The Finite Element Method

Heiner Igel

Department of Earth and Environmental Sciences
Ludwig-Maximilians-University Munich

The Finite-Element Method: 1D Elastic Wave Equation

1D Elastic Wave Equation

Apply the Galerkin principle to the 1D elastic wave equation

$$\rho \partial_t^2 u = \partial_x \; \mu \; \partial_x u + f \; .$$

where again we omit space and time-dependencies. From now on we assume that the properties of the medium, density ρ, and shear modulus μ are both space-dependent. We obtain the weak form as

$$\int_D \rho \; \partial_t^2 \; u \; \varphi_j \; dx = \int_D \partial_x \; \mu \; \partial_x \; u \; \varphi_j \; dx + \int_D f \; \varphi_j \; dx \; .$$

Integration by parts of the term containing the space derivatives leads to

$$\int_D \partial_x \; \mu \; \partial_x \; u \; \varphi_j \; dx = \left[\mu \partial_x \; u \; \varphi_j \right] - \int_D \mu \; \partial_x \; u \; \partial_x \; \varphi_j \; dx$$

1D Elastic Wave Equation

Assuming a stress-free boundary condition leads to

$$\int_D \rho \, \partial_t^2 \, u \, \varphi_j \, dx + \int_D \mu \, \partial_x \, u \, \partial_x \, \varphi_j \, dx = \int_D f \, \varphi_j \, dx$$

where u is the continuous unknown displacement field. We replace the exact displacement field by an approximation $\overline{u}$ of the form

$$u(x,t) \rightarrow \overline{u}(x,t) = \sum_{i=1}^{N} u_i(t) \, \varphi_i(x)$$

where the coefficients u_i are expected to correspond to a discrete representation of the solution field. The wave equation bevomes

$$\int_D \rho \, \partial_t^2 \, \overline{u} \, \varphi_j \, dx + \int_D \mu \, \partial_x \, \overline{u} \, \partial_x \, \varphi_j \, dx = \int_D f \, \varphi_j \, dx$$

1D Elastic Wave Equation

Turning the continuous weak form into a system of linear equations

$$\int_D \rho\, \partial_t^2 \left(\sum_{i=1}^{N} u_i(t)\, \varphi_i(x) \right) \varphi_j \, dx$$
$$+ \int_D \mu\, \partial_x \left(\sum_{i=1}^{N} u_i(t)\, \varphi_i(x) \right) \partial_x\, \varphi_j \, dx$$
$$= \int_D f\, \varphi_j \, dx \, .$$

Changing the order of integration and summation we obtain

$$\sum_{i=1}^{N} \partial_t^2 u_i \int_D \rho\, \varphi_i\, \varphi_j \, dx + \sum_{i=1}^{N} u_i \int_D \mu\, \partial_x\, \varphi_i\, \partial_x\, \varphi_j \, dx = \int_D f\, \varphi_j \, dx$$

using the fact that the unknown coefficients u_i only depend on time.

1D Elastic Wave Equation

Using matrix-vector notation with the following definitions for the time-dependent solution vector of displacement values $\mathbf{u}(t)$, mass matrix $\mathbf{M}$, the already well-known stiffness matrix $\mathbf{K}$, and the source vector $\mathbf{f}$:

$$\mathbf{u}(t) \rightarrow u_i(t)$$

$$\mathbf{M} \rightarrow M_{ij} = \int_D \rho \, \varphi_i \, \varphi_j \; dx$$

$$\mathbf{K} \rightarrow K_{ij} = \int_D \mu \, \partial_x \, \varphi_i \, \partial_x \, \varphi_j dx$$

$$\mathbf{f} \rightarrow f_j = \int_D f \, \varphi_j \; dx \; .$$

1D Elastic Wave Equation

Thus we can write the system of equations as

$$\ddot{\mathbf{u}}\mathbf{M} + \mathbf{u}\mathbf{K} = \mathbf{f}$$

or with transposed system matrices as

$$\mathbf{M}^T\ddot{\mathbf{u}} + \mathbf{K}^T\mathbf{u} = \mathbf{f}\ .$$

For the second time-derivative we use a standard finite-difference approximation

$$\ddot{\mathbf{u}} = \partial_t^2\mathbf{u} \approx \frac{\mathbf{u}(t+dt) - 2\mathbf{u}(t) + \mathbf{u}(t-dt)}{dt^2}$$

1D Elastic Wave Equation

Replacing the original partial derivative with respect to time to obtain

$$\mathbf{M}^T \left[\frac{\mathbf{u}(t + dt) - 2\mathbf{u}(t) + \mathbf{u}(t - dt)}{dt^2} \right] = \mathbf{f} - \mathbf{K}^T \mathbf{u} \; .$$

Starting from an initial state $\mathbf{u}(t = 0) = 0$ we can determine the displacement field at time $t + dt$ by

$$\mathbf{u}(t + dt) = dt^2 (\mathbf{M}^T)^{-1} \left[\mathbf{f} - \mathbf{K}^T \mathbf{u} \right] + 2\mathbf{u}(t) - \mathbf{u}(t - dt) \; .$$

The System Matrices

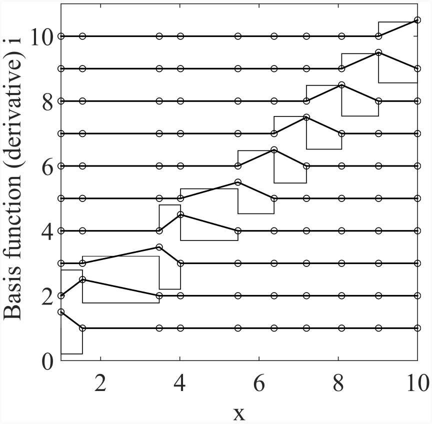
To calculate the entries of the system matrices we transform the space coordinate into a local system

$$\xi = x - x_i$$
$$h_i = x_{i+1} - x_i \ .$$

However, now we allow the element size h_i to vary. With the definition above element i is defined in the interval $x \in [x_i, x_{i+1}]$. In the local coordinate system the basis functions are defined by

$$\varphi_i(\xi) = \begin{cases} \frac{\xi}{h_{i-1}} & \text{for } -h_{i-1} < \xi \leq 0 \\ 1 - \frac{\xi}{h_i} & \text{for } 0 < \xi < h_i \\ 0 & \text{elsewhere} \end{cases}$$

The System Matrices



with the corresponding derivatives

$$\partial_x \varphi_i(\xi) = \begin{cases} \frac{1}{h_{i-1}} & \text{for } -h_{i-1} < \xi \leq 0 \\ -\frac{1}{h_i} & \text{for } 0 < \xi < h_i \\ 0 & \text{elsewhere} \end{cases}$$

Example of a finite-element domain with irregular element sizes h_i. The basis functions (thick solid lines) are illustrated with the normalized derivatives (thin solid lines).

The Mass Matrix

Looking at the global definition of the mass matrix **M** with components

$$M_{ij} = \int_D \rho\, \varphi_i\, \varphi_j\, dx$$

the only non-zero entries are around the diagonal and are of components $M_{i,i-1}$ $M_{i,i}$ and $M_{i,i+1}$ for $i = 2, ..., N-1$. Elements M_{11} and M_{NN} have to be treated separately. For the diagonal elements we obtain

$$M_{ii} = \int_D \rho\, \varphi_i\, \varphi_i\, dx \;=\; \int_{D_\xi} \rho\, \varphi_i\, \varphi_i\, d\xi$$

in the local coordinate system.

The Mass Matrix

Integration has to be carried out over the elements to the left and right of the boundary points x_i. We thus obtain

$$M_{ii} = \rho_{i-1} \int_{-h_{i-1}}^{0} \left(\frac{\xi}{h_{i-1}} + 1 \right)^2 d\xi + \rho_i \int_0^{h_i} \left(1 - \frac{\xi}{h_i} \right)^2 d\xi$$

$$= \frac{1}{3} \left(\rho_{i-1} \; h_{i-1} + \rho_i \; h_i \right)$$

For the off-diagonal elements the basis functions overlap only in one element

$$M_{i,i-1} = \rho_{i-1} \int_{-h_{i-1}}^{0} \left(\frac{\xi}{h_{i-1}} + 1 \right) \frac{-\xi}{h_{i-1}} d\xi = \frac{1}{6} \, \rho_{i-1} \; h_{i-1}$$

or

$$M_{i,i+1} = \rho_i \int_0^{h_i} \frac{\xi}{h_i} \left(1 - \frac{\xi}{h_i} \right) d\xi = \frac{1}{6} \, \rho_i \; h_i \; .$$

The Mass Matrix

Just to illustrate the banded nature of the mass matrix, assuming constant element size h and density ρ the mass matrix is given by

$$\mathbf{M} = \frac{\rho\, h}{6} \begin{pmatrix} \ddots & & & & 0 \\ 1 & 4 & 1 & & \\ & 1 & 4 & 1 & \\ & & 1 & 4 & 1 \\ 0 & & & & \ddots \end{pmatrix}$$

Note: In the general case with varying element size the mass matrix is not symmetric.

Stiffness matrix

The same concepts apply to the stiffness matrix. We move to the local coordinate system by

$$K_{ij} = \int_D \mu \, \partial_x \varphi_i \, \partial_x \varphi_j \, dx \; = \int_{D_\xi} \mu \, \partial_\xi \varphi_i \, \partial_\xi \varphi_j \, d\xi$$

to obtain for a diagonal element, assuming constant shear modulus μ inside each element

$$\begin{aligned} K_{ii} &= \mu_{i-1} \int_{-h_{i-1}}^{0} \left(\frac{1}{h_{i-1}} \right)^2 d\xi + \mu_i \int_0^{h_i} \left(-\frac{1}{h_i} \right)^2 d\xi \\ &= \frac{\mu_{i-1}}{h_{i-1}} + \frac{\mu_i}{h_i} \end{aligned}$$

Stiffness matrix

For the off-diagonal elements

$$K_{i,i+1} = \mu_i \int_0^{h_i} \left(-\frac{1}{h_i}\right)\left(\frac{1}{h_i}\right) d\xi = -\frac{\mu_i}{h_i}$$

$$K_{i,i-1} = \mu_{i-1} \int_{-h_{i-1}}^{0} \left(-\frac{1}{h_{i-1}}\right)\left(\frac{1}{h_{i-1}}\right) d\xi = -\frac{\mu_{i-1}}{h_{i-1}}$$

while all other elements of the stiffness matrix are zero. For example, assuming constant shear modulus and element size the stiffness matrix reads

$$\mathbf{K} = \frac{\mu}{h} \begin{pmatrix} \ddots & & & & 0 \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & -1 & 2 & -1 \\ 0 & & & \ddots & \end{pmatrix}.$$

Simulation Example

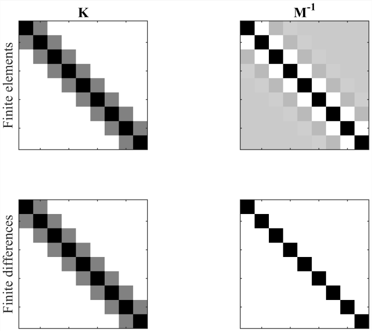
Parameter	Value
x_{max}	10000m
nx	1000
v_s	3000 m/s
ρ	2500 kg/m^3
h	10 m
eps	0.5
f_0	20 Hz

```python
# Time extrapolation
# -----------------------------------------
for it in range(nt):
    (...)

    # -----------------------------------
    # Finite Difference Method
    pnew = (dt**2) * Mf @ (D @ p + f/dx*src[it]) + 2*p - pold
    pold, p = p, pnew
```

Python code segment for a finite-difference algorithm in matrix-vector form.

Simulation Example



The structure of the system matrices for the finite-element method are compared with the finite-difference method formulated with matrix-vector operations. **Top row:** Stiffness and inverse mass matrix for the finite element method. **Bottom row:** Stiffness (differential) matrix and diagonal mass matrix for the finite-difference method.

Simulation Example

Mass matrix for the general case of varying element size

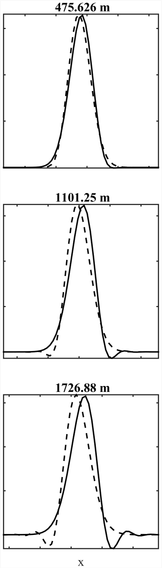
```
# ---------------------------------------------------------------
# Mass matrix M_ij (Eq 6.56)
# ---------------------------------------------------------------
M = np.zeros((nx,nx), dtype=float)
for i in range(1, nx-1):
    for j in range (1, nx-1):
        if j==i:
            M[i,j] = (ro[i-1]*h[i-1] + ro[i]*h[i])/3
        elif j==i+1:
            M[i,j] = ro[i]*h[i]/6
        elif j==i-1:
            M[i,j] = ro[i-1]*h[i-1]/6
        else:
            M[i,j] = 0

# Corner elements
M[0,0] = ro[0]*h[0]/3
M[nx-1,nx-1] = ro[nx-1]*h[nx-2]/3
# Invert M
Minv = np.linalg.inv(M)
(...)
```

Finite element time extrapolation

```
# ---------------------------------------------------------------
# Time extrapolation
# ---------------------------------------------------------------
for it in range(nt):
    # --------------------------------------------------------
    # Finite Element Method
    unew=(dt**2)*Minv@(f*src[it]-K@u)+2*u-uold
    uold, u = u, unew
            (...)
```

Simulation Example



Snapshots of the displacement wavefield calculated with the finite-element method (solid line) are compared with the finite-difference method (dotted line) at various distances from the source using the same parameters. The length of the window is 500m.

h - adaptivity

h - adaptivity: The simplicity with which the element size can vary due to geometrical features or the velocity model.
Think about an Earth model in which the seismic velocities have strong variations.

- P-velocity in the oceans (1.5 km/s)
- P-velocities at the core-mantle boundary (13 km/s)
- Any numerical scheme with globally constant element size has to be accurate for the shortest wavelength
- $\Longrightarrow$ Regions with higher velocities will be oversampled

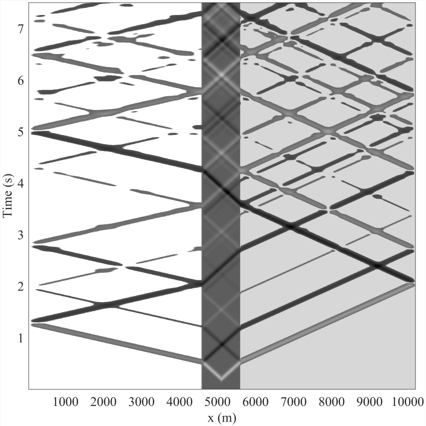
h - adaptivity

	Left	Middle	Right
x	4600m	1000m	4600m
v_s	$6000\frac{m}{s}$	$1500\frac{m}{s}$	$3000\frac{m}{s}$
dx	40m	10m	20m
ρ	$2500\frac{kg}{m^3}$	$2500\frac{kg}{m^3}$	$2500\frac{kg}{m^3}$

Parameter	Value
nt	18000
dt	3.3 ms
f_0	5 Hz
eps	0.5

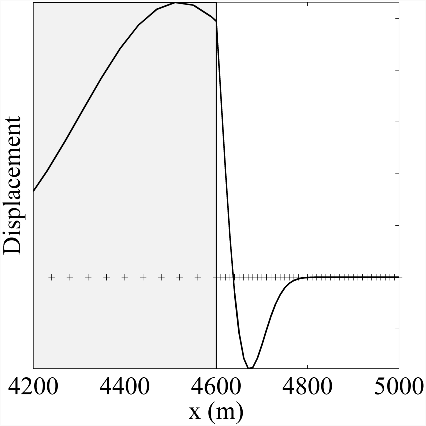
We demonstrate this in the 1D case with a strongly heterogeneous velocity model in which the number of grid points per wavelength is kept constant in the entire physical domain. The model mimics the situation in a fault zone with a central low-velocity zone (damage zone) with different material properties on the two sides of the fault.

h - adaptivity



Snapshots of displacement values are shown as a function time. Where displacement amplitudes are below a threshold, the velocity models is shown in gray scale. Note the polarity change of the reflections at the boundaries and the slope of the signals in the $x - t$ plane indicated their velocities.

h - adaptivity



Detail of the finite-element simulation with varying element size at one of the domain boundaries. The crosses indicate the element boundaries and the changing element size. Note the continuous but non-differentiable behaviour of the displacement field at the interface.

Shape Functions in 1D and 2D

1D

Let us recall how we replaced the originally continuous unknown field $u(x)$ by a sum over some basis functions φ_i

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

denoting the coefficients of the basis functions by c_i. Mapping all elements to a local coordinate system such

$$\xi = \frac{x - x_i}{x_{i+1} - x_i}$$

where our reference element is defined with $\xi \in [0, 1]$.

Linear Shape Functions

We put ourselves at element level and assume that our unknown function $u(\xi)$ is linear

$$u(\xi) = c_1 + c_2\xi$$

where c_i are real coefficients. Each element has two node points, namely the element boundaries at $\xi_{1,2} = 0, 1$. This leads to the following conditions and solutions for coefficients c_i

$$\begin{aligned} u_1 &= c_1 \rightarrow c_1 = u_1 \\ u_2 &= c_1 + c_2 \rightarrow c_2 = -u_1 + u_2 \end{aligned}.$$

Linear Shape Functions

This can also be written in matrix notation which will help us when dealing with high-order systems. We obtain

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

and using matrix inversion

$$\begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} .$$

Linear Shape Functions

With appropriate matrix and vector definitions this can be written as

$$\mathbf{u} = \mathbf{A}\mathbf{c} \rightarrow \mathbf{c} = \mathbf{A}^{-1}\mathbf{u}$$

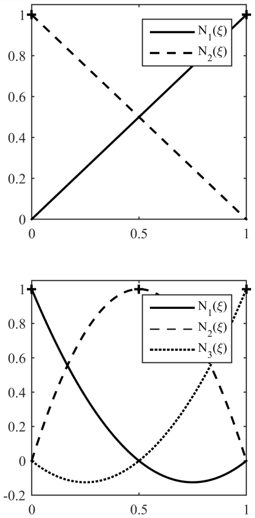
implying that to obtain coefficients **c** we need to calculate the inverse of **A**.

$$\begin{aligned} u(\xi) &= u_1 + (-u_1 + u_2)\xi \\ &= u_1(1 - \xi) + u_2\xi \\ &= u_1 N_1(\xi) + u_2 N_2(\xi) \end{aligned}$$

where we introduced a novel concept, the shape functions $N_i(\xi)$ with the following form

$$N_1(\xi) = 1 - \xi \quad , \quad N_2(\xi) = \xi \ .$$

Linear Shape Functions



The sum over the weighted shape function of general order N gives the approximate *continuous* representation of the solution field $u(\xi)$ inside the element.

$$u(\xi) = \sum_{i=1}^{N} u_i N_i(\xi)$$

Top: Linear shape functions as used in the development of the finite-element solution to static and dynamic elastic problems. Node points are indicated by crosses. **Bottom:** Quadratic shape functions requiring one more node point at the center of the element.

Quadratic Shape Functions

Describing our solution field by quadratic functions requires

$$u(\xi) = c_1 + c_2\xi + c_3\xi^2$$

where we added one more node point at the center of the element $\xi_{1,2,3} = 0, \frac{1}{2}, 1$. With these node locations we obtain

$$\begin{aligned} u_1 &= c_1 \\ u_2 &= c_1 + 0.5c_2 + 0.25c_3 \\ u_3 &= c_1 + c_2 + c_3 \end{aligned}$$

and after inverting the resulting system matrix **A**

$$(A)^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{bmatrix}$$

Quadratic Shape Functions

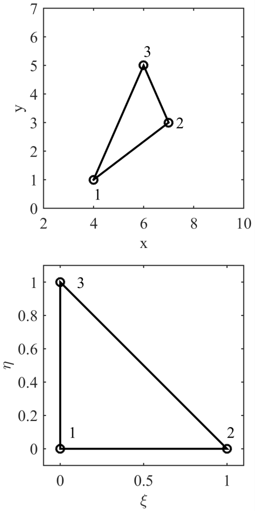
We can represent the final quadratic solution field inside the element with

$$
\begin{aligned}
u(\xi) =& c_1 + c_2\xi + c_3\xi^2 \\
=& u_1(1 - 3\xi + 2\xi^2) + \\
=& u_2(4\xi - 4\xi^2) + \\
=& u_3(-xi + 2\xi^2)
\end{aligned}
$$

resulting in the following shape functions

$$
\begin{aligned}
N_1(\xi) &= 1 - 3\xi + 2\xi^2 \\
N_2(\xi) &= 4\xi - 4\xi^2 \\
N_3(\xi) &= -\xi + 2\xi^2
\end{aligned}
$$

Triangular Shape Functions



Triangular elements. Mapping of physical coordinates (x, y, top) to a local reference frame (bottom) with coordinates ξ, η.

Triangular Shape Functions

To perform the integration operations when calculating the system matrices we move to the local coordinate system $\xi, \eta \in [0, 1]$ (sometimes the reference space is chosen to be $[-1, 1]$) through

$$x = x_1 + (x_2 - x_1)\xi + (x_3 - x_1)\eta$$
$$y = y_1 + (y_2 - y_1)\xi + (y_3 - y_1)\eta \ .$$

We seek to describe a linear function inside our triangle, therefore

$$u(\xi, \eta) = c_1 + c_2\xi + c_3\eta \ .$$

Triangular Shape Functions

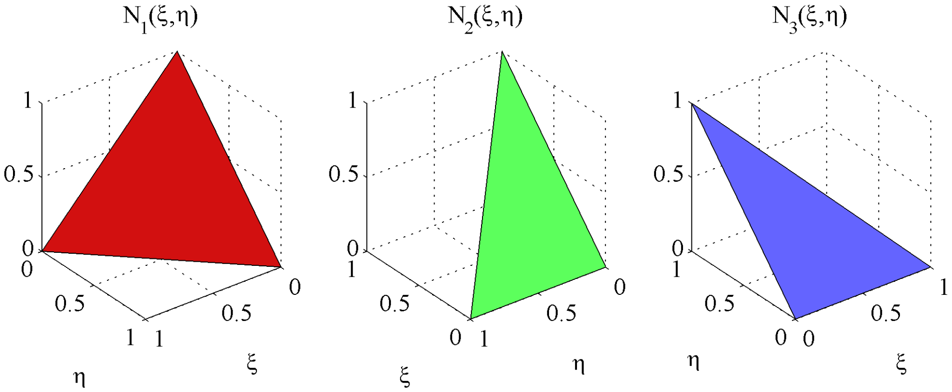
We only know our function at the corners of the reference triangle, therefore the constraints for coefficients c_i are

$$\begin{aligned} u_1 &= u(0,0) = c_1 \\ u_2 &= u(1,0) = c_1 + c_2 \\ u_3 &= u(0,1) = c_1 + c_3 \quad . \end{aligned}$$

This leads - using the same matrix inversion approach described above to - the following shape functions for triangular elements

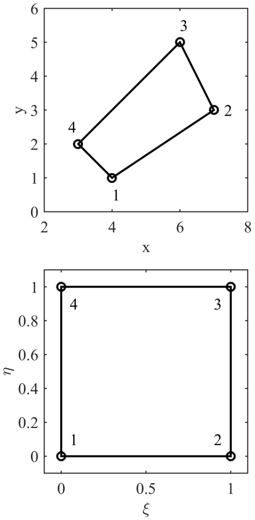
$$\begin{aligned} N_1(\xi,\eta) &= 1 - \xi - \eta \\ N_2(\xi,\eta) &= \xi \\ N_3(\xi,\eta) &= \eta \; . \end{aligned}$$

Triangular Shape Functions



Triangular elements. The three corner nodes lead to an equivalent number of shape functions $N_i(\xi, \eta)$ with unit value at one of the corners.

Rectangular Shape Functions



Quadrilateral elements. Mapping of physical coordinates (x, y, top) to a local reference frame (bottom) with coordinates ξ, η.

Rectangular Shape Functions

Accordingly, shape functions can be derived for general quadrilateral elements.
We map space to a local coordinate system through

$$x = x_1 + (x_2 - x_1)\xi + (x_4 - x_1)\eta + (x_3 - x_2)\xi\eta$$

$$y = y_1 + (y_2 - y_1)\xi + (y_4 - y_1)\eta + (y_3 - y_2)\xi\eta \; .$$

Requiring linear behaviour of the function inside the element

$$u(\xi, \eta) = c_1 + c_2\xi + c_3\eta + c_4\xi\eta$$

we obtain the following shape functions

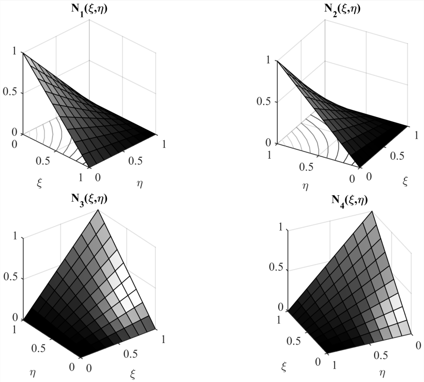
$$N_1(\xi, \eta) = (1 - \xi)(1 - \eta)$$

$$N_2(\xi, \eta) = \xi(1 - \eta)$$

$$N_3(\xi, \eta) = \xi\eta$$

$$N_4(\xi, \eta) = (1 - \xi)\eta \quad .$$

Rectangular Shape Functions



Rectangular reference elements. The four shape functions $N_i(\xi,\eta)$ with unity value at one of the corners.

Summary

- The finite-element method was originally developed mostly for static structural engineering problems.
- The *element* concept relates to describing the solution field in an analogous way inside each element, thereby facilitating the required calculations of the system matrices.
- The finite-element approach can in principle be applied to elements of arbitrary shape. Most used shapes are triangles (tetrahedra) or quadrilateral (hexahedral) structures.
- The finite-element method is a series expansion method. The continuous solution field is replaced by a finite sum over (not necessarily orthogonal) basis functions.
- For static elastic problems or the elastic wave propagation problem finite-element analysis leads to a (large) system of linear equations. In general, the matrices are of size $N \times N$ where N is the number of degrees of freedom.
- Because of the specific interpolation properties of the basis functions, their coefficients take the meaning of the values of the solution field at specific node points.

Summary

- In an initialization step the global stiffness and mass matrices have to be calculated. They depend on integrals over products of basis functions and their derivatives.
- If equation parameters (e.g., elastic parameters, density) vary inside elements, then numerical integration has to be performed.
- The stress-free surface condition can be implicitly solved. This is a major advantage for example for the simulation of surface waves.
- The classic finite-element method plays a minor role in seismology as its high-order sister, the spectral-element method, is more efficient.

Comprehension questions

1. In which community was the finite-element method primarily developed? Give some typical problems.
2. What are weak and strong forms of partial differential equations? Give examples.
3. Discuss the pros and cons of the finite-element method vs. low-order finite-difference methods.
4. Present and discuss problem classes that can be handled well with the finite-volume method, compare to problems better handled with other methods.
5. Compare the spatial discretization strategies of finite-element and finite-difference methods.
6. Describe the concept of shape functions.
7. Discuss qualitatively (use sketches) the use of basis functions. Compare with the pseudospectral method.
8. Is the finite-element method a global or a local scheme?
9. Why does the finite-element method require the solution of a (possible huge) system of linear equations? What is the consequence for parallel computing?
10. Why is the classic linear finite-element method not so much used for seismological research today?

Theoretical questions

11 The scalar advection equation is simply

$$\partial_t q(x,t) + c(x)\partial_x q(x,t) = 0$$

where $q(x,t)$ is the scalar quantity to be advected and $c(x)$ is the advection velocity. Write down the weak form of this equation and perform integration by parts. What happens to the anti-derivative? Does it cancel out at the boundaries like in the 1D elastic wave equation? Note: This is the point of departure for the discontinuous Galerkin method).

12 Are the linear basis functions

$$\varphi_i(x) = \begin{cases} \frac{x - x_{i-1}}{x_i - x_{i-1}} & \text{for } x_{i-1} < x \leq x_i \\ \frac{x_{i+1} - x}{x_{i+1} - x_i} & \text{for } x_i < x < x_{i+1} \\ 0 & \text{elsewhere} \end{cases}$$

orthogonal?

13 Calculate all entries of the stiffness matrix $A_{ij} = \int_D \mu \partial_x \, \varphi_i \, \partial_x \, \varphi_j$ for a static elastic problem with $\mu = 70 GPa$ and $h = 1 m$ for a problem with $n = 5$ degrees of freedom.

Theoretical questions

14 A finite-element system has the following parameters: Element sizes $h = [1, 3, 0.5, 2, 4]$, density $\rho = [2, 3, 2, 3, 2] kg/m^3$. Calculate the entries of the mass matrix given by $M_{ij} = \int_D \rho\, \varphi_i\, \varphi_j\, dx$ using linear basis functions.

15 H-adaptivity. For the simulation with varying velocities and element size, calculate the time step required for $\epsilon = 0.5$ in each of the subdomains. Discuss the result.

16 Follow the approach of the derivation of shape functions and derive the cubic case in 1D: $u(x) = c_1 + c_2\xi + c_3\xi^2 + c_4\xi^3$. What are key differences to quadratic and linear cases?

17 Derive the quadratic shape functions $N(\xi, \eta)$ for 2D triangles with the following node points:

$$P_1(0, 0), P_2(1, 0), P_3(0, 1),$$
$$P_4(1/2, 0), P_5(1/2, 1/2), P_6(0, 1/2)$$

18 Derive the quadratic shape functions $N(\xi, \eta)$ for 2D rectangles with the following node points:

$$P_1(0, 0), P_2(1/2, 0), P_3(1, 0), P_4(1, 1/2),$$
$$P_5(1, 1), P_6(1/2, 1), P_7(0, 1), P_8(0, 1/2)$$

Programming Exercise

19 Static elastic case. Extend the formulation given to arbitrary element sizes (calculation of stiffness matrix and source vector). Make examples and compare with the regular grid finite-difference solution (relaxation method).

20 1D elastic case. Determine numerically the stability limit and compare with the finite-difference solution.

21 Initialize a strongly heterogeneous velocity model with spatially varying element size. Try to match the results with a regular grid finite-difference implementation of the same model. Discuss the two approaches in terms of time step, run time, memory usage.

22 Plot the high-order 2D shape functions derived in the problems above.