Scientific Computing: An Introductory Survey

Chapter 11 – Partial Differential Equations

Prof. Michael T. Heath

Department of Computer Science
University of Illinois at Urbana-Champaign

Outline

Partial Differential Equations

Partial differential equations (PDEs) involve partial derivatives with respect to more than one independent variable

Independent variables typically include one or more space dimensions and possibly time dimension as well

Partial Differential Equations, continued

For simplicity, we will deal only with single PDEs (as opposed to systems of several PDEs) with only two independent variables, either

- two space variables, denoted by x and y, or
- one space variable denoted by x and one time variable denoted by t

Partial derivatives with respect to independent variables are denoted by subscripts, for example

- $u_t = \partial u/\partial t$
- $u_{xy} = \partial^2 u/\partial x \partial y$

Example: Advection Equation

Advection equation

$$\boxed{u_t = -c\,u_x}$$

where c is nonzero constant

Unique solution is determined by initial condition

$$u(0,x) = u_0(x), \qquad -\infty < x < \infty$$

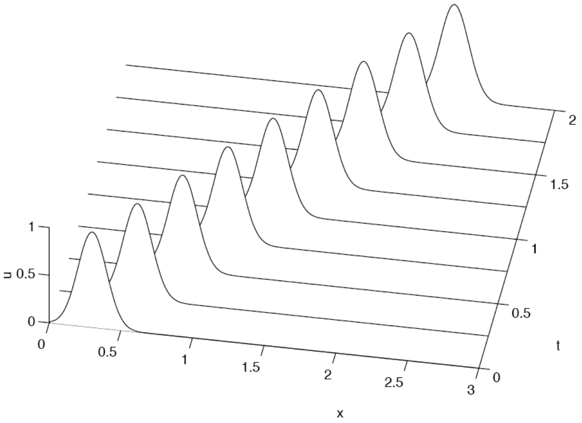
where u_0 is given function defined on $\mathbb{R}$

We seek solution $u(t,x)$ for $t \geq 0$ and all $x \in \mathbb{R}$

From chain rule, solution is given by $u(t,x) = u_0(x - c\,t)$

Solution is initial function u_0 shifted by $c\,t$ to right if $c > 0$, or to left if $c < 0$

Example, continued



Typical solution of advection equation, with initial function "advected" (shifted) over time < interactive example >

Classification of PDEs

Order of PDE is order of highest-order partial derivative appearing in equation

For example, advection equation is first order

Important second-order PDEs include

- *Heat equation* : $u_t = u_{xx}$
- *Wave equation* : $u_{tt} = u_{xx}$
- *Laplace equation* : $u_{xx} + u_{yy} = 0$

Classification of PDEs, continued

Second-order linear PDEs of general form

$$au_{xx} + bu_{xy} + cu_{yy} + du_x + eu_y + fu + g = 0$$

are classified by value of *discriminant* $b^2 - 4ac$

$b^2 - 4ac > 0$: *hyperbolic* (e.g., wave equation)

$b^2 - 4ac = 0$: *parabolic* (e.g., heat equation)

$b^2 - 4ac < 0$: *elliptic* (e.g., Laplace equation)

Classification of PDEs, continued

Classification of more general PDEs is not so clean and simple, but roughly speaking

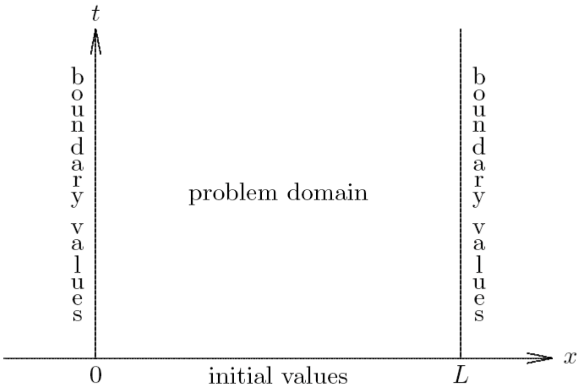
Hyperbolic PDEs describe time-dependent, conservative physical processes, such as convection, that *are not* evolving toward steady state

Parabolic PDEs describe time-dependent, dissipative physical processes, such as diffusion, that *are* evolving toward steady state

Elliptic PDEs describe processes that have already reached steady state, and hence are time-independent

Time-Dependent Problems

Time-dependent PDEs usually involve both initial values and boundary values



Semidiscrete Methods

One way to solve time-dependent PDE numerically is to discretize in space but leave time variable continuous

Result is system of ODEs that can then be solved by methods previously discussed

For example, consider heat equation

$$u_t = c\, u_{xx}, \qquad 0 \le x \le 1, \qquad t \ge 0$$

with initial condition

$$u(0, x) = f(x), \qquad 0 \le x \le 1$$

and boundary conditions

$$u(t, 0) = 0, \qquad u(t, 1) = 0, \qquad t \ge 0$$

Semidiscrete Finite Difference Method

Define spatial mesh points $x_i = i\Delta x$, $i = 0, \dots, n+1$, where $\Delta x = 1/(n+1)$

Replace derivative u_{xx} by finite difference approximation

$$u_{xx}(t, x_i) \approx \frac{u(t, x_{i+1}) - 2u(t, x_i) + u(t, x_{i-1})}{(\Delta x)^2}$$

Result is system of ODEs

$$y_i'(t) = \frac{c}{(\Delta x)^2} \left(y_{i+1}(t) - 2y_i(t) + y_{i-1}(t) \right), \quad i = 1, \dots, n$$

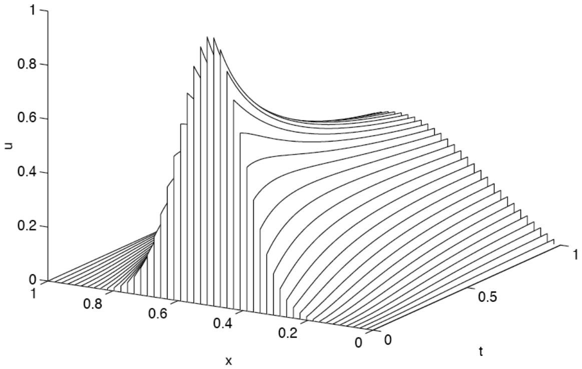
where $y_i(t) \approx u(t, x_i)$

From boundary conditions, $y_0(t)$ and $y_{n+1}(t)$ are identically zero, and from initial conditions, $y_i(0) = f(x_i)$, $i = 1, \dots, n$

We can therefore use ODE method to solve IVP for this system — this approach is called *Method of Lines*

Method of Lines

Method of lines uses ODE solver to compute cross-sections of solution surface over space-time plane along series of lines, each parallel to time axis and corresponding to discrete spatial mesh point



Fully Discrete Methods

Fully discrete methods for PDEs discretize in both time and space dimensions

In fully discrete finite difference method, we

- replace continuous domain of equation by discrete mesh of points
- replace derivatives in PDE by finite difference approximations
- seek numerical solution as table of approximate values at selected points in space and time

Fully Discrete Methods, continued

In two dimensions (one space and one time), resulting approximate solution values represent points on solution *surface* over problem domain in space-time plane

Accuracy of approximate solution depends on step sizes in both space and time

Replacement of all partial derivatives by finite differences results in system of algebraic equations for unknown solution at discrete set of sample points

Discrete system may be linear or nonlinear, depending on underlying PDE

Fully Discrete Methods, continued

With initial-value problem, solution is obtained by starting with initial values along boundary of problem domain and marching forward in time step by step, generating successive rows in solution table

Time-stepping procedure may be explicit or implicit, depending on whether formula for solution values at next time step involves only past information

We might expect to obtain arbitrarily good accuracy by taking sufficiently small step sizes in time and space

Time and space step sizes cannot always be chosen independently of each other, however

Example: Heat Equation

Consider heat equation

$$u_t = c\, u_{xx}, \qquad 0 \le x \le 1, \qquad t \ge 0$$

with initial and boundary conditions

$$u(0, x) = f(x), \qquad u(t, 0) = \alpha, \qquad u(t, 1) = \beta$$

Define spatial mesh points $x_i = i\Delta x$, $i = 0, 1, \ldots, n + 1$, where $\Delta x = 1/(n + 1)$, and temporal mesh points $t_k = k\Delta t$, for suitably chosen Δt

Let u_i^k denote approximate solution at (t_k, x_i)

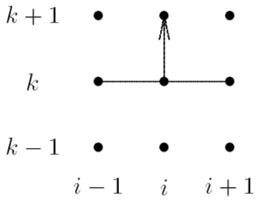
Heat Equation, continued

Replacing u_t by forward difference in time and u_{xx} by centered difference in space, we obtain

$$\frac{u_i^{k+1} - u_i^k}{\Delta t} = c\,\frac{u_{i+1}^k - 2u_i^k + u_{i-1}^k}{(\Delta x)^2}, \quad \text{or}$$

$$u_i^{k+1} = u_i^k + c\,\frac{\Delta t}{(\Delta x)^2}\left(u_{i+1}^k - 2u_i^k + u_{i-1}^k\right), \quad i = 1, \ldots, n$$

Stencil: pattern of mesh points involved at each level



Heat Equation, continued

Boundary conditions give us $u_0^k = \alpha$ and $u_{n+1}^k = \beta$ for all k, and initial conditions provide starting values $u_i^0 = f(x_i)$, $i = 1, \ldots, n$

So we can march numerical solution forward in time using this *explicit* difference scheme

Local truncation error is $\mathcal{O}(\Delta t) + \mathcal{O}((\Delta x)^2)$, so scheme is first-order accurate in time and second-order accurate in space

< interactive example >

Example: Wave Equation

Consider wave equation

$$u_{tt} = c\, u_{xx}, \qquad 0 \le x \le 1, \qquad t \ge 0$$

with initial and boundary conditions

$$u(0, x) = f(x), \qquad u_t(0, x) = g(x)$$

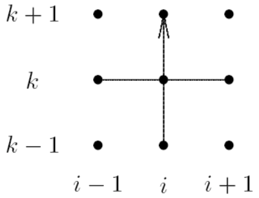
$$u(t, 0) = \alpha, \qquad u(t, 1) = \beta$$

Example: Wave Equation, continued

With mesh points defined as before, using centered difference formulas for both u_{tt} and u_{xx} gives finite difference scheme

$$\frac{u_i^{k+1} - 2u_i^k + u_i^{k-1}}{(\Delta t)^2} = c\,\frac{u_{i+1}^k - 2u_i^k + u_{i-1}^k}{(\Delta x)^2}, \quad \text{or}$$

$$u_i^{k+1} = 2u_i^k - u_i^{k-1} + c\left(\frac{\Delta t}{\Delta x}\right)^2 \left(u_{i+1}^k - 2u_i^k + u_{i-1}^k\right), \; i = 1, \ldots, n$$



Wave Equation, continued

Using data at two levels in time requires additional storage

We also need u_i^0 and u_i^1 to get started, which can be obtained from initial conditions

$$u_i^0 = f(x_i), \qquad u_i^1 = f(x_i) + (\Delta t)g(x_i)$$

where latter uses forward difference approximation to initial condition $u_t(0, x) = g(x)$

< interactive example >