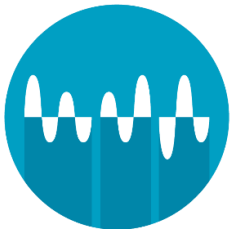


Biomedical Ultrasound Group

k-Wave short course – Part 2 Numerical methods

Bradley Treeby and Ben Cox

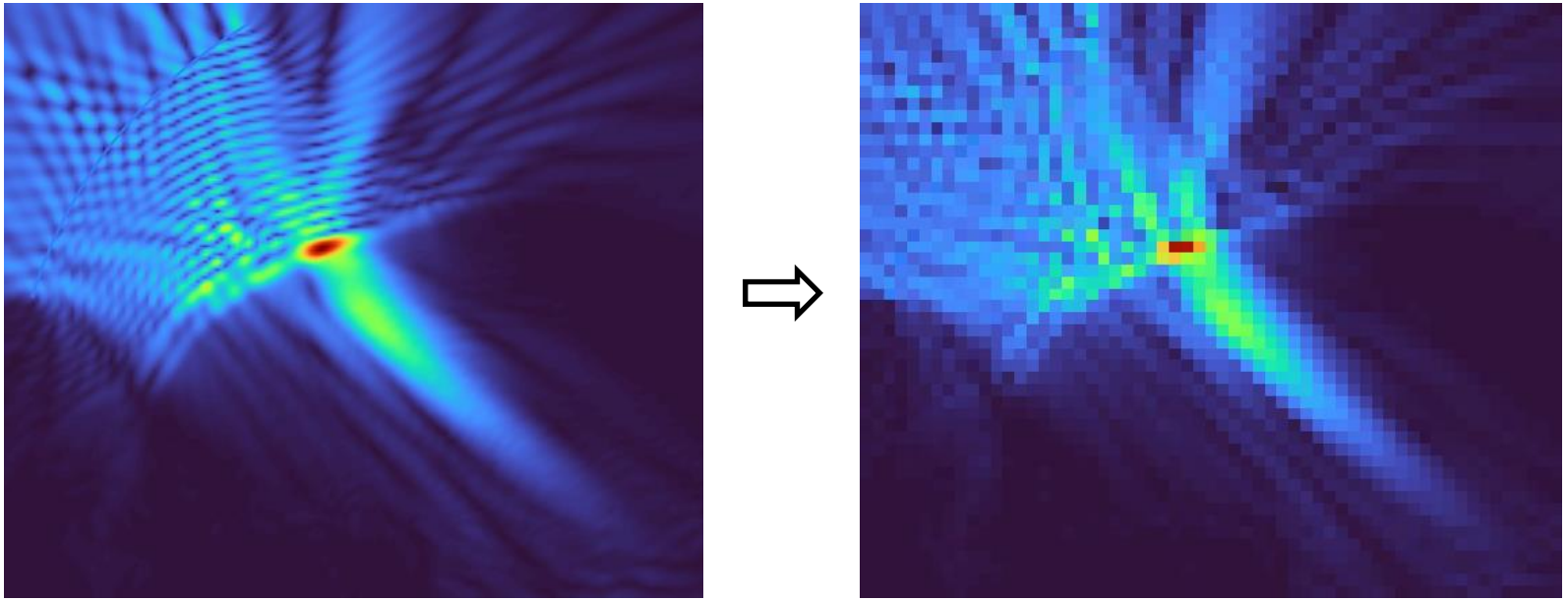
Biomedical Ultrasound Group (BUG)

Department of Medical Physics and Biomedical Engineering

University College London

Numerical methods

- Numerical methods (e.g., finite differences) provide a way to solve the wave equation when analytical solutions are not possible
- The general idea is to divide a continuous system into a (in this case, evenly distributed) mesh of grid points



Finite difference schemes

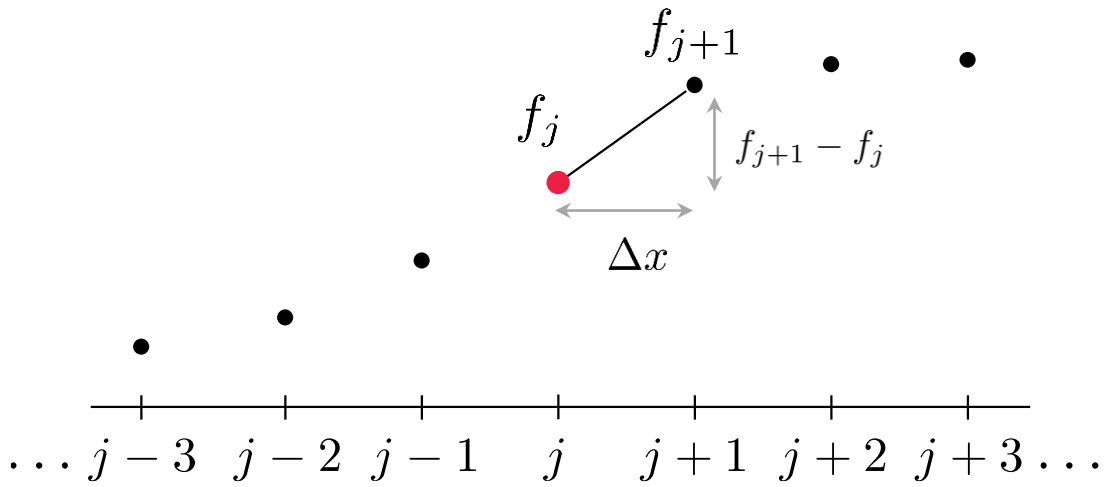
- The grid points represent the positions in space where the solution to the wave equation is obtained
- The continuous derivatives in the wave equation, e.g., $\partial^2/\partial t^2$ and ∇^2 are then replaced with discrete approximations, e.g., finite differences
- We are going to solve the 1D wave equation

$$\frac{\partial^2 p}{\partial t^2} = c_0^2 \frac{\partial^2 p}{\partial x^2}$$

- First with finite differences, and then with the pseudospectral time domain method used in k-Wave

Finite difference schemes

- Consider the calculation of $\frac{\partial f}{\partial x}$ at the grid point j

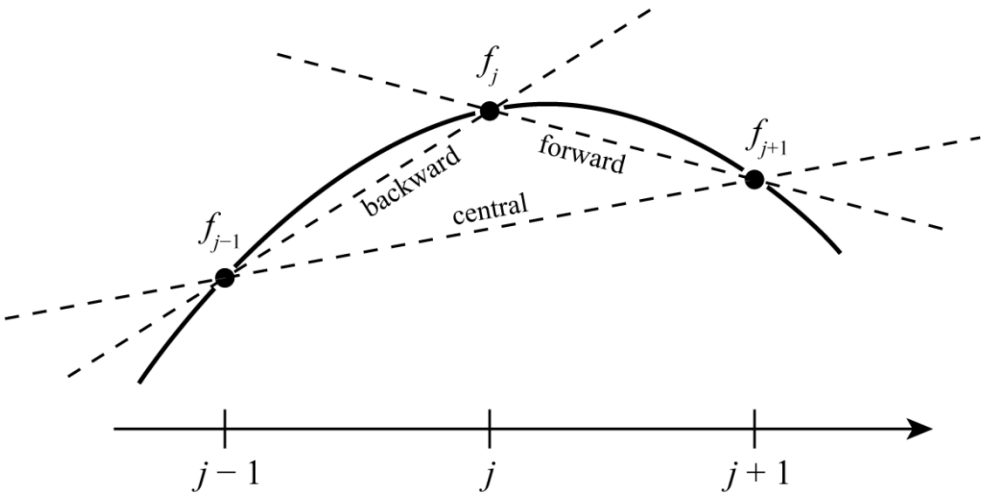


$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{f_{j+1} - f_j}{\Delta x}$$

- This is called a **first-order accurate forward-difference** scheme

Finite difference schemes

- Other approximations are also possible



forward difference

$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{f_{j+1} - f_j}{\Delta x}$$

backward difference

$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{f_j - f_{j-1}}{\Delta x}$$

central difference

$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{f_{j+1} - f_{j-1}}{2\Delta x}$$

- Higher order schemes using more grid points also exist

Finite difference schemes

- Similarly for second order derivatives

$$\left.\frac{\partial^2 f}{\partial x^2}\right|_j \approx \frac{f_{j-1} - 2f_j + f_{j+1}}{\Delta x^2}$$

- This is a **second-order accurate central difference scheme**
- The same thing holds for time derivatives

$$\left.\frac{\partial^2 f}{\partial t^2}\right|_n \approx \frac{f^{n-1} - 2f^n + f^{n+1}}{\Delta t^2}$$

- We're now ready to solve the wave equation!

Solving the wave equation

- We will use the notation p_j^n , where j is the spatial index, and n is the time index (this becomes $p_{i,j}^n$ in 2D)

$$\frac{\partial^2 p}{\partial t^2} = c_0^2 \frac{\partial^2 p}{\partial x^2}$$

- Replacing derivatives with finite differences

$$\frac{p_j^{n+1} - 2p_j^n + p_j^{n-1}}{\Delta t^2} = c_0^2 \frac{p_{j+1}^n - 2p_j^n + p_{j-1}^n}{\Delta x^2}$$

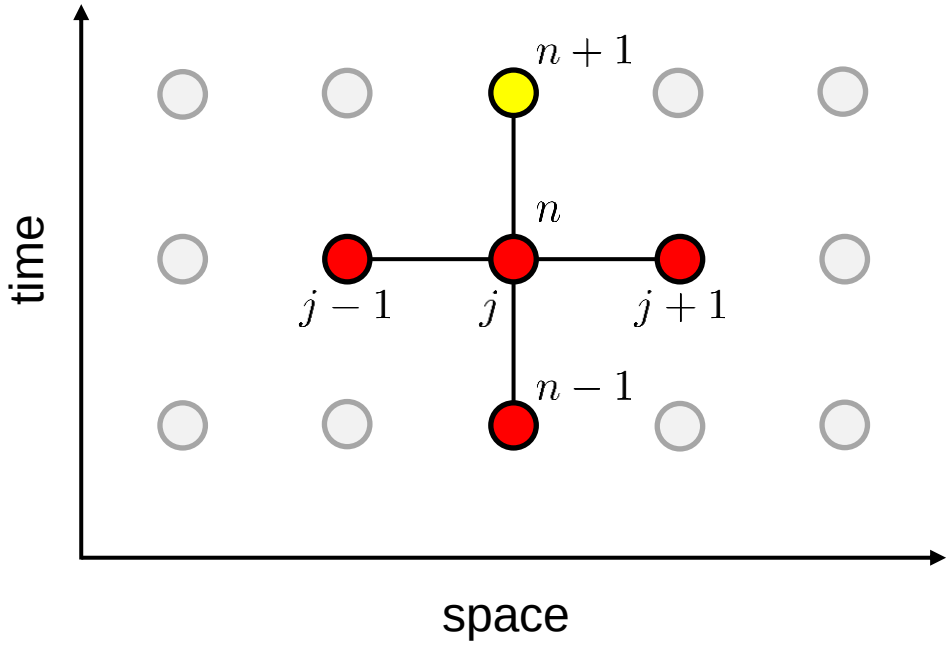
$$p_j^{n+1} = 2p_j^n - p_j^{n-1} + \left(\frac{c_0 \Delta t}{\Delta x}\right)^2 \left(p_{j+1}^n - 2p_j^n + p_{j-1}^n\right)$$

- We can use this to calculate the evolution of the pressure field over time

Finite difference stencil

- The new pressure value at each grid point depends on the previous two values in time, and the previous value either side in space

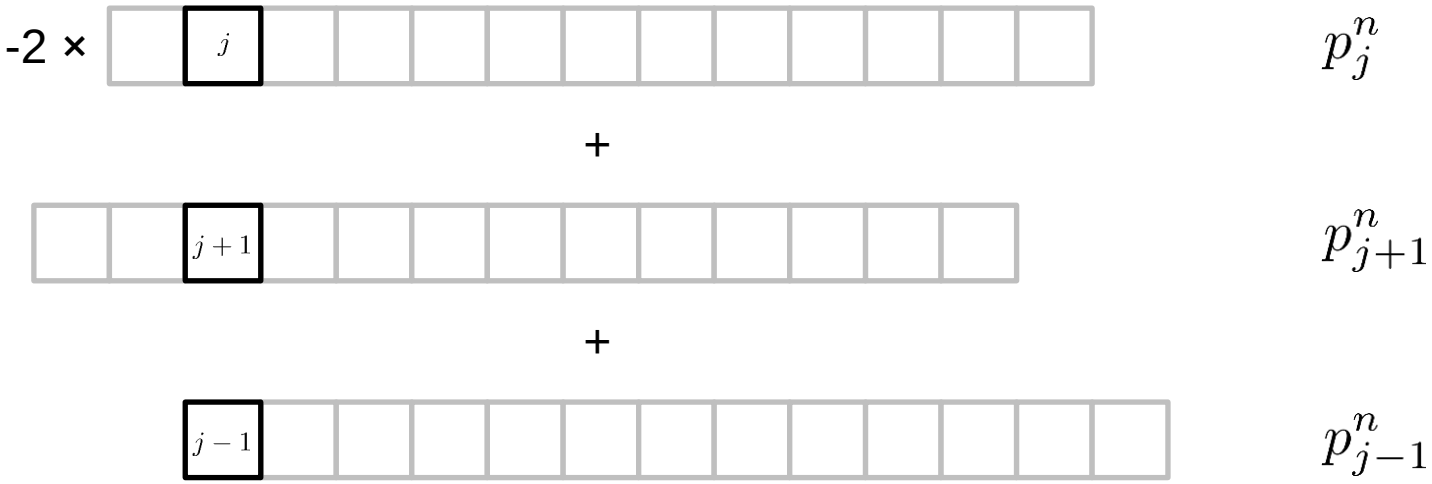
$$p_j^{n+1} = 2p_j^n - p_j^{n-1} + \left(\frac{c_0 \Delta t}{\Delta x}\right)^2 \left(p_{j+1}^n - 2p_j^n + p_{j-1}^n\right)$$



finite difference stencil

Spatial derivative

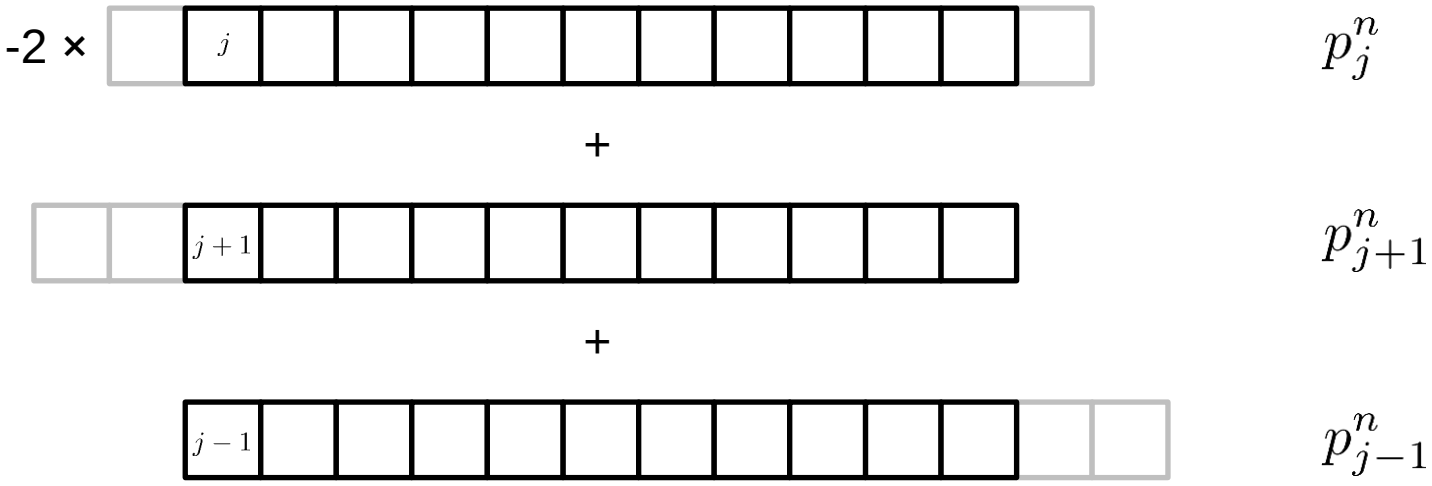
- If p is stored as a vector, we can calculate the spatial derivative at every point in a vectorised way



$$-2p_j^n + p_{j+1}^n + p_{j-1}^n$$

Spatial derivative

- If p is stored as a vector, we can calculate the spatial derivative at every point in a vectorised way



$$-2p_j^n + p_{j+1}^n + p_{j-1}^n$$

- In MATLAB code

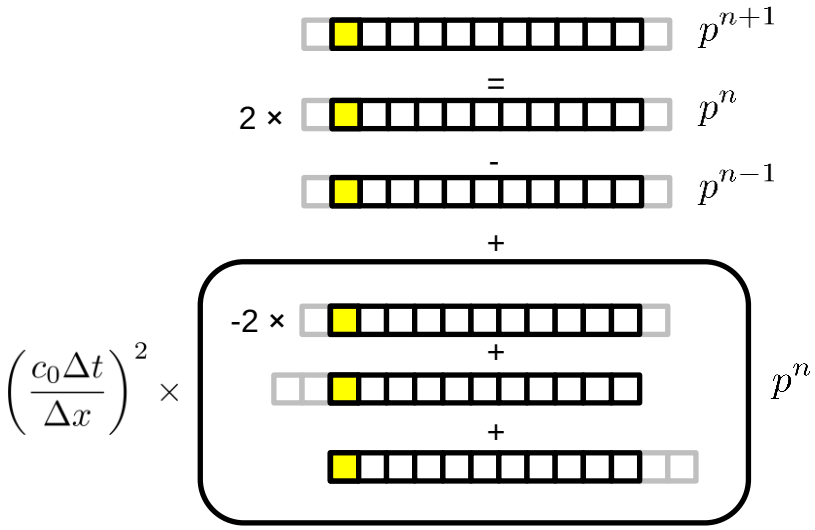
```
-2*p(2:end-1) + p(3:end) + p(1:end-2)
```

Complete loop

for n = 1 to Nt

(1) update p^{n+1}

$$p_j^{n+1} = 2p_j^n - p_j^{n-1} + \left(\frac{c_0 \Delta t}{\Delta x}\right)^2 \left(p_{j+1}^n - 2p_j^n + p_{j-1}^n\right)$$



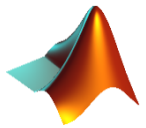
(2) copy $p^{n+1} \rightarrow p^n$ and $p^n \rightarrow p^{n-1}$

end

MATLAB code – FDTD

```
% define literals (hard coded numbers)
Nx  = 100;              % number of grid points
dx  = 1e-3;             % grid point spacing (m)
c0  = 1500;             % sound speed (m/s)
Nt  = 50;               % number of time steps
CFL = 0.5;              % Courant-Friedrichs-Lewy number

% set the time step based on the CFL number
dt = CFL * dx / c0;     % size of time step (s)

% set the initial pressure to be a Gaussian
p_n = exp(-( (1:Nx) - Nx/2).^2 / (2 * (Nx/30)^2));

% set pressure at time step (n - 1) to be equal to (n)
p_nm1 = p_n;

% set pressure at time step (n + 1) to be zero
p_np1 = zeros(size(p_n));

% calculate pressure in a loop
for n = 1:Nt

    % calculate the new value for the pressure at time step (n + 1)
    p_np1(2:end-1) = 2*p_n(2:end-1) - p_nm1(2:end-1) + ...
        (c0 * dt / dx)^2 * ( p_n(1:end-2) - 2*p_n(2:end-1) + p_n(3:end) );

    % copy the value for the pressure at time step (n) to (n - 1)
    p_nm1 = p_n;

    % copy the value for the pressure at time step (n + 1) to (n)
    p_n = p_np1;

end
```

time loop has just three lines of code!



wave_equation_1D_finite_differences.m

Convergence

- The finite difference scheme uses an *approximation* to the derivatives in the wave equation, e.g.,

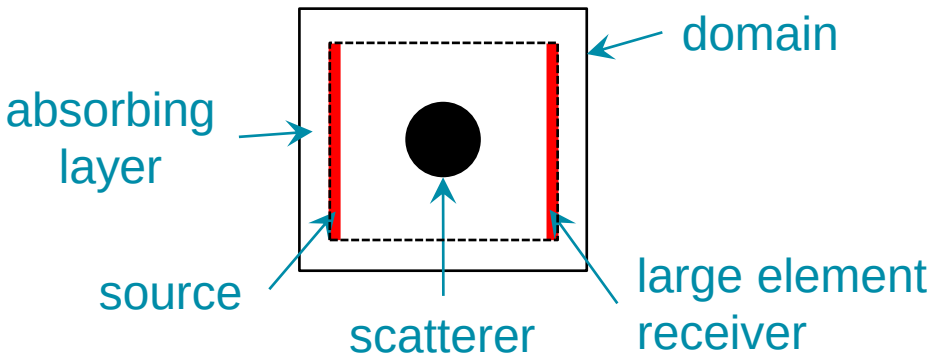
$$\frac{\partial^2 f(x)}{\partial x^2} \approx \frac{f(x - \Delta x) - 2f(x) + f(x + \Delta x)}{\Delta x^2}$$

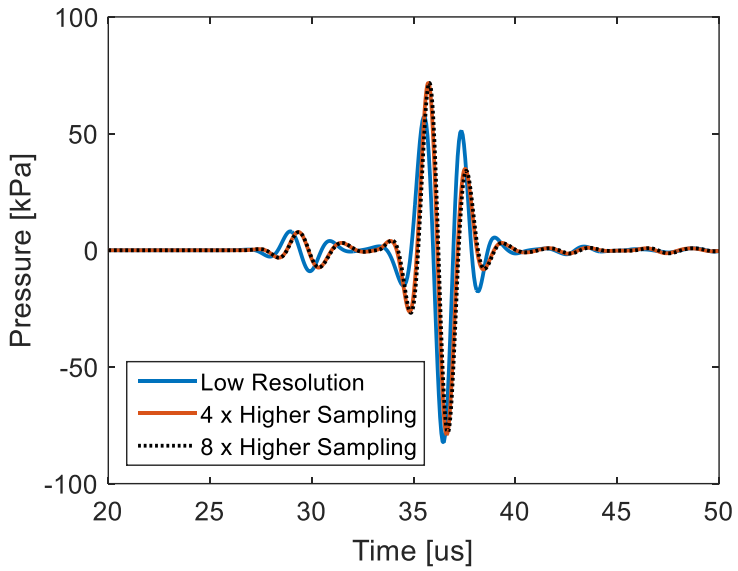
- How do we know when this is accurate?
- We can gradually decrease Δx and Δt until the simulation result converges (doesn't change any more)
- This works because

$$\lim_{\Delta x \to 0} \frac{f(x - \Delta x) - 2f(x) + f(x + \Delta x)}{\Delta x^2} = \frac{\partial^2 f(x)}{\partial x^2}$$

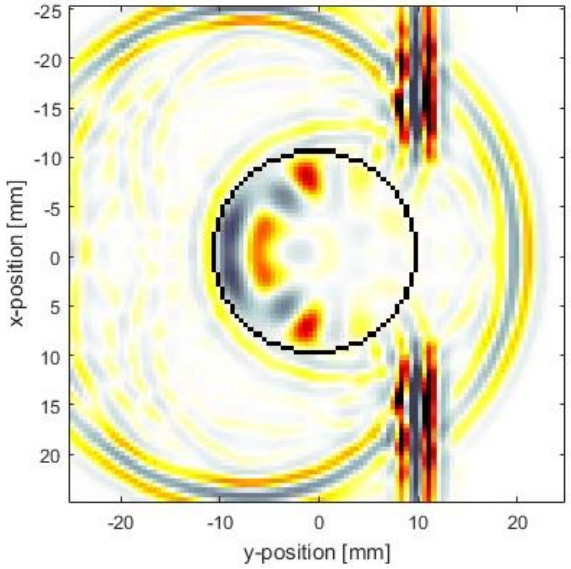
(i.e., the scheme is consistent)

Convergence Example

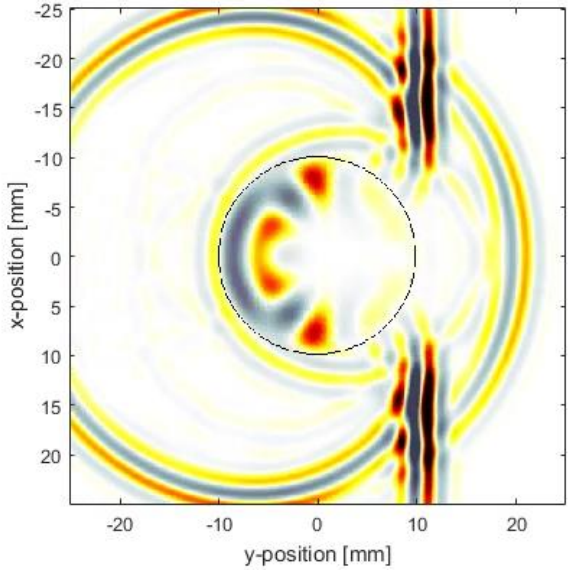




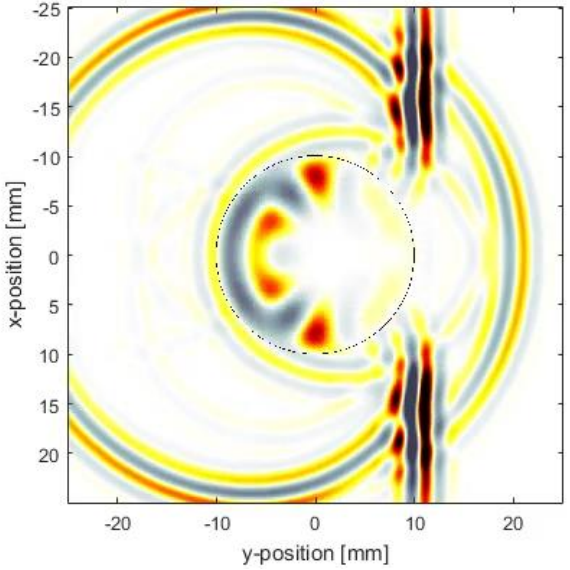
Low resolution simulation



4 x higher sampling



8 x higher sampling



Stability

- Many numerical methods have a restriction on the size of the time step before it becomes unstable
- For the finite difference scheme used here, in 1D the limit is

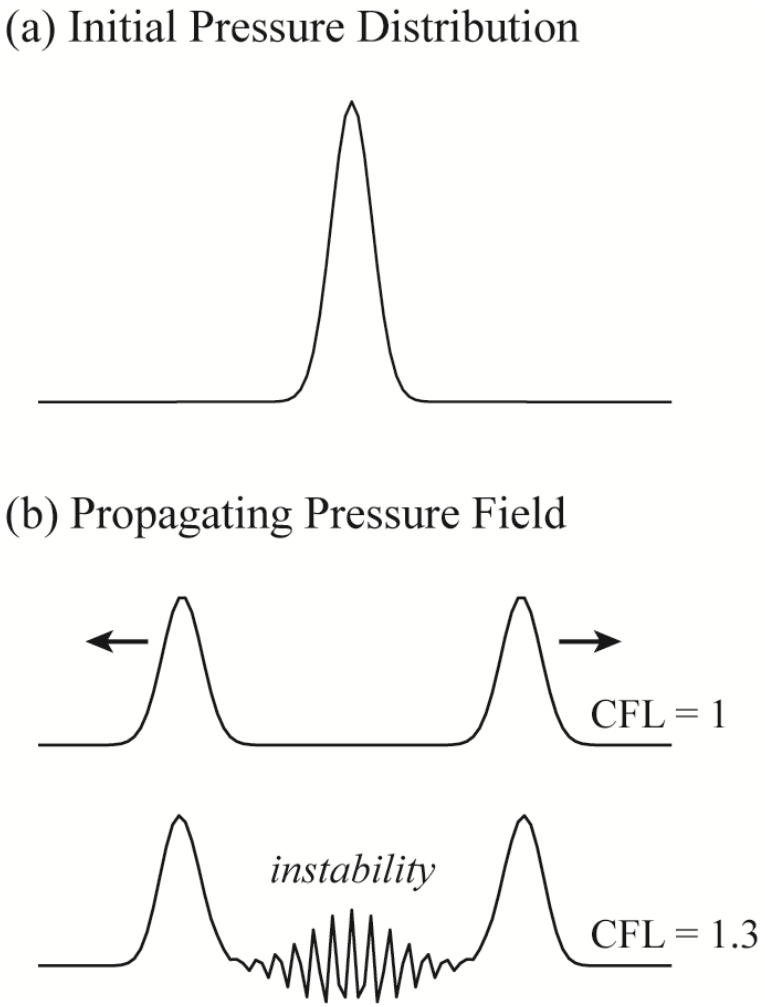
$$\frac{c_0 \Delta t}{\Delta x} \leq 1$$

- In 1D, this non-dimensional ratio is sometimes called the **Courant-Friedrichs-Lewy** (CFL) number, where

$$\mathrm{CFL} = \frac{c_0 \Delta t}{\Delta x}$$

- We will make frequent use of this expression

Example of instability in 1D



wave_equation_1D_finite_differences_unstable.m

Lax equivalence theorem

- When given a numerical model, it is often hard to prove convergence by looking at the equations, however:

A consistent and stable numerical scheme is always convergent

- This is the **Lax equivalence theorem** and links three concepts
  - **Consistency:** the numerical equations mathematically reduce to the continuous equations in the limit $\Delta x, \Delta t \to 0$
  - **Stability:** the error in the numerical solution doesn't grow without bound
  - **Convergence:** the numerical solution approaches the correct answer as Δx and Δt are reduced

Numerical dispersion

- In practice, we must use a finite Δx and Δt
- How can we can analyse the numerical errors introduced by the discretisation?

- You can consider the errors in two ways:

1. Finding a slightly incorrect solution to the correct equation

2. Finding the correct solution to a slightly incorrect equation

Numerical dispersion – FDTD

- Consider solution to the 1D wave equation using second-order accurate central differences

$$\frac{\partial^2 p}{\partial t^2} = c_0^2 \frac{\partial^2 p}{\partial x^2} \quad \rightarrow \quad \frac{p_j^{n+1} - 2p_j^n + p_j^{n-1}}{\Delta t^2} = c_0^2 \frac{p_{j+1}^n - 2p_j^n + p_{j-1}^n}{\Delta x^2}$$

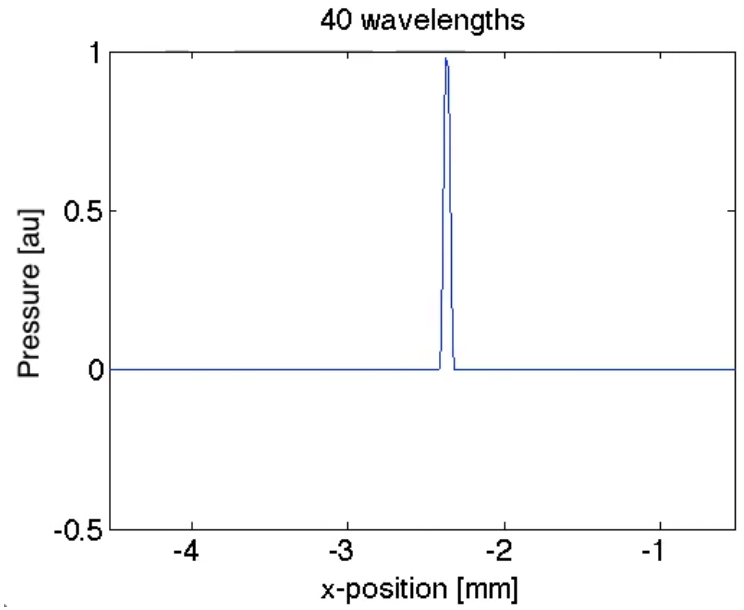
- Because of the truncation error in the FD approximation, the equation solved exactly is actually

$$\frac{\partial^2 p}{\partial t^2} = c_0(k)^2 \frac{\partial^2 p}{\partial x^2} \qquad \text{where} \qquad c(k) = \frac{\overbrace{\text{sinc}(k\Delta x/2)}^{\text{error from space discretisation}}}{\underbrace{\text{sinc}(c_0 k \Delta t/2)}_{\text{error from time discretisation}}} c_0$$

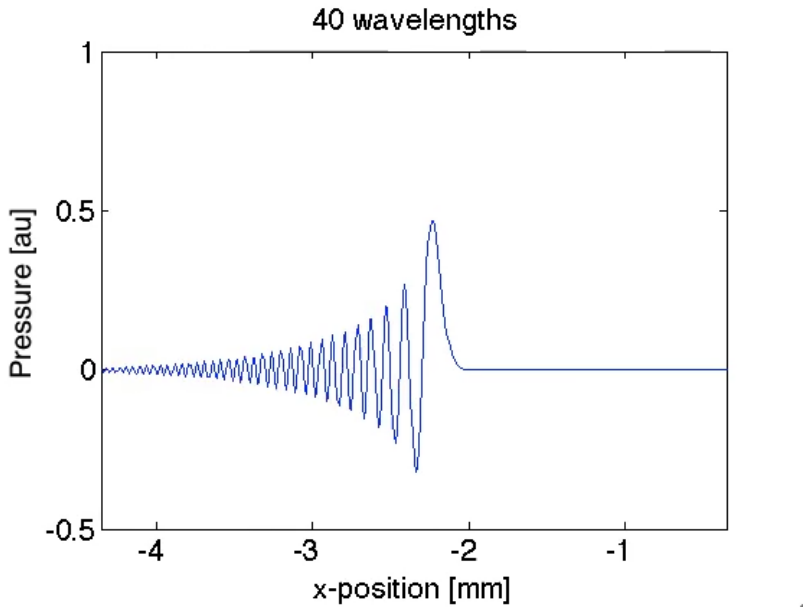
- Results in a **frequency dependent sound speed**

Numerical dispersion

- Propagation of a broadband pulse $\sin^3(2\pi f_s t)$ with 10 grid points per wavelength



Exact Solution

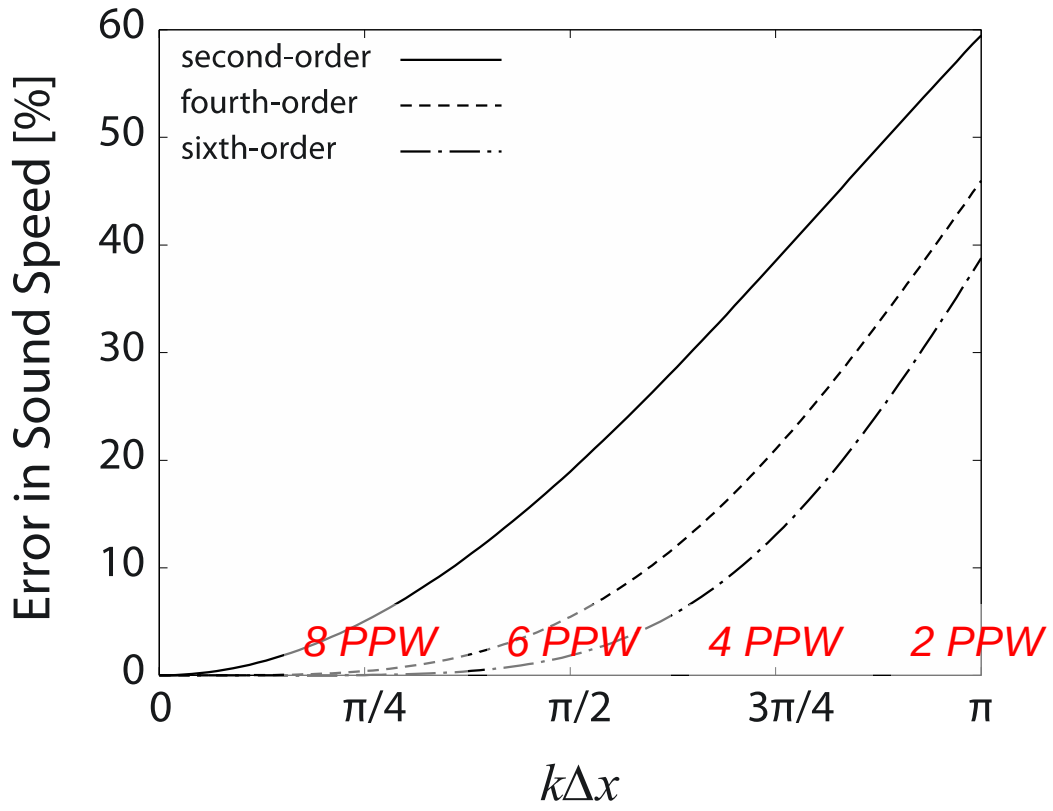


Finite Difference Solution

- Wave changes shape because of **numerical dispersion**

Numerical dispersion

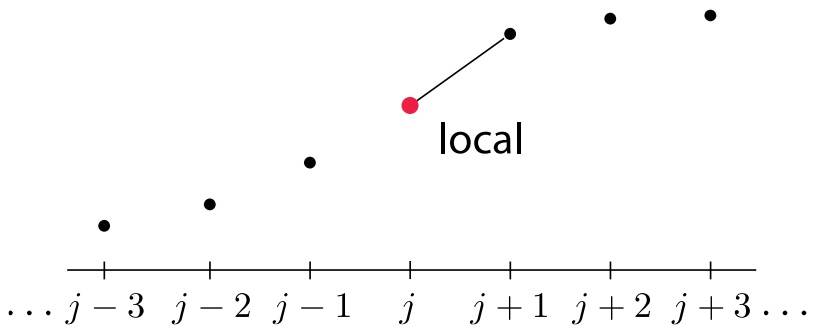
- Error in sound speed due to spatial FD discretisation:



- Errors **accumulate**, so larger grids need more PPW

How can we avoid numerical dispersion?

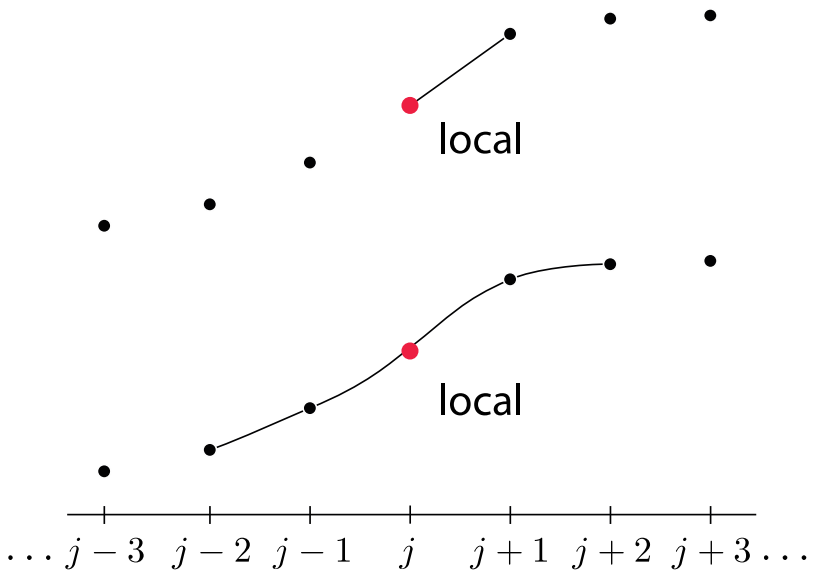
- Consider the calculation of $\frac{\partial f}{\partial x}$ at the grid point j



$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{f_{j+1} - f_j}{\Delta x}$$

How can we avoid numerical dispersion?

- Consider the calculation of $\frac{\partial f}{\partial x}$ at the grid point j

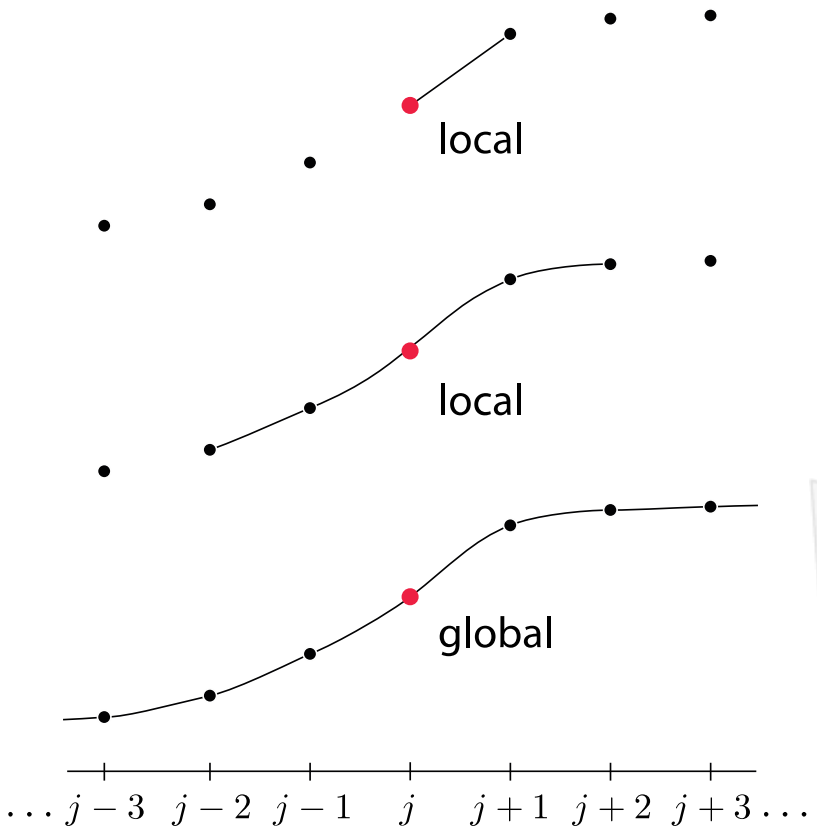


$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{f_{j+1} - f_j}{\Delta x}$$

$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{\frac{1}{12} f_{j-2} - \frac{2}{3} f_{j-1} + \frac{2}{3} f_{j+1} - \frac{1}{12} f_{j+2}}{\Delta x}$$

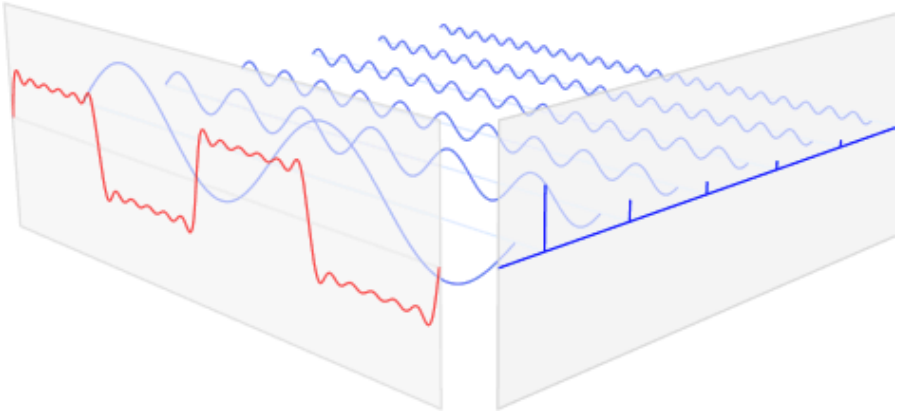
How can we avoid numerical dispersion?

- Consider the calculation of $\frac{\partial f}{\partial x}$ at the grid point j



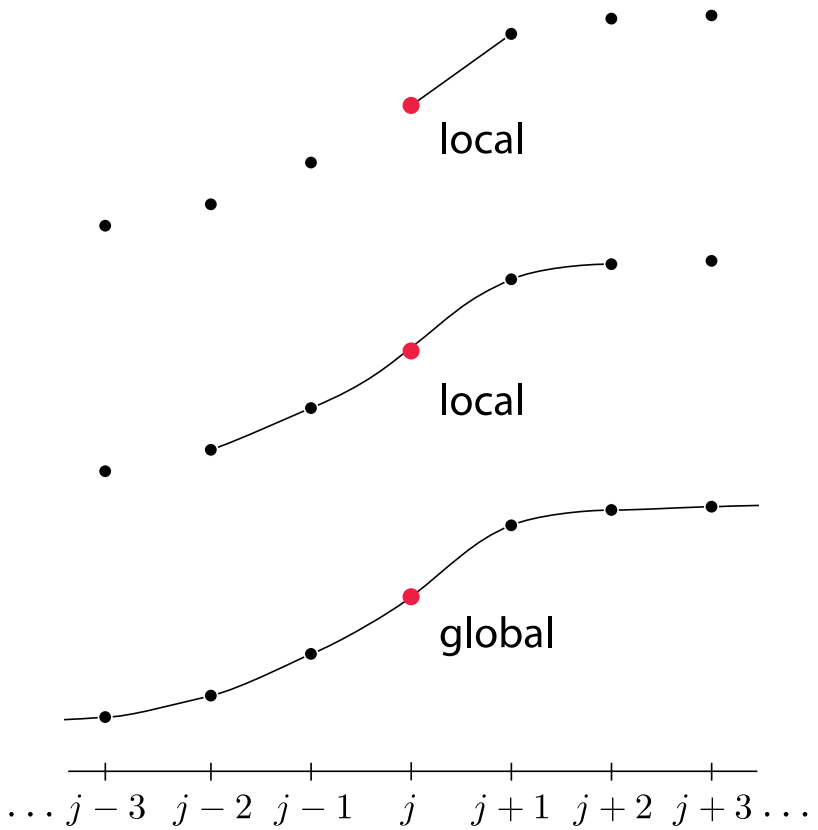
$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{f_{j+1} - f_j}{\Delta x}$$

$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{\frac{1}{12} f_{j-2} - \frac{2}{3} f_{j-1} + \frac{2}{3} f_{j+1} - \frac{1}{12} f_{j+2}}{\Delta x}$$



How can we avoid numerical dispersion?

- Consider the calculation of $\frac{\partial f}{\partial x}$ at the grid point j



$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{f_{j+1} - f_j}{\Delta x}$$

$$\left.\frac{\partial f}{\partial x}\right|_j \approx \frac{\frac{1}{12} f_{j-2} - \frac{2}{3} f_{j-1} + \frac{2}{3} f_{j+1} - \frac{1}{12} f_{j+2}}{\Delta x}$$

$$\frac{\partial f}{\partial x} \approx \mathcal{F}^{-1}\left\{ik_x \mathcal{F}\left\{f\right\}\right\}$$

- Eliminates dispersion from spatial gradient

Fourier collocation spectral method

- Consider the Fourier transform pair

$$\hat{f}(k_x) = \frac{1}{2\pi} \int f(x) e^{-ik_x x}\, dx = \mathcal{F}\{f(x)\}$$

$$f(x) = \int \hat{f}(k_x) e^{ik_x x}\, dx = \mathcal{F}^{-1}\left\{\hat{f}(k_x)\right\}$$

- The Fourier transform of a derivative is then

$$\frac{\partial}{\partial x} f(x) = \int (ik_x) \hat{f}(k_x) e^{ik_x x}\, dx = \mathcal{F}^{-1}\{ik_x \mathcal{F}\{f(x)\}\}$$

Fourier collocation spectral method

- Alternatively, express function as a sum of sinusoids

$$f(x) = \sum_{k_x=-\infty}^{\infty} A_k e^{ik_x x} = \sum_{k_x=-\infty}^{\infty} A_k \left(\cos(k_x x) + i \sin(k_x x)\right)$$

- Differentiate each sinusoid

$$\frac{\partial}{\partial x} f(x) = \sum_{k_x=-\infty}^{\infty} ik_x A_k e^{ik_x x}$$

- Using the Fourier transform to calculate the amplitudes

$$\frac{\partial}{\partial x} f(x) = \mathcal{F}^{-1}\left\{ik_x \mathcal{F}\left\{f(x)\right\}\right\}$$

Fourier collocation spectral method

- Higher order derivatives are calculated in the same way

$$\frac{\partial}{\partial x} f(x) = \mathcal{F}^{-1}\left\{ik_x \mathcal{F}\left\{f(x)\right\}\right\}$$

$$\frac{\partial^2}{\partial x^2} f(x) = \mathcal{F}^{-1}\left\{-k_x^2 \mathcal{F}\left\{f(x)\right\}\right\}$$

$$\nabla^2 f(x) = \mathcal{F}^{-1}\left\{-k^2 \mathcal{F}\left\{f(x)\right\}\right\}$$

$$(-\nabla^2)^a f(x) = \mathcal{F}^{-1}\left\{k^{2a} \mathcal{F}\left\{f(x)\right\}\right\}$$

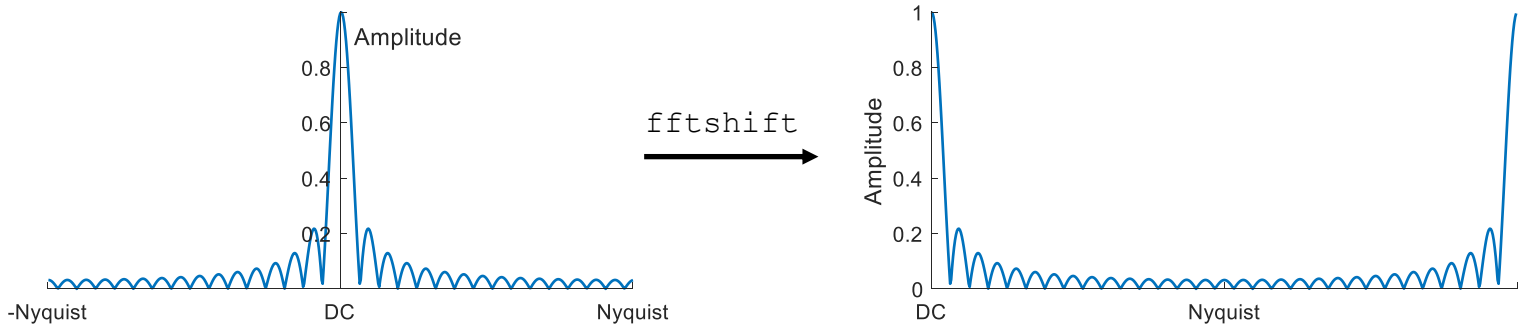
where $k^2 = \mathbf{k} \cdot \mathbf{k} = k_x^2 + k_y^2 + k_z^2$

Fourier collocation spectral method

- In the discrete case, the fast Fourier transform is used, and the wavenumbers are given by

$$k_x = \begin{cases} \left[-\frac{\mathrm{N}_x}{2}, -\frac{\mathrm{N}_x}{2}+1, \ldots, \frac{\mathrm{N}_x}{2}-1\right] \frac{2\pi}{\Delta x\, \mathrm{N}_x} & \text{if } \mathrm{N}_x \text{ is even} \\ \left[-\frac{(\mathrm{N}_x-1)}{2}, -\frac{(\mathrm{N}_x-1)}{2}+1, \ldots, \frac{(\mathrm{N}_x-1)}{2}\right] \frac{2\pi}{\Delta x\, \mathrm{N}_x} & \text{if } \mathrm{N}_x \text{ is odd} \end{cases}$$

- Note, in MATLAB (which uses FFTW), the wavenumbers or frequency components are ordered starting from 0!



Solving the wave equation

- We will use the notation p^n, where n is the time index (all spatial positions are solved at once)

$$\frac{\partial^2 p}{\partial t^2} = c_0^2 \frac{\partial^2 p}{\partial x^2}$$

- Replacing derivatives

$$\frac{p^{n+1} - 2p^n + p^{n-1}}{\Delta t^2} = c_0^2 \mathcal{F}^{-1}\left\{-k_x^2 \mathcal{F}\left\{p^n\right\})\right\}$$

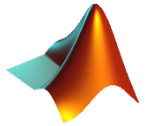
$$p^{n+1} = 2p^n - p^{n-1} + \Delta t^2 c_0^2 \mathcal{F}^{-1}\left\{-k_x^2 \mathcal{F}\left\{p^n\right\}\right\}$$

- We can use this to calculate the evolution of the pressure field over time

MATLAB code – PSTD

```
% set the initial pressure to be a Gaussian
p_n = exp(-( (1:Nx) - Nx/2).^2 / (2 * (Nx/30)^2));

% set pressure at time step (n - 1) to be equal to (n)
p_nm1 = p_n;

% set pressure at time step (n + 1) to be zero
p_np1 = zeros(size(p_n));

% define the set of wavenumbers
kx = (-pi/dx):2*pi/(dx*Nx):(pi/dx - 2*pi/(dx*Nx));

% shift the order of the wavenumbers (the FFT in MATLAB assumes the
% frequency axis starts at DC)
kx = ifftshift(kx);

% calculate pressure in a loop
for n = 1:Nt

    % calculate the new value for the pressure at time step (n + 1)
    p_np1 = 2*p_n - p_nm1 + dt^2 * c0.^2 * real(ifft( -k.^2 .* fft(p_n)));

    % copy the value for the pressure at time step (n) to (n - 1)
    p_nm1 = p_n;

    % copy the value for the pressure at time step (n + 1) to (n)
    p_n = p_np1;

end
```

spatial derivative calculation



wave_equation_1D_pstd.m

Numerical dispersion – PSTD

- 1D wave equation solved using the pseudospectral time domain method (PSTD)

$$\frac{\partial^2 p}{\partial t^2} = c_0^2 \frac{\partial^2 p}{\partial x^2} \quad \rightarrow \quad \frac{p^{n+1} - 2p^n + p^{n-1}}{\Delta t^2} = c_0^2 \mathcal{F}^{-1}\left\{-k_x^2 \mathcal{F}\left\{p^n\right\}\right\}$$

- Because of the truncation error in the FD approximation, the equation solved exactly is actually

$$\frac{\partial^2 p}{\partial t^2} = c_0(k)^2 \frac{\partial^2 p}{\partial x^2} \qquad \text{where} \qquad c(k) = \underbrace{\frac{1}{\operatorname{sinc}\left(c_0 k \Delta t/2\right)}}_{\text{error from time discretisation}} c_0$$

- Error from space discretization is eliminated!

k-space pseudospectral method

- The *k*-space pseudospectral method applies additional correction $\kappa = \mathrm{sinc}\left(c_0 k \Delta t/2\right)$ in the spatial frequency domain to reduce dispersion from temporal gradient
- Example: 1D homogeneous linear wave equation

$$\frac{\partial^2 p}{\partial t^2} = c_0^2 \frac{\partial^2 p}{\partial x^2}$$

- *k*-space solution is given by

k-space operator

$$\frac{p^{n+1} - 2p^n + p^{n-1}}{\Delta t^2} = c_0^2 \mathcal{F}^{-1}\left\{-\kappa^2 k_x^2 \mathcal{F}\left\{p^n\right\}\right\}$$

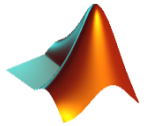
- Solution is exact and unconditionally stable

MATLAB code – k-space PSTD

```
% define the set of wavenumbers
kx = (-pi/dx):2*pi/(dx*Nx):(pi/dx - 2*pi/(dx*Nx));

% shift the order of the wavenumbers (the FFT in MATLAB assumes the
% frequency axis starts at DC)
kx = ifftshift(kx);

% compute the k-space operator
k = abs(kx);
kappa = sinc(c0 * k * dt / 2);

% calculate pressure in a loop
for n = 1:Nt

    % calculate the new value for the pressure at time step (n + 1)
    p_np1 = 2*p_n - p_nm1 ...
        + dt^2 * c0.^2 * real(ifft( -kappa.^2 .* k.^2 .* fft(p_n)));

    % copy the value for the pressure at time step (n) to (n - 1)
    p_nm1 = p_n;

    % copy the value for the pressure at time step (n + 1) to (n)
    p_n = p_np1;

end
```

NOTE: MATLAB sinc function in the signal processing toolbox has a non-standard definition

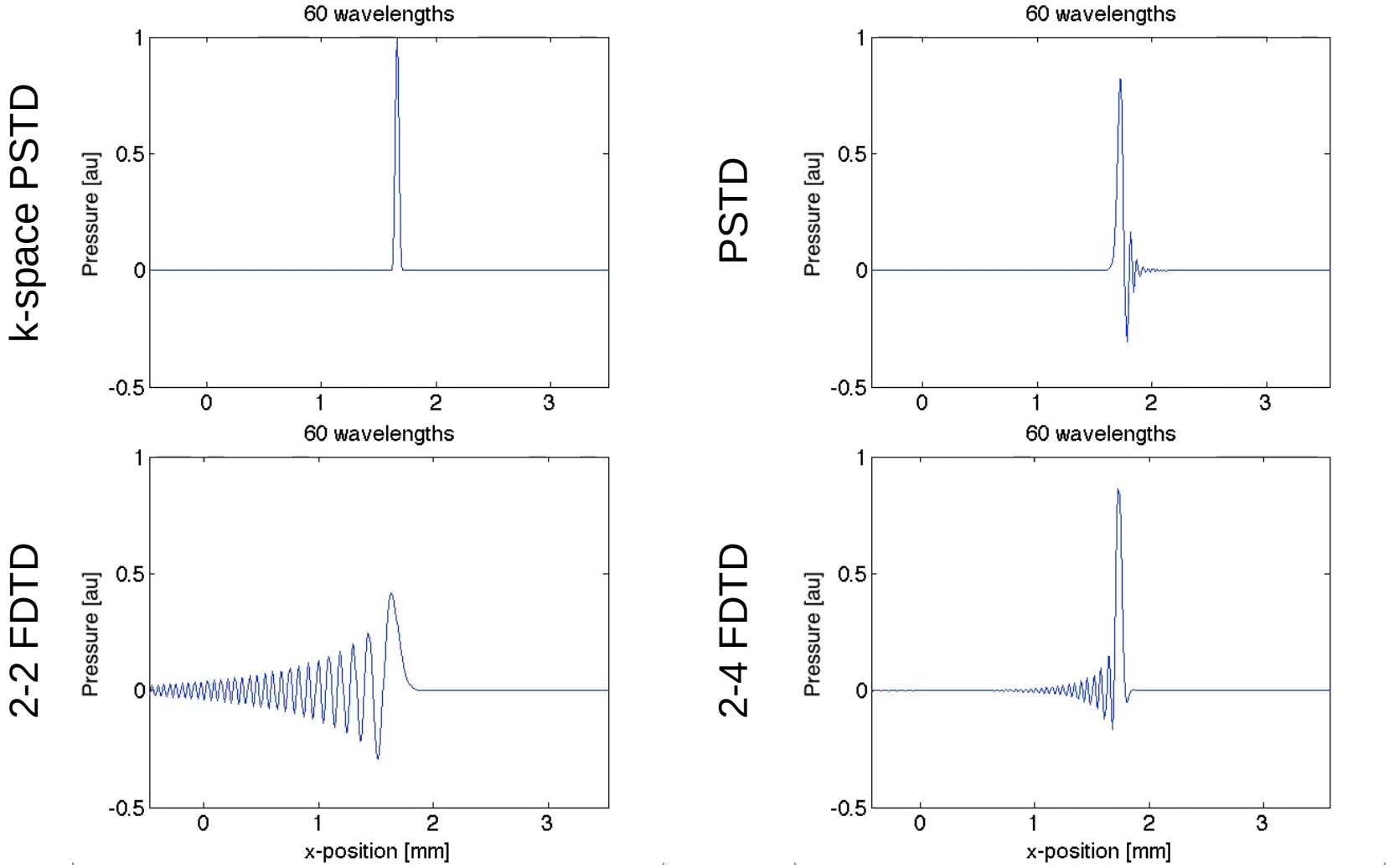
k-space correction



wave_equation_1D_kspace.m

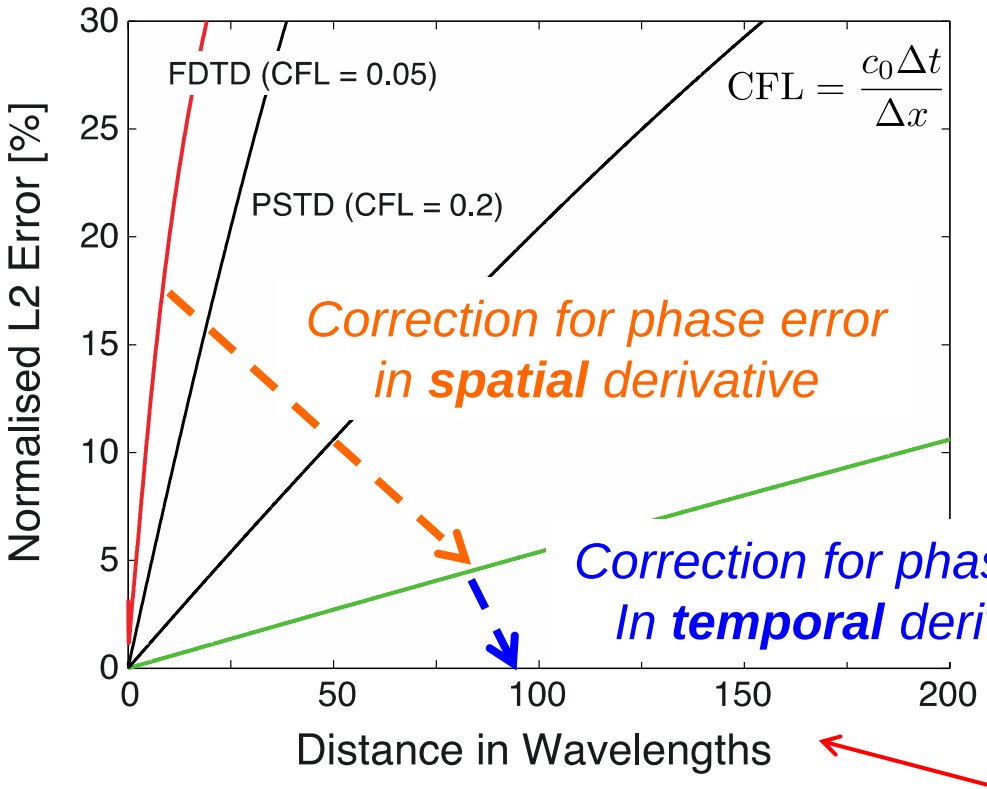
Numerical dispersion comparison

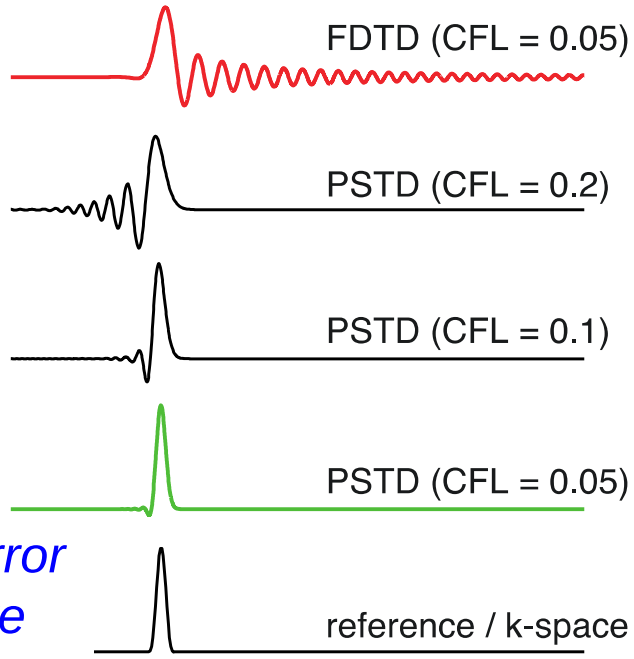
- Propagation of a broadband pulse



Numerical dispersion comparison

- Propagation of a pulse $\sin^3(2\pi f_s t)$ defined by $\sin^3(2\pi f_s t)$ in a linear, lossless medium using 20 PPW

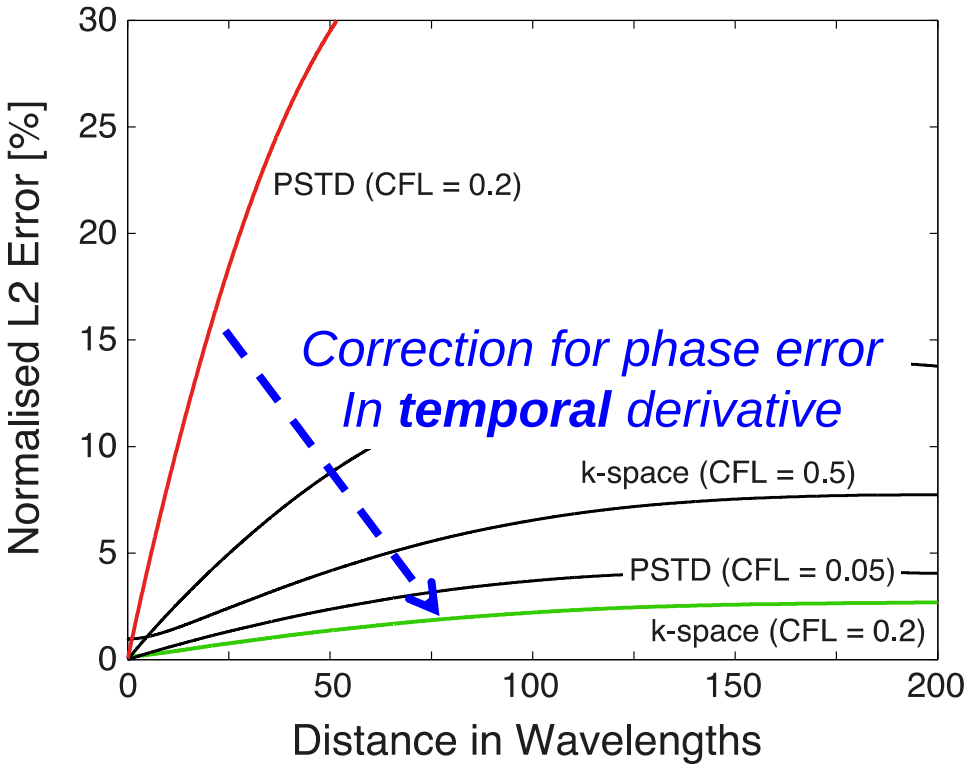


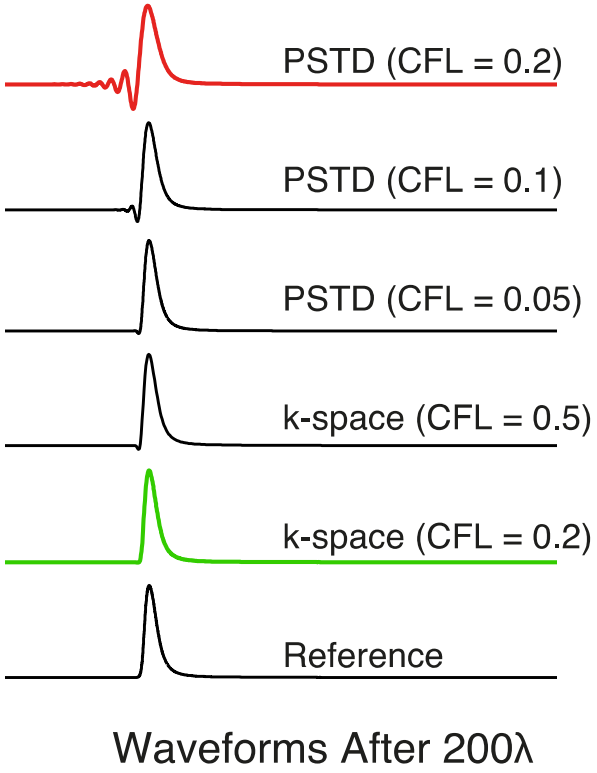


Errors accumulate, so larger domains need more PPW

Numerical dispersion comparison

- Propagation of a pulse defined by $\sin^3(2\pi f_s t)$ in a nonlinear, absorbing medium using 20 PPW





First-order equations

- Same principle can be applied to solve coupled first-order acoustic equations

momentum conservation

$$\frac{\partial u}{\partial t} = -\frac{1}{\rho_0}\frac{\partial p}{\partial x} \quad \rightarrow \quad \frac{u^{n+\frac{1}{2}} - u^{n-\frac{1}{2}}}{\Delta t} = -\frac{1}{\rho_0}\mathcal{F}^{-1}\left\{ik_x\kappa\mathcal{F}\left\{p^n\right\}\right\}$$

mass conservation

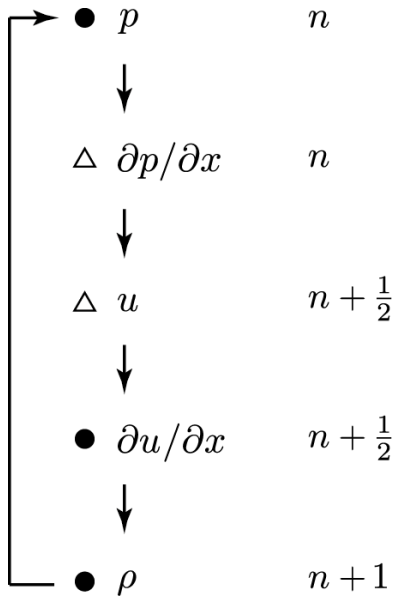
$$\frac{\partial \rho}{\partial t} = -\rho_0\frac{\partial u}{\partial x} \quad \rightarrow \quad \frac{\rho^{n+1} - \rho^n}{\Delta t} = -\rho_0\mathcal{F}^{-1}\left\{ik_x\kappa\mathcal{F}\left\{u^{n+\frac{1}{2}}\right\}\right\}$$

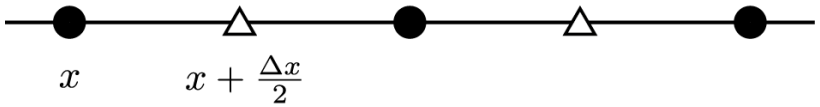
pressure density

$$p = c_0^2\rho \quad \rightarrow \quad p^{n+1} = c_0^2\rho^{n+1}$$

Grid staggering

- The pressure and velocity are calculated at alternating (staggered) time points
- Convention is to also use staggered spatial points – this significantly improves accuracy for heterogeneous media





Grid staggering

- Staggering can be applied using the shift property of the Fourier transform

$$\mathcal{F}\left\{f(x+\Delta x)\right\} = e^{ik_x \Delta x}\hat{f}(k_x)$$

- The term $e^{ik_x \Delta x}$ applied in Fourier space shifts the result by Δx

First-order equations on staggered grid

- Derivative, k-space correction, and shift operators are all applied in the spatial frequency domain

momentum conservation

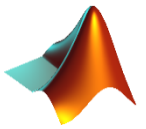
$$\frac{\partial u}{\partial t} = -\frac{1}{\rho_0}\frac{\partial p}{\partial x} \quad \rightarrow \quad \frac{u^{n+\frac{1}{2}} - u^{n-\frac{1}{2}}}{\Delta t} = -\frac{1}{\rho_0}\mathcal{F}^{-1}\left\{ik_x\kappa e^{ik_x\Delta x/2}\mathcal{F}\left\{p^n\right\}\right\}$$

mass conservation

$$\frac{\partial \rho}{\partial t} = -\rho_0\frac{\partial u}{\partial x} \quad \rightarrow \quad \frac{\rho^{n+1} - \rho^n}{\Delta t} = -\rho_0\mathcal{F}^{-1}\left\{ik_x\kappa e^{-ik_x\Delta x/2}\mathcal{F}\left\{u^{n+\frac{1}{2}}\right\}\right\}$$

pressure density

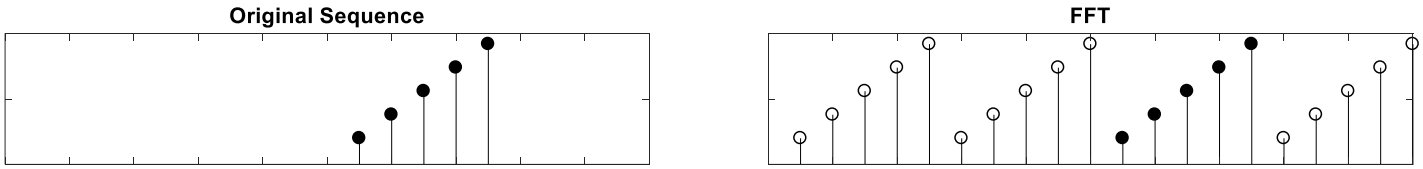
$$p = c_0^2\rho \quad \rightarrow \quad p^{n+1} = c_0^2\rho^{n+1}$$



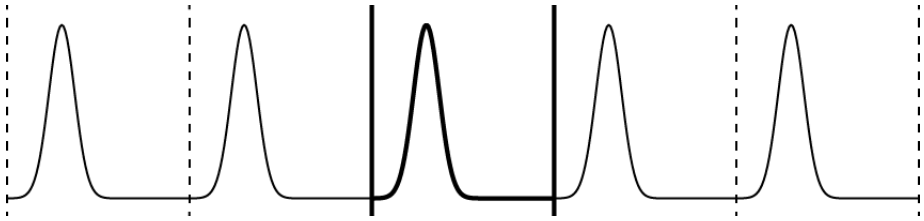
wave_equation_1D_kspace_first_order.m

Perfectly matched layer

- The FFT represents the pressure field as a sum of sines and cosines
- Therefore it implicitly assumes the sequence is part of a periodic sequence



- This leads to ‘wrapping’ in propagation models, which can be understood as a periodically repeated sound field

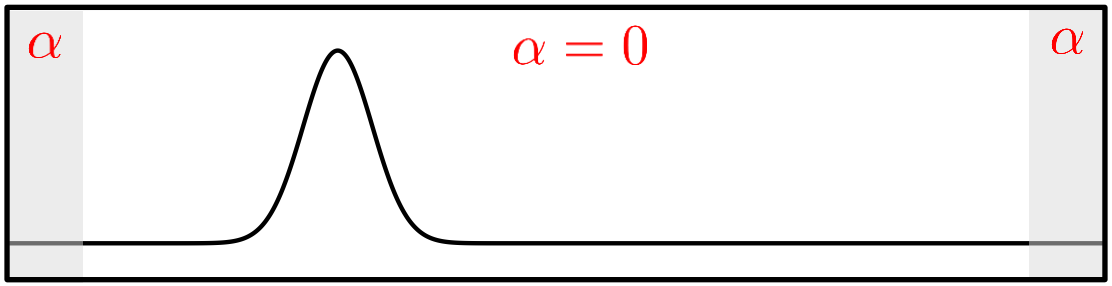


wave_equation_1D_kspace_first_order_wrapping.m

Perfectly matched layer

- An infinite domain can be approximated by using a **perfectly matched layer**
- An absorption term is added to the equations, where the absorption is only non-zero within a boundary layer

$$\frac{\partial u}{\partial t} = -\frac{1}{\rho_0}\frac{\partial p}{\partial x} - \alpha u \qquad \frac{\partial \rho}{\partial t} = -\rho_0 \frac{\partial u}{\partial x} - \alpha \rho$$



- In 2D and 3D, this requires splitting the density field

Perfectly matched layer

- Equations are transformed by multiplying by $e^{\alpha t}$ and rearranging

$$\frac{\partial u}{\partial t} = -\frac{1}{\rho_0}\frac{\partial p}{\partial x} - \alpha u \quad \rightarrow \quad \frac{\partial (e^{\alpha t} u)}{\partial t} = -e^{\alpha t}\frac{1}{\rho_0}\frac{\partial p}{\partial x}$$

$$\frac{\partial \rho}{\partial t} = -\rho_0\frac{\partial u}{\partial x} - \alpha \rho \quad \rightarrow \quad \frac{\partial (e^{\alpha t} \rho)}{\partial t} = -e^{\alpha t}\rho_0\frac{\partial u}{\partial x}$$

- This allows higher absorption values to be used while retaining stability

Perfectly matched layer

- Discrete equations are then given by

momentum conservation

$$u^{n+\frac{1}{2}} = e^{-\alpha \Delta t/2} \left(e^{-\alpha \Delta t/2} u^{n-\frac{1}{2}} - \Delta t \frac{1}{\rho_0} \mathcal{F}^{-1} \left\{ i k_x \kappa e^{i k_x \Delta x/2} \mathcal{F} \left\{ p^n \right\} \right\} \right)$$

mass conservation

$$\rho^{n+1} = e^{-\alpha \Delta t/2} \left(e^{-\alpha \Delta t/2} \rho^n - \Delta t \rho_0 \mathcal{F}^{-1} \left\{ i k_x \kappa e^{-i k_x \Delta x/2} \mathcal{F} \left\{ u^{n+\frac{1}{2}} \right\} \right\} \right)$$

pressure density

$$p^{n+1} = c_0^2 \rho^{n+1}$$

perfectly matched layer

spatial derivative

k-space correction

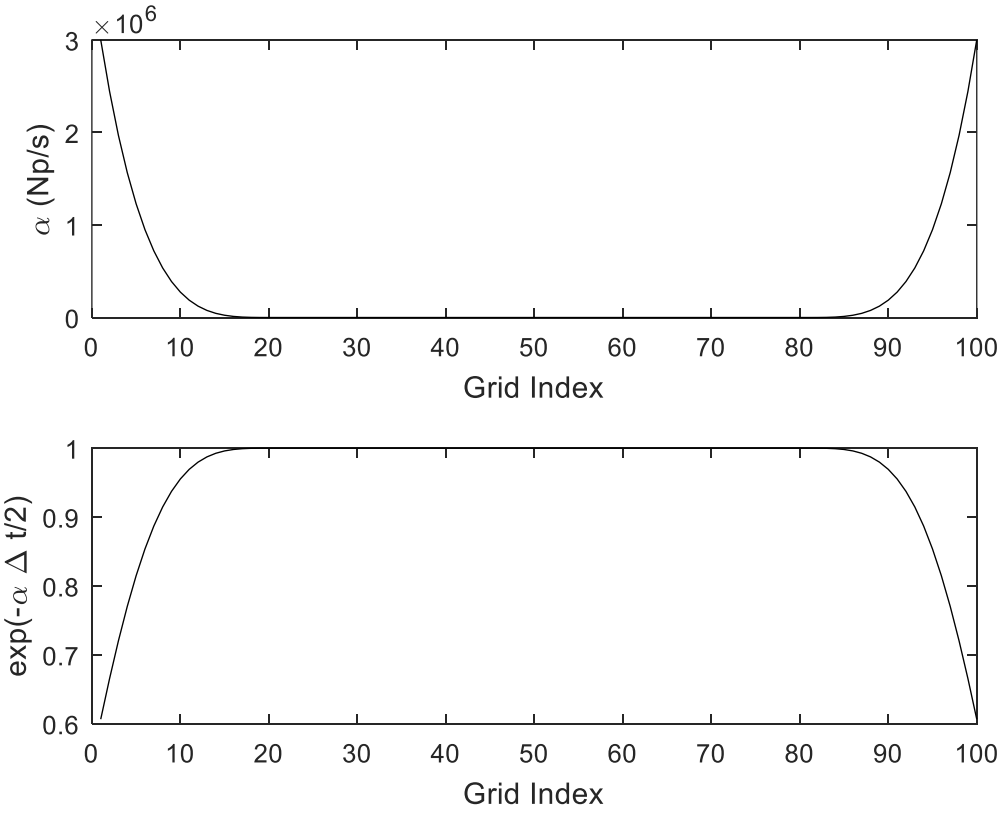
staggered grid shift

Perfectly matched layer

- Absorption profile within the PML is graded to avoid reflections

$$\alpha_j = \alpha_{\max} \left(\frac{j}{J} \right)^4$$

$$e^{-\alpha \Delta t/2}$$



wave_equation_1D_kspace_first_order_pml.m

Summary

- We have now covered all of numerical details used in k-Wave for a homogeneous medium in 1D!
- Simulations in 2D and 3D are very similar
- k-Wave also accounts for
  - Absorption following a frequency power law
  - Nonlinear wave propagation
  - Heterogeneous material properties (sound speed, density, acoustic absorption coefficient, nonlinearity parameter)

- Similar principles apply for the elastic wave model and the heat diffusion model in k-Wave