Numerical Methods

Alberto Ruiz-Biestro

July 24, 2024

Contents

1 Finite Difference Methods

A great resource for finite difference methods is the book by Smith [1]. Almost all of the information and methods can be traced back to that book.

1.1 Wave Equation

Hyperbolic equations are generally the description of vibrational problems. The (scalar) wave equation is the model hyperbolic equation,

$$\frac{\partial^2 u}{\partial t^2} - c^2 \Delta u = f, \tag{1}$$

where c is the speed determined by the medium and f is a source. Considering the homogeneous version of the previous equation. It will be useful to write it in operator form,

$$D_t^2 u - c^2 \left(D_x^2 + D_y^2 + D_z^2\right) u = 0. \tag{2}$$

We proceed as usual in discretizing the derivatives with a *centered differences scheme* that has an error $\mathcal{O}((h^2)^n)$. Our description of said equation will be at gridpoints $(p\,h, q_i\,k)$, with h being the time step and k the spatial step size[1]. It will be useful to realize that a (cartesian) centered difference scheme has a general form of a sum of the value of the field at a set of gridpoints $\{x_i\}$ times a set of weights $\{w_i\}$,

$$D_x u \Rightarrow \frac{1}{k_x}(\cdots + w_{i-1}u_{i-1} + w_i u_i + w_{i+1}u_{i+1} + \dots), \tag{3}$$

From Wikipedia one gets the required weights for a second order derivative (see Table 1).

Table 1: Weights for discretized $\partial^2/\partial x^2$, with error $O(h^n)$ at grid index.

n \ idx	-3	-2	-1	0	1	2	3
2			1	-2	1		
4		$-1/12$	$16/12$	$-30/12$	$16/12$	$-1/12$	
6	$2/180$	$-27/180$	$270/180$	$-490/180$	$270/180$	$-27/180$	$2/180$

1.1.1 2D operator

Discretizing a cartesian operator in a cartesian grid translates into summing the respective weights in distinct axes, since $\Delta = \sum_j \partial^2/\partial x_j^2$. For example, for our two dimensional $D_x^2 + D_y^2$, an

[1] It need not be the same, but the discretization takes a more complicated form (non uniform grids)

approximation of $\mathcal{O}(h^2)$[2] requires the sum of $[1,-2,1]$ and $[1,2,1]^T$. One finds the 2D version of the discretized wave equation to be,

$$\frac{1}{h^2}\left(u_{i,j}^{t-1}-2u_{i,j}^{t}+u_{i,j}^{t+1}\right)-\frac{c^2}{h^2}\left(\underbrace{\begin{bmatrix}0&1&0\\1&-4&1\\0&1&0\end{bmatrix}}_{\mathbf{D}^2}\cdot\begin{bmatrix}&u_{i,j+1}^{t}&\\u_{i-1,j}^{t}&u_{i,j}^{t}&u_{i+1,j}^{t}\\&u_{i,j-1}^{t}&\end{bmatrix}\right)=0, \tag{4}$$

where $\cdot$ implies the dot product. Alternatively, one calls the spatial discretization operator $\mathbf{D}^2$ and thus writes its product as $\mathbf{D}^2\cdot\mathbf{u}$. This expression already hints at the fact that one effectively *convolves* the operator $\mathbf{D}^2$ with the lattice points, $\mathbf{D}^2 * [u]^t$. This is what's done usually in image filters and graphics processing, and what is described in Algorithm 1.

Solving for u^{t+1}, one finds

$$u_{i,j}^{t+1}=\alpha^2\left(\mathbf{D}^2\cdot\mathbf{u}\right)+2u_{i,j}^{t}-u_{i,j}^{t-1}, \tag{5}$$

where $\alpha\equiv ch/k$. One has the freedom of choosing each approximation's error, whether to use more coefficients for D_t^2 or for $\mathbf{D}^2$, or even take it to an $N-$dimensional routine by using a general tensor form of $\mathbf{D}^2$. One also notices that exchanging $u_{i,j}^{t+1}\leftrightarrow u_{i,j}^{t-1}$ has no effect whatsoever in the final expression.

As with all centered differences, one can kick-start the time-stepping through an explicit Euler scheme,

$$u_{i,j}^{t+1}=u_{i,j}^{t}+h(c^2\mathbf{D}^2\cdot\mathbf{u}),$$

which just implies having to state the initial conditions for $u_{i,j}^{(0)}$ and $u_{i,j}^{(-1)}$ (should one start at $t=0$).

Adding a source f requires no further changes other than adding its value $f_{i,j,k}$ at time t. The algorithm for the time evolution of the wave equation in a domain Ω, assuming a Dirichlet boundary conditions, and V to be the initial perturbation, can be seen in Algorithm 1.

Algorithm 1 FDTD for wave equation

$u^{t-1}\leftarrow 0$
$u^{t}\leftarrow V$
$u^{t+1}\leftarrow 0$
while $N<N_{\max}$ **do**
  $u^{t+1}=\alpha^2(\mathbf{D}^2 * u^t)+2u^t-u^{t-1}$
  $u^{t+1}(\mathbf{r}\in\partial\Omega)=0$ ▷ Enforce Dirichlet B.C.
  $u^{t-1}\leftarrow u^t$
  $u^{t}\leftarrow u^{t+1}$ ▷ Update fields
end while

[2]Also known as a 5-point lattice

1.1.2 Image filters

Using an image filtering library has the advantage of an easy implementation of periodic boundary conditions. In Julia, such function is given by `imfilter(im,ker,"circular")`. Furthermore, it allows generalization to N dimensions, which makes it easy to read and work with. However, such operations can get expensive memory-wise, which is why it might also be recommended to use `for` loops. For Julia, this is no issue, since `for` loops should be as fast as Fortran code.

References

[1] Gordon D Smith. *Numerical solution of partial differential equations: finite difference methods*. Oxford university press, 1985.

1.2 Ginzburg-Landau equation

Ignoring some stability issues with the centered difference method outlined before, once we discretize the Δ operator, we can apply it to any other equation. Consider the equation

$$\frac{\partial \phi}{\partial t} = \phi - |\phi|^2\phi + \Delta\phi, \tag{6}$$

where we define $f(\phi) \equiv \phi - |\phi|^2\phi$ for convenience. Using a simple Euler approximation

$$\frac{1}{h}\left(\phi_{i,j}^{t-1} - 2\phi_{i,j}^{t} + \phi_{i,j}^{t+1}\right) = f_{i,j}^{t} + \frac{1}{k^2}(\mathbf{D}^2 * \phi)_{i,j}^{t}$$

1.3 Crank-Nicholson method

2 Boundary-Wall method

2.1 BWM for Schrödinger's equation

2.2 BWM for other equations