Sample BEAMER Presentation

Matt Kinsinger[1]

ARIZONA STATE UNIVERSITY
SCHOOL OF MATHEMATICAL AND STATISTICAL SCIENCES

May 9, 2017

[1] mkinsing@asu.edu

Introduction

- Heat as a function of 1-dimensional space and time: $u(x,t)$
- Notation for u evaluated at discretized points in space and time

$$u(x_i, t_n) = u_i^n$$

- The heat equation, a second order partial differential equation

$$\frac{\partial u}{\partial t} = k\frac{\partial^2 u}{\partial x^2}$$

- Three approximation strategies
  - Euler's method
  - Improved Euler's method
  - Implicit Time Stepping

Euler's Method

$$u_i^{n+1} \approx u_i^n + \frac{k\Delta t}{\Delta x^2}\left[u_{i+1}^n - 2u_i^n + u_{i-1}^n\right]$$

- This can be derived using Taylor series expansions in each variable

Improved Euler's method

- Approximate the slopes at the endpoints of each time interval
  - $\widetilde{K}_{i+1}$ is found using an approximation for u_i^{n+1}
- Use the average of these slopes to linearly approximate the solution over the interval

$$u_i^{n+1} = u_i^n + \Delta t \left[\frac{K_i + \widetilde{K}_{i+1}}{2} \right]$$

- $K_i \approx$ slope at u_i^n
- $\widetilde{K}_{i+1} \approx$ slope at u_i^{n+1}

- Pieces to the approximation
  - $K_i = \frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{k}{\Delta x^2}\left[u_{i-1}^n - 2u_i^n + u_{i+1}^n\right]$
  - $\tilde{u}_i^{n+1} = u_i^n + \frac{k\Delta t}{\Delta x^2}\left[u_{i-1}^n - 2u_i^n + u_{i+1}^n\right]$
  - $\widetilde{K}_{i+1} = \frac{u_i^{n+2} - u_i^{n+1}}{\Delta t} = \frac{k}{\Delta x^2}\left[\tilde{u}_{i-1}^{n+1} - 2\tilde{u}_i^{n+1} + \tilde{u}_{i+1}^{n+1}\right]$

Implicit Time Stepping

- Benefits
  - Increased stabilty of approximation
  - Able to handle larger diffusivity constants
- Drawbacks
  - Need to solve a linear system $Ax = b$

- Setting up the approximation
  - Use the Taylor's series expansion, but the RHS of our final approximation expression is in terms of $u_x^{t_{n+1}}$ rather than $u_x^{t_n}$.

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{k}{\Delta x^2}\left[u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}\right]$$

$$u_i^n = u_i^{n+1} - \frac{k\Delta t}{\Delta x^2}\left[u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}\right]$$

$$u_i^n = u_i^{n+1}\left[1 + 2r\right] - r\left[u_{i-1}^{n+1} + u_{i+1}^{n+1}\right] \qquad , r = \frac{k\Delta t}{\Delta x^2}$$

  - We are using the value at the *next* time step

- Fix n (time) and let i (space) float over all of the interior points of our rod
  - Boundary conditions $u(x_1, t)$ and $u(x_N, t)$ are known for all t
  - Interior points

$$
\begin{aligned}
u_2^n &= u_2^{n+1}\left[1+2r\right] - r\left[u_1^{n+1} + u_3^{n+1}\right] \\
u_3^n &= u_3^{n+1}\left[1+2r\right] - r\left[u_2^{n+1} + u_4^{n+1}\right] \\
u_4^n &= u_4^{n+1}\left[1+2r\right] - r\left[u_3^{n+1} + u_5^{n+1}\right] \\
&\,. \\
&\,. \\
u_{N-2}^n &= u_{N-2}^{n+1}\left[1+2r\right] - r\left[u_{N-3}^{n+1} + u_5^{N-1}\right] \\
u_{N-1}^n &= u_{N-1}^{n+1}\left[1+2r\right] - r\left[u_{N-2}^{n+1} + u_N^{n+1}\right]
\end{aligned}
$$

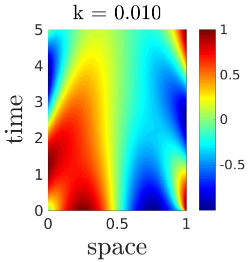
We move our known terms u_1^{n+1} and u_N^{n+1} to the RHS.

Recognizing the pattern we build the linear system:

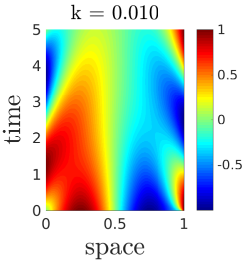
$$\begin{pmatrix} 1+2r & -r & 0 & 0 & . & . & 0 \\ -r & 1+2r & -r & 0 & 0 & . & 0 \\ 0 & -r & 1+2r & -r & 0 & . & 0 \\ . & & . & . & . & & . \\ . & & . & . & . & & . \\ . & & . & . & . & & . \\ 0 & 0 & . & . & -r & 1+2r & -r \\ 0 & 0 & . & . & 0 & -r & 1+2r \end{pmatrix} \begin{pmatrix} u_2^{n+1} \\ u_3^{n+1} \\ u_4^{n+1} \\ . \\ . \\ . \\ u_{N-2}^{n+1} \\ u_{N-1}^{n+1} \end{pmatrix} = \begin{pmatrix} u_2^n + ru_1^{n+1} \\ u_3^n \\ u_4^n \\ . \\ . \\ . \\ u_{N-2}^n \\ u_{N-1}^n + ru_N^{n+1} \end{pmatrix}$$

Comparing plots

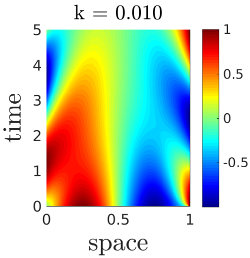
- We need the initial temperature profile of the rod and the temperatures for all points in time of the endpoints of the rod
  - We will give these values in our MatLab code
- Example 1
  - Length of rod, L: 1 meter
  - Number of points: 50
  - Total time, t_f: 5 seconds
  - Number of time points: 2000
  - Diffusivity constant: k
  - $r = k\frac{\Delta t}{\Delta x^2}$ $r < 0.50$ required
  - $u(1,:) = \sin\left(\frac{2\pi}{t_f}\ t\right)$ Left endpoint boundary conditions
  - $u(N,:) = \cos\left(\frac{2\pi}{t_f}\ t\right)$ Right endpoint boundary conditions
  - $u(:,1) = \sin\left(\frac{2\pi}{L}\right)\ x$ Initial conditions



Euler's



Improved Euler's



Implicit Time Stepping

Exploring Error

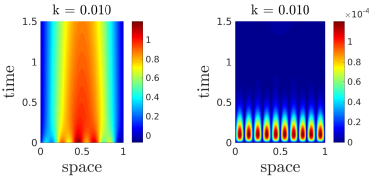
- Simplify IC and Boundary conditions so the we can solve for an explicit solution
  - IC
    $$u(x,0) = \sin(\pi x) + 0.2\sin(10\pi x)$$
  - Boundary conditions
    $$u(0,t) = u(1,t) = 0$$
  - Guess a solution to $u_t = ku_{xx}$

$$u(t,x) = e^{-\pi^2 kt}\sin(\pi x) + 0.2e^{-(10\pi)^2 kt}\sin(10\pi x)$$

$$\begin{aligned}
u_t &= \frac{\partial}{\partial t}\Big[u(x,t)\Big] \\
&= \frac{\partial}{\partial t}\Big[e^{-\pi^2 kt}\sin(\pi x) + 0.2e^{-(10\pi)^2 kt}\sin(10\pi x)\Big] \\
&= -\pi^2 k e^{-\pi^2 kt}\sin(\pi x) + 0.2\left[-(10\pi)^2 k\right] e^{-(10\pi)^2 kt}\sin(10\pi x) \\
&= k\Big[-\pi^2 e^{-\pi^2 kt}\sin(\pi x) + 0.2\left[-(10\pi)^2 k\right] e^{-(10\pi)^2 t}\sin(10\pi x)\Big] \\
&= k\frac{\partial^2 u}{\partial x^2}\Big[e^{-\pi^2 kt}\sin(\pi x) + 0.2e^{-(10\pi)^2 kt}\sin(10\pi x)\Big] \\
&= k\frac{\partial^2 u}{\partial x^2}\Big[u(x,t)\Big] \\
&= ku_{xx}
\end{aligned}$$

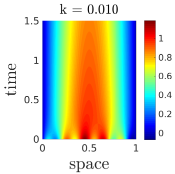
It is a solution! We can use this to measure the error in our approximations!

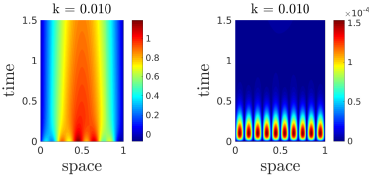
Plots with $M = 2000, N = 200, k = 0.01 \rightarrow r = 0.2972$



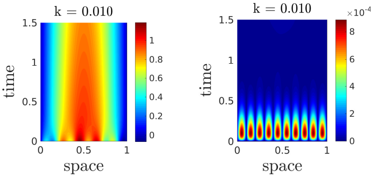
Exact solution

Euler's





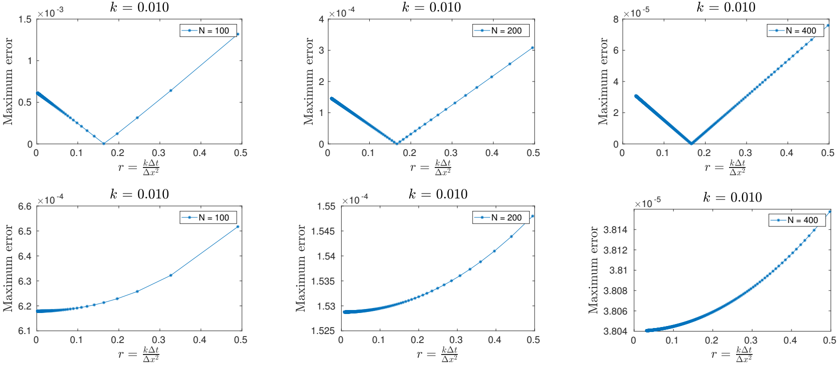
Improved Euler's



Implicit

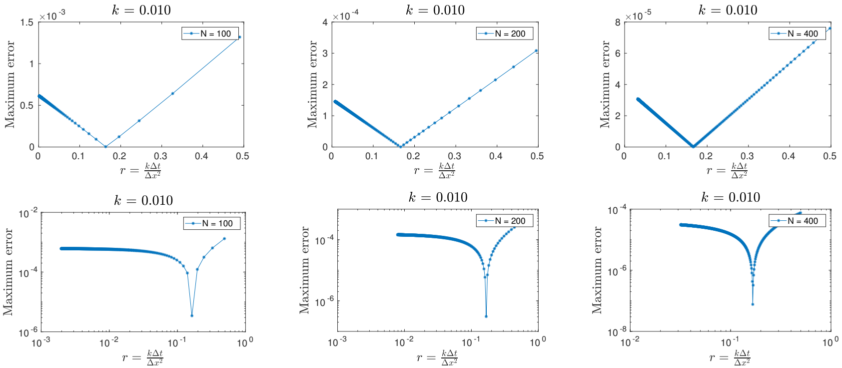
At which point in time should we sample the error across the rod?

- t large and $t \approx 0$ the error is nearly zero
- Error appears non-trivial at $t \approx 0.10$ seconds
- Experiment:
    - A given number of sample points in space
    - Range through different numbers of sample points in time
        - $r = \frac{k\Delta t}{\Delta x^2} \downarrow \quad \text{as } \Delta t \downarrow$



Implicit time stepping: The code ran forever

Log plot to get a closer look at Euler's method approximation



At some point just above $r = 0.10$ we see the error becoming very small!

Intersting...Why?

Remarks

Acknowledgments