AMSC/CMSC 661 Scientific Computing II

Spring 2010

Solution of Parabolic Partial Differential Equations

Part 2: Computation

Dianne P. O'Leary

These notes are based on the 2003 textbook of Stig Larsson and Vidar Thomée.

Solving parabolic PDEs numerically

As you would expect, there are two methods:

- Finite differences
- Finite elements

It takes just a bit of thought to realize that it is not very natural to use finite elements in the time dimension.

- So popular algorithms use finite differences in time (for the t variable) and either finite differences or finite elements in space (for the x variables).
- In the space dimensions, we just have an elliptic operator to discretize, so we know all about that:
  - If $x \in \mathcal{R}^1$ or if $\Omega \subset \mathcal{R}^d$ is very simple, then finite differences can be used.
  - Otherwise, finite elements will probably be a better choice.

A brief discussion of finite difference methods

Reference: small pieces of Chapter 9

Consider the simplest possible parabolic PDE:

$$u_t = u_{xx}$$

with $x \in \mathcal{R}$ and initial conditions $u(x,0) = v(x)$ given.

(pp 129-131) Consider the simplest discretization: $u_j^n \approx u(jh, nk)$, using Forward Euler for time and a central difference for space:

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{h^2}.$$

We march forward with this: given values at n, we solve for values at $n+1$:

$$u_j^{n+1} = ru_{j+1}^n + (1-2r)u_j^n + ru_{j-1}^n$$

where $r = k/h^2$ is the mesh ratio.

We have a stability estimate for the continuous problem:

$$\|u(\cdot,t)\|_{\mathcal{C}} \leq \|v\|_{\mathcal{C}}.$$

What about our discretization?

$$u_j^{n+1} = ru_{j+1}^n + (1-2r)u_j^n + ru_{j-1}^n$$

If $1-2r > 0$, then

$$|u_j^{n+1}| \leq r|u_{j+1}^n| + (1-2r)|u_j^n| + r|u_{j-1}^n| \leq (r+(1-2r)+r)\max_j |u_j^n| = \max_j |u_j^n|,$$

so we have the same stability estimate.

The restriction that $1-2r > 0$, so $r = k/h^2 < 1/2$, is a special case of von Neumann's stability condition.

What about Backward Euler? (p140)

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{h^2}.$$

so

$$-ru_{j+1}^{n+1} + (1+2r)u_j^{n+1} - ru_{j-1}^{n+1} = u_j^n$$

where $r = k/h^2$ is the mesh ratio.

This gives us a linear system of equations to solve:

$$\mathbf{B}\mathbf{u}^{n+1} = \mathbf{u}^n.$$

Is the method stable? We need to look at the eigenvalues of **B**.

A very useful result about eigenvalues of matrices

Gershgorin's Circle Theorem (not in your book)
Let **A** be an $n \times n$ matrix. For $i = 1, \ldots, n$, let C_i be a circle in the complex plane, centered at a_{ii} with radius $\sum_{i \neq j} |a_{ij}|$. Then all of the eigenvalues of **A** lie in the union of the circles C_i.

Proof: See (for example) Golub and Van Loan, p320.

Example: For our matrix **B**, the main diagonal elements are all equal to $1+2r$, so this is the center of all of the circles. The radii are at most $2r$. So all of the eigenvalues of **B** lie either at $\lambda = 1$ or outside the unit circle, and therefore all eigenvalues of $\mathbf{B}^{-1}$ lie either at $\lambda = 1$ or within.

We can verify that $\lambda = 1$ is not an eigenvalue of $\mathbf{B}$. (For example, Cholesky runs to completion on $\mathbf{B} - \mathbf{I}$ and produces two nonsingular factors.) Therefore, all eigenvalues of $\mathbf{B}^{-1}$ are inside the unit circle.

Therefore, since $\mathbf{u}^{n+1} = \mathbf{B}^{-1}\mathbf{u}^n = \mathbf{B}^{-2}\mathbf{u}^{n-1} = \ldots = \mathbf{B}^{-(n+1)}\mathbf{u}^0$,

$$\mathbf{u}^{n+1} \to \mathbf{0},$$

and the method is unconditionally stable without a von Neumann condition on the relation between h and k.

(But for accuracy, we need to take k much smaller than h, since the error is proportional to $O(k) + O(h^2)$.)

What about Crank-Nicolson? (pp142-143)

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{1}{2}\left(\frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{h^2} + \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{h^2}\right).$$

so

$$-\frac{r}{2}u_{j+1}^{n+1} + (1+r)u_j^{n+1} - \frac{r}{2}u_{j-1}^{n+1} = \frac{r}{2}u_{j+1}^n + (1-r)u_j^n + \frac{r}{2}u_{j-1}^n$$

which we can write in matrix form as

$$\mathbf{B}\mathbf{u}^{n+1} = \mathbf{A}\mathbf{u}^n.$$

where $\mathbf{B}$ has $1 + r$ on its main diagonal, $-r/2$ above and below, and $\mathbf{A}$ has $1 - r$ on its main diagonal and $r/2$ above and below.

Grinding the linear algebra crank (details not important for this course), we could show that the eigenvalues of $\mathbf{B}^{-1}\mathbf{A}$ are all inside the unit circle and Crank-Nicolson is unconditionally stable.

The error is proportional to $O(k^2) + O(h^2)$.

What you need to know in order to use finite differences for parabolic PDEs:

- There is a von Neumann stability restriction on the mesh sizes when explicit methods such as forward Euler are used.
- Implicit methods like backward Euler and Crank-Nicholson avoid this, but, as we know, they are more difficult to compute with. Further, we still need to keep both h and k small enough so that the error term is sufficiently small for our purposes.
- For the mixed Initial-Boundary value problem (for example, when $x \in [0, 1]$), then we use the same discretization (Euler, for example)

  $$u_j^{n+1} = ru_{j+1}^n + (1-2r)u_j^n + ru_{j-1}^n$$

  but plug in boundary values when the subscript is 0 or $1/h$.

- Read Chapter 9 if you ever need to use these methods.

Finite element methods

Reference: Chapter 10, especially pp. 149-150

Consider the parabolic Initial-Boundary Value Problem

$$\begin{array}{rcll} u_t(x,t) - \Delta u(x,t) & = & f(x,t) & (x,t) \in \Omega \times R_+ \\ u & = & 0 & (x,t) \in \Gamma \times R_+ \\ u(x,0) & = & v(x) & x \in \Omega \end{array}$$

where $\Omega \subset \mathcal{R}^2$ and $\Delta u = u_{xx} + u_{yy}$.

We discretize in two steps:

- Use the Galerkin finite element method on Δu.
- Use finite differences on u_t.

Step 1a: Galerkin

Let t be fixed and choose $\phi \in H_0^1$. Then we take

$$u_t(x,t) - \Delta u(x,t) = f(x,t)$$

and integrate its product with ϕ to get

$$(u_t, \phi) - (\Delta u, \phi) = (f, \phi).$$

Use integration by parts to get

$$(u_t, \phi) + a(u, \phi) = (f, \phi).$$

Step 1b: Finite elements

Galerkin equation:

$$(u_t, \phi) + a(u, \phi) = (f, \phi).$$

Let

$$u(x,t) \approx u_h(x,t) = \sum_{j=1}^{M} \alpha_j(t)\phi_j(x)$$

where the functions ϕ_j form a basis for the finite element space (piecewise linears, for example). Then substituting for u, our Galerkin equation becomes

$$\sum_{j=1}^{M} \alpha_j'(t)(\phi_j, \phi) + \sum_{j=1}^{M} \alpha_j(t) a(\phi_j, \phi) = (f, \phi).$$

Repeating,

$$\sum_{j=1}^{M} \alpha_j'(t)(\phi_j, \phi) + \sum_{j=1}^{M} \alpha_j(t) a(\phi_j, \phi) = (f, \phi).$$

We get a system of equations by letting $\phi = \phi_k$, for $k = 1, \ldots, M$:

$$\mathbf{B}\boldsymbol{\alpha}'(t) + \mathbf{A}\boldsymbol{\alpha}(t) = \mathbf{g}(t)$$

where

- $\boldsymbol{\alpha}(t)$ is a vector with M components $\alpha_j(t)$.
- $a(\phi_k, \phi_j) = \int_\Omega \bigtriangledown\phi_k \cdot \bigtriangledown\phi_j$, and this is element (k, j) of the stiffness matrix **A**.
- $\mathbf{g}(t)$ is a vector with kth component $(f, \phi_k) = \int_\Omega f(x, t)\phi_k(x)dx$.
- **B** is the mass matrix with element (k, j) equal to (ϕ_k, ϕ_j).

$$\mathbf{B}\boldsymbol{\alpha}'(t) + \mathbf{A}\boldsymbol{\alpha}(t) = \mathbf{g}(t)$$

This is a system of ordinary differential equations with initial values

$$\alpha_j(0) = v(x_j).$$

Recall that the stiffness matrix is symmetric and positive definite.
So is the mass matrix, since it is clearly symmetric and for any $\mathbf{z}$,

$$\mathbf{z}^T\mathbf{B}\mathbf{z} = \sum_{k=1}^{M}\sum_{j=1}^{M}(\phi_k, \phi_j)z_k z_j = \|\sum_{k=1}^{M} z_k\phi_k\|^2 \geq 0,$$

and zero only if $\mathbf{z} = \mathbf{0}$. (The first equality comes from the definition of matrix-vector product, and the value is equal to the norm of $\sum_{k=1}^{M} z_k\phi_k$. Since the ϕ_k are linearly independent, this is zero only if all $z_k = 0$.)

Therefore, if our integrator requires it, we can express

$$\mathbf{B}\boldsymbol{\alpha}'(t) + \mathbf{A}\boldsymbol{\alpha}(t) = \mathbf{g}(t)$$

as

$$\boldsymbol{\alpha}'(t) = -\mathbf{B}^{-1}\mathbf{A}\boldsymbol{\alpha}(t) + \mathbf{B}^{-1}\mathbf{g}(t).$$

This helps to establish existence and uniqueness of the solution.

In the rest of Section 10.1, the authors use what we already know about elliptic finite element methods to derive estimates for the parabolic problem.

Step 2: Discretize in time

We have a system of M ordinary differential equations

$$\mathbf{B}\boldsymbol{\alpha}'(t) = -\mathbf{A}\boldsymbol{\alpha}(t) + \mathbf{g}(t).$$

with $\boldsymbol{\alpha}(0)$ given.

In some sense we are finished; plug the ODEs into your favorite solver (e.g., `ode23s`) and let it go.

But because M is typically quite large (perhaps 10^6 if $d = 2$), it is worthwhile to have options to solve the ODEs by less expensive (although less sophisticated) methods.

So pull out

- Euler
- Backward Euler
- Crank-Nicholson

and use them on the system.

A few notes about using Euler

$$\mathbf{B}\boldsymbol{\alpha}'(t) = -\mathbf{A}\boldsymbol{\alpha}(t) + \mathbf{g}(t).$$

becomes

$$\mathbf{B}\,\frac{\boldsymbol{\alpha}(t+k) - \boldsymbol{\alpha}(t)}{k} = -\mathbf{A}\boldsymbol{\alpha}(t) + \mathbf{g}(t).$$

This isn't really explicit unless the mass matrix $\mathbf{B}$ is diagonal. So sometimes we lump the mass, replacing $\mathbf{B}$ by a diagonal matrix $\bar{\mathbf{B}}$ with the same row-sums.

This sounds like an awful idea, until we stop to realize:

- We have already made an approximation in computing the elements of $\mathbf{B}$ by numerical integration, and we are free to choose the integration formula in a convenient way.
- So instead of the barycentric integration rule we discussed before, let's use a nodal rule, that only evaluates the functions at the nodes of the triangles. Any such rule gives us a diagonal matrix $\mathbf{B}$!

A few notes about Backward Euler and Crank-Nicolson

Backward Euler, used in place of Forward, gives a method that is stable for a wider choice of h, k values, but not unconditionally stable, because of the behavior of our approximation to Δu.

Crank-Nicolson is unconditionally stable for this problem. See the definition of the method at the bottom of p. 158 and read Theorem 10.6.