Numerical Solutions to PDEs

Lecture Notes #15
— Second Order Equations —
Boundary Conditions; 2D and 3D

Peter Blomgren,
⟨blomgren.peter@gmail.com⟩

Department of Mathematics and Statistics
Dynamical Systems Group
Computational Sciences Research Center
San Diego State University
San Diego, CA 92182-7720

http://terminus.sdsu.edu/

Spring 2018

We started looking at problems with more than one time-derivative, *e.g.* the wave equation, and the Euler-Bernoulli beam equation.

Many of our previous definitions and theorems go through without change: consistency, convergence, and order of accuracy; however, the **definition and machinery for checking stability** had to be modified a little.

Previously: Second Order Equations,   2 of 2

The stability definition was modified to allow for **linear growth** in the ℓ_2-norm over time (to match the growth of the PDE), and in the **von Neumann** analysis we allowed for double roots of the amplification polynomial on the unit circle.

Further, we augmented our definitions of Schur and von Neumann polynomials (with the von-Neumann-Order), so that a finite difference scheme for a second order (time) problem is stable if and only if its amplification polynomial is a von Neumann polynomial of second order.

Boundary Conditions for Second-Order Equations   1 of 2

Since the solutions to the second order wave equation

$$u_{tt} - a^2 u_{xx} = 0$$

consist of two parts moving at characteristic speeds $\pm a$, it is clear that in a finite domain, *e.g.* $0 \leq x \leq 1$, we must specify **one** boundary condition at each boundary.
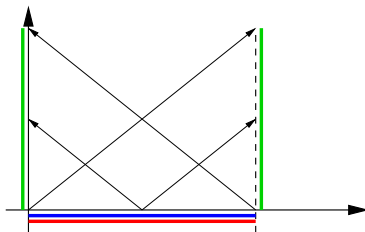

Figure: We must specify two initial conditions *e.g.* $u(0, x) = u_0(x)$, $u_t(0, x) = u_1(x)$, and two boundary conditions *e.g.* $u(t, 0) = f_0(t)$, and $u(t, 1) = f_1(t)$.

The specified boundary conditions can be of Dirichlet type (u specified), or Neumann type (u_x specified), or a combination thereof:

$$\alpha_0 u(0, t) + \beta_0 u_x(0, t) = \tilde{f}_0(x), \quad \min\{|\alpha_0|, |\beta_0|\} > 0$$

$$\alpha_1 u(1, t) + \beta_1 u_x(1, t) = \tilde{f}_1(x), \quad \min\{|\alpha_1|, |\beta_1|\} > 0$$

When $\beta_i = 0$, the numerical implementation of the boundary condition is trivial

$$\alpha_i v_I^n = \tilde{f}_I^n, \quad I = \begin{cases} 0 & \text{when } i = 0 \\ M & \text{when } i = 1 \end{cases}$$

When $\beta_i \neq 0$, then several possibilities present themselves. For a pure Neumann boundary condition at $x = 0$
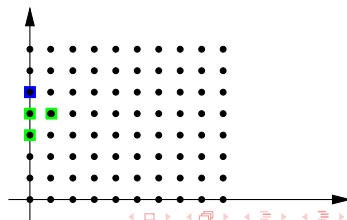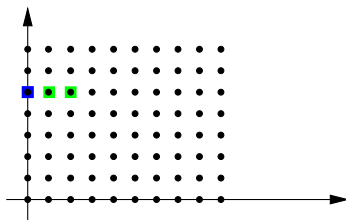
$$u_x(t, 0) = 0, \quad \text{no-flux}$$

We can use

$$v_0^{n+1} = \frac{4v_1^{n+1} - v_2^{n+1}}{3}$$

or

$$v_0^{n+1} = 2v_0^n - v_0^{n-1} - 2a^2\lambda^2(v_0^n - v_1^n)$$

The first formula originates from the second-order accurate
one-sided approximation

$$u_x(0) = \frac{4u(h) - 3u(0) - u(2h)}{2h} + \mathcal{O}\left(h^2\right),$$

and the second from applying the scheme

$$\frac{v_m^{n+1} - 2v_m^n + v_m^{n-1}}{k^2} = a^2 \frac{v_{m+1}^n - 2v_m^n + v_{m-1}^n}{h^2},$$

at $m = 0$, and eliminating the **ghost point** v_{-1}^n using the central
second-order difference

$$\frac{v_1^n - v_{-1}^n}{2h} = 0.$$

First-order one-sided differences should be avoided, since they will
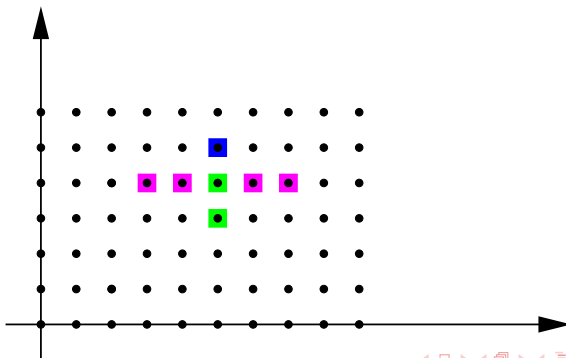degrade the overall accuracy of the scheme.

The scheme

$$\frac{v_m^{n+1} - 2v_m^n + v_m^{n-1}}{k^2} = a^2 \left(1 - \frac{h^2}{12} \delta^2 \right) \delta^2 v_m^n$$

is accurate of order (2,4) for the wave equation $u_{tt} = a^2 u_{xx}$.

If the value on the boundary is specified, then the value next to the boundary can be determined by interpolation, *e.g.*

$$v_1^{n+1} = \frac{1}{4} \left(v_0^{n+1} + 6v_2^{n+1} - 4v_3^{n+1} + v_4^{n+1} \right)$$

which comes from (the numerical BC — $\frac{\partial^4}{\partial x} u = 0$):

$$h^4 \delta_+^4 v_0^{n+1} = 0.$$

Applying this scheme with Neumann/mixed boundary conditions becomes quite challenging; — we can use (1) two layers of "ghost points," v_{-1}^n, and v_{-2}^n, which must be eliminated; or (2) non-symmetric finite differencing in the x-direction. In both settings we (α) have to match the order of the scheme, and (β) analyze the stability [lecture notes #19].

Next, we consider the Euler-Bernoulli equation

$$u_{tt} = -b^2 u_{xxxx}$$

and the second order accurate scheme

$$\frac{v_m^{n+1} - 2v_m^n + v_m^{n-1}}{k^2} = -b^2 \frac{v_{m+2}^n - 4v_{m+1}^n + 6v_m^n - 4v_{m-1}^n + v_{m-2}^n}{h^4}$$

Here, we are going to need 2 boundary conditions at each end-point:
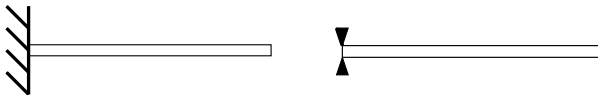

Figure: Illustration of physical boundary conditions, in the left figure the beam is clamped in at $x = 0$, and we have $u(t, 0) = u_x(t, 0) = 0$; in the right figure the beam is fixed in place at $x = 0$ but is allowed to pivot, and we have $u(t, 0) = u_{xx}(t, 0) = 0$. In both cases the right end of the beam is free to move, and the boundary conditions are $u_{xx}(t, L) = u_{xxx}(t, L) = 0$.

Boundary Type	**u**	**u$'$**	**u$''$**	**u$'''$**
Free End			$u'' = 0$	$u''' = 0$
Clamp at End	fixed	fixed		
Simply Supported End	fixed		$u'' = 0$	
Point Force at End			$u'' = 0$	specified
Point Torque at End			specified	$u''' = 0$
	Δu	**Δu$'$**	**Δu$''$**	**Δu$'''$**
Interior Clamp	$\Delta u = 0$	$\Delta u' = 0$		
Interior Simple Support	$\Delta u = 0$	$\Delta u' = 0$	$\Delta u'' = 0$	
Interior Point Force	$\Delta u = 0$	$\Delta u' = 0$	$\Delta u'' = 0$	$\Delta u'''$ specified
Interior Point Torque	$\Delta u = 0$	$\Delta u' = 0$	$\Delta u''$ specified	$\Delta u''' = 0$

Note: Here $\Delta u'' \equiv u''(x_{\text{right}}) - u''(x_{\text{left}})$.

http://en.wikipedia.org/wiki/Euler-Bernoulli_beam_equation

SAN DIEGO STATE
UNIVERSITY

The finite difference implementations of these boundary conditions are quite straight-forward, but require some thought...

Second order accurate approximations for u_x, u_{xx} and u_{xxx} can be used at the boundary points

$$\frac{v_1^n - v_{-1}^n}{2h}, \quad \frac{v_1^n - 2v_0^n + v_{-1}^n}{h^2}, \quad \frac{v_2^n - 2v_1^n + 2v_{-1}^n - v_{-2}^n}{2h^3},$$

$$\frac{v_{M+1}^n - v_{M-1}^n}{2h}, \quad \frac{v_{M+1}^n - 2v_M^n + v_{M-1}^n}{h^2}, \quad \frac{v_{M+2}^n - 2v_{M+1}^n + 2v_{M-1}^n - v_{M-2}^n}{2h^3},$$

after which we must eliminate the values at the "ghost points" v_{-1}^n, v_{-2}^n, v_{M+1}^n, and v_{M+2}^n. It's "just" a "book-keeping" problem!

In terms of definitions and theory, nothing much changes as we move our finite difference schemes into 2D and 3D.

The wave equation in 2D / 3D is given by

$$u_{tt} = a^2 \left(u_{xx} + u_{yy} \right), \quad u_{tt} = a^2 \left(u_{xx} + u_{yy} + u_{zz} \right),$$

and the most straight-forward second order schemes are given by

$$\delta_t^2 v_{\ell,m}^n = a^2 \left(\delta_x^2 v_{\ell,m}^n + \delta_y^2 v_{\ell,m}^n \right)$$

$$\delta_t^2 v_{k,\ell,m}^n = a^2 \left(\delta_x^2 v_{k,\ell,m}^n + \delta_y^2 v_{k,\ell,m}^n + \delta_z^2 v_{k,\ell,m}^n \right),$$

When $\Delta x = \Delta y = \Delta z = h$ the stability conditions for 2D and 3D are

$$a\lambda \leq \frac{1}{\sqrt{2}}, \quad a\lambda \leq \frac{1}{\sqrt{3}}$$

Stability "Footprint" in 2D
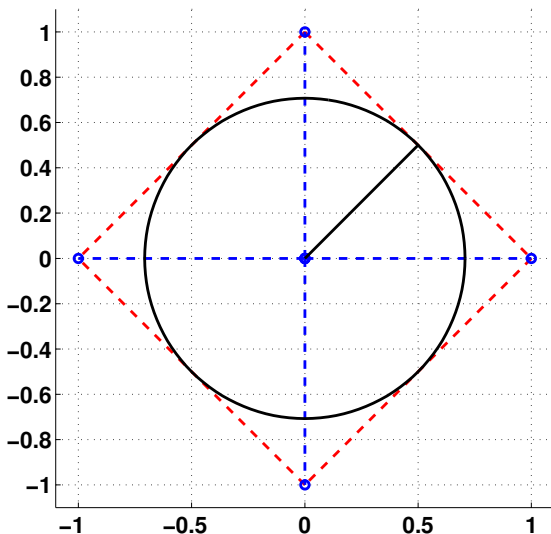
Stability "Footprint" in 3D
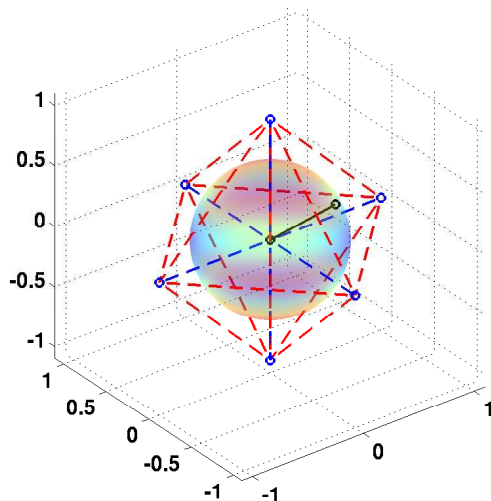
These restrictions can be improved to $a\lambda \leq 1$, by modifying the schemes, here in 2D:

$$\delta_t^2 v_{\ell,m}^n = \frac{1}{4} a^2 \left[\delta_x^2 (v_{\ell,m+1}^n + 2v_{\ell,m}^n + v_{\ell,m-1}^n) \right.$$
$$\left. + \delta_y^2 (v_{\ell+1,m}^n + 2v_{\ell,m}^n + v_{\ell-1,m}^n) \right]$$
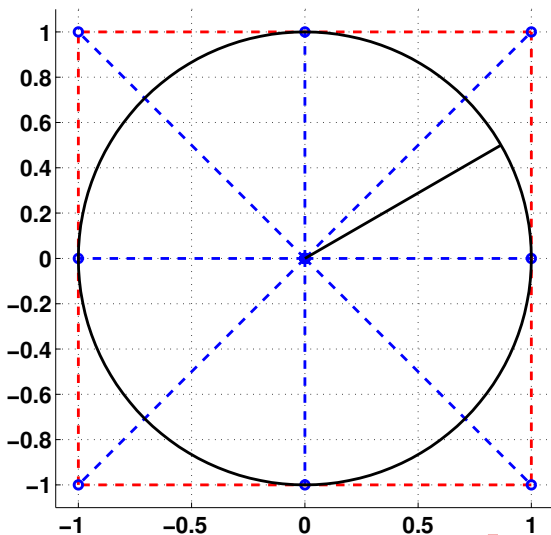
Further, ADI schemes can be developed, *e.g.*

$$\left[1 - \frac{1}{4} k^2 a^2 \delta_x^2 \right] \tilde{v}_{\ell,m}^{n+1/2} = \left[1 + \frac{1}{4} k^2 a^2 \delta_y^2 \right] v_{\ell,m}^n$$
$$\left[1 - \frac{1}{4} k^2 a^2 \delta_y^2 \right] \tilde{v}_{\ell,m}^{n+1} = \left[1 + \frac{1}{4} k^2 a^2 \delta_y^2 \right] \tilde{v}_{\ell,m}^{n+1/2}$$
$$v_{\ell,m}^{n+1} = 2\tilde{v}_{\ell,m}^{n+1} - v_{\ell,m}^{n-1}$$

Stability "Footprint" in 2D

Stability "Footprint" in 3D
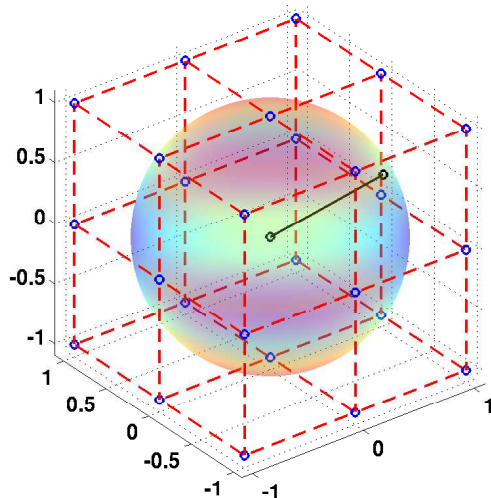
If we look at the amplification factors corresponding to the finite difference scheme for the wave equations, we have

$$g_{\pm} = \left[\left[1 - a^2 \lambda^2 \left(\sin^2 \left(\frac{\theta}{2} \right) + \sin^2 \left(\frac{\phi}{2} \right) \right) \right] \pm i a \lambda \left(\sin^2 \left(\frac{\theta}{2} \right) + \sin^2 \left(\frac{\phi}{2} \right) \right)^{1/2} \right]^2$$

Comparing this with

$$e^{ia(\xi_1^2 + \xi_2^2)^{1/2} k}$$

we can identify the phase velocity, $\alpha(\xi_1, \xi_2)$, from the expression

$$\sin \left[\frac{1}{2} \alpha(\xi_1, \xi_2) k \, (\xi_1^2 + \xi_2^2)^{1/2} \right] = a \lambda \left(\sin^2 \left(\frac{h\xi_1}{2} \right) + \sin^2 \left(\frac{h\xi_2}{2} \right) \right)^{1/2}$$

With a little bit of help from Taylor, we identify

$$\alpha(\xi_1, \xi_2) = a \left[1 - \frac{h^2 |\xi|^2}{24} \left(\cos^4 \beta + \sin^4 \beta - a^2 \lambda^2 \right) + \mathcal{O} \left(h^4 |\xi|^4 \right) \right]$$

where

$$|\xi| = (\xi_1^2 + \xi_2^2)^{1/2}, \quad \beta = \tan^{-1}(\xi_1/\xi_2)$$

This shows that the **phase error** depends on the direction of propagation $\bar{\mathbf{n}} = (\cos \beta, \sin \beta)$.

In most computations this distortion is not visible, unless the grid is very coarse (h large).
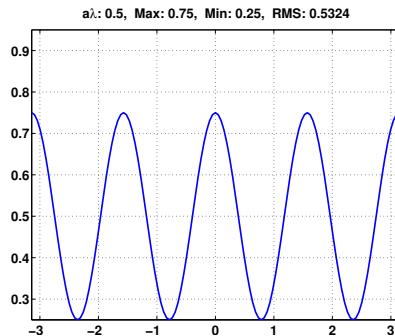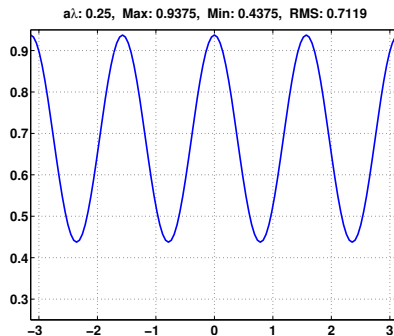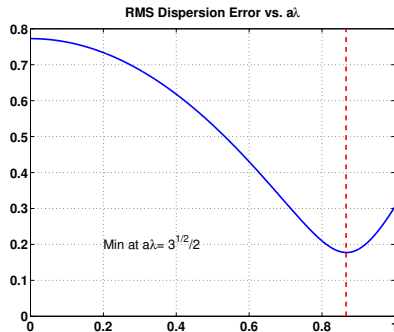
Figure: LEFT: The numerical dispersion factor ($\cos^4 \beta + \sin^4 \beta - a^2\lambda^2$) for angles in $[-\pi, \pi]$ for $a\lambda = \frac{1}{4}$. The RMS-deviation is 0.71. RIGHT: $a\lambda = \frac{1}{2}$, with RMS-deviation at 0.53.
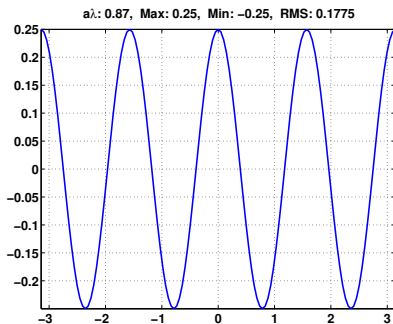
Figure: LEFT: The numerical dispersion factor ($\cos^4 \beta + \sin^4 \beta - a^2\lambda^2$) for angles in $[-\pi, \pi]$ for $a\lambda = \frac{\sqrt{3}}{2}$. The RMS-deviation is 0.18. RIGHT: The RMS Dispersion Error vs. $a\lambda$ has a minimum at $\frac{\sqrt{3}}{2}$.

We use the (2,4)-order scheme

$$
\delta_t^2 v_{\ell,m}^n = \frac{1}{4} a^2 \left[\delta_x^2 (v_{\ell,m+1}^n + 2v_{\ell,m}^n + v_{\ell,m-1}^n) \right.
$$
$$
\left. + \delta_y^2 (v_{\ell+1,m}^n + 2v_{\ell,m}^n + v_{\ell-1,m}^n) \right]
$$

to solve

$$
u_{tt} = a^2 (u_{xx} + u_{yy}), \quad x, y \in [-1, 1]
$$

with

$$
a = 1, \quad u(0, x) = J_0 \left(3\sqrt{(x - 1/2)^2 + (y - 1/2)^2} \right), \quad u_t(0, x) = 0
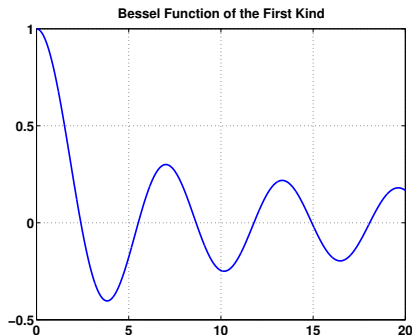$$

and

$$
\Delta x = \Delta y = h = 0.1, \quad \lambda = 0.9
$$

$J_0(r)$ is the Bessel function of the first kind

Bessel Function of the First Kind

We use the exact solution

$$u(t, x, y) = \cos(3t)\, J_0\left(3\sqrt{(x - 1/2)^2 + (y - 1/2)^2}\right)$$

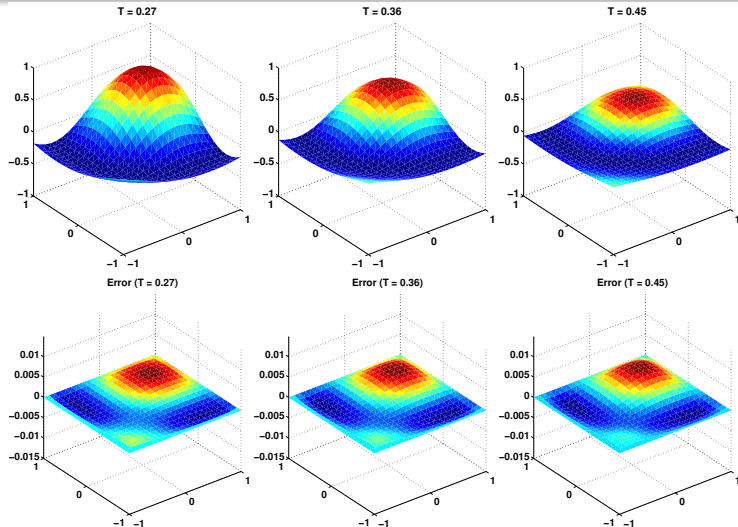to prescribe boundary conditions.

Figure: Snapshots of the solution and error at $T = 0.27$, $T = 0.36$, and $T = 0.45$.
See also the movies **wave2d_soln.mpg**, and **wave2d_err.mpg**