# Summary of Hyperbolic PDEs

- We looked at a simple linear and a nonlinear scalar hyperbolic PDE
  - There is a speed associated with the change of the solution
  - Explicit methods cannot take a step larger than the time it takes for a solution to cross a single zone
    - For nonlinear equations, this speed changes on the grid—you need to find the most restrictive timestep in a zone
  - Upwinding (for linear advection) gives a stable method
    - For nonlinear PDEs, this idea is contained in the solution to the Riemann problem

# Elliptic Problems in Physics

- Gravitational and Electric potentials (Poisson equation)

$$\nabla^2\Phi = 4\pi G\rho \qquad \nabla^2\Phi = -\frac{\rho}{\epsilon}$$

(gravitational potential) (electric potential)

- Helmholtz equation:

$$(\alpha + \nabla \cdot \beta\nabla)\Phi = f$$

  - Often arises by discretizing or separating out time in a PDE

## Elliptic Problems in Physics

- Sometimes we have a system with different PDE types.
- Fluid dynamics w/ self gravity

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho U) = 0$$

$$\frac{\partial \rho U}{\partial t} + \nabla \cdot (\rho U U) + \nabla p = \rho \nabla \Phi \qquad \nabla^2 \Phi = 4\pi G \rho$$

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot (\rho U E + U p) = \rho U \cdot \nabla \Phi$$

- Incompressible hydrodynamics

$$U_t + U \cdot \nabla U + \nabla p = 0 \qquad \nabla \cdot U = 0$$

(constraint on velocity)

## Elliptic PDEs and Boundary Conditions

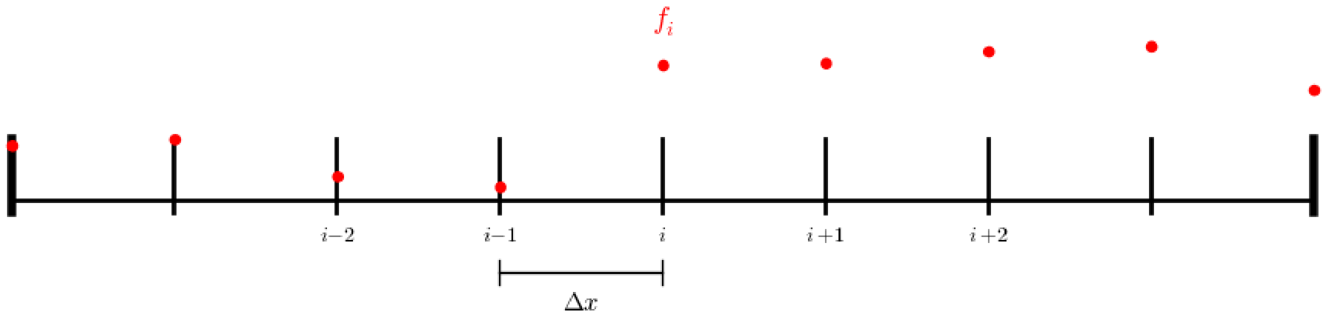
- There is no time-dependence in these equations
- Field responds instantly to the boundary conditions and source
  - There is no propagation speed as in the hyperbolic / advection equations we studied
  - Treatment of boundary conditions becomes essential

## Relaxation Methods

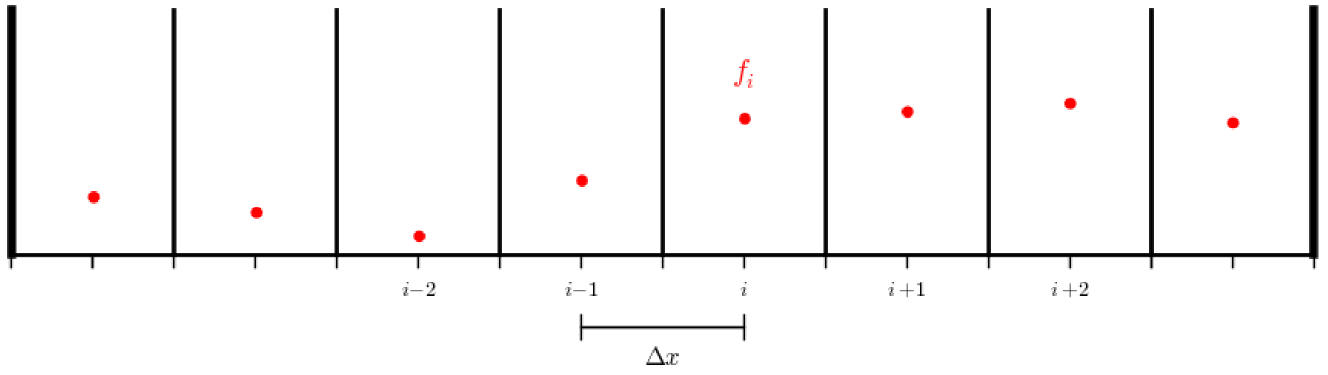
- We'll see that for a broad class of elliptic problems, relaxation methods (iterative) are easy to implement
  - Multigrid is a technique that we'll study to accelerate the convergence of relaxation methods
- There is an excellent book, *The Multigrid Tutorial*, that gives a great introduction to these methods and the math behind them
  - We'll follow this a bit, but focus on just some of the main results
  - We'll do things in terms of cell-centered finite-difference / finite-volume grids (the text focuses on finite-difference)
    - Differences come up in boundary conditions and transferring the problem through a hierarchy of grids (as we'll see shortly)
  - Some PDF notes are linked online

# Grid Types

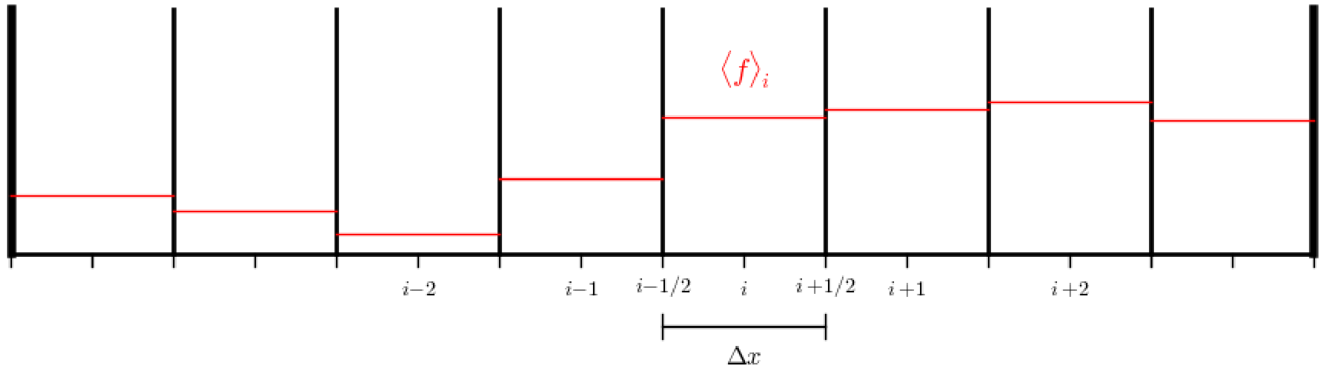
finite-difference



cell-centered finite-difference



finite-volume



## Grid Types

- Major difference between grid types:
  - Does the data exist precisely on the boundary?
  - Since boundary conditions critically affect the solution, we need to keep the centering of the data in mind
- Cell-centered finite-difference vs. Finite-volume:
  - To second-order accuracy, we can treat the cell-averages as centered in the zone

$$\langle f \rangle_i = f(x_i) + \mathcal{O}(\Delta x^2)$$

    - Blackboard derivation...
  - Our methods will be second-order accurate for both of these grid types

## Model Problem

- Consider a one-dimensional Poisson equation:

$$\phi'' = f$$

  - This is a second-order ODE, so we need 2 boundary conditions
    - Dirichlet: $\phi(a) = A$
    - Neumann: $\phi'(a) = C$
  - In two or more dimensions, this would be a PDE
- We already saw the shooting method for solving this (when we studied ODEs)
  - That does not translate to multi-dimensions easily

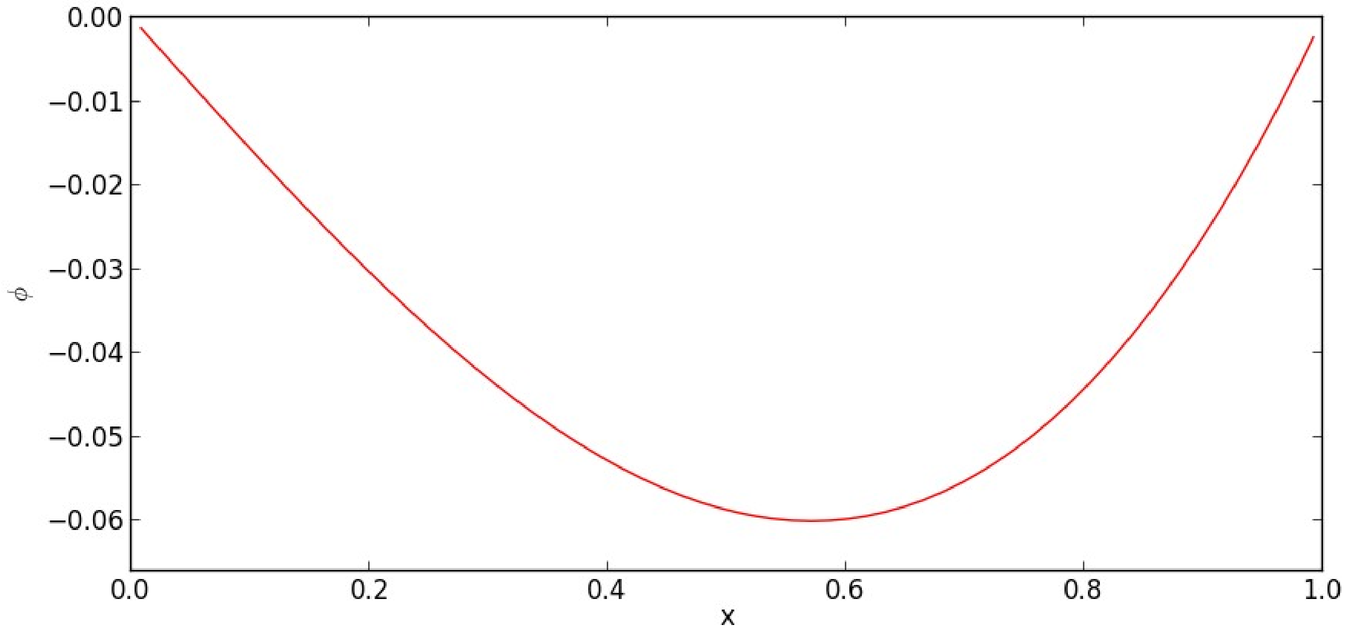
# Model Problem

- To allow us to test things, let's pick something with an analytic solution

$$\phi'' = \sin(x) \qquad \phi(0) = 0,\ \phi(1) = 0$$

  - Solution:

$$\phi(x) = -\sin(x) + x\sin(1)$$



# Relaxation

- Recall from our lecture on derivatives that a second-order accurate difference for the second derivative is:

$$\phi_i'' = \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2}$$

  - True on F-D or F-V grids
- Our 1-d Poisson equation becomes:

$$\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2} = f_i$$

  - Solve for a single zone:

$$\phi_i = \frac{1}{2}(\phi_{i+1} + \phi_{i-1} - \Delta x^2 f_i)$$

  - Set of coupled algebraic equations (think matrices)

# Relaxation

- Instead of a direct matrix solve, we'll use an iterative method
  - We are just shy of being diagonally dominant, nevertheless, these methods converge
- Jacobi iteration
  - Pick initial guess: $\phi_i^{(0)}$
  - Improve the guess through relaxation:

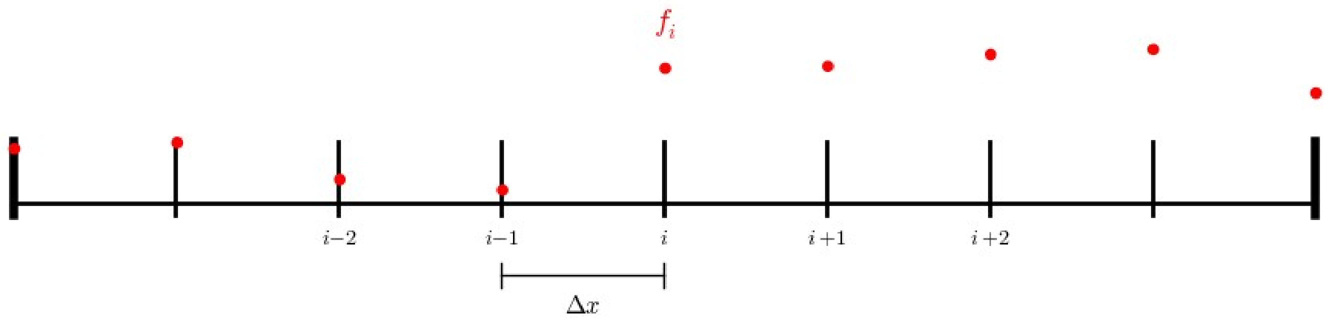
$$\phi_i^{(k+1)} = \frac{1}{2}(\phi_{i+1}^{(k)} + \phi_{i-1}^{(k)} - \Delta x^2 f_i)$$

  - Assess the error, if needed iterate...
- Gauss-Seidel iteration
  - Use new data as it becomes available:

$$\phi_i^{(k+1)} \leftarrow \frac{1}{2}(\phi_{i+1}^{(k)} + \phi_{i-1}^{(k)} - \Delta x^2 f_i)$$

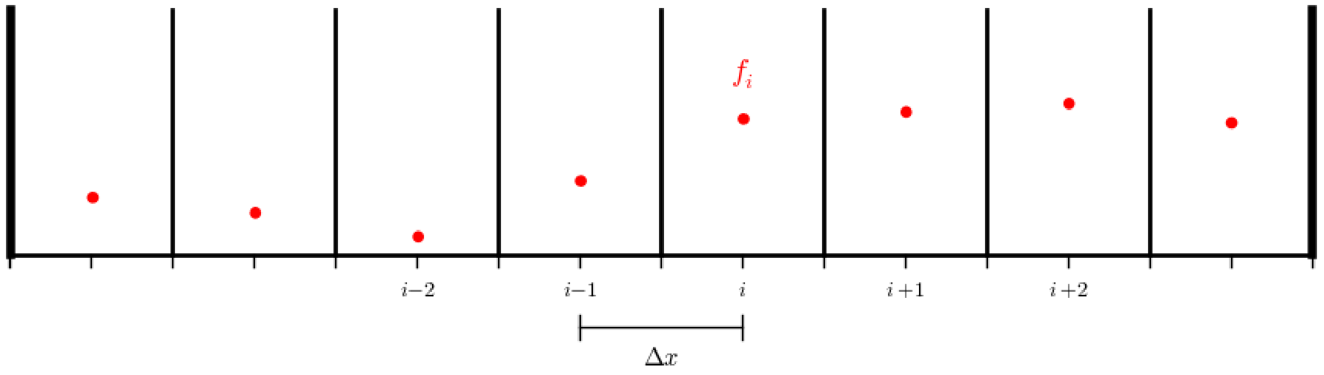
# Boundary Conditions: FV vs. FD

- F-D grid:



  - We have a point exactly on the boundary—iterate only over the interior points

- F-V or cc F-D grid:



  - Must interpolate to the boundary

# Finite-Volume BCs

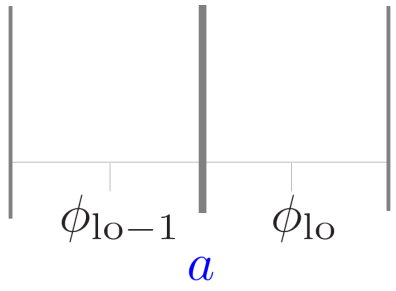
- Dirichlet: we need the value on the boundary itself to satisfy the boundary condition:

$$\phi(a) = A$$

- Use ghost cells to extend past the physical domain
- Interpolation to fill the ghost cell:
  - Naive guess:

$$\phi_{\mathrm{lo}-1} = A$$

  - Second-order accurate:



$$A = \frac{\phi_{\mathrm{lo}} + \phi_{\mathrm{lo}-1}}{2}$$

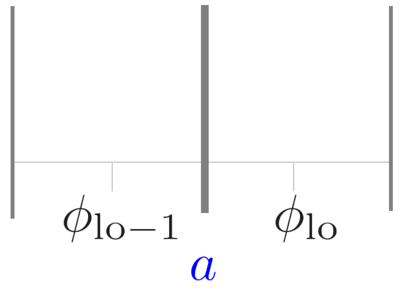
# Finite-Volume BCs

- Neumann: we need the gradient, centered a the boundary, to match the given value

$$\phi'(a) = B$$

- Use ghost cells to extend past the physical domain
- Centered difference at boundary:

$$\phi_a = \frac{\phi_{\mathrm{lo}} - \phi_{\mathrm{lo}-1}}{\Delta x}$$



  - This is second-order accurate

# Solvable Boundary Conditions

- Integrate our Poisson equation over the domain:

$$\int_\Omega \nabla^2 \phi \, d\Omega = \int_{\partial\Omega} \nabla\phi \cdot \mathbf{n} \, dS = \int_\Omega f \, d\Omega$$

  - Consider homogeneous Neumann BCs on all sides, $\nabla\phi \cdot \mathbf{n} = 0$
    - Our source must satisfy:

$$\int_\Omega f \, d\Omega = 0$$

  - Likewise, with periodic BCs all around, the flux in one end of the domain is the flux out the other end, so again, we require

$$\int_\Omega f \, d\Omega = 0$$

  - We will not converge if our source is not consistent with the boundary conditions

## Solvable Boundary Conditions

- Another way to see this:
  - Consider a 1-d Laplace equation $\phi'' = 0$
  - Solution is just a line: $\phi = ax + b$
  - If you specify different inhomogeneous Neumann BCs on each end, then you are giving conflicting values for the slope—unsolvable!

# Error and Norms

- There are many different norms that can be used to determine the error
- General p-norm:

$$\|e\|_p = \left( \Delta x \sum_{i=1}^{N} |e_i|^p \right)^{1/p}$$

  - We already saw the L2 norm
  - Also interesting are the L1 norm:

$$\|e\|_1 = \Delta x \sum_{i=1}^{N} |e_i|$$

  - And the inf norm:

$$\|e\|_\infty = \max_i |x_i|$$

# Error and Norms

- The norm gives us a single number with which to measure if we are converged
  - The choice of norm should not matter—if we converge, we should converge in all norms
  - L2 falls somewhere between L1 and the inf-norm in magnitude
  - L1 and L2 are more “global”—all  values contribute

# Error and Norms

- We still need to define the error that we are taking the norm of
  - For our test problems, we can compare to the analytic solution, but that's not general
  - Only other measure: how well we satisfy the discrete equation—this is the residual

$$r_i \equiv f_i - \frac{\phi_{i+1}^{(k)} - 2\phi_i^{(k)} + \phi_{i-1}^{(k)}}{\Delta x^2}$$

  - We use the source norm to provide a size to compare to. Stop when:

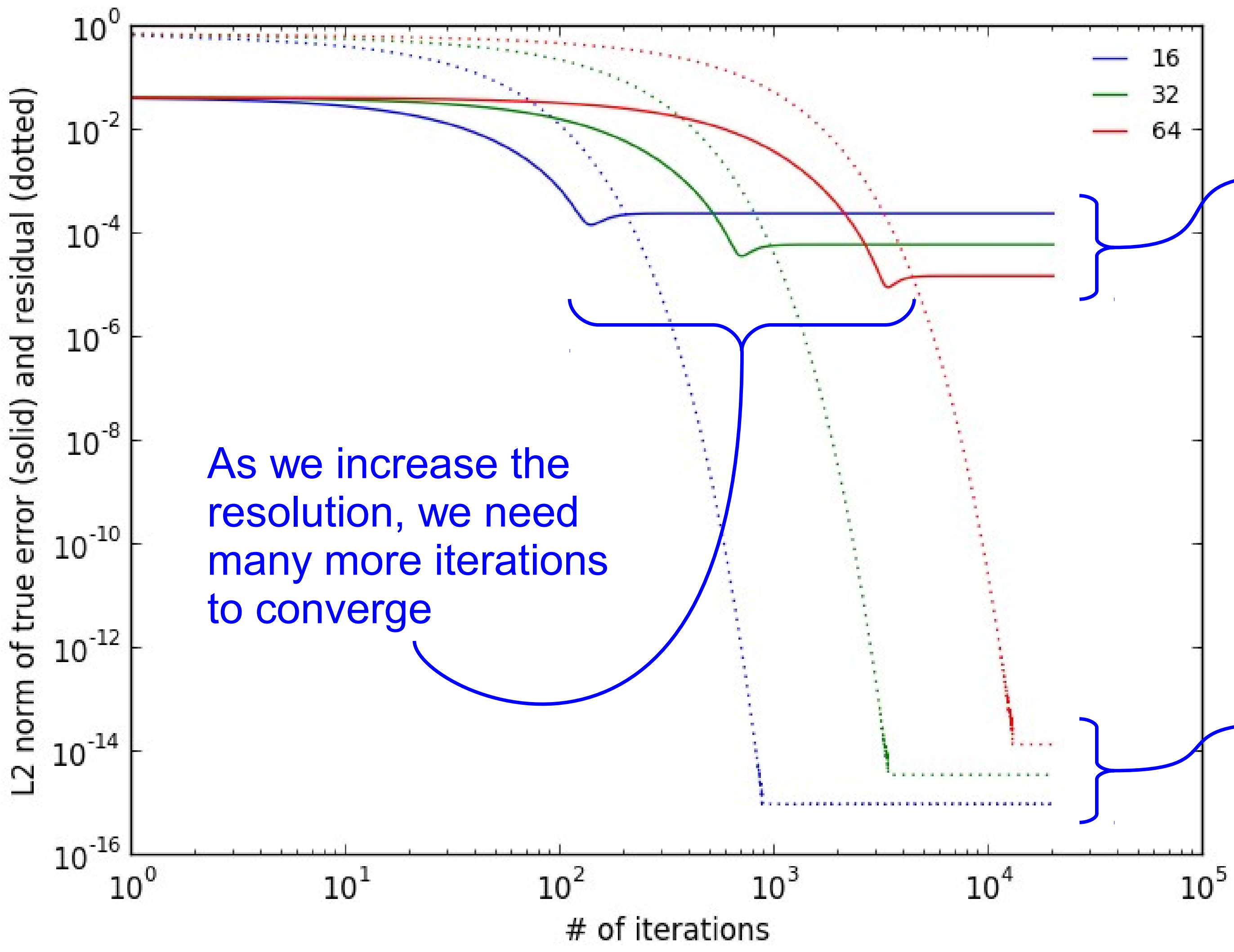
$$\|r\| < \epsilon \|f\|$$

## Implementation

- We need to fill the ghost cells after each iteration to reflect the change in the solution
- Let's start by doing a fixed number of iterations
- Let's look at the code...

# Convergence

- Residual error vs. true error using Gauss-Seidel (red-black)

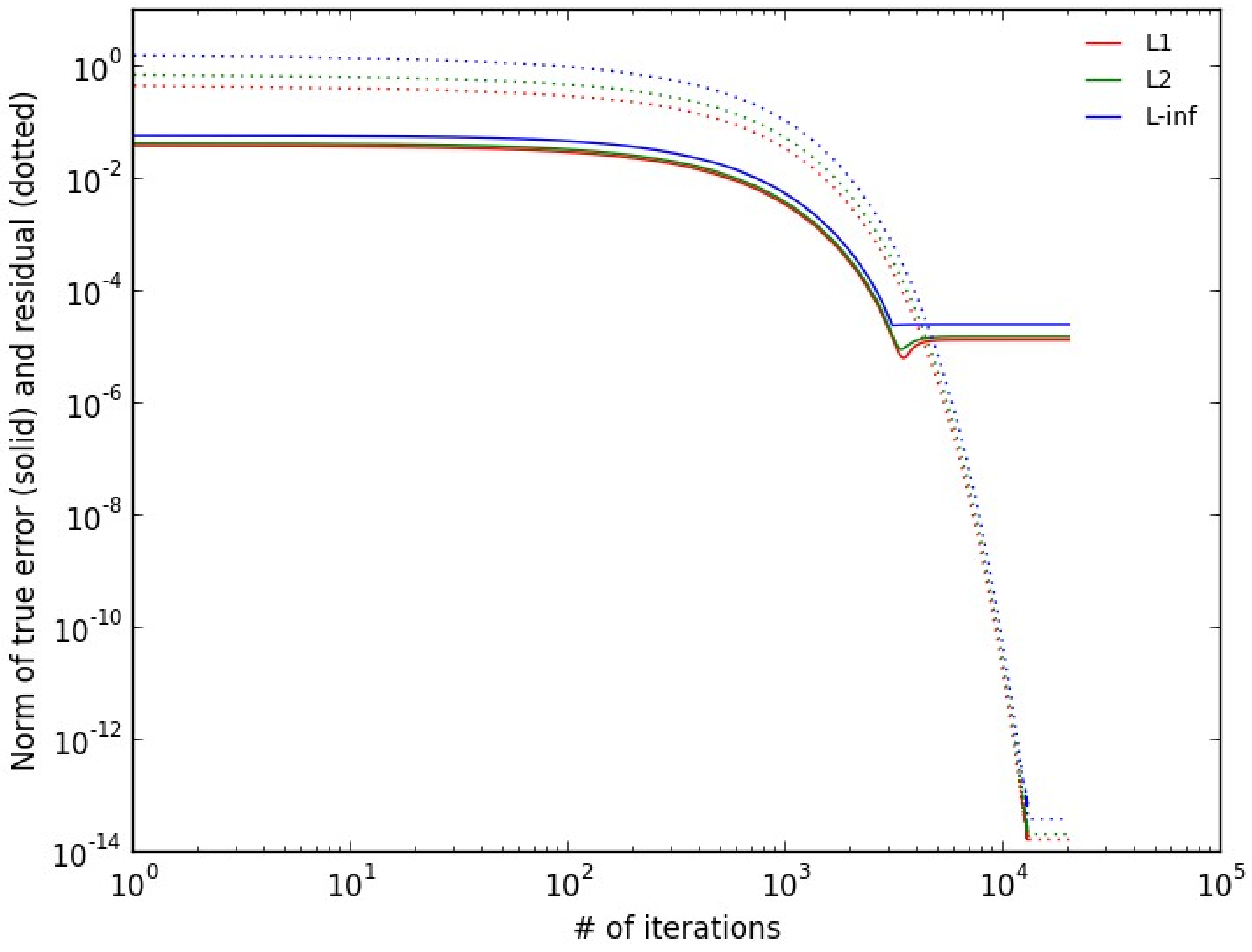


As we increase the resolution, we need many more iterations to converge

The true error stalls at a relatively high value—this is the truncation error of our method.

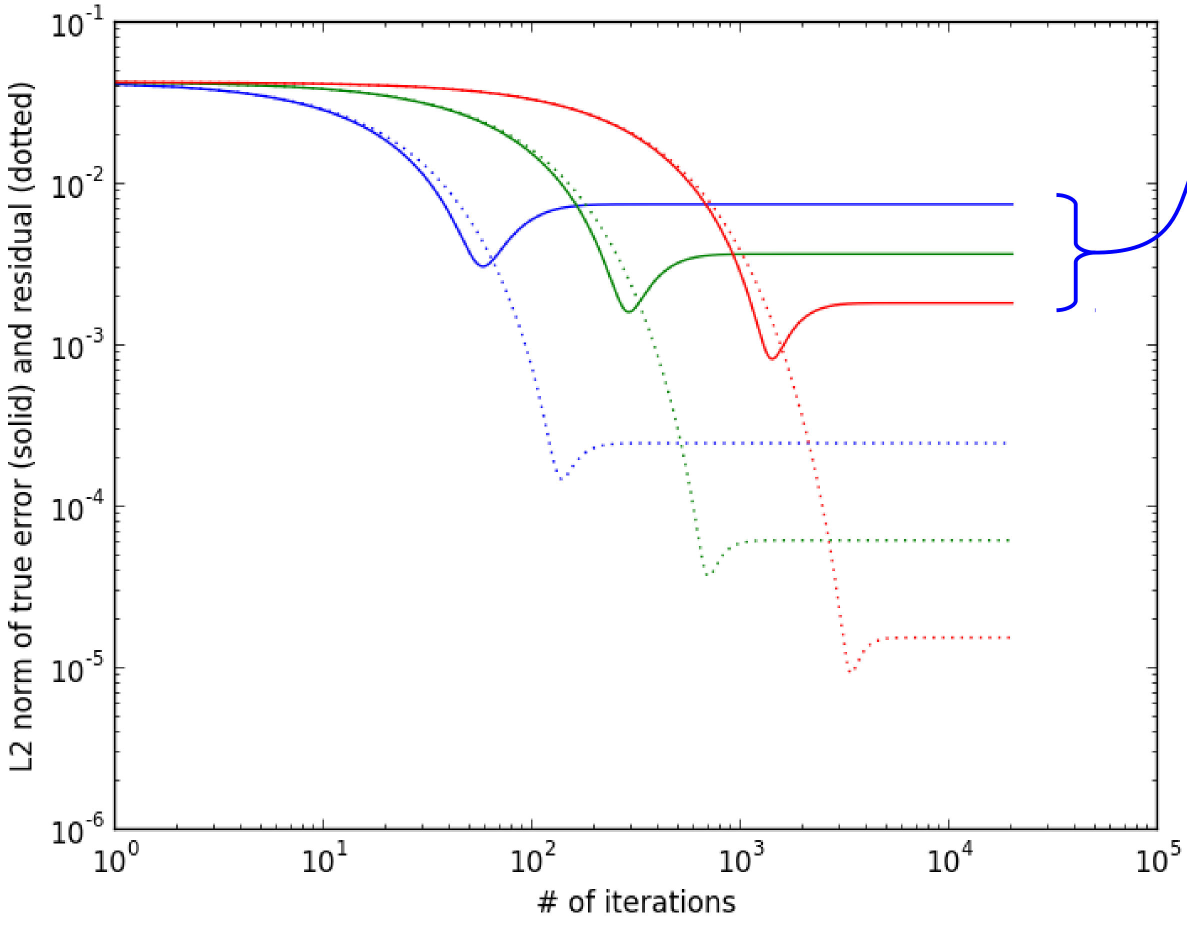
The residual error stalls only near machine epsilon—our solution satisfies our discrete equation "exactly"

# Convergence with Different Norms



# Effect of Boundary Conditions

- What if we didn't take into account the BCs properly?
  - i.e. Set the ghost cell value to the BC value instead of interpolate to the actual boundary



First-order accurate convergence with naive BCs

Look at the convergence!

# Behavior of Different Modes

- Consider Laplace's equation:

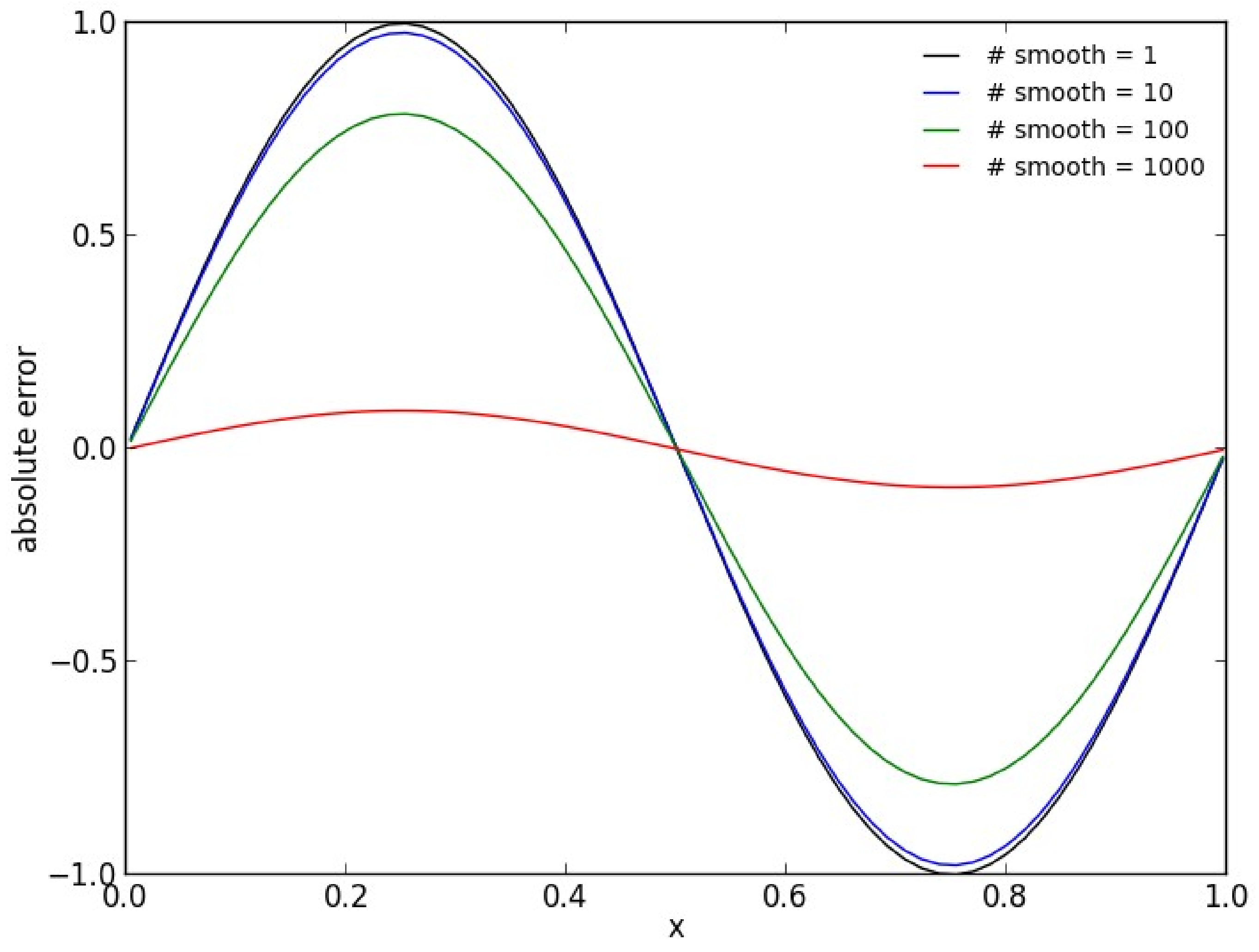
$$\nabla^2 \phi = 0\,, \qquad \phi(0) = 0,\ \phi(1) = 0$$

  - The solution is simply: $\phi(x) = 0$
- Pick a single mode sine wave as an initial guess:

$$\phi^{(0)}(x) = \sin(2\pi m x)$$

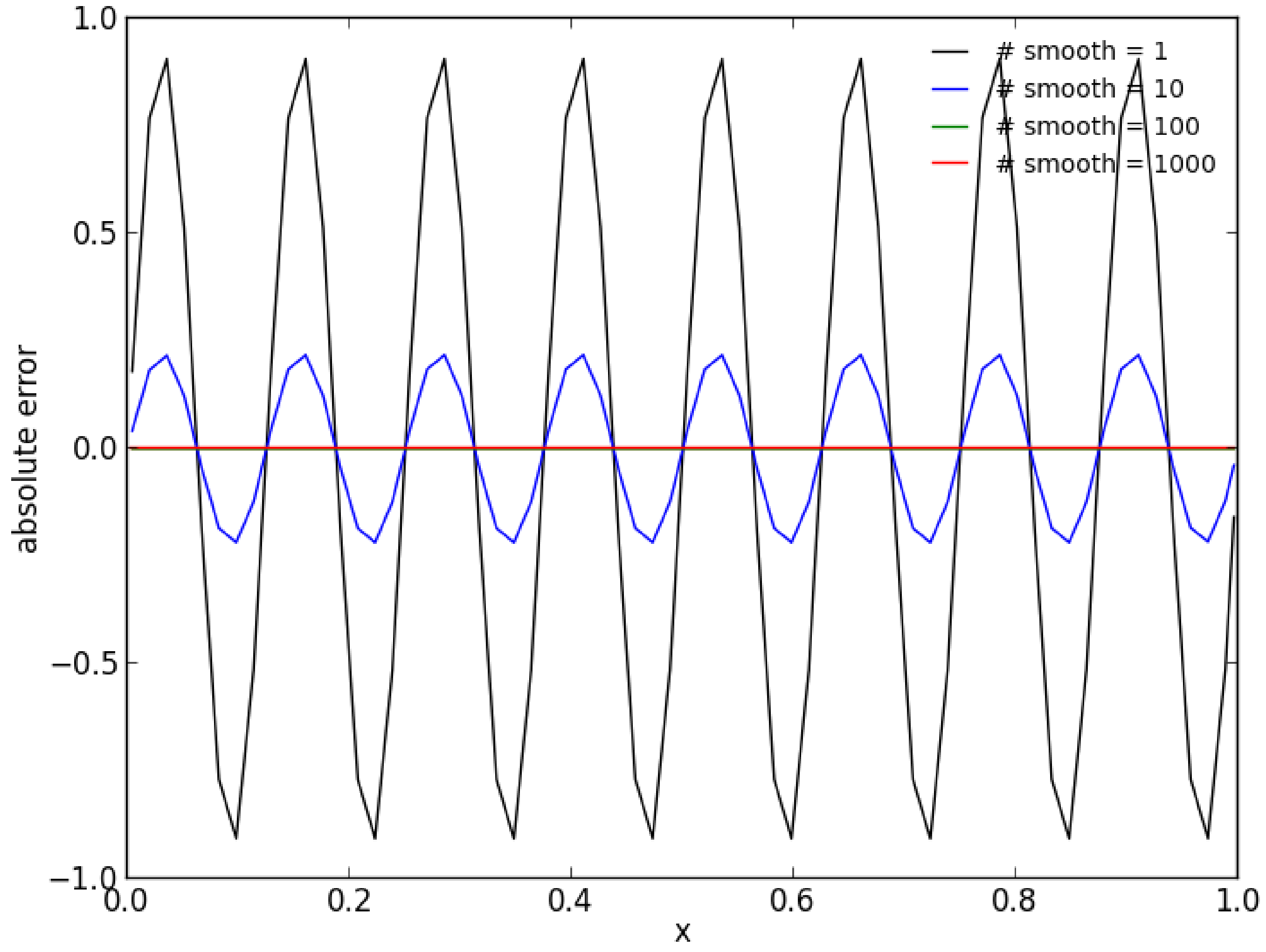
  - The error after X iterations is simply $\phi(x)$
- Let's look at how different modes behave

# M=1 Mode



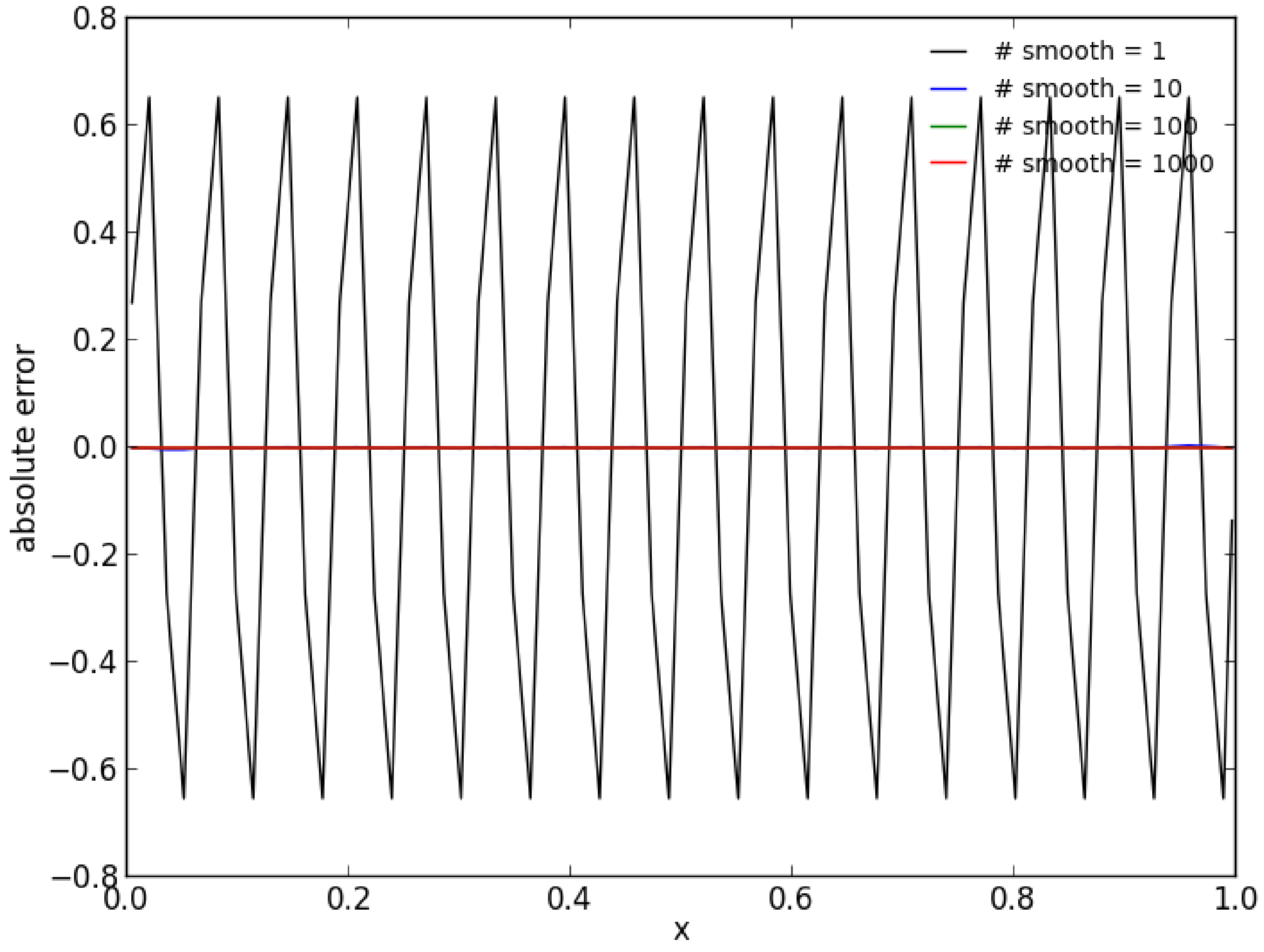
Initial guess after 1, 10, 100, and 1000 smoothings for m = 1 and 128 zones
Not much progress...

# M=8 Mode



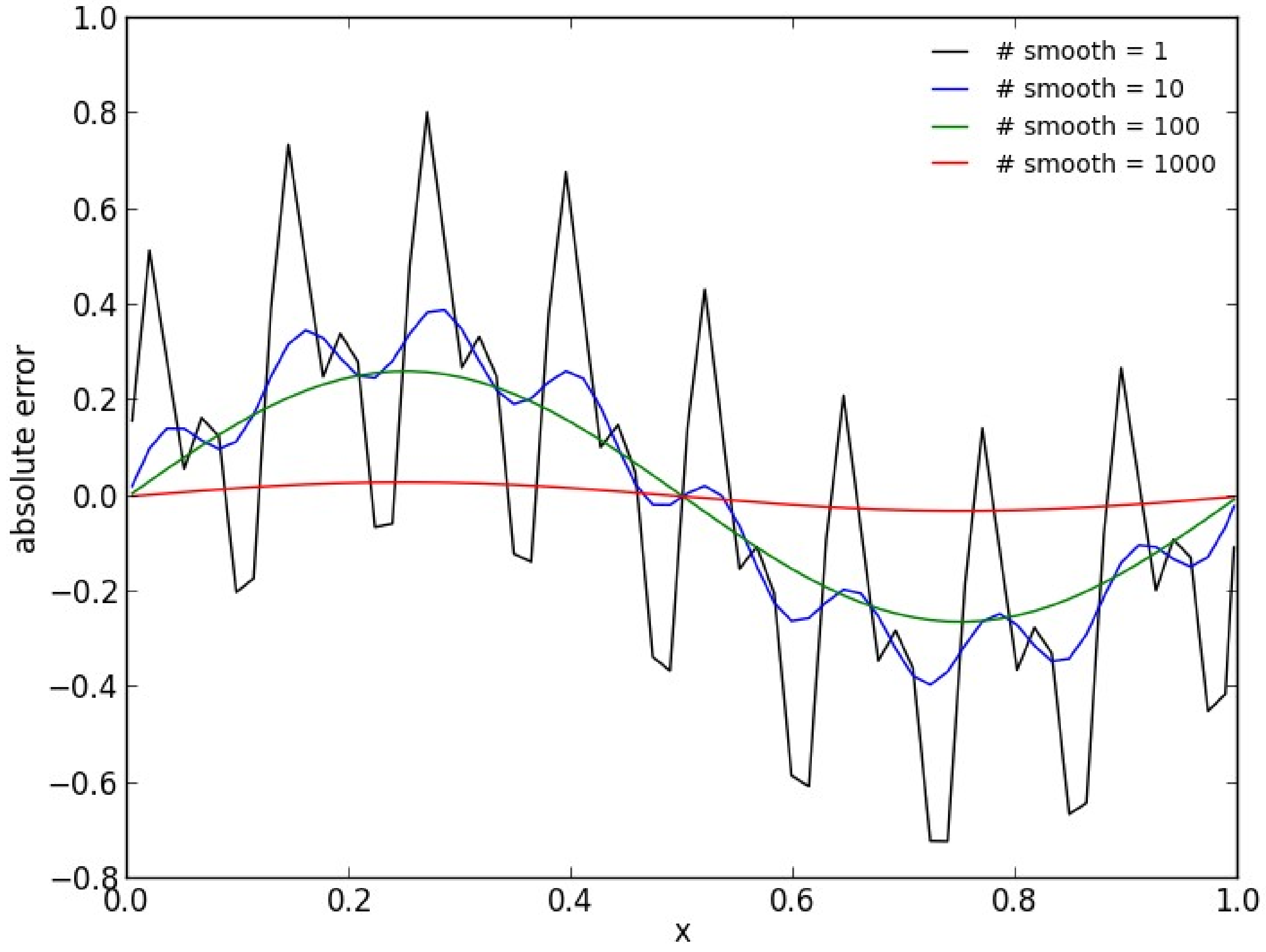
Initial guess after 1, 10, 100, and 1000 smoothings for m = 8 and 128 zones
Here we see that after 100 smoothings, the error is mostly gone

# M=16 Mode



Initial guess after 1, 10, 100, and 1000 smoothings for m = 16 and 128 zones
Now after 10 smoothings, the error is small

# Multiple Modes



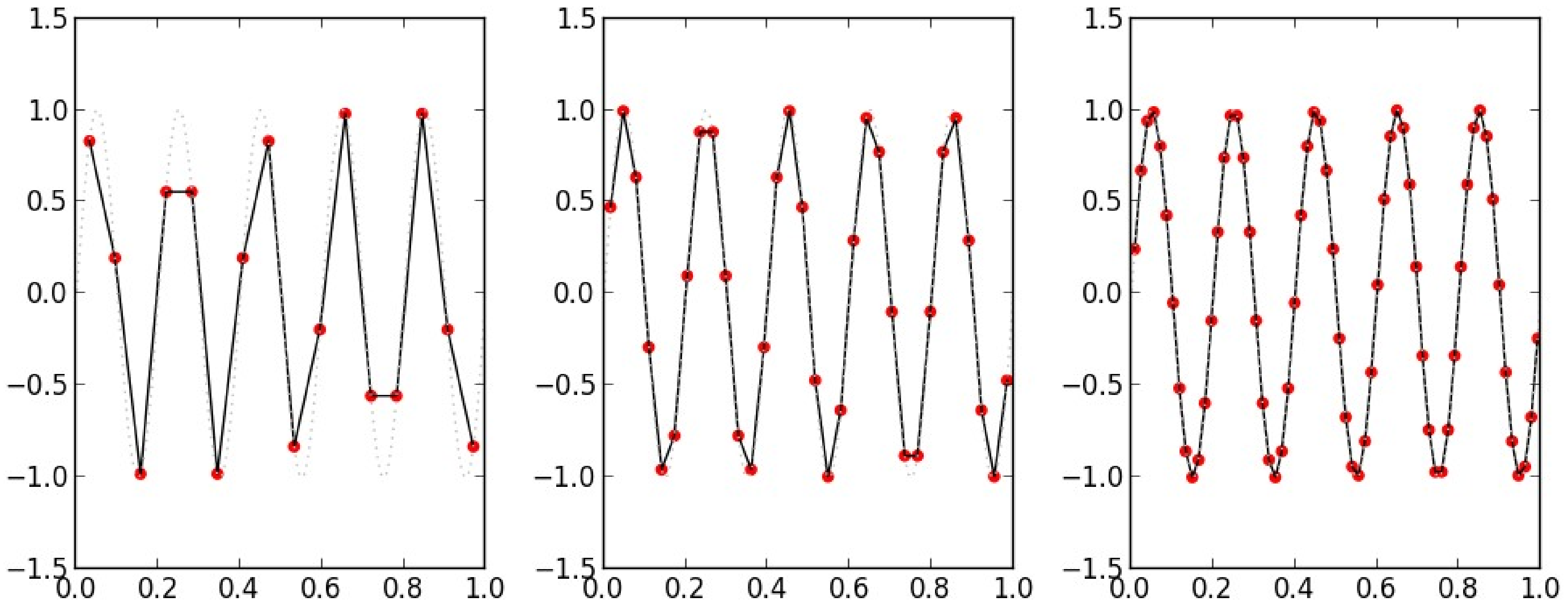
Initial guess after 1, 10, 100, and 1000 smoothings for an initial guess consisting of m=1, 8, and 16 modes (equally weighted) and 128 zones
Notice that the highest wavelength errors disappear fastest.

## Relaxation Observations

- Observe that the higher-frequency (shorter wavelength) errors smooth away fastest
  - Here we measure the wavelength in terms of number of zones
- Every zone is linked to every other zone
  - If an error has a wavelength of N zones, then N iterations are required to communicate across it
- Our PDE is linear
  - Each mode evolves independent of the others

# Coarsening

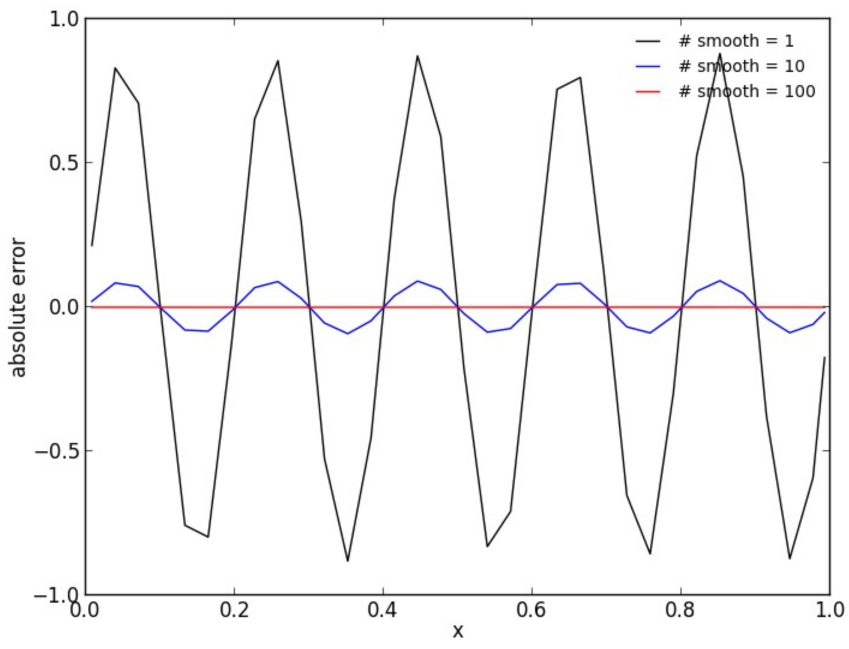
- On a coarser grid, long wavelength errors appear to have a shorter wavelength
- Consider m = 5 mode on a 64, 32, and 16 zone grid

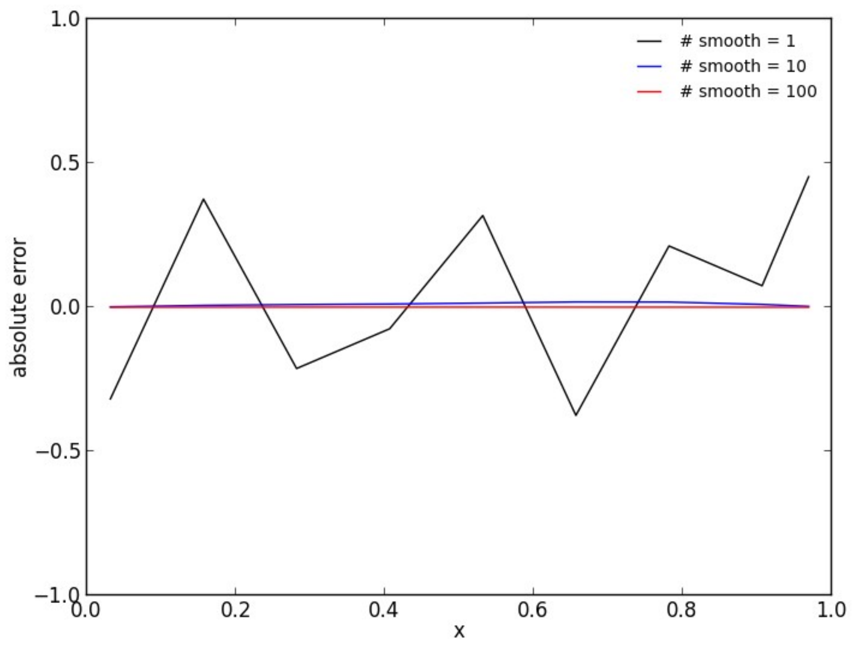


  - Notice that on a 64 zone mesh the error appears smooth, but on a 16 zone mesh, it is very oscillatory

# Coarsening

- Since the error appear higher frequency on the coarser grid, we expect that we will solve with fewer iterations

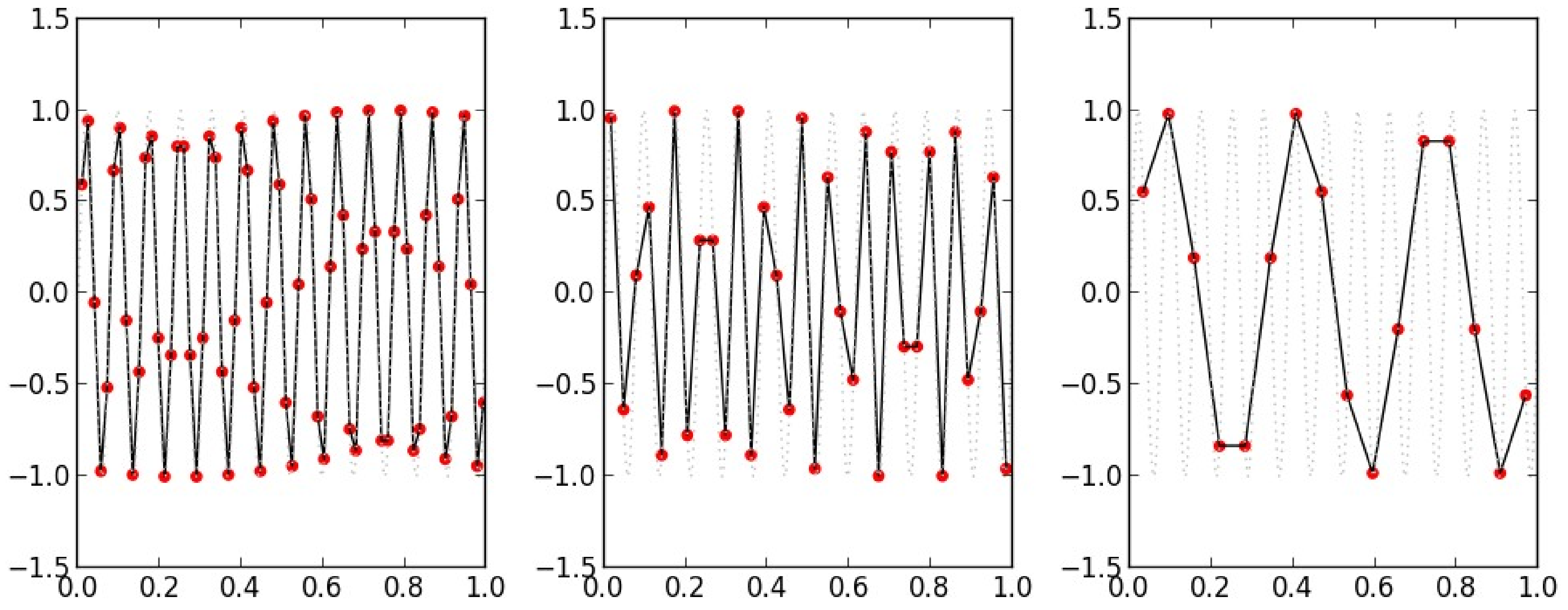




Error for m = 5 mode with 64 zones (left) and 16 zones (right) after 1, 10, and 100 iterations

# Aliasing

- However, if we coarsen a short wavelength error, it can appear to have a longer wavelength on the coarse grid—this is aliasing



m = 13 mode on a 64, 32, and 16 zone grid

## On To Multigrid

- Multigrid is a method to accelerate the convergence of relaxation
  - It eliminates the short wavelength errors on the original grid
  - Coarsens the problem and eliminates the formerly long wavelength errors on the new coarser grid
- Multigrid relies on a method to move the solution up and down a hierarchy of grids