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
- More dimensions complicate problem formulation: we can have pure initial value problem, pure boundary value problem, or mixture of both
- Equation and boundary data may be defined over irregular domain

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*

$$u_t = -c\,u_x$$

  where c is nonzero constant

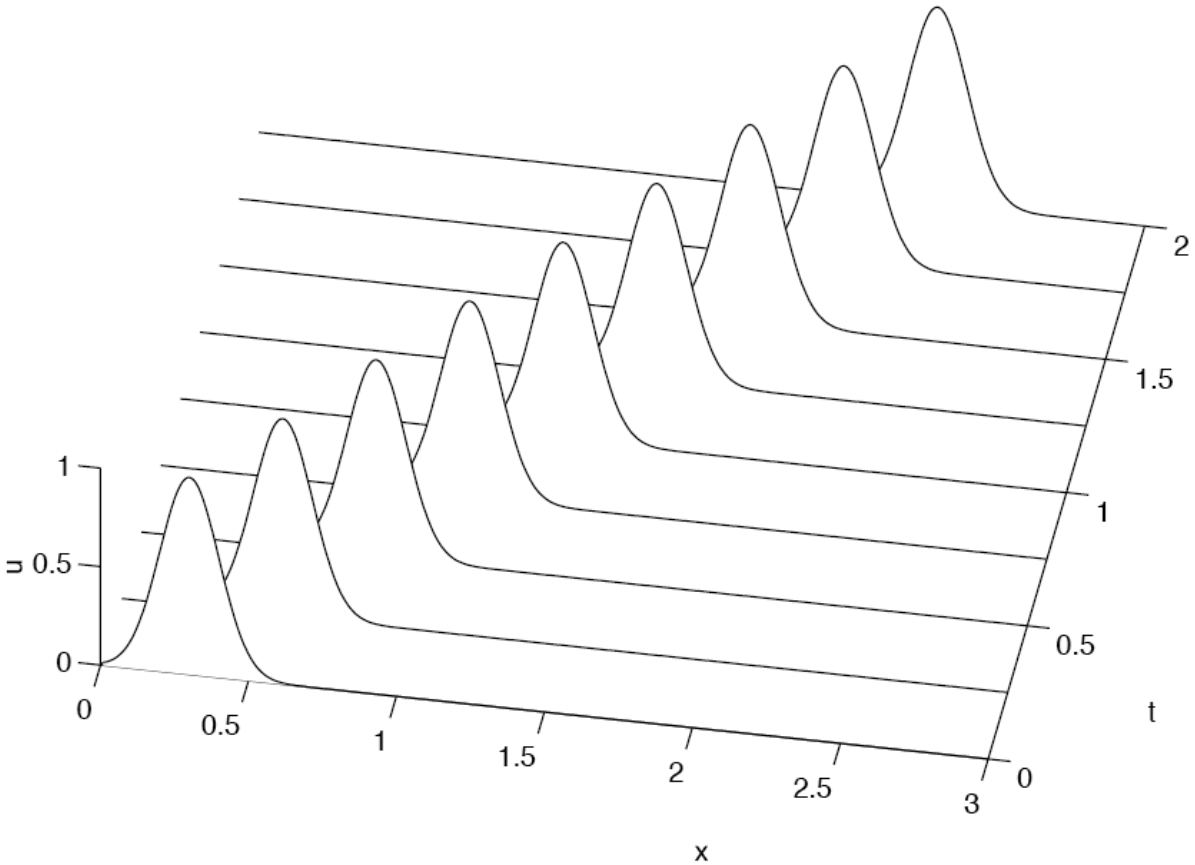
- Unique solution is determined by initial condition

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

  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 $c < 0$

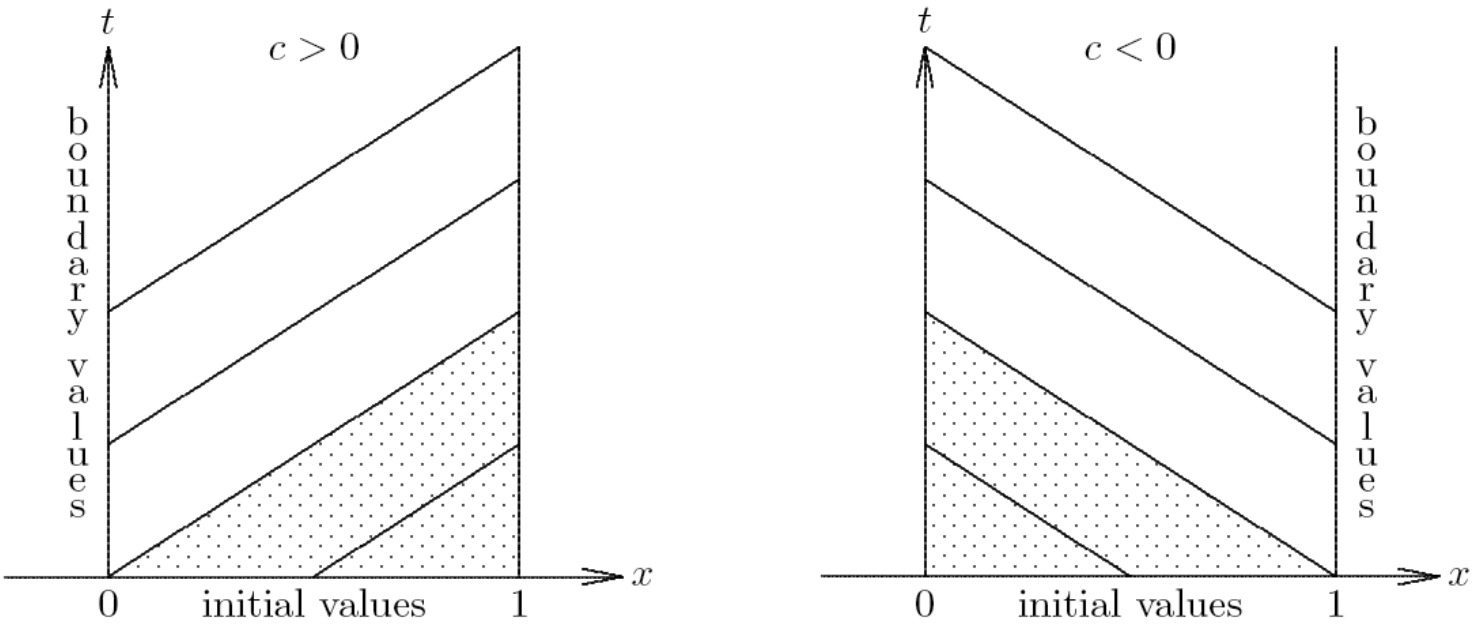
Example, continued



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

Characteristics

- *Characteristics* for PDE are level curves of solution
- For advection equation $u_t = -c\,u_x$, characteristics are straight lines of slope c



- Characteristics determine where boundary conditions can or must be imposed for problem to be well-posed

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)

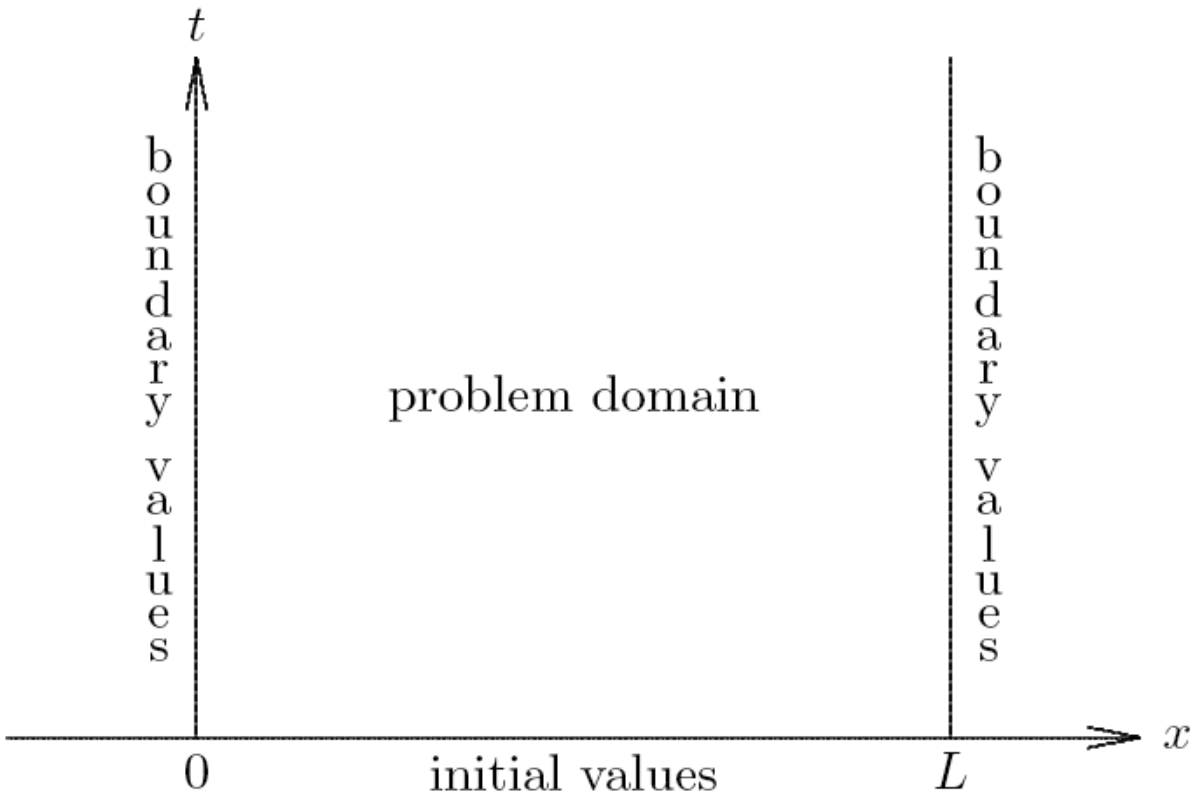
Classification of PDEs, continued

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 $x_i = i\Delta x$, $i = 0, \ldots, n+1$, 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

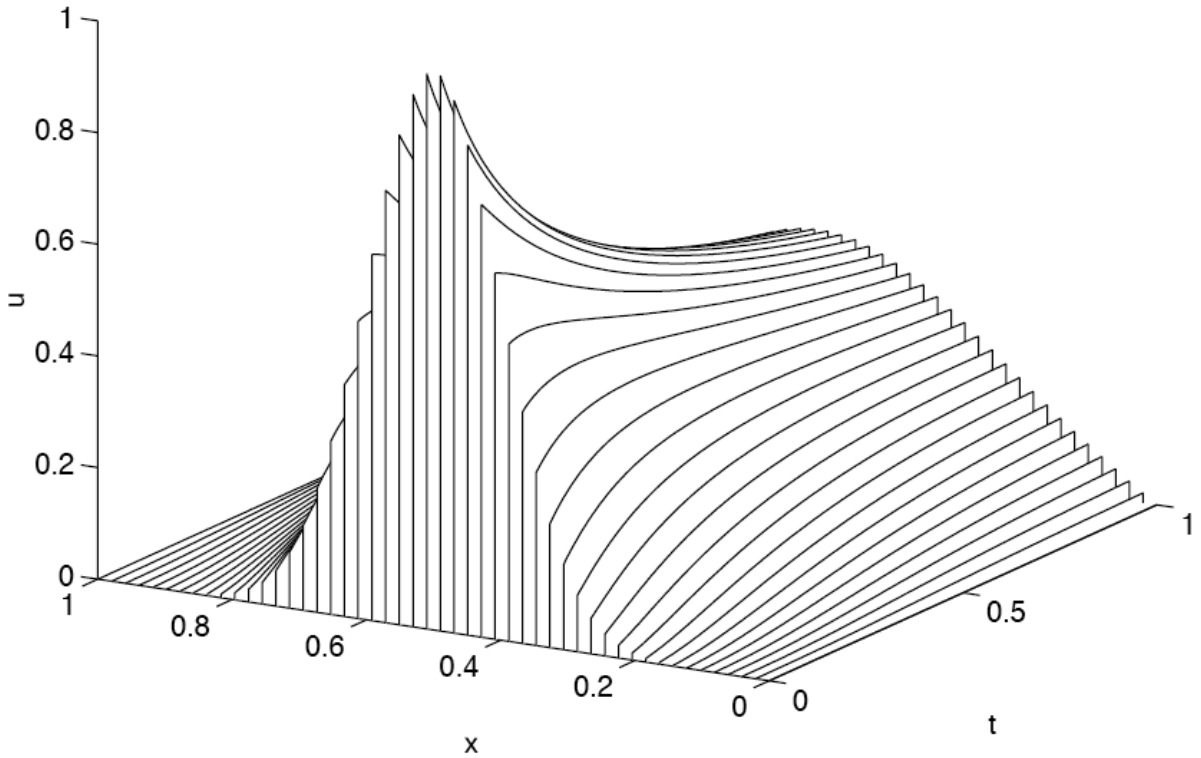
$$y_i'(t) = \frac{c}{(\Delta x)^2}\left(y_{i+1}(t) - 2y_i(t) + y_{i-1}(t)\right), \quad i = 1, \ldots, n$$

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, $y_i(0) = f(x_i)$, $i = 1, \ldots, n$
- 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



Stiffness

- Semidiscrete system of ODEs just derived can be written in matrix form

$$\boldsymbol{y}' = \frac{c}{(\Delta x)^2} \begin{bmatrix} -2 & 1 & 0 & \cdots & 0 \\ 1 & -2 & 1 & \cdots & 0 \\ 0 & 1 & -2 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & -2 \end{bmatrix} \boldsymbol{y} = \boldsymbol{A}\boldsymbol{y}$$

- Jacobian matrix $\boldsymbol{A}$ of this system has eigenvalues between $-4c/(\Delta x)^2$ and 0, which makes ODE very *stiff* as spatial mesh size Δx becomes small
- This stiffness, which is typical of ODEs derived from PDEs, must be taken into account in choosing ODE method for solving semidiscrete system

Semidiscrete Collocation Method

- Spatial discretization to convert PDE into system of ODEs can also be done by spectral or finite element approach
- Approximate solution is expressed as linear combination of basis functions, but with time dependent coefficients
- Thus, we seek solution of form

$$u(t,x) \approx v(t,x,\boldsymbol{\alpha}(t)) = \sum_{j=1}^{n} \alpha_j(t)\phi_j(x)$$

  where $\phi_j(x)$ are suitably chosen basis functions

- If we use collocation, then we substitute this approximation into PDE and require that equation be satisfied exactly at discrete set of points x_i

Semidiscrete Collocation, continued

- For heat equation, this yields system of ODEs

$$\sum_{j=1}^{n} \alpha_j'(t)\phi_j(x_i) = c\sum_{j=1}^{n} \alpha_j(t)\phi_j''(x_i)$$

  whose solution is set of coefficient functions $\alpha_i(t)$ that determine approximate solution to PDE

- Implicit form of this system is not explicit form required by standard ODE methods, so we define $n \times n$ matrices $\boldsymbol{M}$ and $\boldsymbol{N}$ by

$$m_{ij} = \phi_j(x_i), \qquad n_{ij} = \phi_j''(x_i)$$

Semidiscrete Collocation, continued

- Assuming M is nonsingular, we then obtain system of ODEs

$$\boldsymbol{\alpha}'(t) = c\,\boldsymbol{M}^{-1}\boldsymbol{N}\boldsymbol{\alpha}(t)$$

  which is in form suitable for solution with standard ODE software

  - As usual, M need not be inverted explicitly, but merely used to solve linear systems

- Initial condition for ODE can be obtained by requiring solution to satisfy given initial condition for PDE at mesh points x_i

- Matrices involved in this method will be sparse if basis functions are "local," such as B-splines

Semidiscrete Collocation, continued

- Unlike finite difference method, spectral or finite element method does not produce approximate values of solution u directly, but rather it generates representation of approximate solution as linear combination of basis functions
- Basis functions depend only on spatial variable, but coefficients of linear combination (given by solution to system of ODEs) are time dependent
- Thus, for any given time t, corresponding linear combination of basis functions generates cross section of solution *surface* parallel to spatial axis
- As with finite difference methods, systems of ODEs arising from semidiscretization of PDE by spectral or finite element methods tend to be stiff

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)

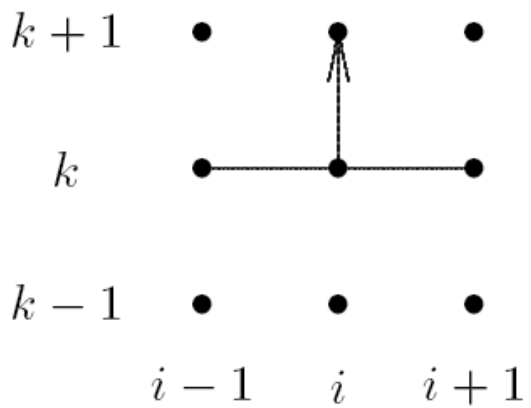
Heat Equation, continued

- 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

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

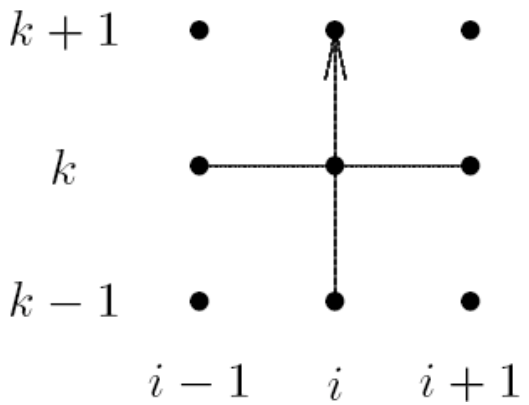
$$u(t,0) = \alpha, \qquad u(t,1) = \beta$$

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 >

Stability

- Unlike Method of Lines, where time step is chosen automatically by ODE solver, user must choose time step Δt in fully discrete method, taking into account both accuracy and stability requirements
- For example, fully discrete scheme for heat equation is simply Euler's method applied to semidiscrete system of ODEs for heat equation given previously
- We saw that Jacobian matrix of semidiscrete system has eigenvalues between $-4c/(\Delta x)^2$ and 0, so stability region for Euler's method requires time step to satisfy

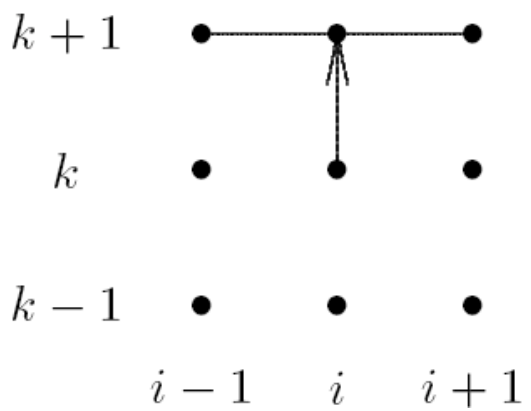
$$\Delta t \leq \frac{(\Delta x)^2}{2\,c}$$

- Severe restriction on time step can make explicit methods relatively inefficient < interactive example >

Implicit Finite Difference Methods

- For ODEs we saw that implicit methods are stable for much greater range of step sizes, and same is true of implicit methods for PDEs
- Applying *backward Euler method* to semidiscrete system for heat equation gives *implicit* finite difference scheme

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



Implicit Finite Difference Methods, continued

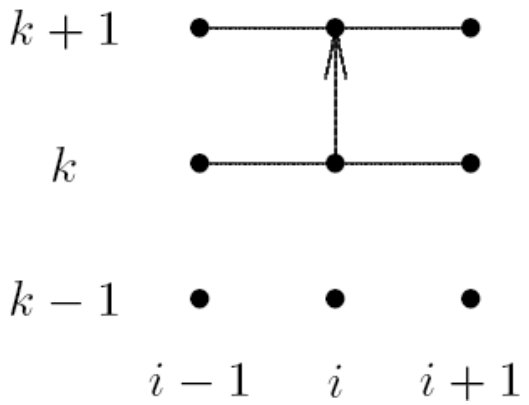
- This scheme inherits unconditional stability of backward Euler method, which means there is no stability restriction on relative sizes of Δt and Δx
- But first-order accuracy in time still severely limits time step

< interactive example >

Crank-Nicolson Method

- Applying *trapezoid method* to semidiscrete system of ODEs for heat equation yields implicit *Crank-Nicolson* method

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



- This method is unconditionally stable and second-order accurate in time < interactive example >

Implicit Finite Difference Methods, continued

- Much greater stability of implicit finite difference methods enables them to take much larger time steps than explicit methods, but they require more work per step, since system of equations must be solved at each step
- For both backward Euler and Crank-Nicolson methods for heat equation in one space dimension, this linear system is tridiagonal, and thus both work and storage required are modest
- In higher dimensions, matrix of linear system does not have such simple form, but it is still very sparse, with nonzeros in regular pattern