in this sense. Any DAE (1) can be written in

the semi-explicit form by introducing a new variable $z=x'$. However, the index of the new DAE is increased by one. Finally, it is important to note, as the following example illustrates, that in general the index may depend also on a particular solution and not only on the form of the DAE.

Example. Consider the DAE system for $x=(x1,x2,x3)T$

$$x'100===x3x2(1–x2)x1x2+x3(1–x2)–t.(9)$$

The second equation has two solutions $x2=0$ and $x2=1$. If the continuity of $x2$ is given, then $x2$ does not switch between these two values. It is easy to see that if $x2=0$, then the system is in semi-explicit form and has index-1, while for the case $x2=1$, the system has index-2 and unlike the index-1 case, no initial value of $x1$ is required.

Now if one replaces the algebraic equation involving $x2$ by $x'2=0$, then the index of the new DAE system depends on the initial condition. If $x2(0)=1$ the index is 2, otherwise the index is 1.

Special DAE forms

The general DAE system (1) can include problems which are not well-defined in mathematical sense, as well as problems which will result in failure for any direct discretization method (see the Numerical Solution Section). Fortunately, many of the higher-index problems encountered in practice can be expressed as a combination of more restrictive structures of ODEs coupled with constraints. One of the more important classes of systems are the *Hessenberg forms* and are given below.

- **Hessenberg index-1**

  $$y'0==f(t,y,z)g(t,y,z),(10)$$

  where the Jacobian gz is assumed to be nonsingular for all t. This is just a semi-explicit index-1 DAE system mentioned above. Semi-explicit index-1 DAEs are very closely related to implicit ODEs. After solving for z in the algebraic equation (using the implicit function theorem, it can be done in

principle), substituting z into the differential equation yields the so-called underlying ODE in y (although no uniqueness is guaranteed). However, for various reasons, this procedure is not always recommended in practice for numerical solution.

- **Hessenberg index-2**

  y′0==f(t,y,z)g(t,y),(11)

  where gyfz is assumed to be nonsingular for all t. Note that the algebraic variable z is absent from the second equation. This is a pure index-2 DAE and all algebraic variables play the role of index-2 variables. An example arising from modeling incompressible fluid flow by discretized Navier-Stokes equations is given in Ascher et al.(1998).

Numerical solution

Numerical approaches for the solution of DAEs can be divided into roughly two classes: (i) direct discretizations of the given system and (ii) methods which involve a reformulation (e.g. index reduction), combined with a discretization. The desire for as direct a discretization as possible arises because a reformulation may be costly, it may require more input from the user, and it may involve more user intervention. The reason for the popularity of reformulation approaches is that, as it turns out, direct discretizations are limited in their utility essentially to index-1, index-2 Hessenberg, and index -3 Hessenberg DAE systems.

Fortunately, many DAEs encountered in practical applications are either index-1 or, if higher-index, can be expressed as a simple combination of Hessenberg systems. However, some worse-case difficulties may occur and the most robust direct applications of numerical ODE methods do not always work as one might hope, even for these restricted classes of problems. For a DAE of index greater than two it is usually best to use one of the index-reduction techniques to solve the problem in a lower-index form.

Differential equations such as

y′εz′==f(t,y,z)g(t,y,z),(12)

where ε is a small parameter are called singularly perturbed ODE systems. When the parameter ε is set to be 0 , (12) becomes the DAE (2). Since the system (12) is (in general) very stiff for small ε , it is natural to consider methods for stiff ODEs for the direct discretization of the limit DAE, and for DAEs of the form (1) in general. In particular, ODE methods which have stiff decay such as BDF and Radau collocation methods are useful.

Numerical methods/Direct discretization

- Backward Euler method/Example of instability

The idea of a direct discretization is simple: approximate x and x′ by a discretization formula like multistep methods or Runge-Kutta methods. As an illustration of the use of direct discretization, consider the backward Euler method, the simplest method which has the stiff decay property. Applying the backward difference formula to x′ in (1), a system of N nonlinear equations for xn

F(tn,xn,xn−xn−1hn)=0 for n=1,2,...,(13)

results. Here tn are the time points where we are computing the approximation, xn is the approximation of x(tn) , and hn=tn−tn−1 is the time step or step size. Once this nonlinear equation system has been recursively solved, a numerical solution for (1) is obtained. This method works well for index-1 DAEs, and is particularly appropriate for stiff index-1 DAEs, as well as for stiff ODEs.

For higher-index DAEs this simple method, as well as other methods, does not always work. In the worst case, there are simple higher-index DAE systems with well-defined and stable solutions for which the backward Euler method, and in fact all other multistep and Runge-Kutta methods, are unstable or not even applicable. See Example 10.1 in Ascher et al. (1998). See also a multibody mechanics simulation, where the use of stable

versus unstable numerical methods is visualized. Some practical difficulties may occur as well during solving the nonlinear system (13) for xn given xn−1 . The solution must be accomplished with a type of iterative numerical method such as a Newton method. These technical difficulties are why, in general, a direct discretization of fully implicit DAEs of index higher than one is not recommended. For fully implicit index-1 and semi-explicit index-2 DAEs, it has been shown that the Backward Euler method is first-order accurate, stable and convergent. Detailed discussions and convergence results can be found in Brenan et al. (1996) and Hairer et al. (1998).

- BDF and general linear multistep methods

  Euler is only a first-order method. To get a more accurate solution without taking smaller steps, a higher order method is needed. The constant step-size BDF method applied to a general nonlinear DAE of the form (1) is given by

  F(tn,xn,1β0h∑j=0kαjxn−j)=0,(14)

  where β0 and αj, j=0,1,...,k, are the coefficients of the BDF method. It has been shown that the k-step BDF method of fixed step-size h is convergent of order 0(hk) if all initial values are correct to 0(hk) , and if the Newton iteration on each step is solved to accuracy 0(hk+1) . For general linear multistep methods, similar convergence results have been established, provided that the coefficients of the multistep methods satisfy a set of order conditions which is in addition to the order conditions for ODEs, to attain order greater than 2. These extra conditions are satisfied by BDF methods. For more details, see Brenan et al. (1996).

- Radau collocation and implicit Runge-Kutta methods

  The s-stage implicit Runge-Kutta method applied to the general nonlinear DAE of the form (1) is given by

  F(tn−1+cih,Xni,Kni)Xni==0,xn−1+h∑j=1saijKj,i=1,2,...,s,(15)

  and

$$x_n = x_{n-1} + h\sum_{i=1}^{s} b_i K_{ni}, \quad (16)$$

where c_i, a_{ij}, b_i, $i,j=1,2,\ldots,s$, are the coefficients of the Runge-Kutta method. We assume in addition that the matrix $A=(a_{ij})$ is nonsingular.

For the semi-explicit DAE (2), the formula (15) for the internal stages reads

$$K_{ni} \quad Y_{ni} \quad g(t_{n-1}+c_i h, Y_{ni}, Z_{ni}) = = = f(t_{n-1}+c_i h, Y_{ni}, Z_{ni}), y_{n-1}+h\sum_{j=1}^{s} a_{ij} K_j, 0. \quad i=1,2,\ldots,s$$

It is possible to avoid the quadrature step (16) for the algebraic variables z by making the use of stiffly accurate methods, i.e. Runge-Kutta methods satisfying $b_j = a_{sj}$, $j=1,2,\ldots,s$, . Instead of (16), one simply sets $y_n = Y_{ns}$. As was the case for general multistep methods, there are additional order conditions which the method coefficients must satisfy for the method to attain order greater than 2. For Runge-Kutta methods, the requirement of extra order conditions arises even for semi-explicit index-1 DAEs.

It should also be noted that the implementation of direct discretization methods for DAEs faces some additional practical difficulties such as how to obtain a consistent set of initial conditions, a treatment of the ill-conditioning of iteration matrix, and finally, error estimation and stepsize control for index-2 Hessenberg DAEs.

For some special classes of DAEs such as semi-explicit DAEs in Hessenberg form and ODEs on manifolds, in particular for DAEs arising in multibody mechanics, there exist very efficient and robust numerical methods called stabilized or projected methods. The main idea is first to discretize the differential equations by an appropriate numerical ODE method. This step is followed by a post-stabilization or a coordinate projection step to bring the numerical solution closer to satisfying the constraint, see Eich-Soellner et al. (1998).

For more details on numerics of DAEs, see Ascher et al. (1998), Brenan et al. (1996) and Hairer et al. (1998).

Software

Initial value problems

- The code DASSL by Petzold uses the BDF formulas to solve general index-1 DAEs, see Brenan et al. (1996) for details. Versions for large scale problems (called DASPK) and for sensitivity analysis are also available. Some later versions of DASPK can also solve Hessenberg index-2 DAEs. There is also a code DASPKADJOINT which implements the adjoint method for sensitivity analysis of DAE systems.
- The code RADAU5 by Hairer & Wanner (1998) is based on the 3-stage Radau collocation method. It solves DAEs of the form $Mx'=f(t,x)$, where M is a constant, square matrix which may be singular. The code is applicable to problems of index 1,2,3. The higher-index variables must be identified by the user.
- The code IDA is a part of the software package called SUNDIALS (SUite of Nonlinear and DIfferential/ALgebraic equation Solvers) which was developed by Serban and Hindmarsh at Lawrence Livermore National Laboratory, USA. It is written in C for solving nonlinear DAEs, but derived from the package DASPK which is written in Fortran. A SUNDIALS-related code CPODES (the Coordinate Projection solver for ODEs with invariants), written by Serban, is also available.
- DAEPACK is a software library developed by Paul I. Barton and his group at MIT. DAEPACK is an acronym for Differential-Algebraic Equation Package, however, its scope is not limited to the analysis of DAEs. DAEPACK includes both symbolic and numerical components for modeling as well as for general numerical calculations.
- Other codes: MEXX by Lubich et al. (1992), LIMEX by Deuflhard et al. (1987), GELDA and GENDA by Kunkel et al. (1997) are also available.

Boundary value problems

- The code COLDAE by Ascher et al. (1994) uses projected Gauss

collocation to BVPs for semi-explicit index-2 DAEs.

References

- Ascher U.M.; Petzold L.R. (1998) Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA.
- Brenan K.E.; Campbell S.L.; Petzold L.R. (1996) Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations. Revised and corrected reprint of the 1989 original, with an additional chapter and additional references. Classics in Applied Mathematics, 14. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA.
- Campbell S.L. and Petzold L.R. (1983) Canonical Forms and Solvable Singular Systems of Differential Equations, SIAM J. Alg. Disc. Meth. 4:517-521
- Hairer E.; Wanner G. (1996) Solving Ordinary Differential Equations. II. Stiff and Differential-Algebraic Problems. Second edition. Springer Series in Computational Mathematics, 14. Springer-Verlag, Berlin.
- Petzold L.R. (1982) Differential/Algebraic Equations are not ODEs, SIAM J. Sci. Stat. Comput., 3:367-384
- Rabier P.J.; Rheinboldt W.C. (2002) Theoretical and Numerical Analysis of Differential-Algebraic Equations. Handbook of Numerical Analysis, Vol. VIII, 183--540, Handb. Numer. Anal., VIII, North-Holland, Amsterdam.
- Riaza R. (2008), Differential-Algebraic Systems: Analytical Aspects and Circuit Applications, World Scientific, Singapore.

Internal references

- John Butcher (2007) Runge-Kutta methods. Scholarpedia, 2(9):3147
- Bill Gear (2007) Backward differentiation formulas. Scholarpedia, 2(8):3162

- Ian Gladwell (2008) Boundary value problem. Scholarpedia, 3(1):2853
- Alan C. Hindmarsh and Radu Serban (2007) Sundials equation solvers. Scholarpedia, 2(3):2860
- Zdzislaw Jackiewicz (2007) General linear methods. Scholarpedia, 2(4):2852
- Lawrence F. Shampine and Skip Thompson (2007) Initial value problems. Scholarpedia, 2(3):2861
- Lawrence F. Shampine and Skip Thompson (2007) Stiff systems. Scholarpedia, 2(3):2855

Internal references

- Bill Gear (2007) Backward differentiation formulas. Scholarpedia, 2(8):3162.
- Ian Gladwell (2008) Boundary value problem. Scholarpedia, 3(1):2853.
- James Meiss (2007) Dynamical systems. Scholarpedia, 2(2):1629.
- Zdzislaw Jackiewicz (2007) General linear methods. Scholarpedia, 2(4):2852.
- Lawrence F. Shampine and Skip Thompson (2007) Initial value problems. Scholarpedia, 2(3):2861.
- Kendall E. Atkinson (2007) Numerical analysis. Scholarpedia, 2(8):3163.
- John Butcher (2007) Runge-Kutta methods. Scholarpedia, 2(9):3147.
- Philip Holmes and Eric T. Shea-Brown (2006) Stability. Scholarpedia, 1(10):1838.
- Nicola Guglielmi and Ernst Hairer (2007) Stiff delay equations. Scholarpedia, 2(11):2850.
- Lawrence F. Shampine and Skip Thompson (2007) Stiff systems. Scholarpedia, 2(3):2855.

- Alan C. Hindmarsh and Radu Serban (2007) Sundials equation solvers. Scholarpedia, 2(3):2860.

Recommended reading

- Eich-Soellner E. and Führer C. (1998) Numerical Methods in Multibody Systems. Teubner Verlag, Stuttgart, Germany.
- Griepentrog E., März R. (1986) Differential-Algebraic Equations and Their Numerical Treatment. With German, French and Russian summaries. Teubner-Texte zur Mathematik [Teubner Texts in Mathematics], 88. BSB B. G. Teubner Verlagsgesellschaft, Leipzig.
- Hairer E., Lubich C., and Roche M. (1989) The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods, Lecture Notes in Mathematics No. 1409, Springer-Verlag, Berlin.
- Kunkel P., Mehrmann V. (2006), Differential-Algebraic Equations Analysis and Numerical Solution. EMS Publishing House, Zurich, Switzerland.
- Rabier P.J.; Rheinboldt W.C. (2002) Theoretical and Numerical Analysis of Differential-Algebraic Equations. Handbook of Numerical Analysis, Vol. VIII, 183--540, Handb. Numer. Anal., VIII, North-Holland, Amsterdam.
- Riaza R. (2008), Differential-Algebraic Systems: Analytical Aspects and Circuit Applications, World Scientific, Singapore.

External links

- Linda Petzold's website
- Stephen L. Campbell's website
- Vu Hoang Linh's website

See also

Dynamical Systems, Numerical Analysis, Stability, Stiff Delay Equations