Aliasing

Aliasing

- Aliasing is another source of error due to discretization.
- We have seen that the solutions of hyperbolic equations are basically waves. Each wave has a wavelength. However, we cannot resolve all these wavelengths.
  - Grid-based algorithms: finite spacing between points
  - Spectral-like methods: truncation at a finite number of wavenumbers/modes

Grid-Based Aliasing

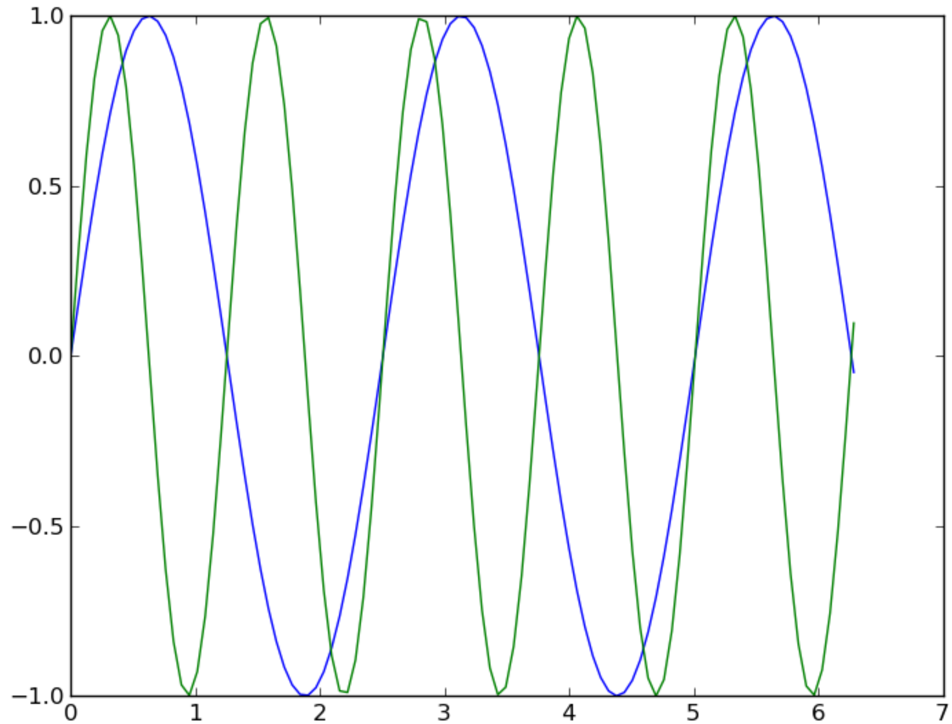
- A grid has a maximum resolved wavenumber that is determined by the grid spacing:

$$k_{\max} = \frac{2\pi}{2\Delta x}$$

- Higher wavenumbers (shorter wavelengths) cannot be resolved.

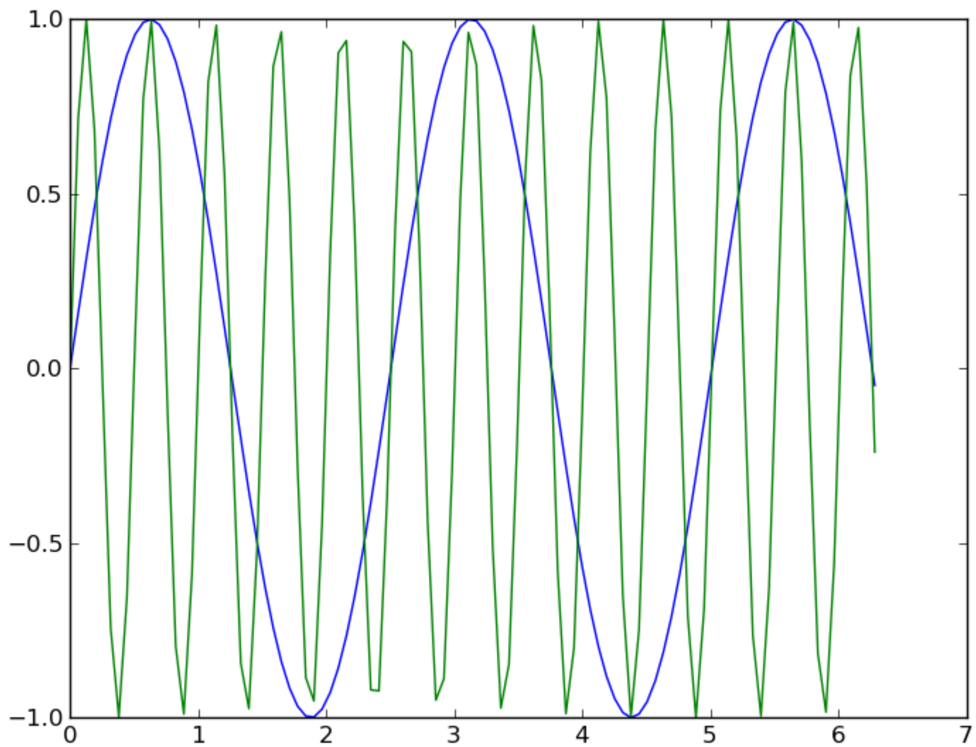
Example

- Waves sin(i/2π) and sin(2i/2π) can be distinguished on the grid:



Example (Continued)

- Waves $\sin(i/2\pi)$ and $\sin(5i/2\pi)$ might not be distinguishable on the grid:



Consequences of Aliasing

- Aliasing causes power to be fed back from higher, unresolved wavelengths into lower, resolved wavelengths.
- Nonlinear interactions between wavenumbers can cause power to reflect from unresolved wavenumbers into resolved ones.
- When severe, aliasing can cause the power in the unresolved wavenumbers to build up in the resolved wavenumbers, eventually leading to instability.

Controlling Aliasing

- The usual method to control aliasing error is to use a *hyperviscosity*. This is an artificial viscosity that is designed to suppress high-wavenumber solutions.

Techniques for Multiple Dimensions/Operators

Operator Splitting

- If we have a PDE that can be represented as

$$\frac{\partial u}{\partial t} = L(u)$$

- Where *L* represents an operator, then if we can write *L* as

  $L(u)=L_1(u)+L_2(u)+\ldots L_n(u)$

  we can solve the PDE by *operator splitting*

Operators

- Specifically, we break each timestep into multiple steps

$$\frac{du}{dt} = L_1(u) \Rightarrow u = u_1$$

$$\frac{du_1}{dt} = L_2(u) \Rightarrow u = u_2$$

$$\vdots$$

$$\frac{du_n}{dt} = L_n(u) \Rightarrow u = u_n$$

- Boundary conditions must be applied for each subproblem.
- This method is first-order accurate in time so we use a first order timestepping method such as Euler or backwards Euler.

Example

- Step 1

$$\frac{u^{n+1/2} - u^n}{\Delta t} + L_1(u^{n+1/2}) = f^{n+1/2}$$

- Step 2

$$\frac{u^{n+1} - u^{n+1/2}}{\Delta t} + L_2(u^{n+1}) = 0$$

Partial-Timestep Splitting

- We can also advance through *partial* timesteps for each suboperator.
- Example:

$$\frac{\partial u}{\partial t} + L_1(u) + L_2(u) = 0$$

- We solve $u_t = -L_1(u)$ for t^n to $t^{n+1/2}$
- Then we solve $u_t = -L_2(u^{n+1/2})$ for $t^{n+1/2}$ to t^n
- This is second order in time so at minimum a second-order timestepping method (e.g. leapfrog or a predictor-corrector) must be used.

Example

- Step 1

$$\frac{u^{n+1/4} - u^n}{\Delta t / 2} + L_1(u^{n+1/4}) = f^{n+1/2}$$

- Step 2

$$\frac{u^{n+1/2} - u^{n+1/4}}{\Delta t / 2} + L_2(u^{n+1/2}) = 0$$

- Step 3

$$\frac{u^{n+1} - u^n}{\Delta t} + L(u^{n+1/2}) = f^{n+1/2}$$

Directional Splitting

- For spatial dimensions higher than two, we must use the higher-dimensional Taylor series to derive methods. In particular, the first few terms of a 2D Taylor series are

$$f(x,y) = f(x_0,y_0) + (x-x_0)\frac{\partial f}{\partial x}(x_0,y_0) + (y-y_0)\frac{\partial f}{\partial y}(x_0,y_o)$$
$$+\frac{(x-x_0)^2}{2}\frac{\partial^2 f}{\partial x^2}(x_0,y_0) + \frac{(y-y_0)^2}{2}\frac{\partial^2 f}{\partial y^2}(x_0,y_o)$$
$$+(x-x_0)(y-y_0)\frac{\partial^2 f}{\partial x \partial y}(x_0,y_o)$$

- Note the cross term f_{xy}

Cross Term

- There is no particular benefit to be gained from computing the cross term. Instead we usually use *directional splitting* and omit it.
- In general, we would update along each direction, then average the results. In two dimensions, with symmetric x and y discretizations, this is equivalent to averaging. We should also alternate directions, e.g. xy yx xy yx and so forth, but over a large number of timesteps (again, for 2D) this works out to pretty much the same thing as xy xy xy xy …

Splitting

- First do x then y, or y then x, but apply the second directional operator to the result of the first, not to the original value of u.
- In particular:

$$u^* = [f - L_x(\Delta t)]u_{i,j}^n$$

$$u_{i,j}^{n+1} = [f - L_y(\Delta t)]u^*$$