Numerical methods

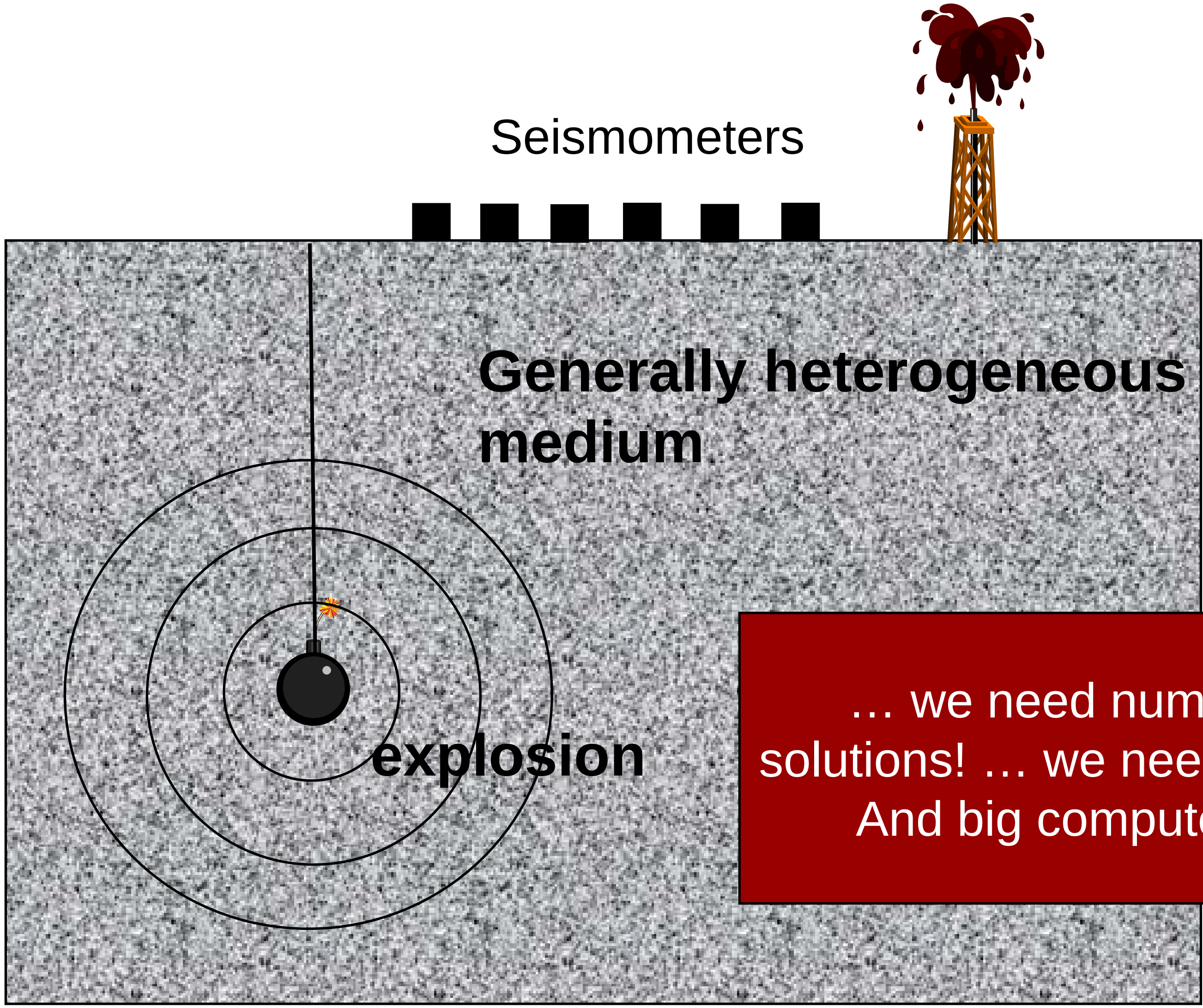
Specific methods:

- Finite differences
- Pseudospectral methods
- Finite volumes

… applied to the acoustic wave equation …

Why numerical methods

Example: seismic wave propagation



… we need numerical solutions! … we need grids! … And big computers …

Partial Differential Equations in Geophysics

$$\partial_t^2 p = c^2 \Delta p + s$$

$$\Delta = (\partial_x^2 + \partial_y^2 + \partial_z^2)$$

P	pressure
c	acoustic wave speed
s	sources

The acoustic wave equation

- seismology
- acoustics
- oceanography
- meteorology

$$\partial_t C = k\Delta C - \mathbf{v} \bullet \nabla C - RC + p$$

C	tracer concentration
k	diffusivity
v	flow velocity
R	reactivity
p	sources

Diffusion, advection, Reaction

- geodynamics
- oceanography
- meteorology
- geochemistry
- sedimentology
- geophysical fluid dynamics

Numerical methods: properties

Finite differences →

- time-dependent PDEs
- seismic wave propagation
- geophysical fluid dynamics
- Maxwell's equations
- Ground penetrating radar

-> robust, simple concept, easy to parallelize, regular grids, explicit method

Finite elements →

- static and time-dependent PDEs
- seismic wave propagation
- geophysical fluid dynamics
- all problems

-> implicit approach, matrix inversion, well founded, <u>irregular grids</u>, more complex algorithms, engineering problems

Finite volumes →

- time-dependent PDEs
- seismic wave propagation
- mainly fluid dynamics

-> robust, simple concept, <u>irregular grids</u>, explicit method

Other Numerical methods:

Particle-based methods

- lattice gas methods
- molecular dynamics
- granular problems
- fluid flow
- earthquake simulations

-> very heterogeneous problems, nonlinear problems

Boundary element methods

- problems with boundaries (rupture)
- based on analytical solutions
- only discretization of planes

--> good for problems with special boundary conditions (rupture, cracks, etc)

***Pseudo*spectral methods**

- orthogonal basis functions, special case of FD
- spectral accuracy of space derivatives
- wave propagation, GPR

-> regular grids, explicit method, problems with strongly heterogeneous media

What is a finite difference?

Common definitions of the derivative of f(x):

$$\partial_x f = \lim_{dx \to 0} \frac{f(x+dx) - f(x)}{dx}$$

$$\partial_x f = \lim_{dx \to 0} \frac{f(x) - f(x-dx)}{dx}$$

$$\partial_x f = \lim_{dx \to 0} \frac{f(x+dx) - f(x-dx)}{2dx}$$

These are all correct definitions in the limit dx->0.

But we want dx to remain **FINITE**

What is a finite difference?

The equivalent ***approximations*** of the derivatives are:

$$\partial_x f^+ \approx \frac{f(x+dx)-f(x)}{dx}$$ forward difference

$$\partial_x f^- \approx \frac{f(x)-f(x-dx)}{dx}$$ backward difference

$$\partial_x f \approx \frac{f(x+dx)-f(x-dx)}{2dx}$$ centered difference

The big question:

How good are the FD approximations?

$$\approx \neq =$$

This leads us to Taylor series....

Our first FD algorithm (ac1d.m) !

$$\partial_t^2 p = c^2 \Delta p + s$$

$$\Delta = (\partial_x^2 + \partial_y^2 + \partial_z^2)$$

P	pressure
c	acoustic wave speed
s	sources

Problem: Solve the 1D acoustic wave equation using the finite Difference method.

Solution:

$$p(t+dt) = \frac{c^2 dt^2}{dx^2}\left[p(x+dx) - 2p(x) + p(x-dx)\right] + 2p(t) - p(t-dt) + sdt^2$$

Problems: Stability

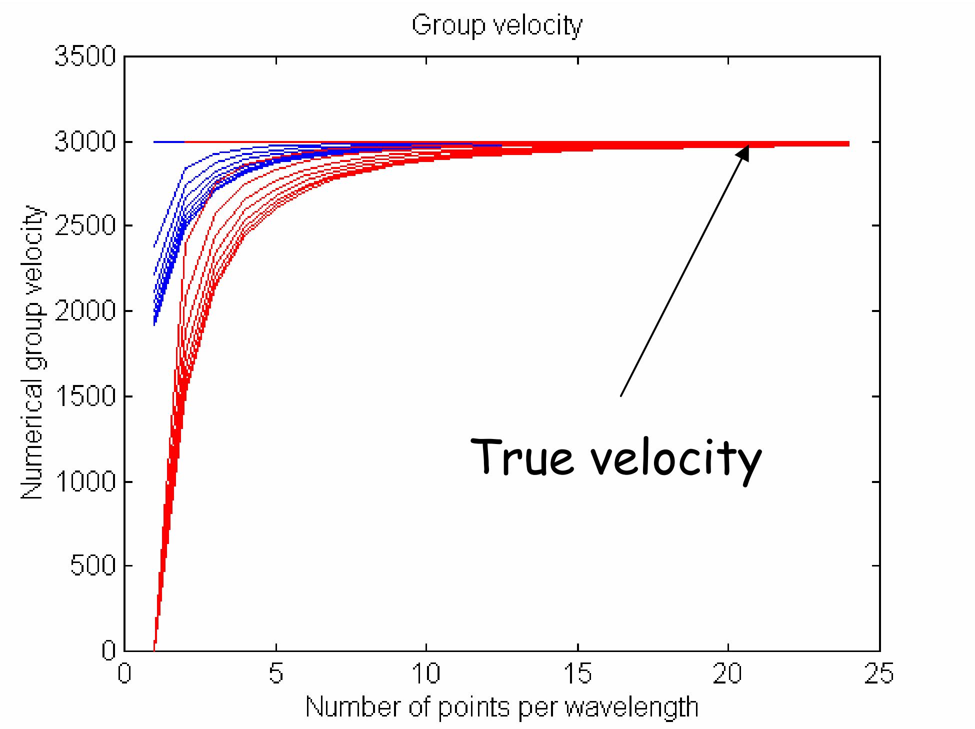
$$p(t+dt) = \frac{c^2 dt^2}{dx^2}\left[p(x+dx) - 2p(x) + p(x-dx)\right] + 2p(t) - p(t-dt) + sdt^2$$

Stability: Careful analysis using harmonic functions shows that a stable numerical calculation is subject to special conditions (conditional stability). This holds for many numerical problems.

$$c\frac{dt}{dx} \leq \varepsilon \approx 1$$

Problems: Dispersion

$$p(t+dt) = \frac{c^2 dt^2}{dx^2}\left[p(x+dx) - 2p(x) + p(x-dx)\right] + 2p(t) - p(t-dt) + sdt^2$$



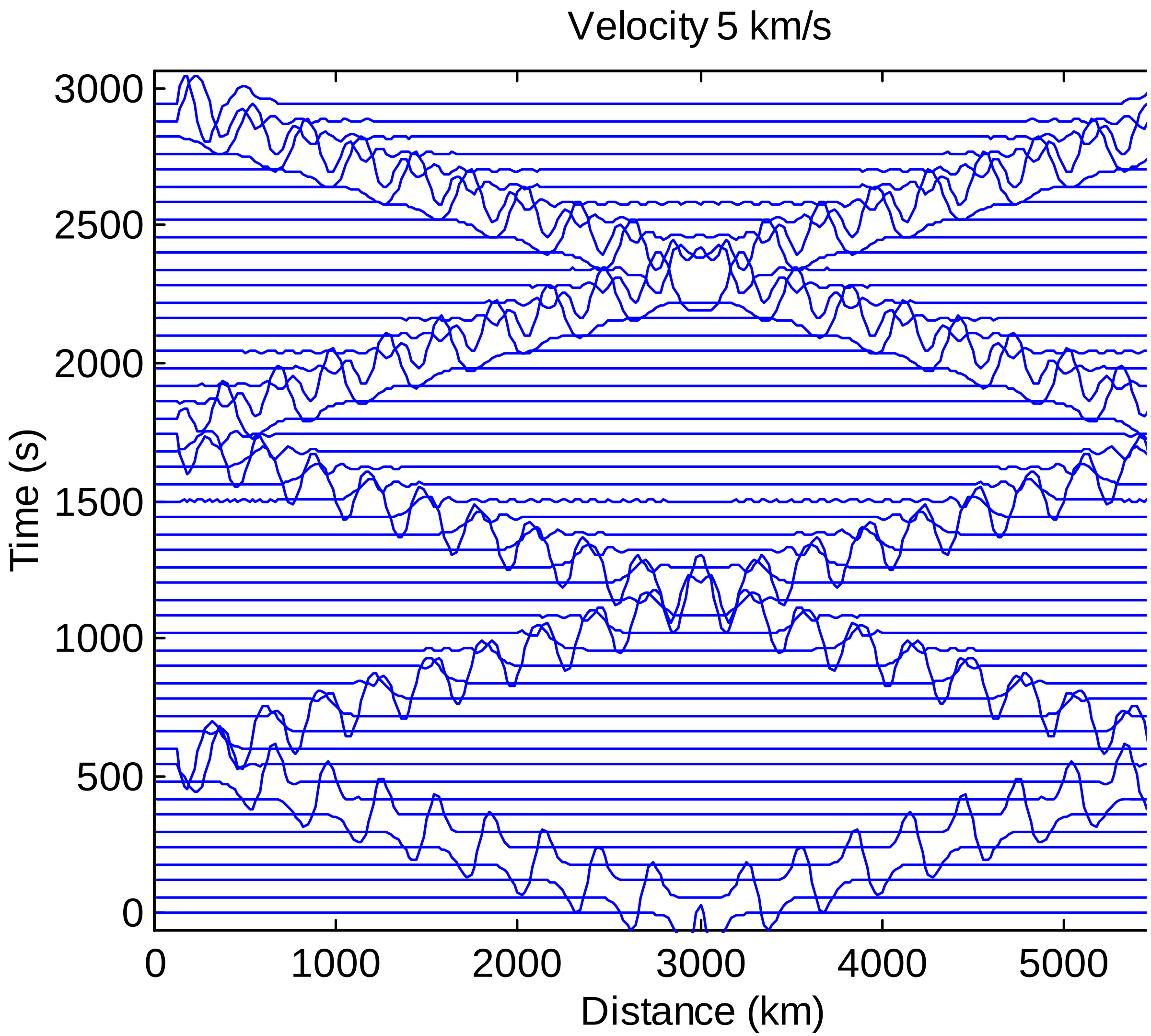
Dispersion: The numerical approximation has artificial dispersion, in other words, the wave speed becomes frequency dependent.
You have to find a frequency bandwidth where this effect is small.
The solution is to use a sufficient number of grid points per wavelength.

Our first FD code!

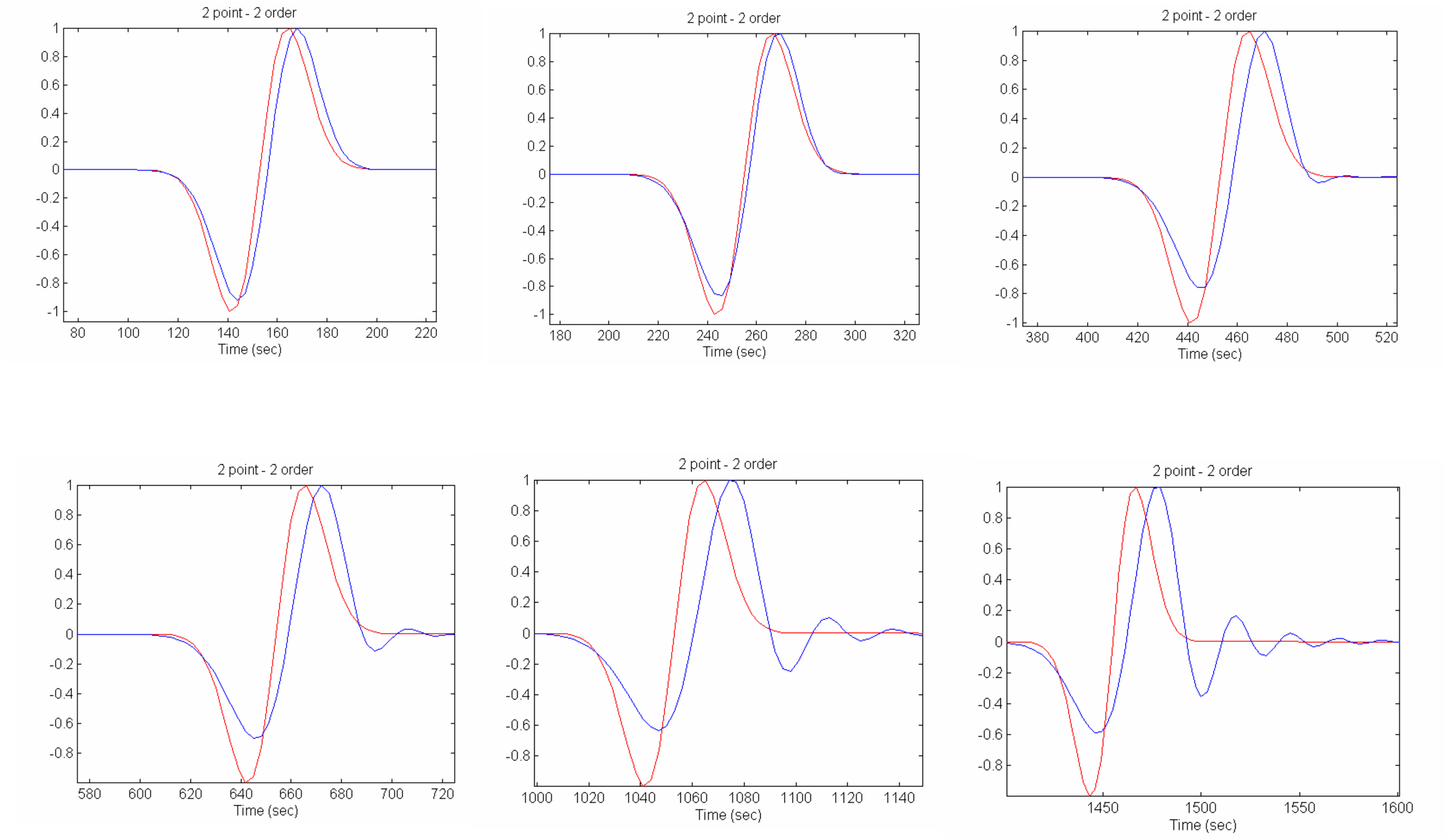
$$p(t+dt) = \frac{c^2 dt^2}{dx^2}\left[p(x+dx) - 2p(x) + p(x-dx)\right] + 2p(t) - p(t-dt) + sdt^2$$

```
% Time stepping

for i=1:nt,

   % FD

   disp(sprintf(' Time step : %i',i));

   for j=2:nx-1
      d2p(j)=(p(j+1)-2*p(j)+p(j-1))/dx^2; % space derivative
   end
   pnew=2*p-pold+d2p*dt^2;                % time extrapolation
   pnew(nx/2)=pnew(nx/2)+src(i)*dt^2;     % add source term
   pold=p;                          % time levels
   p=pnew;
   p(1)=0;                 % set boundaries pressure free
   p(nx)=0;

   % Display
   plot(x,p,'b-')
   title(' FD ')
   drawnow

end
```

Snapshot Example



Seismogram Dispersion



Finite Differences - Summary

- Conceptually the most simple of the numerical methods and can be learned quite quickly
- Depending on the physical problem FD methods are conditionally stable (relation between time and space increment)
- FD methods have difficulties concerning the accurate implementation of boundary conditions (e.g. free surfaces, absorbing boundaries)
- FD methods are usually explicit and therefore very easy to implement and efficient on parallel computers
- FD methods work best on regular, rectangular grids

The Fourier Method

- What is a *pseudo*-spectral Method?
- Fourier Derivatives
- The Fast Fourier Transform (FFT)
- The Acoustic Wave Equation with the Fourier Method
- Comparison with the Finite-Difference Method
- Dispersion and Stability of Fourier Solutions

Spectral solutions to time-dependent PDEs are formulated in the frequency-wavenumber domain and solutions are obtained in terms of spectra (e.g. seismograms). This technique is particularly interesting for geometries where partial solutions in the ω-k domain can be obtained analytically (e.g. for layered models).

In the pseudo-spectral approach - in a finite-difference like manner - the PDEs are solved pointwise in physical space (x-t). However, the space derivatives are calculated using orthogonal functions (e.g. Fourier Integrals, Chebyshev polynomials). They are either evaluated using matrix-matrix multiplications or the fast Fourier transform (FFT).

.. let us recall the definition of the derivative using Fourier integrals ...

$$\partial_x f(x) = \partial_x \left(\int_{-\infty}^{\infty} F(k) e^{-ikx} dk \right)$$

$$= -\int_{-\infty}^{\infty} ikF(k) e^{-ikx} dk$$

... we could either ...

1) perform this calculation in the space domain by convolution

2) actually transform the function f(x) in the k-domain and back

... the latter approach became interesting with the introduction of the Fast Fourier Transform (FFT). What's so fast about it ?

The FFT originates from a paper by Cooley and Tukey (1965, Math. Comp. vol 19 297-301) which revolutionised all fields where Fourier transforms where essential to progress.

The discrete Fourier Transform can be written as

$$\hat{u}_k = \frac{1}{N} \sum_{j=0}^{N-1} u_j e^{-2\pi i k j / N}, k = 0,1,\ldots, N-1$$

$$u_k = \sum_{j=0}^{N-1} \hat{u}_j e^{2\pi i k j / N}, k = 0,1,\ldots, N-1$$

... this can be written as matrix-vector products ...
for example the inverse transform yields ...

$$\begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \dots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \dots & \omega^{2N-2} \\ \vdots & \vdots & & & & \vdots \\ \vdots & \vdots & & & & \vdots \\ 1 & \omega^{N-1} & \dots & \dots & \dots & \omega^{(N-1)^2} \end{bmatrix} \begin{bmatrix} \hat{u}_0 \\ \hat{u}_1 \\ \hat{u}_2 \\ \vdots \\ \vdots \\ \hat{u}_{N-1} \end{bmatrix} = \begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ \vdots \\ u_{N-1} \end{bmatrix}$$

.. where ...

$$\omega = e^{2\pi i / N}$$

... the FAST bit is recognising that the full matrix - vector multiplication can be written as a few sparse matrix - vector multiplications (for details see for example Bracewell, the Fourier Transform and its applications, MacGraw-Hill) with the effect that:

<u>Number of multiplications</u>

full matrix	FFT
N^2	$2N\log_2 N$

this has enormous implications for large scale problems. Note: the factorisation becomes particularly simple and effective when N is a highly composite number (power of 2).

Number of multiplications

Problem	full matrix	FFT	Ratio full/FFT
1D (nx=512)	2.6×10^5	9.2×10^3	28.4
1D (nx=2096)			94.98
1D (nx=8384)			312.6

.. the right column can be regarded as the speedup of an algorithm when the FFT is used instead of the full system.

let us take the acoustic wave equation with variable density

$$\frac{1}{\rho c^2}\partial_t^2 p = \partial_x\left(\frac{1}{\rho}\partial_x p\right)$$

the left hand side will be expressed with our standard centered finite-difference approach

$$\frac{1}{\rho c^2 dt^2}\left[p(t+dt) - 2p(t) + p(t-dt)\right] = \partial_x\left(\frac{1}{\rho}\partial_x p\right)$$

... leading to the extrapolation scheme ...

$$p(t+dt) = \rho c^2 dt^2 \boxed{\partial_x \left(\frac{1}{\rho} \partial_x p \right)} + 2p(t) - p(t-dt)$$

where the space derivatives will be calculated using the Fourier Method. The highlighted term will be calculated as follows:

$$P_j^n \rightarrow \text{FFT} \rightarrow \hat{P}_\upsilon^n \rightarrow ik_\upsilon \hat{P}_\upsilon^n \rightarrow \text{FFT}^{-1} \rightarrow \partial_x P_j^n$$

multiply by 1/ρ

$$\frac{1}{\rho} \partial_x P_j^n \rightarrow \text{FFT} \rightarrow \left(\frac{1}{\rho} \partial_x \hat{P} \right)_\upsilon^n \rightarrow ik_\upsilon \left(\frac{1}{\rho} \partial_x \hat{P} \right)_\upsilon^n \rightarrow \text{FFT}^{-1} \rightarrow \partial_x \left(\frac{1}{\rho} \partial_x P_j^n \right)$$

... then extrapolate ...

$$p(t+dt) =$$

$$\rho c^2 dt^2 \boxed{\left(\partial_x \left(\frac{1}{\rho} \partial_x p \right) + \partial_y \left(\frac{1}{\rho} \partial_y p \right) + \partial_z \left(\frac{1}{\rho} \partial_z p \right) \right)}$$

$$+ 2p(t) - p(t-dt)$$

.. where the following algorithm applies to each space dimension ...

$$P_j^n \rightarrow \mathrm{FFT} \rightarrow \hat{P}_\upsilon^n \rightarrow ik_\upsilon \hat{P}_\upsilon^n \rightarrow \mathrm{FFT}^{-1} \rightarrow \partial_x P_j^n$$

$$\frac{1}{\rho} \partial_x P_j^n \rightarrow \mathrm{FFT} \rightarrow \left(\frac{1}{\rho} \partial_x \hat{P} \right)_\upsilon^n \rightarrow ik_\upsilon \left(\frac{1}{\rho} \partial_x \hat{P} \right)_\upsilon^n \rightarrow \mathrm{FFT}^{-1} \rightarrow \partial_x \left(\frac{1}{\rho} \partial_x P_j^n \right)$$

Comparison with finite differences - Algorithm

let us compare the core of the algorithm - the calculation of the derivative (Matlab code)

```
function df=fder1d(f,dx,nop)
% fDER1D(f,dx,nop) finite difference
% second derivative

nx=max(size(f));

n2=(nop-1)/2;

if nop==3; d=[1 -2 1]/dx^2; end
if nop==5; d=[-1/12 4/3 -5/2 4/3 -1/12]/dx^2; end

df=[1:nx]*0;

for i=1:nop;
df=df+d(i).*cshift1d(f,-n2+(i-1));
end
```

... and the first derivative using FFTs ...

```
function df=sder1d(f,dx)
% SDER1D(f,dx) spectral derivative of vector
nx=max(size(f));


% initialize k
kmax=pi/dx;
dk=kmax/(nx/2);
for i=1:nx/2, k(i)=(i)*dk; k(nx/2+i)=-kmax+(i)*dk; end
k=sqrt(-1)*k;

% FFT and IFFT
ff=fft(f); ff=k.*ff; df=real(ifft(ff));
```

.. simple and elegant ...

... with the usual *Ansatz*

$$p_j^n = e^{i(kjdx - n\omega dt)}$$

we obtain

$$\partial_x^2 p_j^n = -k^2 e^{i(kjdx - \omega ndt)}$$

$$\partial_t^2 p_j^n = -\frac{4}{dt^2}\sin^2\frac{\omega dt}{2} e^{i(kjdx - \omega ndt)}$$

... leading to

$$k = \frac{2}{cdt}\sin\frac{\omega dt}{2}$$

$$k = \frac{2}{cdt} \sin \frac{\omega dt}{2}$$

$$\omega = \frac{2}{dt} \sin^{-1}\left(\frac{kcdt}{2}\right)$$

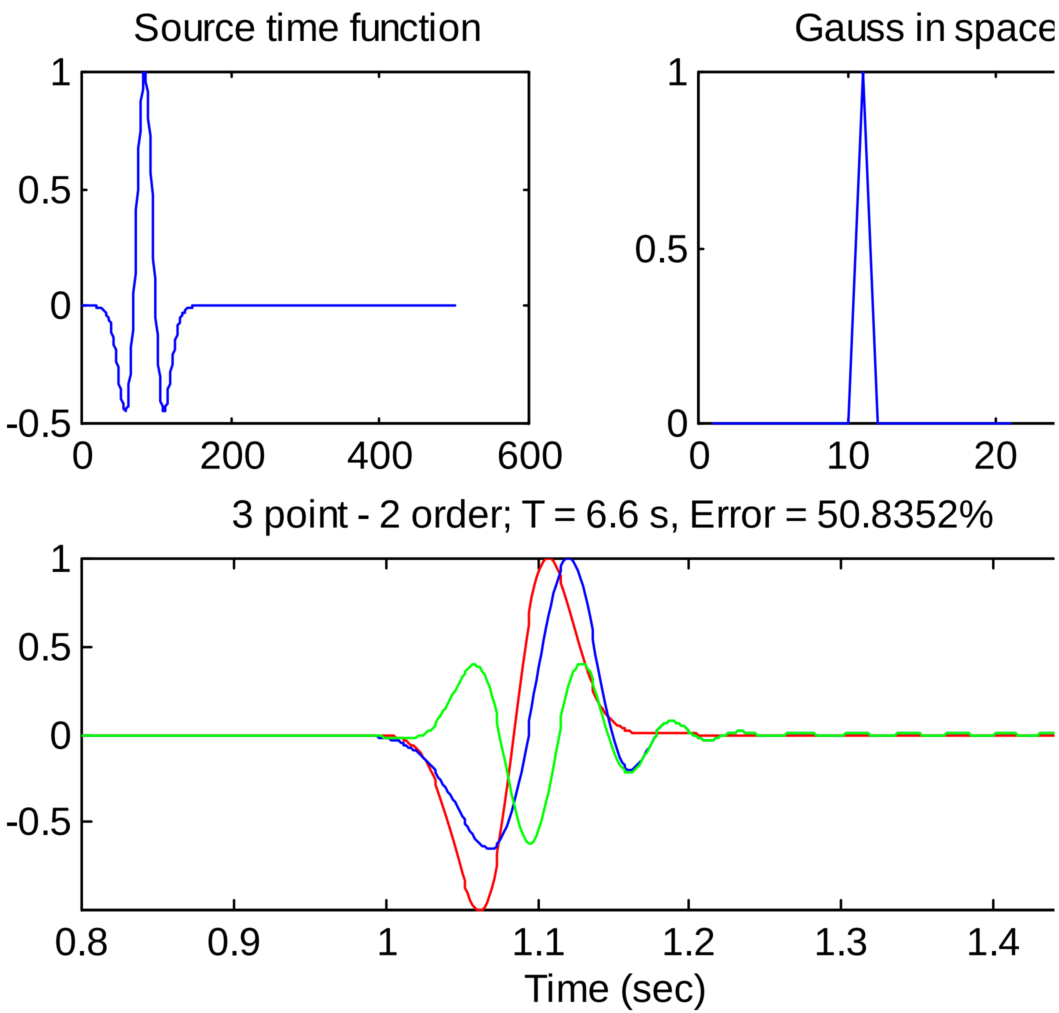
What are the consequences?

a) when dt << 1, $\sin^{-1}$ (kcdt/2) ≈kcdt/2 and w/k=c
-> practically no dispersion
b) the argument of asin must be smaller than one.

$$\frac{k_{\max} cdt}{2} \leq 1$$

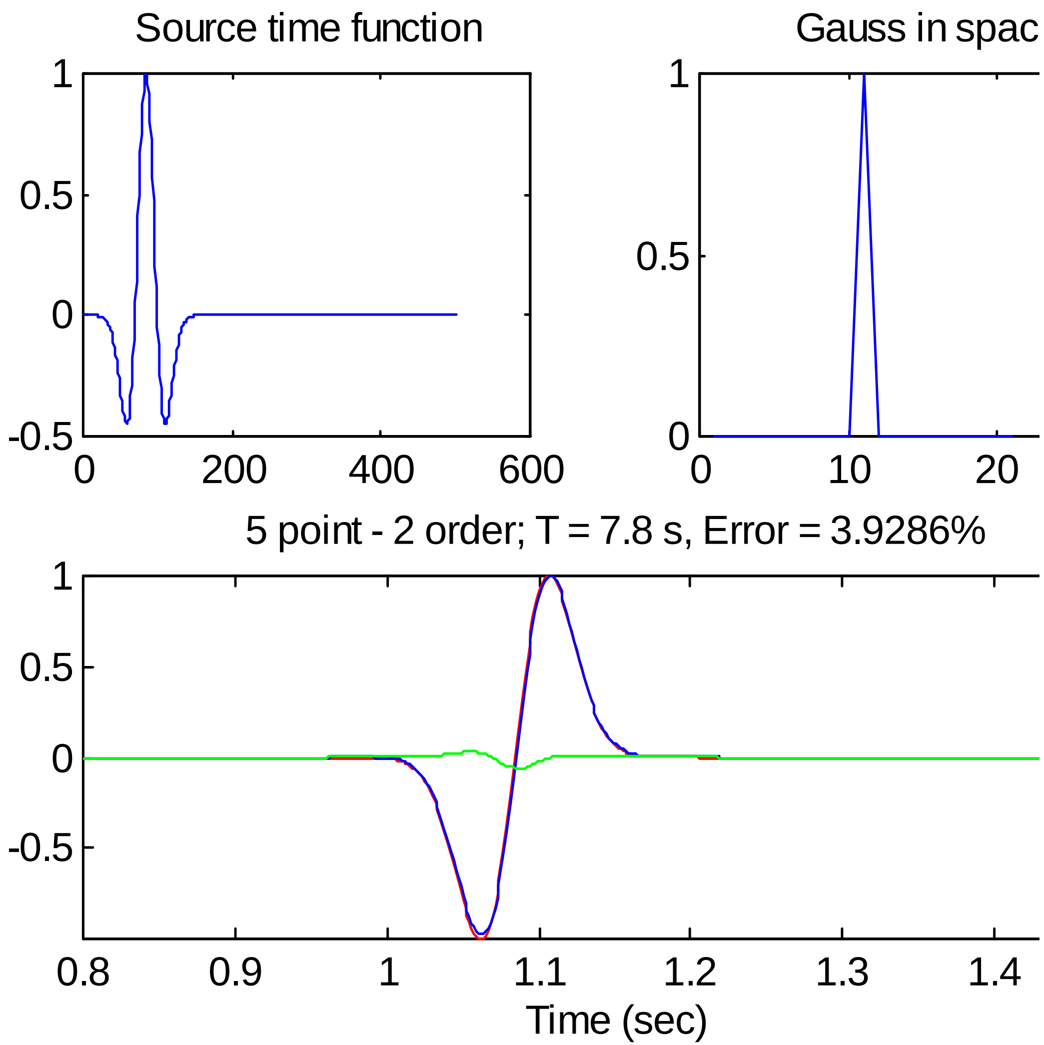
$$cdt / dx \leq 2 / \pi \approx 0.636$$

Fourier Method - Comparison with FD - 10Hz



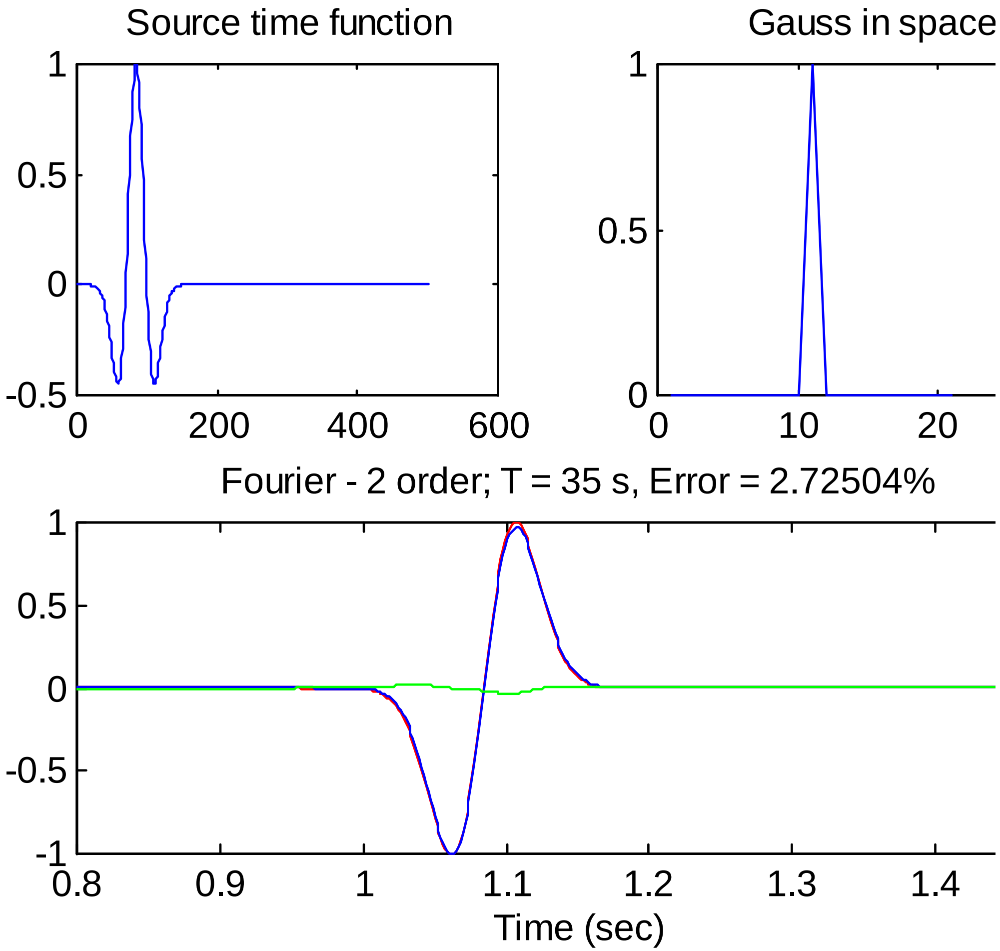
Example of acoustic 1D wave simulation.
FD 3 -point operator
red-analytic; blue-numerical; green-difference

Fourier Method - Comparison with FD - 10Hz



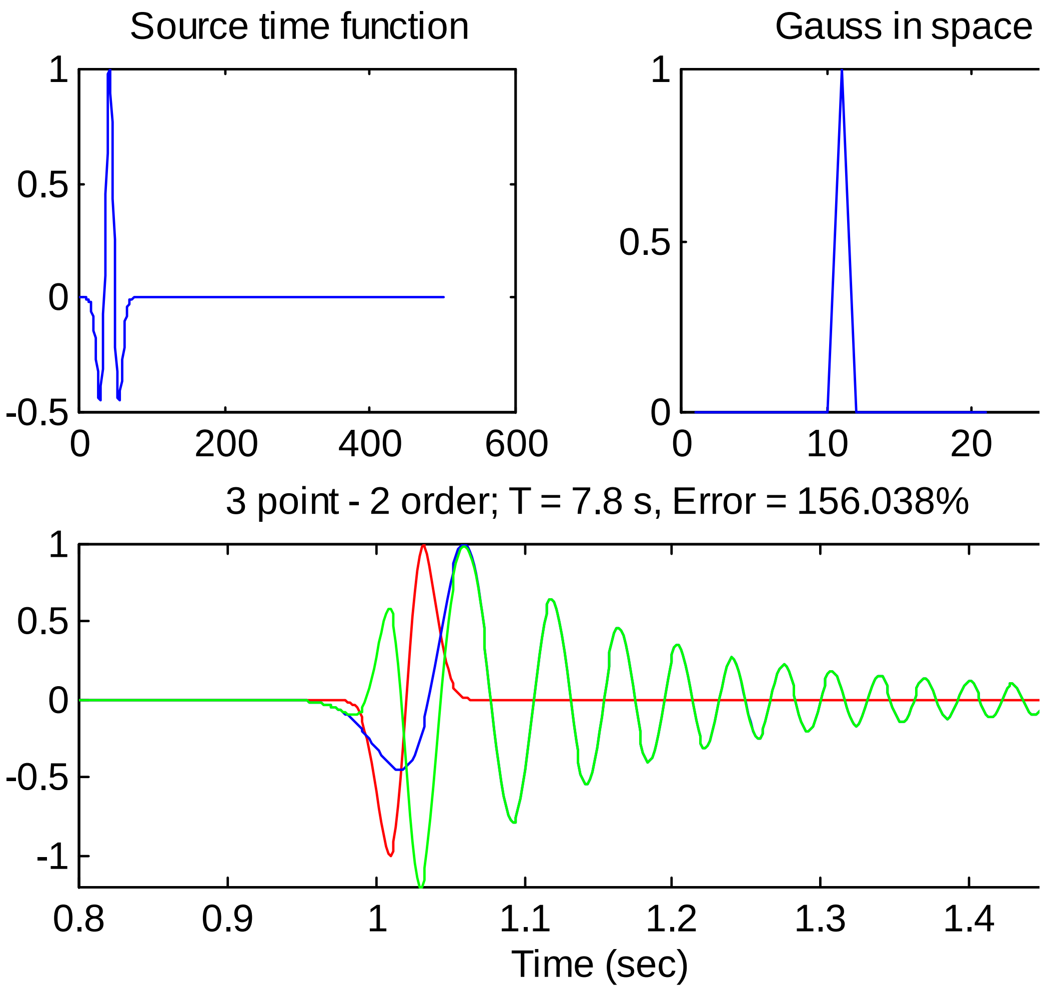
Example of acoustic 1D wave simulation.
FD 5 -point operator
red-analytic; blue-numerical; green-difference

Fourier Method - Comparison with FD - 10Hz

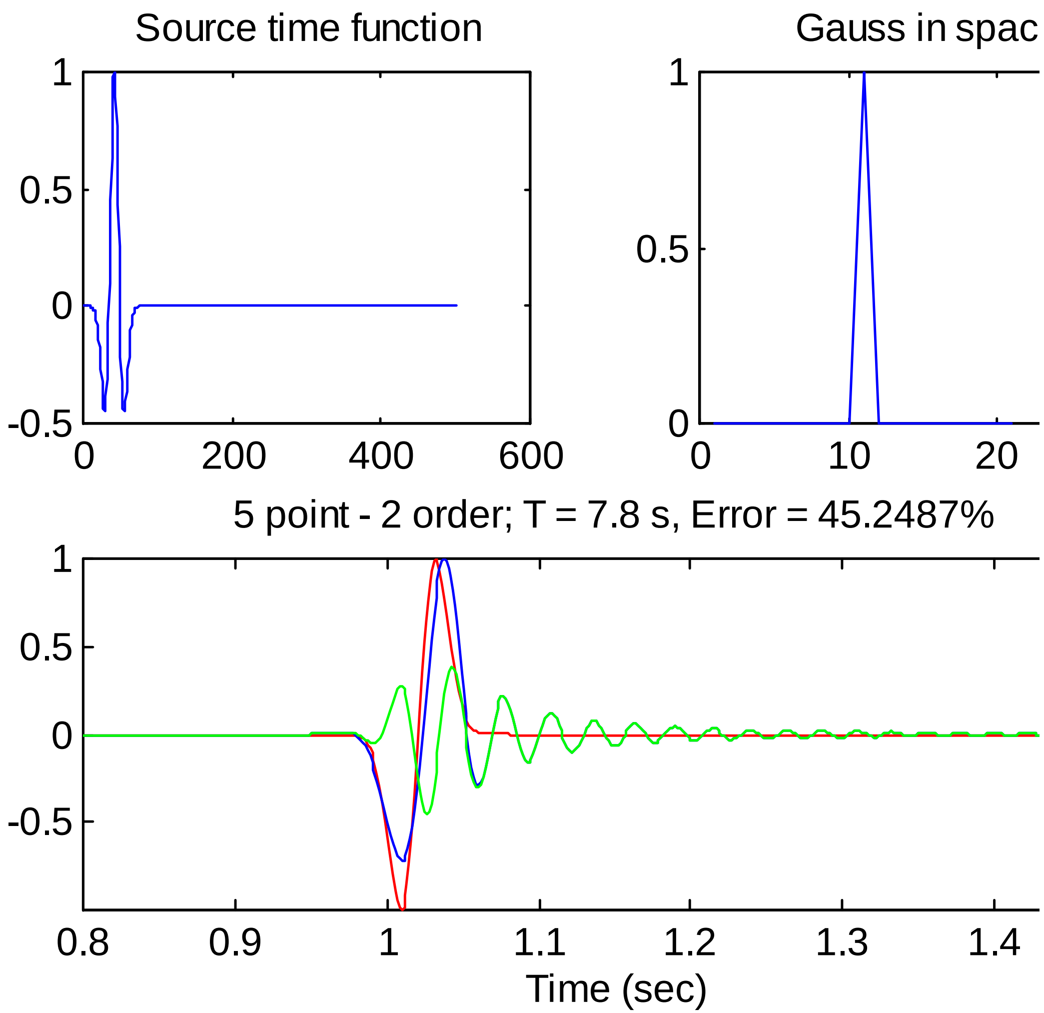


Example of acoustic 1D wave simulation.
Fourier operator
red-analytic; blue-numerical; green-difference

Fourier Method - Comparison with FD - 20Hz

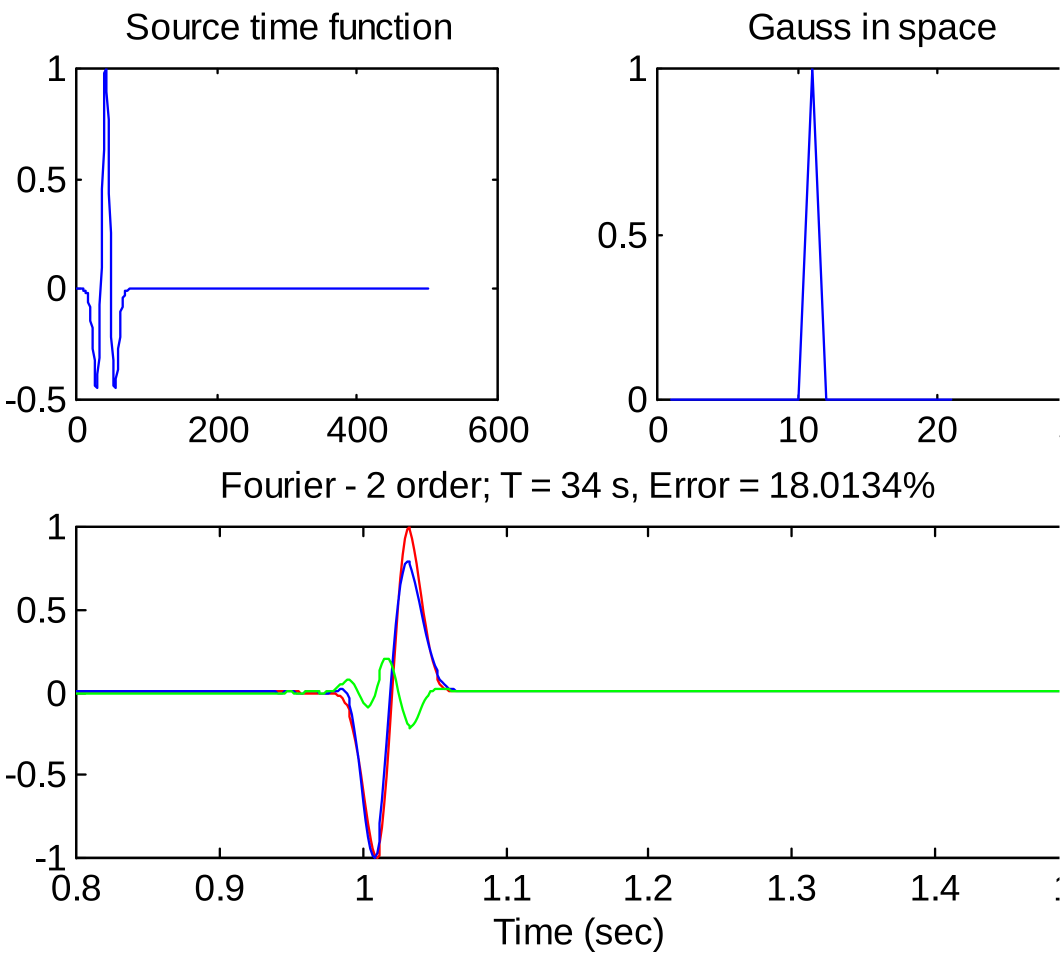


Example of acoustic 1D wave simulation.
FD 3 -point operator
red-analytic; blue-numerical; green-difference



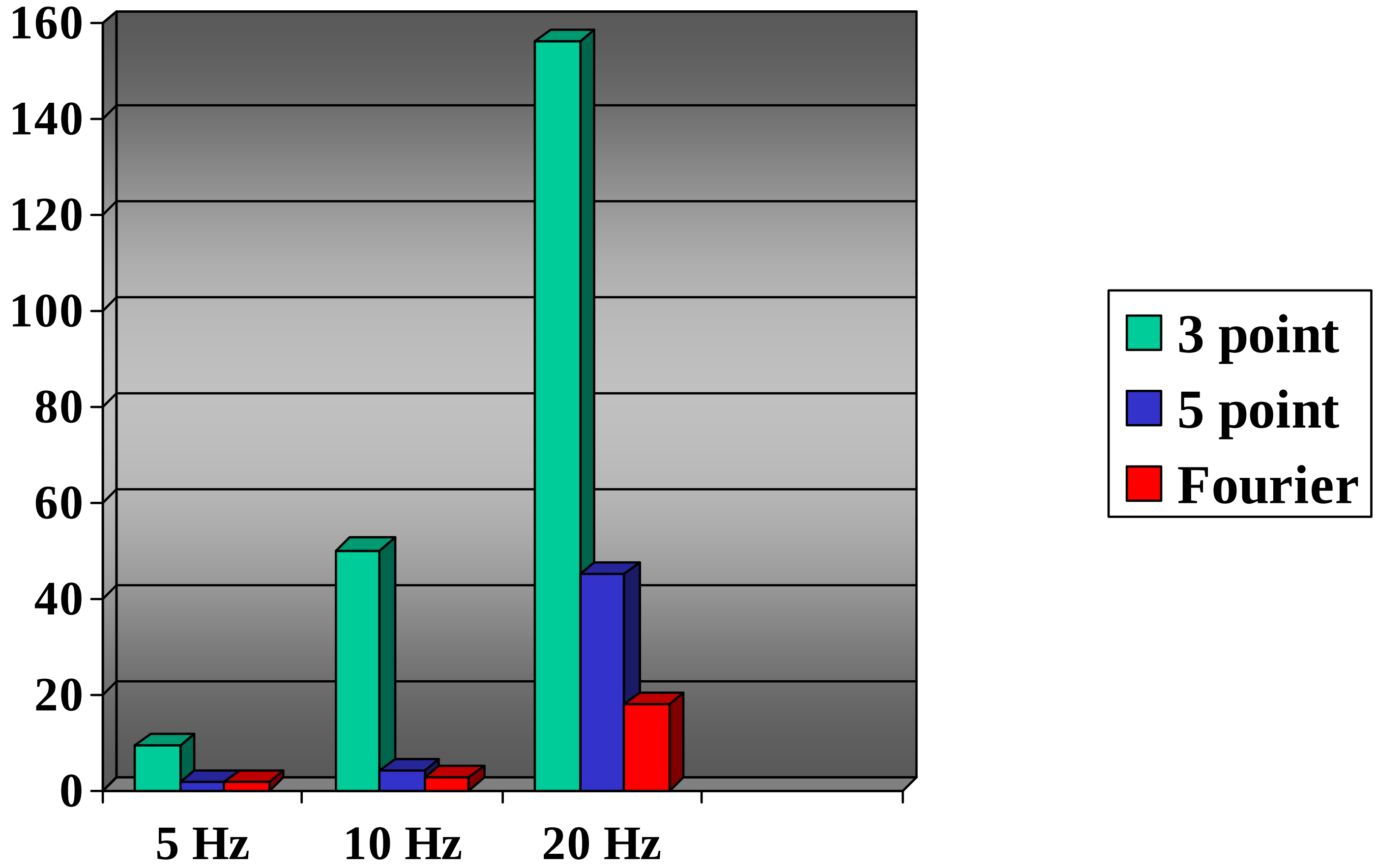
Example of acoustic 1D wave simulation.
FD 5 -point operator
red-analytic; blue-numerical; green-difference

Fourier Method - Comparison with FD - 20Hz



Example of acoustic 1D wave simulation.
Fourier operator
red-analytic; blue-numerical; green-difference

Difference (%) between numerical and analytical solution
as a function of propagating frequency



Simulation time
5.4s
7.8s
33.0s

The concept of Green's Functions (impulse responses) plays an important role in the solution of partial differential equations. It is also useful for numerical solutions. Let us recall the acoustic wave equation

$$\partial_t^2 p = c^2 \Delta p$$

with Δ being the Laplace operator. We now introduce a delta source in space and time

$$\partial_t^2 p = \delta(\underline{x})\delta(t) + c^2 \Delta p$$

the formal solution to this equation is

$$p(\underline{x}, t) = \frac{1}{4\pi c^2} \frac{\delta(t - |\underline{x}|/c)}{|\underline{x}|}$$

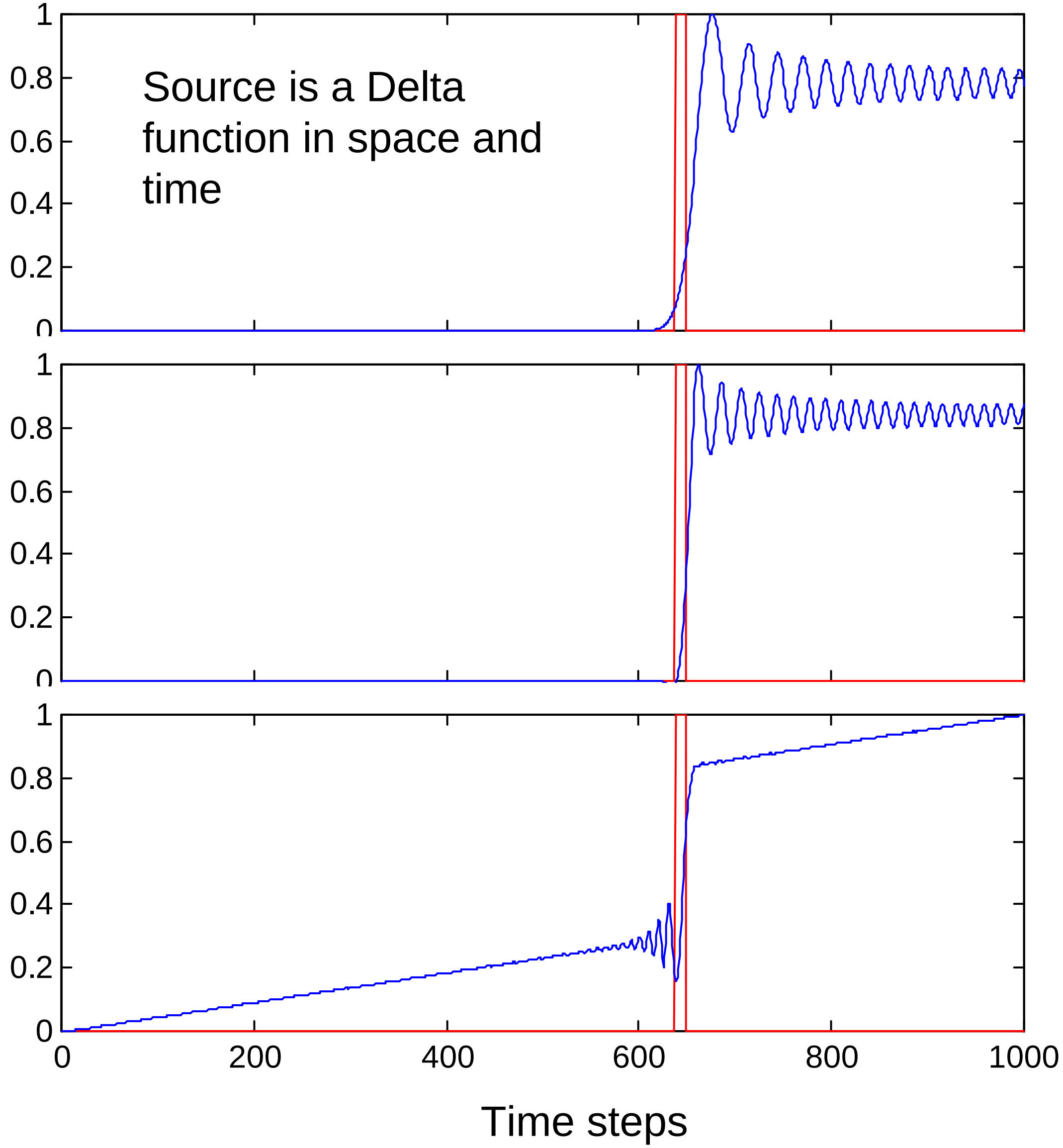
(Full proof given in Aki and Richards, Quantitative Seismology, Freeman+Co, 1981, p. 65)

$$p(\underline{x},t) = \frac{1}{4\pi c^2} \frac{\delta(t - |\underline{x}|/c)}{|\underline{x}|}$$

In words this means (in 1D and 3D but not in 2D, why?) , that in homogeneous media the same source time function which is input at the source location will be recorded at a distance r, but with amplitude proportional to 1/r.

An arbitrary source can evidently be constructed by summing up different delta - solutions. Can we use this property in our numerical simulations?

What happens if we solve our numerical system with delta functions as sources?



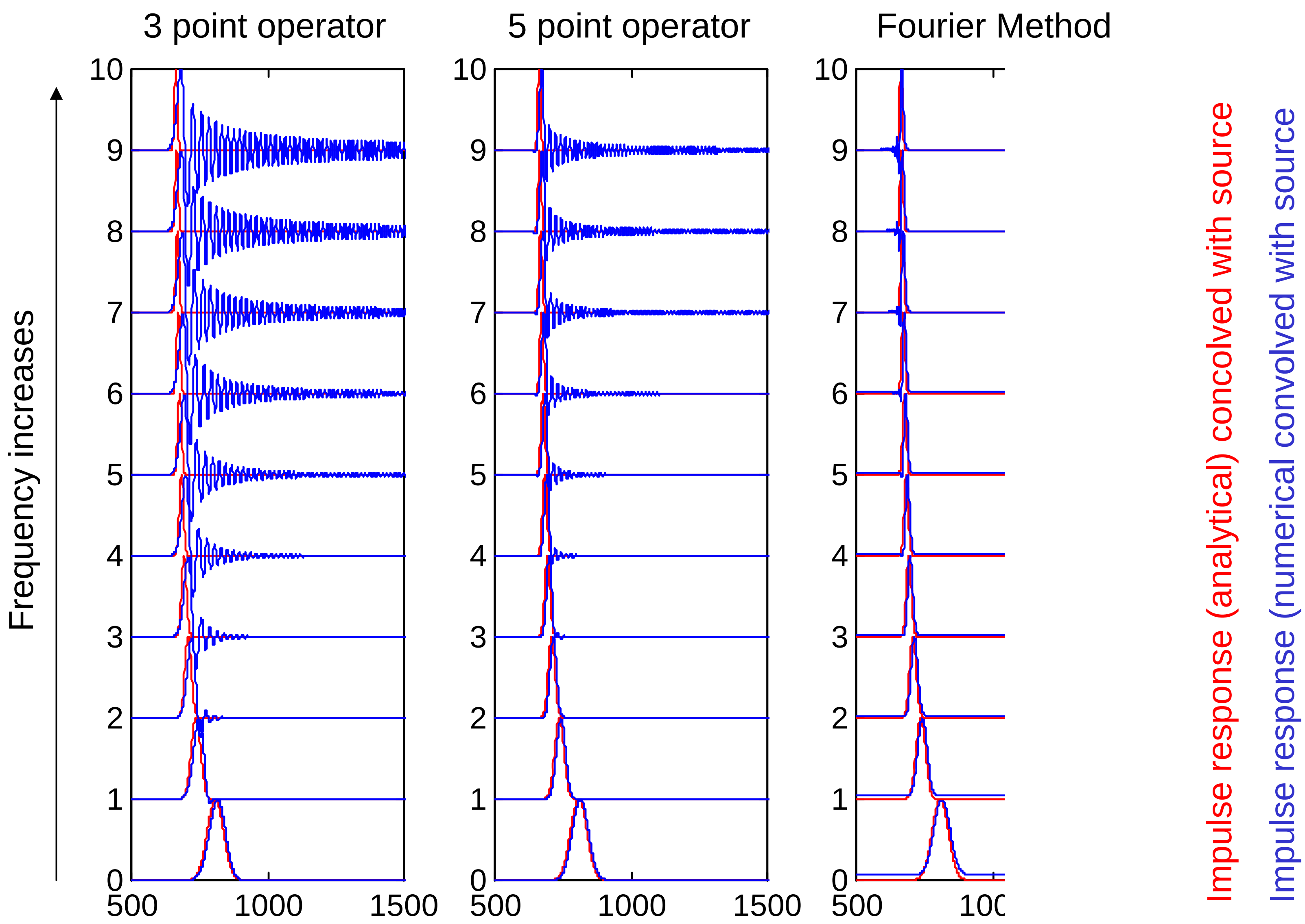
3 point operator

5 point operator

Fourier Method

Impulse response (analytical)

Impulse response (numerical



3 point operator
5 point operator
Fourier Method
Frequency increases
0
1
2
3
4
5
6
7
8
9
10
500
1000
1500
Impulse response (analytical) concolved with source
Impulse response (numerical convolved with source

Fourier Method - Summary

The Fourier Method can be considered as the limit of the finite-difference method as the length of the operator tends to the number of points along a particular dimension.

The space derivatives are calculated in the wavenumber domain by multiplication of the spectrum with *ik.* The inverse Fourier transform results in an exact space derivative up to the Nyquist frequency.

The use of Fourier transform imposes some constraints on the smoothness of the functions to be differentiated. Discontinuities lead to Gibb's phenomenon.

As the Fourier transform requires periodicity this technique is particular useful where the physical problems are periodical (e.g. angular derivatives in cylindrical problems).

Finite Elements – the concept

How to proceed in FEM analysis:

- Divide structure into pieces (like LEGO)
- Describe behaviour of the physical quantities in each element
- Connect (assemble) the elements at the nodes to form an approximate system of equations for the whole structure
- Solve the system of equations involving unknown quantities at the nodes (e.g. displacements)
- Calculate desired quantities (e.g. strains and stresses) at selected elements

Finite Elements – Why?

FEM allows discretization of bodies with arbitrary shape. Originally designed for problems in static elasticity.

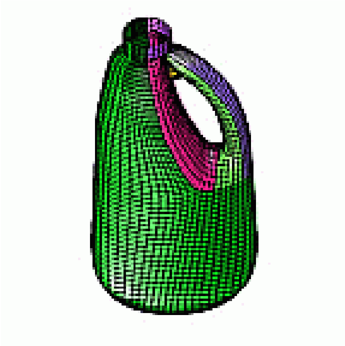
FEM is the most widely applied computer simulation method in engineering.

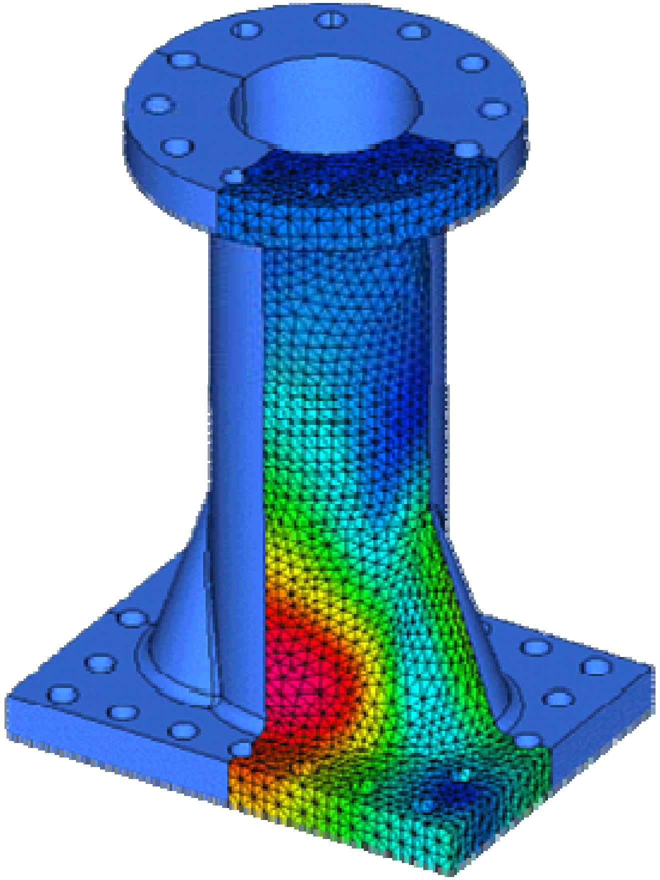
Today spectral elements is an attractive new method with applications in seismology and geophysical fluid dynamics

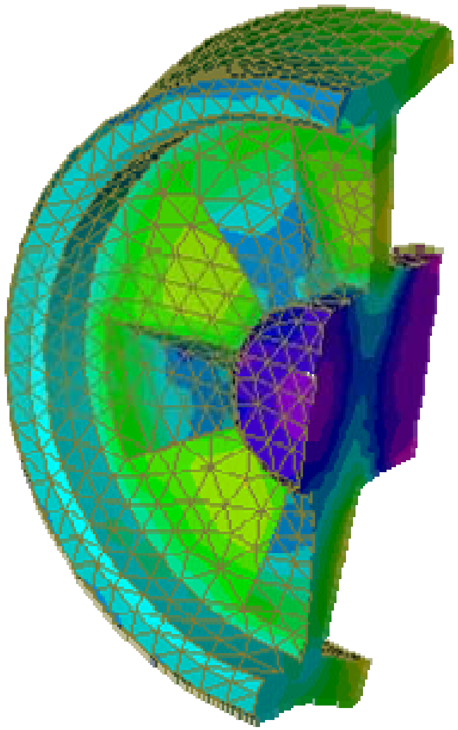
The required grid generation techniques are interfaced with graphical techniques (CAD).

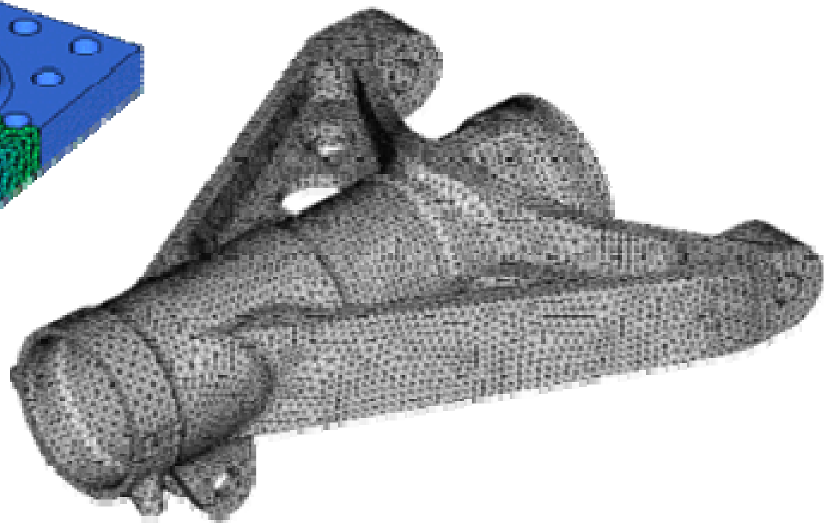
Today a large number of commercial FEM software is available (e.g. ANSYS, SMART, MATLAB, ABACUS, etc.)

Finite Elements - Examples



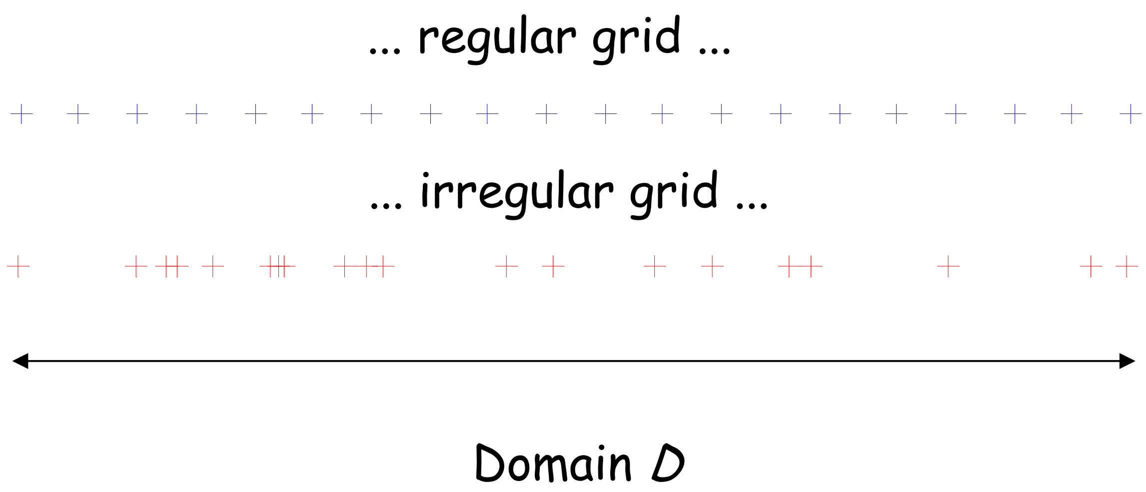






Discretization – finite elements

As we are aiming to find a numerical solution to our problem it is clear we have to discretize the problem somehow. In FE problems – similar to FD – the functional values are known at a discrete set of points.



The key idea in FE analysis is to approximate all functions in terms of basis functions φ, so that

$$\boxed{u \approx \tilde{u} = \sum_{i=1}^{N} c_i \varphi_i}$$

Finite elements – basic formulation

Let us start with a simple linear system of equations

$$\boxed{\mathbf{Ax} = \mathbf{b}} \qquad | * \mathbf{y}$$

and observe that we can generally multiply both sides of this equation with **y** without changing its solution. Note that **x**,**y** and **b** are vectors and **A** is a matrix.

$$\rightarrow \mathbf{yAx} = \mathbf{yb} \qquad \mathbf{y} \in \Re^{\mathbf{n}}$$

We first look at Poisson's equation (e.g., wave equation without time dependence)

$$\boxed{-\Delta u(x) = f(x)}$$

where u is a scalar field, f is a source term and in 1-D

$$\Delta = \nabla^2 = \frac{\partial^2}{\partial x^2}$$

Formulation – Poisson's equation

We now multiply this equation with an arbitrary function v(x), (dropping the explicit space dependence)

$$-\Delta u v = f v$$

... and integrate this equation over the whole domain. For reasons of simplicity we define our physical domain *D* in the interval [0, 1].

$$-\int_D \Delta u v = \int_D f v$$

$$-\int_0^1 \Delta u v dx = \int_0^1 f v dx$$

... why are we doing this? ... be patient ...

Partial Integration

... partially integrate the left-hand-side of our equation ...

$$-\int_0^1 \Delta u v dx = \int_0^1 f v dx$$

$$-\int_0^1 \Delta u v dx = \boxed{\left[\nabla u v\right]_0^1} + \int_0^1 \nabla v \nabla u dx$$

we assume for now that the derivatives of u at the boundaries vanish so that for our particular problem

$$-\int_0^1 \Delta u v dx = \int_0^1 \nabla v \nabla u dx$$

... so that we arrive at ...

$$\int_0^1 \nabla u \nabla v dx = \int_0^1 f v dx$$

... with u being the unknown function. This is also true for our approximate numerical system

$$\int_0^1 \nabla \tilde{u} \nabla v dx = \int_0^1 f v dx$$

... where ...

$$\tilde{u} = \sum_{i=1}^{N} c_i \varphi_i$$

was our choice of approximating u using basis functions.

The basis functions

we are looking for functions φ_i with the following property

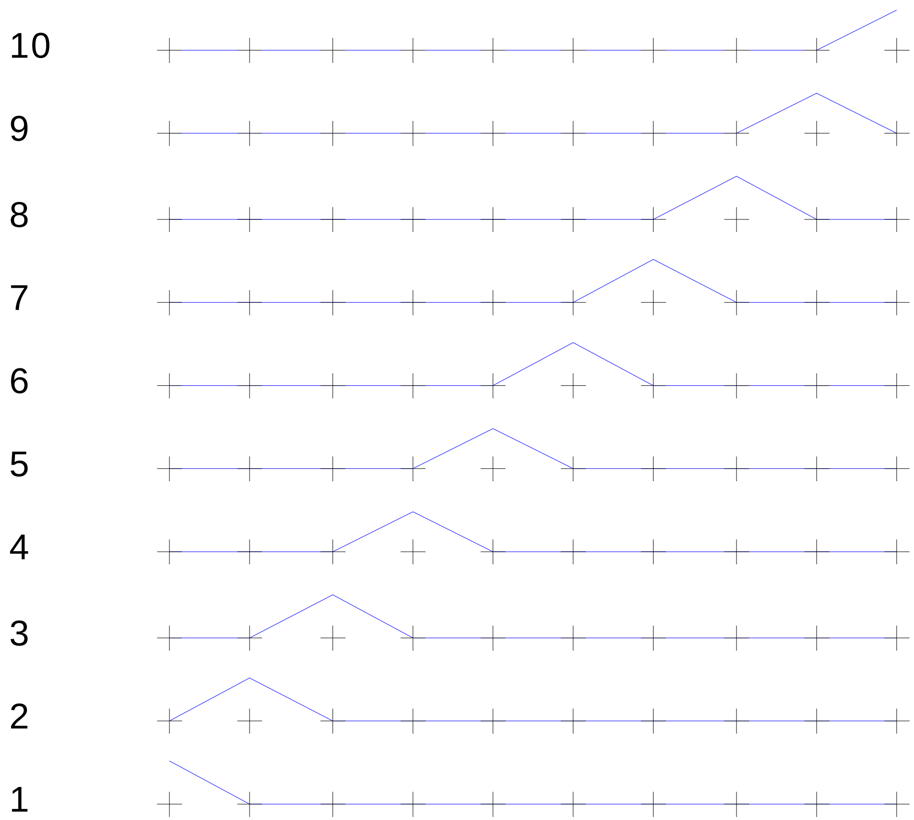
$$\varphi_i(x) = \begin{cases} 1 & for \quad x = x_i \\ 0 & for \quad x = x_j,\ j \neq i \end{cases}$$

... otherwise we are free to choose any function ...

The simplest choice are of course linear functions:

+ grid nodes

blue lines – basis functions φ_i



The discrete system

The ingredients:

$$v = \varphi_k$$

$$\tilde{u} = \sum_{i=1}^{N} c_i \varphi_i$$

$$\int_0^1 \nabla \tilde{u} \nabla v dx = \int_0^1 f v dx$$

$$\int_0^1 \nabla \left(\sum_{i=1}^{n} c_i \varphi_i \right) \nabla \varphi_k dx = \int_0^1 f \varphi_k dx$$

... leading to ...

The discrete system

... the coefficients c_k are constants so that for one particular function φ_k this system looks like ...

$$\boxed{\sum_{i=1}^{n} c_i \int_0^1 \nabla \varphi_i \nabla \varphi_k dx = \int_0^1 f \varphi_k dx}$$

... probably not to your surprise this can be written in matrix form

$$\boxed{b_i A_{ik} = g_k}$$

$$\boxed{A_{ik}^T b_i = g_k}$$

The solution

... with the even less surprising solution

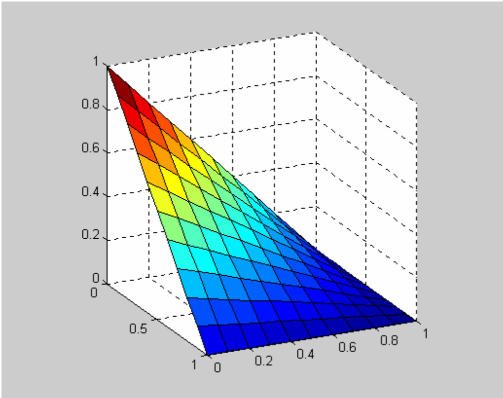
$$\boxed{b_i = \left(A_{ik}^T\right)^{-1} g_k}$$

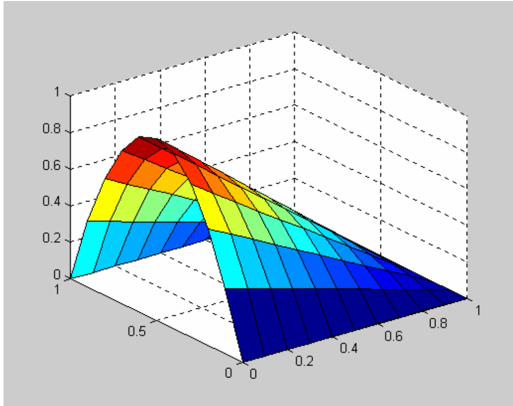
remember that while the b_i's are really the coefficients of the basis functions these are the actual function values at node points i as well because of our particular choice of basis functions.

Basis functions and element

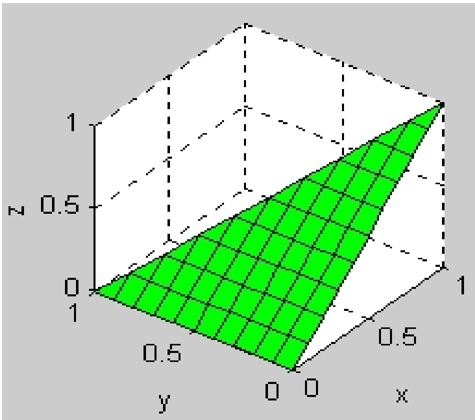
Linear | Quadratic

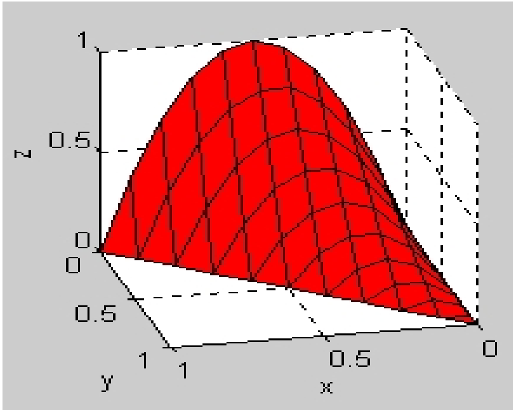
Trangles





Quadrangles





The Acoustic Wave Equation 1-D

How do we solve a time-dependent problem such as the acoustic wave equation?

$$\partial_t^2 u - v^2 \Delta u = f$$

where v is the wave speed.
using the same ideas as before we multiply this equation with an arbitrary function and integrate over the whole domain, e.g. [0,1], and after partial integration

$$\int_0^1 \partial_t^2 u \varphi_j dx - v^2 \int_0^1 \nabla u \nabla \varphi_j dx = \int_0^1 f \varphi_j dx$$

.. we now introduce an approximation for u using our previous basis functions...

The Acoustic Wave Equation 1-D

$$u \approx \tilde{u} = \sum_{i=1}^{N} c_i(t)\varphi_i(x)$$

note that now our coefficients are time-dependent!
... and ...

$$\partial_t^2 u \approx \partial_t^2 \tilde{u} = \partial_t^2 \sum_{i=1}^{N} c_i(t)\varphi_i(x)$$

together we obtain

$$\left[\sum_i \partial_t^2 c_i \int_0^1 \varphi_i \varphi_j dx\right] + v^2 \left[\sum_i c_i \int_0^1 \nabla\varphi_i \nabla\varphi_j dx\right] = \int_0^1 f\varphi_j$$

which we can write as ...

Time extrapolation

$$\underbrace{\left[\sum_i \partial_t^2 c_i \int_0^1 \varphi_i \varphi_j dx\right]}_{\substack{M \\ \text{mass matrix}}} + v^2 \underbrace{\left[\sum_i c_i \int_0^1 \nabla \varphi_i \nabla \varphi_j dx\right]}_{\substack{A \\ \text{stiffness matrix}}} = \underbrace{\int_0^1 f \varphi_j}_{b}$$

... in Matrix form ...

$$\boxed{M^T \ddot{c} + v^2 A^T c = g}$$

... remember the coefficients c correspond to the actual values of u at the grid points for the right choice of basis functions ...

How can we solve this time-dependent problem?

FD extrapolation

$$M^T\ddot{c} + v^2 A^T c = g$$

... let us use a finite-difference approximation for the time derivative ...

$$M^T\left(\frac{c_{k+1} - 2c + c_{k-1}}{dt^2}\right) + v^2 A^T c_k = g$$

... leading to the solution at time t_{k+1}:

$$c_{k+1} = \left[(M^T)^{-1}(g - v^2 A^T c_k)\right] dt^2 + 2c_k - c_{k-1}$$

we already know how to calculate the matrix A but how can we calculate matrix M?

Matrix assembly

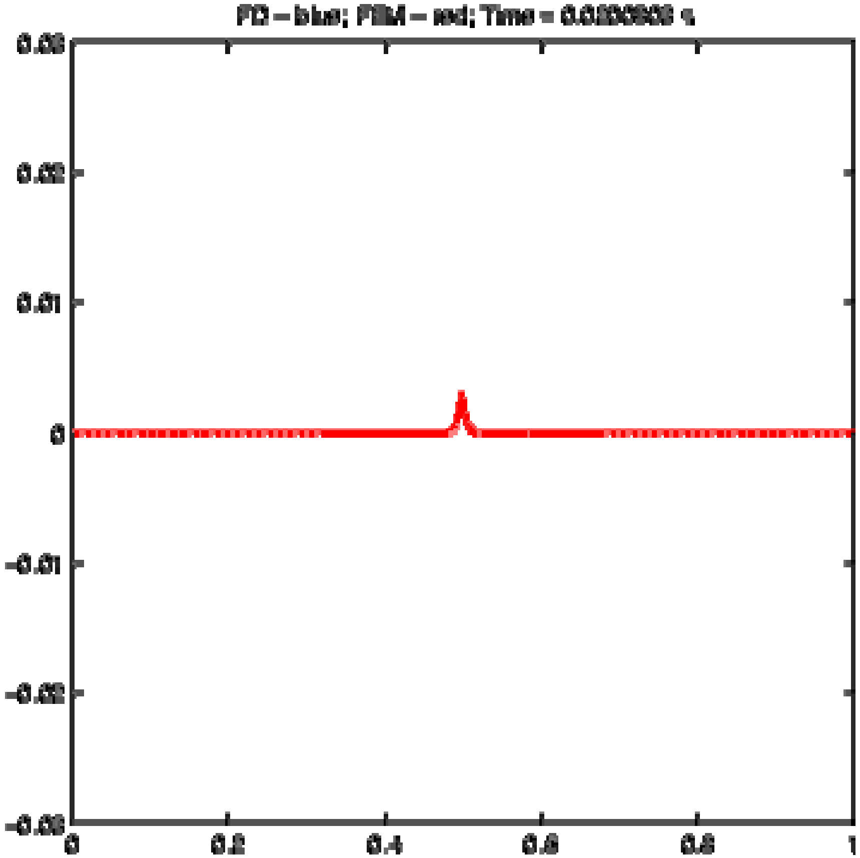
M_{ij}

```
% assemble matrix Mij

M=zeros(nx);

for i=2:nx-1,
   for j=2:nx-1,
      if i==j,
         M(i,j)=h(i-1)/3+h(i)/3;
      elseif j==i+1
         M(i,j)=h(i)/6;
      elseif j==i-1
         M(i,j)=h(i)/6;
      else
         M(i,j)=0;
      end
   end
end
```

A_{ij}

```
% assemble matrix Aij

A=zeros(nx);

for i=2:nx-1,
   for j=2:nx-1,
      if i==j,
         A(i,j)=1/h(i-1)+1/h(i);
      elseif i==j+1
         A(i,j)=-1/h(i-1);
      elseif i+1==j
         A(i,j)=-1/h(i);
      else
         A(i,j)=0;
      end
   end
end
```

Numerical example - regular grid



This is a movie obtained with the sample Matlab program: femfd.m

Finite Elements - Summary

- FE solutions are based on the "weak form" of the partial differential equations
- FE methods lead in general to a linear system of equations that has to be solved using matrix inversion techniques (sometimes these matrices can be diagonalized)
- FE methods allow rectangular (hexahedral), or triangular (tetrahedral) elements and are thus more flexible in terms of grid geometry
- The FE method is mathematically and algorithmically more complex than FD
- The FE method is well suited for elasto-static and elasto-dynamic problems (e.g. crustal deformation)