Numerical Solutions to PDE’s
Mathematical Modelling Week 5
Kurt Bryan

Introduction

Let’s start by recalling a simple numerical scheme for solving ODE’s. Suppose we have an ODE \( u'(t) = f(t, u(t)) \) for some unknown function \( u(t) \) (\( f \) is specified), with initial condition \( u(0) = u_0 \). Choose some small number \( h_t \), the so-called *stepsize*, and use it to approximate \( u'(t) \) as

\[
    u'(t) \approx \frac{u(t + h_t) - u(t)}{h_t}
\]

a *finite-difference* approximation to the derivative of \( u \). The smaller \( h_t \) is, the better the approximation.

Now define \( t_i = ih_t \) and also \( u(t_i) = u_i \). Take the original DE and replace \( t \) with \( t_i \), \( u(t_i) \) with \( u_i \), and the derivative \( u'(t_i) \) with the appropriate finite difference approximation to obtain

\[
    \frac{u_{i+1} - u_i}{h_t} \approx f(t_i, u_i).
\]

This can be re-arranged into

\[
    u_{i+1} \approx u_i + h_t f(t_i, u_i).
\]

Equation (1) gives us a recipe for approximating \( u(t) \). We start off knowing \( u_0 = u(0) \). We can use (1) to approximate \( u_1 = u(h_t) \). With an estimate of \( u_1 \), we can then approximate \( u_2 \), and then \( u_3 \) and so on. This is just good old Euler’s method.

There are much more sophisticated ways to solve ODE’s, but philosophically they all work in pretty much the same way: Knowing the value of \( u_i \) (and maybe early values), we attempt to extrapolate the solution into the future by making use of the DE. The same idea works for PDE’s.
Finite Differencing for PDE’s

Consider the advection equation

$$\frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = 0 \quad (2)$$

for some function $u(x,t)$ on the half-line $x > 0$ with boundary and initial conditions

$$u(x,0) = f(x), \quad (3)$$
$$u(0,t) = g(t), \quad (4)$$

for some functions $f(x)$ and $g(t)$. The number $c$ is the wave speed and is a positive constant. Although $u$ can be found explicitly, we are going to consider a numerical method for approximating $u$. For a more complicated equation in which $c$ is no longer constant, or even depends on $u$, a numerical solution will be the only option.

Let’s suppose that we’re interested in the solution on the interval $0 \leq x \leq 1$. We will replace the partial derivatives of $u$ by finite-difference approximations. Choose $n + 1$ equally spaced points $x_0, x_1, \ldots, x_n$ in the interval $[0, 1]$ of the form $x_i = i/n$. Let $h_x = 1/n$ denote the spacing between the points. Let’s also divide time $t$ up into increments $h_t$ by setting $t_j = h_{t,j}$ where $h_t$ is some “small” number. The partial derivatives for $u$ can then be approximated as

$$\frac{\partial u}{\partial t}(x_i, t_j) \approx \frac{u(x_i, t_{j+1}) - u(x_i, t_j)}{h_t},$$
$$\frac{\partial u}{\partial x}(x_i, t_j) \approx \frac{u(x_i, t_j) - u(x_{i-1}, t_j)}{h_x}.$$

As $h_x$ and $h_t$ get smaller the approximations typically get better—they’re $o(h_x)$ and $o(h_t)$, at least if $u$ is differentiable enough. Let’s use the notation $u_{ij}$ to mean $u(x_i, t_j)$. In this case we have

$$\frac{\partial u}{\partial t}(x_i, t_j) \approx \frac{u_{i,j+1} - u_{ij}}{h_t},$$
$$\frac{\partial u}{\partial x}(x_i, t_j) \approx \frac{u_{ij} - u_{i-1,j}}{h_x}.$$
Take these expressions and substitute them into the advection equation (and replace “≈” with “=”) to obtain

\[
\frac{u_{i,j+1} - u_{ij}}{h_t} + c \frac{u_{ij} - u_{i-1,j}}{h_x} = 0.
\]

Notice how this finite-difference equation mirrors the original differential equation. We can solve for \(u_{i,j+1}\) as

\[
u_{i,j+1} = \left(1 - c \frac{h_t}{h_x}\right) u_{ij} + c \frac{h_t}{h_x} u_{i-1,j}.
\]

Equation (5) is the basis of a reasonable numerical method for computing the solution to the original differential equation. Repeated application of (5) let’s us estimate the solution \(u(x,t)\) at any later time. For example, we know \(u(x,0) = f(x)\), where \(f\) is a given function, so that \(u_{i,0} = f(x_i)\) is known for all \(i\) from 0 to \(n\). We can estimate \(u(x_i,t_1) \approx u_{i,1}\) for \(1 \leq i \leq n\) by using (5) with \(j = 0\); all terms on the right side are known. We compute \(u_{0,1}\) from the boundary condition, as \(u_{0,1} = g(h_t)\). Once the numbers \(u_{i,1}\) are known we apply equation (5) with \(j = 1\) to compute \(u_{i,2}\), and then \(u_{i,3}, \text{etc.},\) while using the boundary condition to compute \(u_{0,j+1}\). Such a method for solving a PDE is called an explicit time-marching method—repeated application of (5) marches the solution forward in time.

Here’s a graphical way to look at what we’re doing.

This figure is called the stencil for the numerical method, and it pictorially illustrates what equation (5) is doing—estimating \(u(x, t_{j+1})\) in terms of \(u(x_i, t_j)\) and \(u(x_{i-1}, t_j)\).
Exercises:

1. Explain why the scheme in equation (5) is exact (for the advection equation) if we choose \( h_t \) and \( h_x \) so that \( c h_t h_x = 1 \).

2. Take \( c = 2, f(x) \equiv 0, \) and \( g(t) = \frac{1}{2} \sin(5t) \). Use \( h_x = 0.1 \) and \( h_t = 0.04 \) in the scheme (5) and solve out to \( t = 1 \) for \( 0 < x < 1 \). You may find the Maple notebook on the class web site useful, or you can write your own code—it’s easy!

   Plot the solution for several times from \( t = 0 \) to \( t = 1 \). Change \( h_t \) to 0.1 and repeated the process, solving out to time \( t = 1.0 \). What happens?

3. Suppose that instead of Dirichlet boundary conditions at \( x = 0 \) we have a Neumann condition
   \[ \frac{\partial u}{\partial x}(0, t) = g(t) \]
   How should this be implemented numerically?

Stability

Problem 2 illustrates that there’s something more to know about implementing equation (5); in certain circumstances the method may well numerically unstable.

One way to understand the problem is via linear algebra. Let \( u_j \) denote the column vector \([u_0, u_1, \ldots, u_n]^T\) (where \( T \) is transpose). Then the iteration in \( j \) embodied by equation (5) can be cast as

\[ u^{j+1} = Au^j + g_j \]

where

\[
A = \begin{bmatrix}
0 & 0 & 0 & 0 & \cdots & 0 \\
\alpha & 1 - \alpha & 0 & 0 & \cdots & 0 \\
0 & \alpha & 1 - \alpha & 0 & \cdots & 0 \\
& & & & & \\
& & & & & \\
0 & 0 & \cdots & 0 & \alpha & 1 - \alpha
\end{bmatrix}
\]

with \( \alpha = \frac{c h_t}{h_x} \) and \( g_j = [g(j h_t), 0, \ldots, 0]^T \).

For simplicity let’s assume we have zero boundary data, so \( g_j = 0 \) for all \( j \). Then we have simply

\[ u^j = A^j u^0. \]
We expect the process will be unstable (and in particular, errors will be
magnified without bound) if $A_j$ grows large in some sense.

One desirable feature is that $A$ should have all eigenvalues of absolute
value less than or equal to one. The reason for this is that the eigenvalues
of $A_j$ are of the form $\lambda^j$ where $\lambda$ is an eigenvalue of $A$. If $|\lambda| > 1$ then
the corresponding eigenvalue of $A_j$ is large, and so errors which are not
orthogonal to the eigenvector are multiplied.

It’s easy to check that the eigenvalues of $A$ are 0 (a simple eigenvalue
if $\alpha \neq 1$) and $1 - \alpha$ (multiplicity $n$). Thus we definitely want $|1 - \alpha| \leq 1,
leading to $0 \leq \alpha \leq 2$, or $0 \leq \frac{ch}{h_x} \leq 2$. This is certainly a condition we should
enforce on $h_t$ and $h_x$.

But actually, that’s not quite good enough. Suppose that $e^j$ is the error
in the $j$th stage of the computation. It would be preferable if $Ae^j$ was no
larger than $e^j$, and this isn’t quite the same as requiring the eigenvalues less
than one.

Let $\|v\|$ denote the Pythagorean length of the vector $v$. Then we want
$\|Ae^j\| \leq \|e^j\|$, or $\frac{\|Ae^j\|}{\|e^j\|} \leq 1$. Since we don’t know what $e^j$ is, we simple
require that
$$\frac{\|Av\|}{\|v\|} \leq 1$$
for all vectors $v$.

Now it’s a fact from linear algebra (easy to prove) that the maximum
value of $\frac{\|Av\|}{\|v\|}$ over all possible vectors $v$ is exactly the largest eigenvalue of
the matrix $AA^T$. In the present case it’s easy to compute that
$$AA^T = \begin{bmatrix}
0 & 0 & 0 & 0 & \cdots & 0 \\
0 & 1 - 2\alpha + 2\alpha^2 & \alpha - \alpha^2 & 0 & \cdots & 0 \\
0 & \alpha - \alpha^2 & 1 - 2\alpha + 2\alpha^2 & \alpha - \alpha^2 & \cdots & 0 \\
0 & 0 & \cdots & \alpha - \alpha^2 & 0 & \alpha - \alpha^2 \\
0 & 0 & \cdots & \alpha - \alpha^2 & 0 & \alpha - \alpha^2 \\
0 & 0 & \cdots & \alpha - \alpha^2 & 0 & \alpha - \alpha^2 \\
\end{bmatrix}
$$
The characteristic polynomial $p(\lambda)$ of this matrix is just $\lambda$ times the character-
istic polynomial of
$$A_1 = \begin{bmatrix}
1 - 2\alpha + 2\alpha^2 & \alpha - \alpha^2 & 0 & \cdots & 0 \\
\alpha - \alpha^2 & 1 - 2\alpha + 2\alpha^2 & \alpha - \alpha^2 & \cdots & 0 \\
\alpha - \alpha^2 & \cdots & \alpha - \alpha^2 & 0 & \alpha - \alpha^2 \\
0 & \cdots & 0 & \alpha - \alpha^2 & 1 - 2\alpha + 2\alpha^2 \\
\end{bmatrix}$$
(just think about expanding the determinant of $A - \lambda I$ along the top row).
The matrix $A_1$ is $n$ by $n$ and can be written as $A_1 = I - (\alpha - \alpha^2)M$ where
\[
M = \begin{bmatrix}
2 & -1 & 0 & \cdots & 0 \\
-1 & 2 & -1 & \cdots & 0 \\
-1 & \vdots & \ddots & \ddots & 0 \\
0 & \cdots & 0 & -1 & 2
\end{bmatrix}
\]
Thus the eigenvalues of $A_1$ are of the form $1 - (\alpha - \alpha^2)\lambda$ where $\lambda$ is an
eigenvalue of $M$. To see this suppose that $Mv = \lambda v$ ($v$ is an eigenvector of $M$). Then $A_1v = (I - (\alpha - \alpha^2)M)v = (1 - (\alpha - \alpha^2)\lambda)v$, i.e., $1 - (\alpha - \alpha^2)\lambda$
is an eigenvalue for $A_1$.

And fortunately, the eigenvalues of $M$ are known in closed form! The
matrix $M$ comes up a lot in finite difference methods. The eigenvalues of
the $n$ by $n$ matrix $M$ are given by $2(1 - \cos(k\pi/(n + 1)))$ for $k = 1$ to
$k = n$. Thus the eigenvalues of the matrix $A$ are 0 and the $n$ numbers
$1 - (\alpha - \alpha^2)(1 - \cos(k\pi/(n + 1)))$.

If we require that all of these eigenvalues have magnitude less than or
equal to one we have
\[-1 \leq 1 - (\alpha - \alpha^2)(1 - \cos(k\pi/(n + 1))) \leq 1.\]
This is easily rearranged to
\[0 \leq (\alpha - \alpha^2)(1 - \cos(\frac{k\pi}{n + 1})) \leq 2.\]
Now since $-1 \leq \cos \leq 1$, the above inequality will be satisfied if $0 \leq 2(\alpha - \alpha^2) \leq 2$, leading immediately to $0 \leq \alpha \leq 1$ or
\[\frac{h_t}{h_x} \leq 1\]
(since all quantities are positive, we needn’t worry about the left inequality).
Equation (6) is called the Courant-Friedrich-Lewy condition, or CFL condi-
tion for short. It’s necessary (and sufficient) for the numerical scheme (for
the advection equation with $c > 0$) to be stable.

The typical way the CFL condition is employed is as follows: We want to
solve the advection equation with a certain spatial resolution, out to some
time $t = T$. We thus choose $h_x$ first. We then choose $h_t$ in accordance to
equation (6), and then march out in time to $t = T$ in steps of size $h_t$. The finer the spatial resolution required (smaller $h_x$) the smaller the time steps must be, with correspondingly greater computational burden.

If other “advective” like problems (such as in problem 4 below) there might not be a clear-cut choice for $h_t$. If, for example, $c$ is variable, we might pick the smallest value for $h_t$ dictated by the CFL condition (6).

4. Repeat problem 2 where $c$ now depends on position, say $c(x) = 2 + \tanh(10(x - 0.5))$. Use $h_x = 0.1$. How small should you choose $h_t$?
   Show a graph of the solution at $t = 1$ with your choice for $h_t$.

5. Repeat problem 2 for the non-linear traffic flow equation

$$\frac{\partial u}{\partial t} + v_m \left(1 - \frac{2u}{u_m}\right) \frac{\partial u}{\partial x} = 0,$$

where $u(x,t)$ is the traffic density at time $t$ and position $x$, $v_m$ is the maximum traffic velocity, and $u_m$ is the maximum traffic density. For simplicity take $v_m = u_m = 1$. You’ll need to work out the appropriate time marching scheme analogous to equation (5). Use $g(t) \equiv 0.2$ and $f(x) = 0.2$ for $x < 1/2$, $f(x) = 0$ for $x \geq 1/2$. Take $h_x = 0.1$ and try to find an appropriate choice for $h_t$—does the CFL condition help estimate a good choice?

What is the physical interpretation of the initial conditions? What happens? Does it make sense?

6. Repeat the last problem but with $g(t) = 0.8$ and $f(x) = 0.8$ for $x < 1/2$, $f(x) = 0$ for $x \geq 1/2$. Can you make it work in any sensible way?

7. Repeat the last problem but with boundary condition $g(t) = 0$ and initial condition $f(x) = 0.5(1 + \tanh(10(x - 0.5)))$. What happens?