

AMSC/CMSC 660 Scientific Computing I

Fall 2006

UNIT 5: Numerical Solution of Ordinary Differential Equations

Dianne P. O'Leary

©2002,2004,2006

---

**The Plan**

- Initial value problems (IVPs) for ordinary differential equations (ODEs)
  - Review 460 ODE notes
  - Hamiltonian systems
- Differential-Algebraic Equations
  - Some basics
  - Some numerical methods
- Boundary value problems for ODEs.
  - Some basics
  - Shooting methods
  - Finite difference methods

---

**References:**

- The 460 notes are based on Chapter 9 of Van Loan's book.
- These notes are based on Parts III and IV of Ascher and Petzold's book.

---

Initial value problems for ordinary differential equations

- Review 460 ODE notes
- Hamiltonian systems

---

**Review 460 ODE notes**

---

**Hamiltonian systems**

In some ODE systems, there is an associated [conservation principle](#), and if possible, we formulate the problem so that conservation is observed.

**Definition:** A [Hamiltonian system](#) is one for which there exists a scalar Hamiltonian function  $H(\mathbf{y})$  so that

$$\mathbf{y}' = \mathbf{D}\nabla_{\mathbf{y}}H(\mathbf{y}),$$

where  $\mathbf{D}$  is a block-diagonal matrix with blocks equal to

$$\mathbf{J} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}.$$

**Example:** Linear harmonic oscillator. Let  $q(t)$  and  $p(t)$  be unknown functions satisfying

$$\begin{aligned} q' &= \omega p \\ p' &= -\omega q \end{aligned}$$

where  $\omega > 0$  is a fixed parameter.

The [Hamiltonian](#) of the system is defined to be

$$H = \frac{\omega}{2}(p^2 + q^2).$$

To verify this, note that if  $\mathbf{y} = [q, p]^T$ , then

$$\nabla_{\mathbf{y}}H(\mathbf{y}) = \begin{bmatrix} \omega q \\ \omega p \end{bmatrix}$$

so that

$$\mathbf{y}' = \begin{bmatrix} \omega p \\ -\omega q \end{bmatrix} = \mathbf{D}\nabla_{\mathbf{y}}H(\mathbf{y}) = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \omega q \\ \omega p \end{bmatrix}.$$

(See <http://scienceworld.wolfram.com/physics/HamiltonsEquations.html> for more information on Hamiltonian systems.)

Note that

$$\begin{aligned} H' &= \frac{\omega}{2}(2pp' + 2qq') \\ &= \frac{\omega}{2}\left(2\frac{q'}{\omega}p' + 2\frac{-p'}{\omega}q'\right) \\ &= 0, \end{aligned}$$

so  $H(t)$  must be constant; in other words, the quantity  $H$  is [conserved](#) or [invariant](#).

We can verify this a different way by writing the general solution to the problem:

$$\begin{bmatrix} q(t) \\ p(t) \end{bmatrix} = \begin{bmatrix} \cos \omega t & \sin \omega t \\ -\sin \omega t & \cos \omega t \end{bmatrix} \begin{bmatrix} q(0) \\ p(0) \end{bmatrix}$$

and computing  $p(t)^2 + q(t)^2$ .

The eigenvalues of the matrix defining the solution are imaginary numbers, so a small perturbation of the matrix can cause the quantity  $H$  to either grow or shrink, and this will not produce a useful solution.

□

Therefore, in solving systems involving Hamiltonians (conserved quantities), it is important to [build conservation into the numerical method](#) whenever possible!

**Example:** If the ODE has the form

$$\begin{aligned} \mathbf{y}' &= \mathbf{f}(t, \mathbf{y}) \\ h(\mathbf{y}) &= 0 \end{aligned}$$

(as in the previous example), then we can rewrite it as

$$\begin{aligned} \mathbf{y}' &= \mathbf{f}(t, \mathbf{y}) - \mathbf{g}(\mathbf{y})z \\ h(\mathbf{y}) &= 0 \end{aligned}$$

where  $z(t)$  is a scalar function (added so that the system is not overdetermined) and  $\mathbf{g}$  is any bounded function whose Jacobian matrix with respect to the variables  $\mathbf{y}$  has an inverse that is bounded away from singularity for all  $t$ .

If we solve this system [exactly](#), then we will get  $z(t) = 0$  and we recover our original solution. But if we solve it [numerically](#), the second equation forces  $z$  to be nonzero in order to keep the solution satisfying the conservation law  $h(\mathbf{y}) = 0$ .

For example, we can rewrite our harmonic oscillator example as

$$\begin{aligned} q' &= \omega p - \omega qz \\ p' &= -\omega q - \omega pz \\ 5 &= \frac{\omega}{2}(p^2 + q^2) \end{aligned}$$

(5 used as an example) □

Adding an invariant, or conservation law, generally changes the ODE system to a system that includes nonlinear equations not involving derivatives – a system of [differential-algebraic equations](#) (DAEs).

**Warning:** Sometimes, adding conservation makes the problem too expensive to solve; for example, if the solution is rapidly oscillating. In such cases, we may decide to allow the conservation law to be violated.

Next we'll consider a little of the theory and computation of DAEs.

---

## Differential-Algebraic Equations

- Some basics
- Some numerical methods

---

### Some basics

The **index** of a DAE is the number of differentiations needed to convert it to an (explicit) system of ordinary differential equations (not necessarily involving only 1st derivatives)

**Examples:** Let  $g$  be a given function, and let  $y$  and  $u$  be the unknown functions.

System	Explicit system	Index
$y = g(t)$	$y' = g'$	1
$y_1 = g(t)$ $y_2 = y_1'$	$y_1' = g', y_1'' = g''$ $y_2' = y_1'', y_2'' = y_1'''$	2
$y = g(t)$ $u = y''$	$y' = g', y'' = g'', y''' = g'''$ $u' = y''', u'' = y'''' , u''' = y'''''$	3

---

### A major difference between DAEs and ODEs

For ODEs, it is easy to count how many initial conditions we need to uniquely determine the solution.

For DAEs, it is not so simple. For each example in the table above, **no** initial conditions are needed.

And even if some initial conditions are necessary, it is difficult to determine whether the conditions given are **consistent** so that a solution exists.

---

### General form of a DAE

$$\mathbf{F}(y, y', t) = \mathbf{0}$$

where the Jacobian of  $\mathbf{F}$  may be singular.

---

### Some numerical methods

The main idea: If we want to solve

$$\mathbf{F}(\mathbf{y}, \mathbf{y}', t) = \mathbf{0},$$

then we can step from known values at  $t = t_n, t_{n-1}, \dots, t_{n-k}$  to unknown values at  $t = t_{n+1}$  using our favorite approximation scheme to replace  $\mathbf{y}'(t_{n+1})$  by

$$\mathbf{y}'(t_{n+1}) \approx \sum_{i=0}^k \alpha_i \mathbf{y}(t_{n-i}).$$

This gives us a nonlinear equation to solve for  $\mathbf{y}_{n+1} \approx \mathbf{y}(t_{n+1})$ :

$$\mathbf{F}(\mathbf{y}_{n+1}, \sum_{i=0}^k \alpha_i \mathbf{y}(t_{n-i}), t_{n+1}) = \mathbf{0}.$$

We can solve this equation using our favorite method (Newton-like, homotopy, ...).

---

### Complications

- **Stability** is an important consideration. The ODE method needs to be chosen carefully; usually a **stiff method** is needed.
- The nonlinear equation **may fail to have a solution**.
- Even if a solution exists, the method you choose for solving the nonlinear equation **may fail to converge**.
- **Automatic control** of order and stepsize is even more difficult than for ODEs.

---

### Bottom line

Don't try to write your own solver for DAEs. Use high-quality software:

- The Matlab ODE solvers handle some DAEs. I believe they are well-written, but I don't have vast personal experience with them.
- There are two high-quality codes:
  - **DASSL** by Linda Petzold solves index-1 problems.
  - **RADAU5** by Hairer and Wanner solves some problems with index up to 3.

- Some basics
- Shooting methods
- finite difference methods

---

### Some basics

In an [initial value problem](#), all of the data values are given at a [single point](#)  $t_0$ .

In a [boundary value problem](#), more than one point is involved.

[Example:](#)

$$\begin{aligned}y'' &= 6y' - ty + y^2 \\y(0) &= 5 \\y(1) &= 0\end{aligned}$$

Find  $y(t)$  for  $t \in (0, 1)$ .

If we convert this to a system of first order equations, we let  $u_1 = y'$ ,  $u_2 = y$ , and

$$\begin{aligned}u_1' &= 6u_1 - tu_2 + u_2^2 \\u_2' &= u_1 \\u_2(0) &= 5 \\u_2(1) &= 0\end{aligned}$$

So we have values of  $u_2$  at 0 and 1. If we had values of  $u_1$  and  $u_2$  at 0, we could use our old (IVP) methods. But now we have a boundary value problem.

---

### What to do?

There are two alternatives:

- adapt our old methods to this problem: shooting.
- develop new methods: finite differences.

---

### Shooting methods

When in doubt, [guess](#). The idea behind shooting methods is to [guess](#) at the missing initial values, integrate the equation using our favorite method, and then use the results to improve our guess.

In fact, we recognize this as a nonlinear system of equations: to solve our example problem,

$$\begin{aligned}u_1' &= 6u_1 - tu_2 + u_2^2 \\u_2' &= u_1 \\u_2(0) &= 5 \\u_2(1) &= 0\end{aligned}$$

we want to solve the nonlinear equation

$$F(z) = 0$$

where  $z$  is the value we give to  $u_1(0)$  and  $F(z)$  is the value that our (IVP) ODE solver returns for  $u_2(1)$ .

So a [shooting method](#) involves using our favorite nonlinear equation solver, with function evaluation through our favorite IVP-ODE solver. Once we find the initial value  $z$ , then the IVP-ODE solver can give us values  $\mathbf{u}(t)$  for any  $t$ .

[Unquiz](#): Write the code to solve this IVP using ODE45 and FZERO.

---

### Warnings

- If the IVP is difficult to solve (for example, stiff), then it will be difficult to get an accurate value of  $z$ .
- Our function evaluation for FZERO is [noisy](#): it includes all of the round-off error and the global discretization error introduced by the IVP-ODE solver. The resulting wiggles in the values of  $F$  can cause the nonlinear equation solver to have trouble finding an accurate solution, and can also introduce multiple solutions where there is really only one.
- If the interval of integration is long, these difficulties can be overwhelming and we need to go to more complicated methods; for example, [multiple shooting](#). See Ascher and Petzold for further discussion.

---

### Finite difference methods

**Unquiz:** Suppose  $y$  has 4 continuous derivatives. Prove that

$$\begin{aligned}y'(t) &= \frac{y(t+h) - y(t-h)}{2h} + O(h^2), \\y''(t) &= \frac{y(t-h) - 2y(t) + y(t+h)}{h^2} + O(h^2)\end{aligned}$$

for small values of  $h$ .  $\square$

Now consider our example problem in its original form:

$$\begin{aligned}y'' &= 6y' - ty + y^2 \\y(0) &= 5 \\y(1) &= 0\end{aligned}$$

Given a large number  $n$  (for example,  $n = 100$ ), let  $h = 1/n$  and define

$$y_j \approx y(jh), \quad j = 0, \dots, n.$$

Then we can approximate our original equation

$$y'' = 6y' - ty + y^2$$

at  $t = t_j$  ( $0 < j < n$ ) by

$$\frac{y_{j-1} - 2y_j + y_{j+1}}{h^2} = 6\frac{y_{j+1} - y_{j-1}}{2h} - t_j y_j + y_j^2.$$

Since we already know that

$$\begin{aligned}y_0 \approx y(0) &= 5 \\y_n \approx y(1) &= 0\end{aligned}$$

we have a system of  $n - 1$  nonlinear equations in  $n - 1$  unknowns and we can solve it using our favorite method.

---

### Final Words

- Initial value problems for ordinary differential equations that arise in practice can be very difficult to solve.
  - Beware of stiff equations.
  - If there is a [conservation law](#) or Hamiltonian, make sure to incorporate it into the formulation. (Otherwise, your customer will be very unhappy with the numerical results.) But be aware that if you don't do this in a smart way, it may cause the ODE solver to take very small steps.

- We have just [touched](#) on the existence, uniqueness, and stability theory for ODEs and DAEs. If you need to solve an important problem, be ready to study these issues further before you go to the computer.
- Numerical solution of DAEs is still an [evolving science](#), so watch the literature if you are working in this field.

---

### References

Charles F. van Loan, *Introduction to Scientific Computing*, Prentice Hall, 2000.

Uri M. Ascher and Linda R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, SIAM Press, Philadelphia, 1998.