Finite difference methods for initial boundary value problems (IBVPs)

Over the last several weeks, we have numerically solved initial value problems and boundary value problems. The subject of the next several lectures will be to combine our tools for IVPs with our tools for solving BVPs. That is, we will develop finite difference methods for initial boundary value problems (IBVPs). This class of differential equations depends on *both* space and time. To enable us to use our previously developed tools, our approach will be to first discretize the equations in space to arrive at an IVP. Doing this allows us to then directly apply our techniques for solving IVPs. We will focus on developing our method for the heat equation—one of the most important canonical IBVPs.

But these same techniques can be extended beyond the heat equation and into a broad range of IBVPs!

This week, we will focus on developing a finite difference method for the heat equation. That is, we will solve the heat equation using local interpolation. Next week, we will switch gears and consider solving the heat equation using a finite element method. That is, we will consider applying local spectral methods to the solve the heat equation.

1 *The heat equation*

The heat equation may be written for a generic n-dimensional problem as

$$\frac{\partial u}{\partial t} = \kappa \nabla^2 u + g(\boldsymbol{x}, t), \quad 0 < t \leq T, \ \boldsymbol{x} \in \Omega \tag{1}$$

where ∇^2 is the Laplace operator in n spatial dimensions and g is a prescribed source term. We will restrict our attention to $n = 1$ for now, so that the heat equation reduces to

For clarity, we explicitly state that we are solving the heat equation over the time window $0 < t \leq T$ and within the domain Ω. Note that as this is an initial boundary value problem, we require *both* an initial condition and appropriate set of boundary conditions to advance the solution in time.

$$\frac{\partial u}{\partial t} = \kappa \frac{\partial^2 u}{\partial x^2} + g(x, t), \quad 0 \leq t \leq T, \ a \leq x \leq b \tag{2}$$

Note that we are defining $\Omega = [a, b]$ for this 1D case.

The initial condition for this problem is relatively straightforward: we apply a constraint on what the solution u is at time $t = 0$. There is more flexibility in the boundary conditions we consider. Let us assume that we have Dirichlet boundary conditions (that is, we prescribe temperature at the endpoints). Thus, our constraints for the problem are

As with the Poisson problem, there are a variety of boundary conditions we could consider here, including mixing a Neumann condition (prescribing the heat flux) at one end and the temperature at the other end.

$$\begin{aligned} u(x, t = 0) &= \eta(x) \\ u(x = a, t) &= g_a(t) \\ u(x = b, t) &= g_b(t) \end{aligned} \tag{3}$$

for some prescribed functions $\eta(x)$, $g_a(t)$, and $g_b(t)$.

2 *Numerical discretization: the method of lines*

How will we go about solving this problem that depends on space and time? We will combine our methods for solving time-dependent problems with those for solving space-dependent problems. The trick to accomplishing this us to first *discretize* the IBVP in space. Doing this will result in an IVP, which we can then solve using our favorite time integrator. This procedure is referred to as the *method of lines*. Remember that our focus this week is on finite difference methods, so we will focus on using local interpolation. Let us see how this process works in detail.

2.1 *Step 1: discretize in space*

If you feel uncomfortable with the presentation in section 2.1, now is a great time to go back to the lecture notes on the 1D Poisson problem. You will find many similarities between the two sets of notes!

We first discretize the spatial domain, or break up the continuous space variable into a finite number of pieces as

$$x_j = a + \frac{(b-a)(j-1)}{n}, \quad j = 1, \ldots, n+1 \tag{4}$$

Here is a reminder of what the spatial discretization looks like.

We must also find a way of approximating our infinite-dimensional solution $u(x,t)$. We will focus on approximating this function in space first, and will do this using local interpolation. That is, we will approximate the spatial dependence of u in terms of a finite number of locally defined basis functions in space.

Buoyed by our success with the Poisson problem, we will continue to use a *centered* representation in terms of Lagrange polynomials to approximate our solution in space. Recall from lecture 15 that if we use a locally defined p^{th} order polynomial, we require p points in addition to x_j. We will use the points $\{x_{j-p/2}, \ldots, x_j, \ldots, x_{j+p/2}\}$.

Recall that this selection of points is specific to centered representations, and requires that p be even. We may, of course, use biased representations such as a *one-sided representation* using the points $\{x_j, \ldots, x_{j+p}\}$. The motivation for using a centered representation for the heat equation is that we expect the information to be equally valuable on the left of a given point as on the right.

We will approximate $u(x,t)$ over the interval $x_{j-p/2} \leq x \leq x_{j+p/2}$ as

$$u(x,t) \approx \sum_{i=j-p/2}^{j+p/2} b_i(t) L_i^{(j)}(x) \tag{5}$$

where the $L_i^{(j)}(x)$ represents our handy Lagrange basis polynomials and the $b_i(t)$ represent the unknown coefficients in our expansion (note that these coefficients must be functions of time, since we require that our Lagrange polynomials to be only functions of x). We have expressed these unknown coefficients as b_i to be as general as possible, but notice that (just as we saw for the Poisson problem) they simplify considerably. In particular, at any instance in time we have

$$\begin{aligned} u(x_j,t) &\approx \sum_{i=j-p/2}^{j+p/2} b_i(t) L_i^{(j)}(x_j) \\ &= b_j(t) \end{aligned} \tag{7}$$

This simplification follows from the beautiful property of the Lagrange polynomials that

$$L_i^{(j)}(x_j) = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases} \tag{6}$$

and therefore that $b_j(t)$ is an approximation of $u(x_j, t)$. Thus, for finite difference methods we may write that

$$u(x,t) \approx \sum_{i=j-p/2}^{j+p/2} u_i(t) L_i^{(j)}(x) \tag{8}$$

where $u_i(t) \approx u(x_i, t)$. Thus, computing the various coefficients $b_i(t)$ is equivalent to calculating the values $u_i(t)$ that approximate that *exact* solution u at the grid point x_i at some instance in time.

2.2 *The initial value problem*

We have replaced the continuous spatial variable x by the $n+1$ points $\{x_1, \ldots, x_{n+1}\}$, and have restricted the spatial dependence of the function u to be a linear combination of the Lagrange polynomials. We will now show that the result of this *spatial discretization* is to create an initial value problem.

Plugging our approximation for $u(x,t)$ (8) into the 1D heat equation (2) gives

$$\begin{aligned}
&\sum_{i=j-p/2}^{j+p/2} \dot{u}_i(t) L_i^{(j)}(x_j) = \kappa \sum_{i=j-p/2}^{j+p/2} u_i(t) \left.\frac{d^2 L_i^{(j)}}{dx^2}\right|_{x=x_j} + g(x_j,t) \qquad (j = 2, \ldots, n) \\
&\implies \dot{u}_j(t) = \kappa \sum_{i=j-p/2}^{j+p/2} u_i(t) \left.\frac{d^2 L_i^{(j)}}{dx^2}\right|_{x=x_j} + g(x_j,t) \qquad (j = 2, \ldots, n)
\end{aligned} \tag{9}$$

Just as we saw for the Poisson problem, do not forget that (9) is not the full story. We need to incorporate boundary conditions to get a solvable system of equations! We will discuss this subtlety in the next section.

This is an initial value problem for advancing the approximate solution at x_j, u_j. This fact is why this procedure of discretizing in space first is called the *method of lines*: the result is an IVP for the solution at each spatial point x_j. So we can conceptually think of the solution at each spatial point evolving along a "line" over time.

Take a moment to appreciate what we have done here: we have taken a complex IBVP that depends on space and time, used local interpolation in space, and simplified the problem to an initial value problem that only depends on time. We can solve this IVP using our favorite time stepping method (e.g., backward Euler, one of the Adams-Bashforth methods, etc.).

This procedure may seem somewhat abstract, as we have left things in terms of a generic polynomial order p and have not addressed the boundary conditions in detail. To provide a concrete example, we will consider the specific case of $p = 2$ next.

That is, we will use a second-order representation in space.

3 *The specific case of $p = 2$*

Let us consider the specific case where $p = 2$. This second-order method is the one we employed during our study of the Poisson problem (see the associated typed lecture notes for a refresher). In this case, we know that the righthand side of (9) simplifies to

$$\frac{1}{\Delta x^2}[u_{j-1}(t) - 2u_j(t) + u_{j+1}(t)] \tag{10}$$

which allows us to write (9) in matrix form as

$$\begin{bmatrix} \dot{u}_2 \\ u_3 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} = \frac{\kappa}{\Delta x^2} \begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix} \begin{bmatrix} u_2 \\ u_3 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} + \begin{bmatrix} g(x_2,t) + \frac{\kappa g_a(t)}{\Delta x^2} \\ g(x_3,t) \\ \vdots \\ g(x_{n-1},t) \\ g(x_n,t) + \frac{\kappa g_b(t)}{\Delta x^2} \end{bmatrix} \tag{11}$$

or more succinctly as $\dot{\boldsymbol{u}} = \boldsymbol{A}\boldsymbol{u} + \boldsymbol{g}$. The initial condition associated with this IVP is

$$\boldsymbol{u}(t=0) = \begin{bmatrix} \eta(x_2) \\ \eta(x_3) \\ \vdots \\ \eta(x_{n-1}) \\ \eta(x_n) \end{bmatrix} \tag{12}$$

This IVP is of the form $\dot{\boldsymbol{u}} = \boldsymbol{f}(\boldsymbol{u}, t)$ with $\boldsymbol{f}(\boldsymbol{u}, t) = \boldsymbol{A}\boldsymbol{u} + \boldsymbol{g}(t)$, and we may therefore advance this solution in time using our favorite time stepping method.

4 *An example*

Let us illustrate the fruits of our labor by applying our second order finite difference method to solve the 1D heat equation with $a = 2, b = 16, \kappa = 1, g(x,t) = g_a(t) = g_b(t) = 0$ and

$$\eta(x) = \exp\left[-\left(\frac{x - \frac{a+b}{2}}{2\sigma}\right)^2\right]$$

where $\sigma = 0.3$. That is, our initial condition is a Gaussian function centered at the middle of our domain and with a standard deviation of 0.3.

In this example, the initial value problem was advanced using a trapezoid method. Figure 1 shows an overlay of the solution at various times obtained using $\Delta t = \Delta x \approx 0.07$. The figure demonstrates the intuitive behavior that the peak value of the solution lessens and

Notice that we have constructed our initial value problem (11) using the truncated form of the linear system. That is, we have omitted u_1 and u_{n+1} from the solution variables that we are computing. This is natural, as u_1 and u_{n+1} are prescribed for all time, and there is no need to solve a differential equation to compute them. Indeed, by contrast, if we had included these variables as unknowns in our system, we would have to modify our initial value problem as

$$\begin{bmatrix} 0 & & & & & & \\ & 1 & & & & & \\ & & 1 & & & & \\ & & & \ddots & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 0 \end{bmatrix} \begin{bmatrix} \dot{u}_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{n-1} \\ u_n \\ u_{n+1} \end{bmatrix} =$$

$$\frac{\kappa}{\Delta x^2} \begin{bmatrix} \frac{\Delta x^2}{\kappa} & & & & & & \\ 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 & \\ & & & & 1 & -2 & 1 \\ & & & & & & \frac{\Delta x^2}{\kappa} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{n-1} \\ u_n \\ u_{n+1} \end{bmatrix}$$

$$+ \begin{bmatrix} -g_a(t) \\ g(x_2,t) \\ g(x_3,t) \\ \vdots \\ g(x_{n-1},t) \\ g(x_n,t) \\ -g_b(t) \end{bmatrix} \tag{13}$$

where the 0 in the $(1,1)$ and $(n+1, n+1)$ indices of the lefthand side matrix have arisen to account for the fact that there is *no* initial value problem for u_1 and u_{n+1}; these variables are prescribed. Because this lefthand side matrix is not invertible, we can not readily transform the linear system to something of the form $\dot{\boldsymbol{u}} = \boldsymbol{f}(\boldsymbol{u}, t)$. As a result, we will use the truncated variant of the linear system for solving these IBVPs.

the x-extent over which the solution is nonzero gets larger as time progresses. This is typical of the *diffusive* behavior associated with the heat equation.

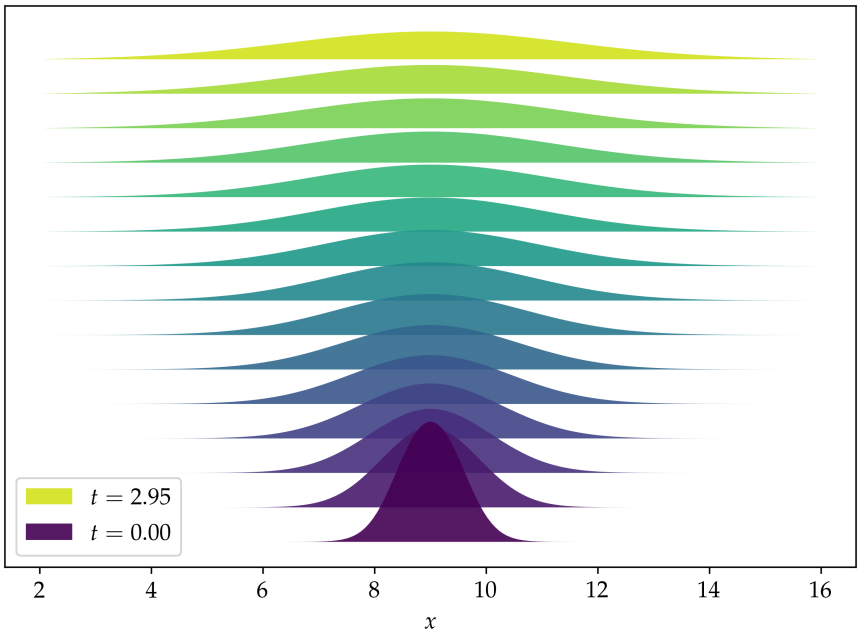

Figure 1: An overlay of the numerical solution of the 1D heat equation at various instances in time. Each plot is displaced in the y-direction by the amount of time that has elapsed.

5 *Some more notes*

We may feel some reassurance that our numerical solution gives us physically intuitive behavior. At the same time, we have left many questions unanswered. Does the numerical solution converge to the true solution? If so, how does this convergence depend on Δt and Δx? Finally, what time stepping methods are best suited for solving the initial value problem (11) obtained by discretizing the equations in space?

Unfortunately, we will not have time to discuss these topics in detail in this class. So here are the punchline answers to those questions:

1. Using a finite difference method in space and time does produce convergent solutions.

2. The convergence rate in space is equal to the order of the spatial discretization used (e.g., for the $p = 2$ case above the convergence rate is $O(\Delta x^2)$). The convergence rate in time is equal to the convergence rate of the time stepping method used for the IVP (e.g., using a backward Euler method would yield an $O(\Delta t)$

While we unfortunately do not have time to go through the details of these convergence properties, a very brief overview is that convergence depends on (you guessed it!) i) using Taylor series of the truncation error and ii) combining this analysis with a notion of stability.

convergence rate). Notice the implications of this: one could use a different order method in space than in time and get different convergence rates in Δx and Δt. This is typically thought of as undesirable, since often we want a solution that is comparably accurate in space and time. So the order of the methods in space and time are typically selected to match.

3. After spatially discretizing the heat equation, it is best to treat the resulting IVP using an implicit method. That is, advancing the IVP associated with the spatially discrete heat equation is best done with a time stepping method that has a large stability region. This consideration is the reason why I used a trapezoid method in the example above. Note that this stiff property of the spatially discrete IVP is a feature of the heat equation, and *not* generic to spatially discrete IBVPs. There are many IVPs that produce non-stiff IVPs when spatially discretized, which may be handled using a favorite explicit time stepping method.

The reason that a time stepping method with a large stability region is useful is that the spatially discrete heat equation is *stiff*: it has some eigenvalues very near the origin but some that extend very far into the left-half complex plane, requiring very small time steps for methods with small stability regions.