Discretizing and Numerically Solving the Wave Equation: documentation for WaveEquationOptimized.py, PhotonGravityWavePerpendicularCase.py, and PhotonGravityWave4D.py

Varadarajan Srinivasan

For a function $f(x)$ whose discretization is represented as a sequence f_n, we define its numerical derivative with respect to x, $\mathrm{nD}_x(f_n)$, as the average of the forward difference and backward difference.

$$\mathrm{nD}_x(f_n) = \frac{1}{2}\left(\frac{f_n - f_{n-1}}{\Delta x} + \frac{f_{n+1} - f_n}{\Delta x}\right) = \frac{f_{n+1} - f_{n-1}}{2\Delta x} \tag{1}$$

For differentiable functions, this approximates the true derivative of the function to an arbitrarily high accuracy as the step size $\Delta x \to 0$. As explained in DiscretizedLaplaceEquation.pdf, this logically produces the one-dimensional numerical second derivative $\mathrm{nD}_x^2(f_n)$ as well as the numerical Laplacian $\mathrm{nD}_{x,y}^2(f_{m,n})$ for our evenly spaced grid ($\Delta x = \Delta y \equiv \Delta$).

$$\mathrm{nD}_x^2(f_n) = \mathrm{nD}(\mathrm{nD}(f_n)) = \frac{f_{n+1} - 2f_n + f_{n-1}}{\Delta^2} \tag{2}$$

$$\mathrm{nD}_{x,y}^2(f_{m,n}) = \frac{f_{m,n+1} + f_{m,n-1} + f_{m+1,n} + f_{m-1,n} - 4f_{m,n}}{\Delta^2} \tag{3}$$

The Wave Equation for displacement f,

$$\frac{\partial^2 f}{\partial t^2} = c^2 \nabla^2 f, \tag{4}$$

can be numerically approximated using our discretizations as

$$\mathrm{nD}_t^2(f_{m,n,t}) = c^2 \mathrm{nD}_{x,y}^2(f_{m,n,t}). \tag{5}$$

Substituting Eq. 2 and Eq. 3 into Eq. 5 gives us

$$\frac{f_{m,n,t+1} - 2f_{m,n,t} + f_{m,n,t-1}}{(\Delta t)^2} = \frac{c^2}{\Delta^2}(f_{m,n+1,t} + f_{m,n-1,t} + f_{m+1,n,t} + f_{m-1,n,t} - 4f_{m,n,t})$$

Rearranging and subtracting one time step from every term gives us the desired iterative formula that solves the wave equation for given boundary conditions

$$f_{m,n,t} = K(f_{m,n+1,t-1} + f_{m,n-1,t-1} + f_{m+1,n,t-1} + f_{m-1,n,t-1} - 4f_{m,n,t-1}) + 2f_{m,n,t-1} - f_{m,n,t-2} \tag{6}$$

where

$$K = c^2 \frac{(\Delta t)^2}{\Delta^2} \tag{7}$$

is a constant which can be thought of as the square of the ratio between the speed of the wave through the plane and the information propagation speed of the numerical method. In each time-step Δt, Eq. 6 transmits information from a distance of exactly 1 grid point in every direction. This means that the information propagation speed is $c_i = \Delta/\Delta t$.

For the numerical method to successfully approximate the wave equation, numerical stability is necessary. That is, the speed of the information propagation throughout our discretization must be no slower than the wave speed $c \leq c_i$. Otherwise, our results will be nonsensical. Noting all these terms are necessarily positive, we see that

$$\frac{c}{c_i} \leq 1 \implies \frac{c\Delta t}{\Delta} \leq 1 \implies K \leq 1 \tag{8}$$

is a necessary condition for numerical stability in our method. We can get a more specific (less restrictive) necessary condition by invoking the time-marching (explicit) case of the Courant–Friedrichs–Lewy condition that

$$\Delta t \left(\frac{c_x}{\Delta x} + \frac{c_y}{\Delta y}\right) \leq C_{max} = 1 \tag{9}$$

which, for our method, becomes

$$\frac{\Delta t}{\Delta}(c_x + c_y) \leq 1 \tag{10}$$

This of course is consistent with Eq. 8 by the Triangle Inequality. Note that even our more restricted condition Eq. 10 is not necessarily a sufficient one. However, the results of the method become obviously nonsensical when the condition to ensure stability is not met and so we need not derive it. Running several tests makes it clear that $K \leq 0.5$ is comfortably sufficient for our purposes with $0.1 \leq K \leq 0.4$ being a good, compromising range. Lower values of K produce more accurate results, but are algorithmically slower.