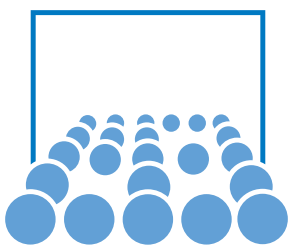
Scientific Computing I

Module 7: An Introduction to Finite Element Methods

Tobias Neckel

Winter 2015/2016



Part I: Introduction to Finite Element Methods

The Model Problem

FEM Main Ingredients

- Weak Forms and Weak Solutions
- Test and Shape Functions

Assembling the System of Equations

- Example Problem: 1D Poisson
- Stiffness Matrix for Linear Nodal Basis Functions

A Road to Theory

Time-Dependent Problems

The Model Problem: Poisson Equation

- 2D Poisson Equation on unit square:

$$\frac{\partial^2}{\partial x^2} u(x,y) + \frac{\partial^2}{\partial y^2} u(x,y) = f(x,y) \qquad \text{in } \Omega = (0,1)^2$$

- Dirichlet boundary conditions:

$$u(x,y) = g(x,y) \qquad \text{on } \partial\Omega$$

The Model Problem: Poisson Equation

- 2D Poisson Equation on unit square:

$$\frac{\partial^2}{\partial x^2} u(x,y) + \frac{\partial^2}{\partial y^2} u(x,y) = f(x,y) \qquad \text{in } \Omega = (0,1)^2$$

- Dirichlet boundary conditions:

$$u(x,y) = g(x,y) \qquad \text{on } \partial\Omega$$

- for general operator L instead of Δ: similar concept

The Model Problem: Poisson Equation

- 2D Poisson Equation on unit square:

$$\frac{\partial^2}{\partial x^2} u(x,y) + \frac{\partial^2}{\partial y^2} u(x,y) = f(x,y) \qquad \text{in } \Omega = (0,1)^2$$

- Dirichlet boundary conditions:

$$u(x,y) = g(x,y) \qquad \text{on } \partial\Omega$$

- for general operator L instead of Δ: similar concept
- **new for FEM**: compute a function to approximate $u(x,y)$ (instead of approximate values at certain grid points)

Finite Elements – Main Ingredients

1. solve *weak form* of PDE to reduce regularity properties

$$u'' = f \quad \longrightarrow \quad \int v' u' \, \mathrm{dx} = \int v f \, \mathrm{dx}$$

$\rightarrow$ allows additional *weak* solutions

Finite Elements – Main Ingredients

1. solve *weak form* of PDE to reduce regularity properties

$$u'' = f \quad \longrightarrow \quad \int v' u' \, \mathrm{dx} = \int v f \, \mathrm{dx}$$

   $\rightarrow$ allows additional *weak* solutions

2. compute a *function* as numerical solution
   $\rightarrow$ search in a function space W_h:

$$u_h = \sum_j u_j \varphi_j(x), \qquad \operatorname{span}\{\varphi_1, \ldots, \varphi_J\} = W_h$$

Finite Elements – Main Ingredients

1. solve *weak form* of PDE to reduce regularity properties

$$u'' = f \quad \longrightarrow \quad \int v' u' \, \mathrm{dx} = \int v f \, \mathrm{dx}$$

   $\rightarrow$ allows additional *weak* solutions

2. compute a *function* as numerical solution
   $\rightarrow$ search in a function space W_h:

$$u_h = \sum_j u_j \varphi_j(x), \qquad \mathrm{span}\{\varphi_1, \ldots, \varphi_J\} = W_h$$

3. find weak solutions of simple form:
   for example piecewise linear functions and choose basis functions with *local support* (“hat functions”)
   $\rightarrow$ leads to system of linear equations

Weak Form and Weak Solutions

The weak form is obtained by 3 steps:

- integrate over the domain
  $\Rightarrow$ from infinetly many equations (strong form has to hold on every point (x, y)), only **one** new is obtained

Weak Form and Weak Solutions

The weak form is obtained by 3 steps:

- integrate over the domain
  $\Rightarrow$ from infinetly many equations (strong form has to hold on every point (x, y)), only **one** new is obtained
- multiply with (any) test function v (from a certain function space V)
  $\Rightarrow$ again, infinitely many equations

Weak Form and Weak Solutions

The weak form is obtained by 3 steps:

- integrate over the domain
  $\Rightarrow$ from infinetly many equations (strong form has to hold on every point (x, y)), only **one** new is obtained
- multiply with (any) test function v (from a certain function space V)
  $\Rightarrow$ again, infinitely many equations
- do “integration by parts” (i.e. use divergence theorem) to play derivatives from solution u to test functions v
  $\Rightarrow$ reduce smoothness assumptions on u

Weak Form and Weak Solutions

The weak form is obtained by 3 steps:

- integrate over the domain
  $\Rightarrow$ from infinetly many equations (strong form has to hold on every point (x, y)), only **one** new is obtained
- multiply with (any) test function v (from a certain function space V)
  $\Rightarrow$ again, infinitely many equations
- do "integration by parts" (i.e. use divergence theorem) to play derivatives from solution u to test functions v
  $\Rightarrow$ reduce smoothness assumptions on u

$\Rightarrow$ "real solution" u also solves the weak form
however, *additional, weak solutions* are now allowed

Weak Form of the Poisson Equation

- Poisson equation with Dirichlet conditions:

$$-\Delta u = f \quad \text{in } \Omega, u = 0 \quad \text{on } \delta\Omega$$

- weak form:

$$-\int_\Omega \Delta u \cdot v \, \mathrm{d}\Omega = \int_\Omega f \cdot v \, \mathrm{d}\Omega \quad \forall v \in V$$

Weak Form of the Poisson Equation

- Poisson equation with Dirichlet conditions:

$$-\Delta u = f \quad \text{in } \Omega, u = 0 \quad \text{on } \delta\Omega$$

- weak form:

$$-\int_\Omega \Delta u \cdot v \, \mathrm{d}\Omega = \int_\Omega f \cdot v \, \mathrm{d}\Omega \quad \forall v \in V$$

- apply divergence theorem:

$$-\int_\Omega \Delta u \cdot v \, \mathrm{d}\Omega = \int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}\Omega - \int_{\partial\Omega} \nabla u \cdot v \, ds$$

- choose functions v such that $v = 0$ on $\partial\Omega$:

$$\int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}\Omega = \int_\Omega f \cdot v \, \mathrm{d}\Omega \quad \forall v \in V$$

Weak Form of the Poisson Equation (2)

- Poisson equation with Dirichlet conditions:

$$-\Delta u = f \quad \text{in } \Omega, u = 0 \quad \text{on } \delta\Omega$$

- transformed into weak form:

$$\int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}\Omega = \int_\Omega f \cdot v \, \mathrm{d}\Omega \quad \forall v \in V$$

- weaker requirements for a solution u:
  twice differentiabale $\rightarrow$ *first derivative integrable*
- $\langle \nabla u, \nabla v \rangle$ a *bilinear form* (i.e. depends on 2 functions u and v); often written as:

$$a(u, v) = \langle f, v \rangle \quad \forall v \in V$$

Test and Shape Functions

- search for solution functions u_h of the form

$$u_h = \sum_j u_j \varphi_j(x)$$

  the $\varphi_j(x)$ are typically called **shape** or **ansatz** functions
- the basis functions $\varphi_j(x)$ build a vector space (i.e., a **function space**) W_h

$$\text{span}\{\varphi_1, \ldots, \varphi_J\} = W_h$$

Test and Shape Functions

- search for solution functions u_h of the form

$$u_h = \sum_j u_j \varphi_j(x)$$

  the $\varphi_j(x)$ are typically called **shape** or **ansatz** functions
- the basis functions $\varphi_j(x)$ build a vector space (i.e., a **function space**) W_h

$$\text{span}\{\varphi_1, \ldots, \varphi_J\} = W_h$$

- insert into weak formulation

$$\int \nabla \left(\sum_j u_j \varphi_j(x) \right) \cdot \nabla v \, \mathrm{dx} = \int f \cdot v \, \mathrm{dx} \quad \forall v \in V$$

Test and Shape Functions (2)

- choose a basis $\{\psi_i\}$ of the *test* space V_h
- then: if all basis functions ψ_i satisfy

$$\int \nabla \left(\sum_j u_j \varphi_j(x) \right) \cdot \nabla \psi_i \, \mathrm{dx} = \int f \cdot \psi_i \, \mathrm{dx} \quad \forall \psi_i$$

  then all $v \in V_h$ satisfy the equation
- the $\{\psi_i\}$ are therefore often called **test functions**

Test and Shape Functions (2)

- choose a basis $\{\psi_i\}$ of the *test* space V_h
- then: if all basis functions ψ_i satisfy

$$\int \nabla\left(\sum_j u_j \varphi_j(x)\right) \cdot \nabla\psi_i \, \mathrm{dx} = \int f \cdot \psi_i \, \mathrm{dx} \quad \forall\psi_i$$

  then all $v \in V_h$ satisfy the equation
- the $\{\psi_i\}$ are therefore often called **test functions**
- we obtain a system of equations for unknowns u_j: one equation for each test function ψ_i

Test and Shape Functions (2)

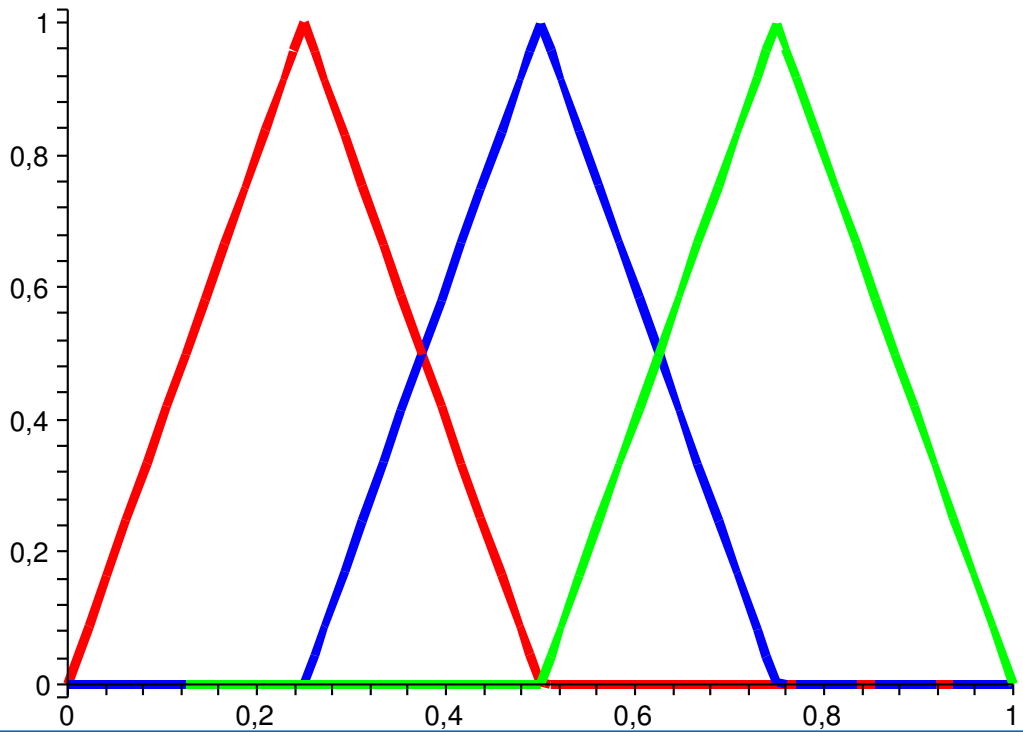
- choose a basis $\{\psi_i\}$ of the *test* space V_h
- then: if all basis functions ψ_i satisfy

$$\int \nabla \left(\sum_j u_j \varphi_j(x) \right) \cdot \nabla \psi_i \, \mathrm{dx} = \int f \cdot \psi_i \, \mathrm{dx} \quad \forall \psi_i$$

  then all $v \in V_h$ satisfy the equation
- the $\{\psi_i\}$ are therefore often called **test functions**
- we obtain a system of equations for unknowns u_j:
  one equation for each test function ψ_i
- V_h is often chosen to be identical to W_h (**Ritz-Galerkin method**)
  $\rightarrow$ we then have as many equations as unknowns

Example: Nodal Basis

$$\varphi_i(x) := \begin{cases} \frac{1}{h}(x - x_{i-1}) & x_{i-1} < x < x_i \\ \frac{1}{h}(x_{i+1} - x) & x_i < x < x_{i+1} \\ 0 & \text{otherwise} \end{cases}$$



Assembling the System of Equations

- Δ linear $\Rightarrow$ linear system of equations

$$\sum_j \underbrace{\left(\int \nabla \varphi_j(x) \cdot \nabla \psi_i(x) \, \mathrm{dx} \right)}_{=:A_{ij}} u_j = \int f \cdot \psi_i \, \mathrm{dx} \quad \forall \psi_i$$

- aim: make matrix A *sparse* $\rightarrow$ most $A_{ij} = 0$

Assembling the System of Equations

- Δ linear $\Rightarrow$ linear system of equations

$$\sum_j \underbrace{\left(\int \nabla \varphi_j(x) \cdot \nabla \psi_i(x) \, \mathrm{dx} \right)}_{=:A_{ij}} u_j = \int f \cdot \psi_i \, \mathrm{dx} \quad \forall \psi_i$$

- aim: make matrix A *sparse* $\rightarrow$ most $A_{ij} = 0$
- approach: local basis functions on a discretisation grid
- ψ_j, φ_j zero everywhere except in grid cells adjacent to grid point x_j
- $A_{ij} = 0$, if ψ_i and φ_j don't overlap

Example Problem: 1D Poisson

- in 1D: $-u''(x) = f(x)$ on $\Omega = (0,1)$,
  hom. Dirichlet boundary cond.: $u(0) = u(1) = 0$
- weak form:

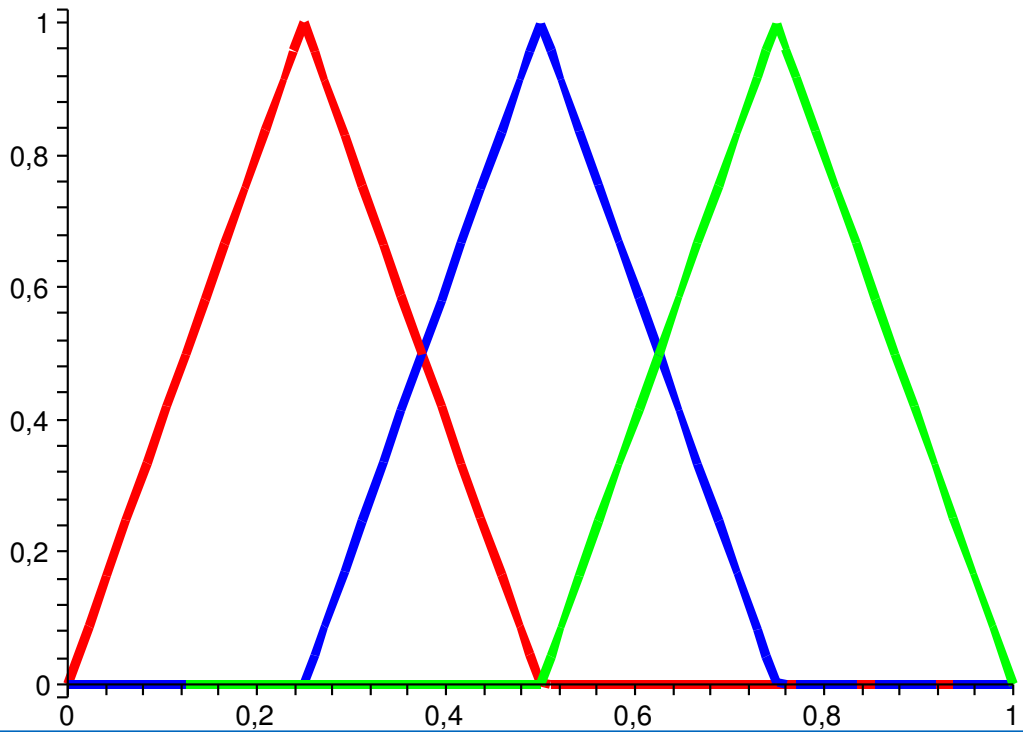
$$\int_0^1 u'(x) \cdot v'(x)\ \mathrm{dx} = \int_0^1 f(x) \cdot v(x)\ \mathrm{dx} \quad \forall v \in V$$

- computational grid:
  $x_i = ih$, (for $i = 1, \ldots, n-1$); mesh size $h = 1/n$
- $V_h = W_h$: piecewise linear functions
  (on intervals $[x_i, x_{i+1}]$)
- weak form reads:

$$\int_0^1 u'(x) \cdot \varphi_i'(x)\ \mathrm{dx} = \int_0^1 f(x) \cdot \varphi_i(x)\ \mathrm{dx} \quad \forall i = 1, \ldots, n-1$$

Nodal Basis

$$\varphi_i(x) := \begin{cases} \frac{1}{h}(x - x_{i-1}) & x_{i-1} < x < x_i \\ \frac{1}{h}(x_{i+1} - x) & x_i < x < x_{i+1} \\ 0 & \text{otherwise} \end{cases}$$



Nodal Basis – Derivatives and Integrals

Compute derivative of basis functions:

$$\varphi_i'(x) := \begin{cases} \frac{1}{h} & x_{i-1} < x < x_i \\ -\frac{1}{h} & x_i < x < x_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

Compute integrals of weak form:

- $\int \varphi_i'(x)\varphi_i'(x)\,\mathrm{dx} = \int_{x_{i-1}}^{x_i} \frac{1}{h}\cdot\frac{1}{h}\,\mathrm{dx} + \int_{x_i}^{x_{i+1}} \left(-\frac{1}{h}\right)\left(-\frac{1}{h}\right)\,\mathrm{dx} = \frac{2}{h}$
- $\int \varphi_i'(x)\varphi_{i+1}'(x)\,\mathrm{dx} = \int_{x_i}^{x_{i+1}} \left(-\frac{1}{h}\right)\frac{1}{h}\,\mathrm{dx} = -\frac{1}{h}$
- $\int \varphi_i'(x)\varphi_{i-1}'(x)\,\mathrm{dx} = -\frac{1}{h}$ (same computation)
- $\int \varphi_i'(x)\varphi_j'(x)\,\mathrm{dx} = 0$ for all $j \neq i, i \pm 1$
  (supports of basis functions do not overlap!)

Nodal Basis – System of Equations

- stiffness matrix:

$$\frac{1}{h}\begin{pmatrix} 2 & -1 & & \\ -1 & 2 & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{pmatrix}$$

- right hand sides (assume $f(x) = \alpha \in \mathbb{R}$):

$$\int_0^1 f(x) \cdot \varphi_i(x)\, \mathrm{dx} = \int_0^1 \alpha \cdot \varphi_i(x)\, \mathrm{dx} = \alpha h$$

- system of equations very similar to finite differences
- note: for other operators L: same approach!
- exercise: compute right-hand-side vector, if $f(x) := \sum f_j \phi_j(x)$

Outlook: A Road to Theory

- weak formulation is equivalent to variational approach: solution u minimises an energy functional

Outlook: A Road to Theory

- weak formulation is equivalent to variational approach: solution u minimises an energy functional
- *best approximation* property:

$$\left\| u - u_h^{FE} \right\|_a \leq \inf_{v \in V_h} \|u - v\|_a$$

  in terms of the norm induced by the bilinear form a (energy norm)
- thus: error bounded by interpolation error (in energy norm)

(details in lecture *Numerical Programming II* ...)

Time-Dependent Problems

Example: 1D Heat Equation

- $u_t = u_{xx} + f$ on domain $\Omega = [0, 1]$ for $t \in [0, t_{\text{end}}]$
- spatial discretisation: weak form

$$\int u_t \cdot v \, \mathrm{dx} = \int u_{xx} \cdot v \, \mathrm{dx} + \int f \cdot v \, \mathrm{dx}$$

$$\frac{\partial}{\partial t} \left(\int u \cdot v \, \mathrm{dx} \right) = \int u_{xx} \cdot v \, \mathrm{dx} + \int f \cdot v \, \mathrm{dx}$$

- spatial discretisation – finite elements:

$$\frac{\partial}{\partial t} \left(M_h u_h \right) = A_h u_h + f_h$$

M_h: mass matrix, A_h: stiffness matrix, $u_h = u_h(t)$

Time-Dependent Problems (2)

Solve a system of ordinary differential equations:

- after spatial discretisation (M_h constant):

$$\frac{\partial}{\partial t} M_h (u_h) = A_h u_h + f_h$$

- u_h a vector of time-dependent functions:

$$u_h = (u_1(t), \ldots, u_i(t), \ldots, u_n(t))^T$$

- usually: approximate M_h by a simpler matrix (diagonal matrix, e.g.) $\rightarrow$ **mass lumping**

Part II: Implementation of Finite Element Methods

Element Stiffness Matrices

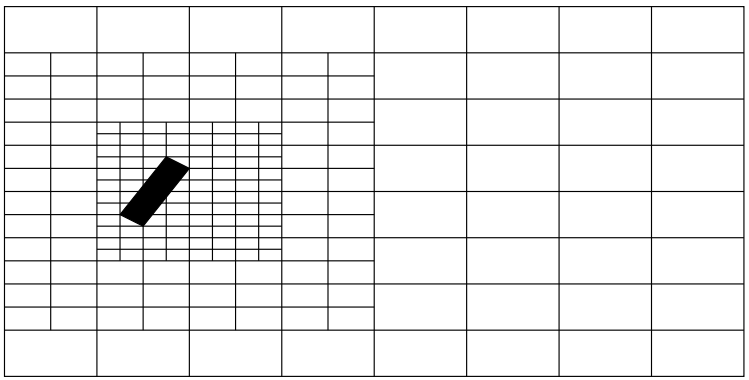
- Element-Oriented Computation of Stiffness Matrices
- Example: 1D Poisson
- Example: 2D Poisson
- Typical Workflow

Reference Elements and Stiffness Matrices

- Element-Oriented Computation on Unstructured Meshes
- Accumulation of Global Stiffness Matrix
- Simple Example: 1D Poisson
- Outlook: Extension to 2D and 3D
- Outlook: Further Components and Aspects of FEM

Element Stiffness Matrices

Finite Element methods are often used for complicated meshes:

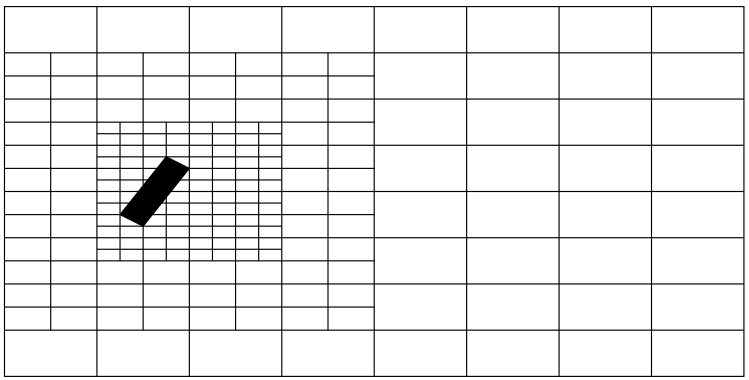


Questions:

- how to set up the system of equations efficiently?
- is it possible to use stencil notation??

Element Stiffness Matrices

Finite Element methods are often used for complicated meshes:



Questions:

- how to set up the system of equations efficiently?
- is it possible to use stencil notation??

$\Rightarrow$ Switch to element stiffness matrices!

Element Stiffness Matrices (2)

- domain Ω subdivided into finite elements $\Omega^{(k)}$:

$$\Omega = \Omega^{(1)} \cup \Omega^{(2)} \cup \cdots \cup \Omega^{(n)}$$

- observation: basis functions are defined element-wisely
- use: $\int_a^b f(x)\,\mathrm{dx} = \int_a^c f(x)\,\mathrm{dx} + \int_c^b f(x)\,\mathrm{dx}$
- element-wise evaluation of the integrals:

$$\int_\Omega \nabla u \cdot \nabla v \,\mathrm{dx} = \sum_k \int_{\Omega^{(k)}} \nabla u \cdot \nabla v \,\mathrm{dx}$$

$$\int_\Omega f \cdot v \,\mathrm{dx} = \sum_k \int_{\Omega^{(k)}} f \cdot v \,\mathrm{dx}$$

Element Stiffness Matrices (3)

- leads to local stiffness matrices for each element:

$$\underbrace{\int_{\Omega^{(k)}} \nabla\phi_j \cdot \nabla\phi_i \, \mathrm{dx}}_{=:A_{ij}^{(k)}}$$

- and respective element systems:

$$A^{(k)}x = b^{(k)}$$

- accumulate to obtain global system:

$$\underbrace{\sum_k A^{(k)}}_{=:A} \; x = \sum_k b^{(k)}$$

Element Stiffness Matrices (4)

Some comments on notation:

- assume: 1D problem, n elements (i.e. intervals)
- in each element only two basis functions are non-zero!
- hence, almost all $A_{ij}^{(k)}$ are zero:

$$A_{ij}^{(k)} = \int_{\Omega^{(k)}} \nabla \phi_j \cdot \nabla \phi_i \, \mathrm{dx}$$

- only 2×2 elements of $A^{(k)}$ are non-zero
- therefore convention to omit zero columns/rows

  $\Rightarrow$ leaves only unknowns that are in $\Omega^{(k)}$

  $\Rightarrow$ notation: write this matrix as $\widehat{A}^{(k)}$ (element stiffness matrix)

Example: 1D Poisson

- $\Omega = [0,1]$ splitted into $\Omega^{(k)} = [x_k, x_{k+1}]$
- nodal basis; leads to element stiffness matrix:

$$\widehat{A}^{(k)} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

- consider case with only three unknowns (i.e., 3 basis functions):

$$A^{(1)} + A^{(2)} = \begin{pmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & -1 & 1 \end{pmatrix}$$

- in stencil notation:

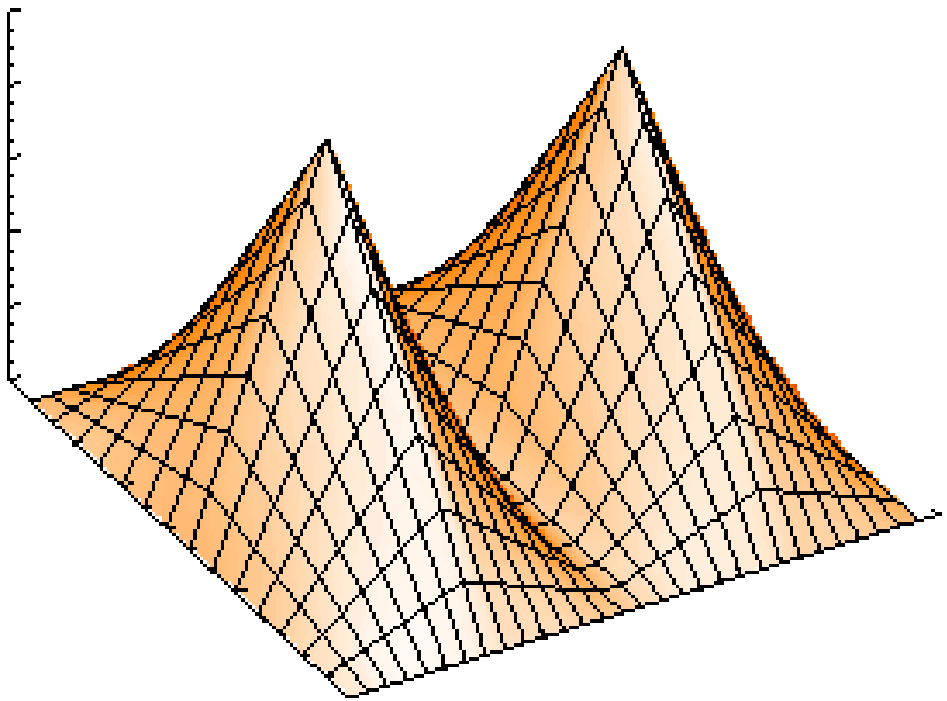
$$[-1 \quad 1^*] + [1^* \quad -1] \rightarrow [-1 \quad 2 \quad -1]$$

Example: 2D Poisson

- $-\Delta u = f$ on domain $\Omega = [0,1]^2$
- splitted into $\Omega^{(i,j)} = [x_i, x_{i+1}] \times [x_j, x_{j+1}]$
- bilinear basis functions

$$\varphi_{ij}(x,y) = \varphi_i(x)\varphi_j(y)$$

- “pagoda” functions



Example: 2D Poisson (2)

- leads to element stiffness matrix:

$$\widehat{A}^{(k)} = \begin{pmatrix} 2 & -\frac{1}{2} & -\frac{1}{2} & -1 \\ -\frac{1}{2} & 2 & -1 & -\frac{1}{2} \\ -\frac{1}{2} & -1 & 2 & -\frac{1}{2} \\ -1 & -\frac{1}{2} & -\frac{1}{2} & 2 \end{pmatrix}$$

- accumulation leads to 9-point stencil

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

- exercise at home: compute $\widehat{A}^{(k)}$ and assemble global matrix (and compare with stencil)

Typical Workflow

1. choose grid:
   - quadratic or cubic cells
   - triangles (structured, unstructured)
   - tetrahedra, etc.
2. set up basis functions for each element $\Omega^{(k)}$; for example, at all nodes $x_i \in \Omega^{(k)}$

$$\begin{aligned} \varphi_i(x_i) &= 1 \\ \varphi_i(x_j) &= 0 \quad \text{for all } j \neq i \end{aligned}$$

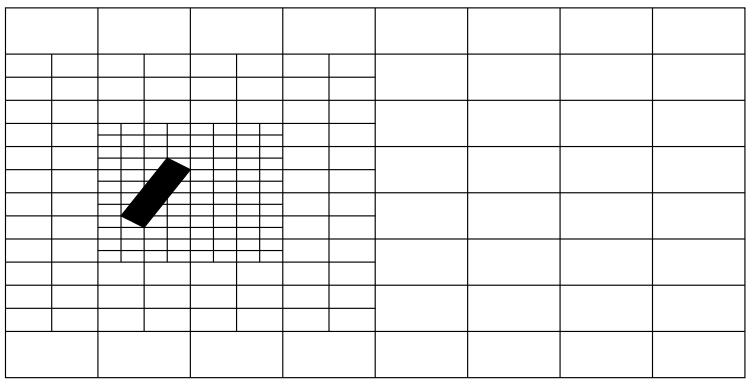
3. for element stiffness matrix, compute all

$$\widehat{A}_{ij}^{(k)} = \int_{\Omega^{(k)}} \nabla\varphi_j \cdot \nabla\varphi_i \, \mathrm{dx}$$

4. accumulate global stiffness matrix A (and rhs)

Element-Oriented Computation on Unstructured Meshes

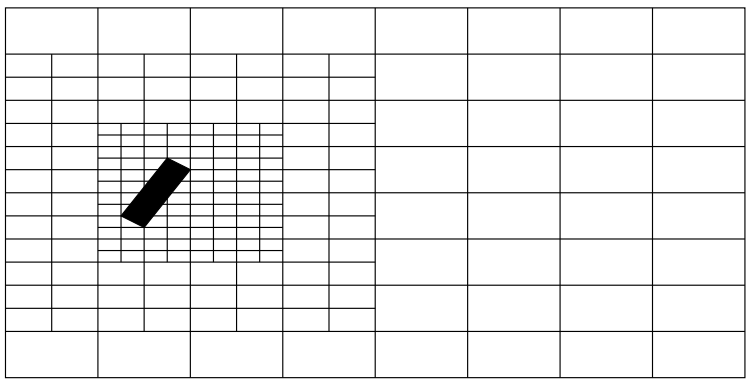
Recall: FEM often used for complicated meshes:



(Re-)computation of element matrices required for each element?

Element-Oriented Computation on Unstructured Meshes

Recall: FEM often used for complicated meshes:

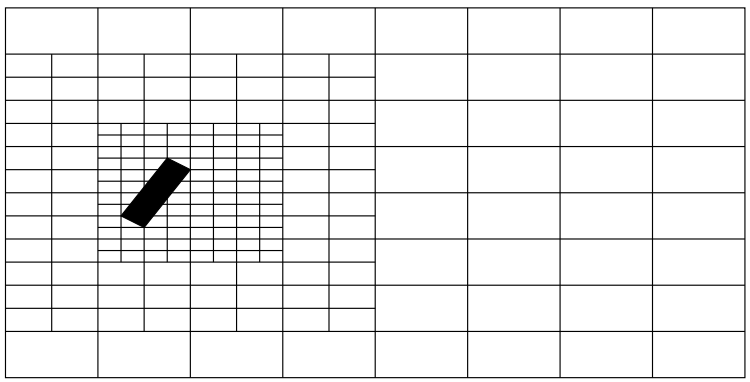


(Re-)computation of element matrices required for each element?

- left grid: no (up to scaling with mesh size)
- right grid: yes?

Element-Oriented Computation on Unstructured Meshes

Recall: FEM often used for complicated meshes:



(Re-)computation of element matrices required for each element?

- left grid: no (up to scaling with mesh size)
- right grid: yes?

$\Rightarrow$ requires a closer look at the computation of element matrices

Element Stiffness Matrices – Notation Revisited

- recall notation for local stiffness matrices $A_{ij}^{(k)}$:

$$A_{ij}^{(k)} = \int_{\Omega^{(k)}} \nabla\phi_j(x) \cdot \nabla\phi_i(x) \, \mathrm{dx}$$

  $\Rightarrow$ which basis functions ϕ_i, ϕ_j belong to element k?
- switch to element-local numbering of basis functions:

$$\widehat{A}_{\mu\nu}^{(k)} = \int_{\Omega^{(k)}} \nabla\phi_\nu^{(k)}(x) \cdot \nabla\phi_\mu^{(k)}(x) \, \mathrm{dx}$$

  where $\phi_\mu^{(k)}(x) = \phi_i(x)$ and $\phi_\nu^{(k)} = \phi_j(x)$ in $\Omega^{(k)}$
- requires a mapping $\phi_\mu^{(k)} \to \phi_i$ (and $\phi_\nu^{(k)} \to \phi_j$),
  i.e., a function $\iota^{(k)}$ with $i = \iota^{(k)}(\mu)$ and $j = \iota^{(k)}(\nu)$

Accumulation of Global Stiffness Matrix

- note the different dimensions of $A^{(k)}$ and $\widehat{A}^{(k)}$: $\widehat{A}^{(k)}$ omits zero rows and columns of $A^{(k)}$
- formulated via a projection matrix:

$$A^{(k)} = \left(P^{(k)}\right)^T \widehat{A}^{(k)} P^{(k)} \qquad \text{with } P^{(k)}_{\mu,i} := \begin{cases} 1 & \text{if } i = \iota^{(k)}(\mu) \\ 0 & \text{otherwise} \end{cases}$$

- recall computation of global stiffness matrix:

$$A := \sum_k A^{(k)} = \sum_k \left(P^{(k)}\right)^T \widehat{A}^{(k)} P^{(k)}$$

Accumulation of Global Stiffness Matrix

- note the different dimensions of $A^{(k)}$ and $\widehat{A}^{(k)}$: $\widehat{A}^{(k)}$ omits zero rows and columns of $A^{(k)}$
- formulated via a projection matrix:

$$A^{(k)} = \left(P^{(k)}\right)^T \widehat{A}^{(k)} P^{(k)} \qquad \text{with } P^{(k)}_{\mu,i} := \begin{cases} 1 & \text{if } i = \iota^{(k)}(\mu) \\ 0 & \text{otherwise} \end{cases}$$

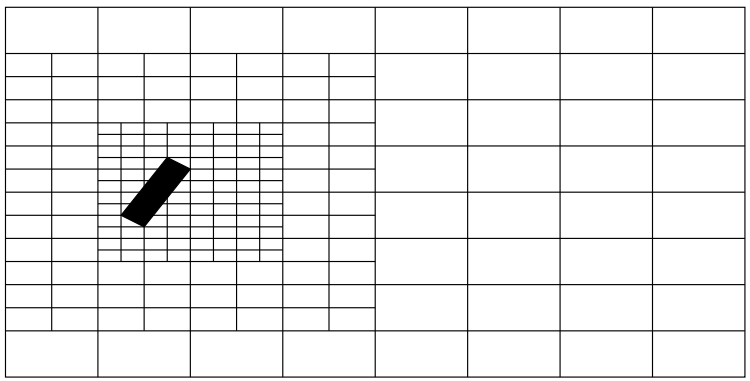
- recall computation of global stiffness matrix:

$$A := \sum_k A^{(k)} = \sum_k \left(P^{(k)}\right)^T \widehat{A}^{(k)} P^{(k)}$$

- hence, for each element:
  1. compute element stiffness matrix $\widehat{A}^{(k)}$
  2. determine position of matrix elements $\widehat{A}^{(k)}$ in $A^{(k)}$ $\rightarrow$ based on mapping $\iota^{(k)}(\mu)$
  3. add non-zero elements of $A^{(k)} = \left(P^{(k)}\right)^T \widehat{A}^{(k)} P^{(k)}$ to global matrix A

Element Stiffness Matrices and Meshes

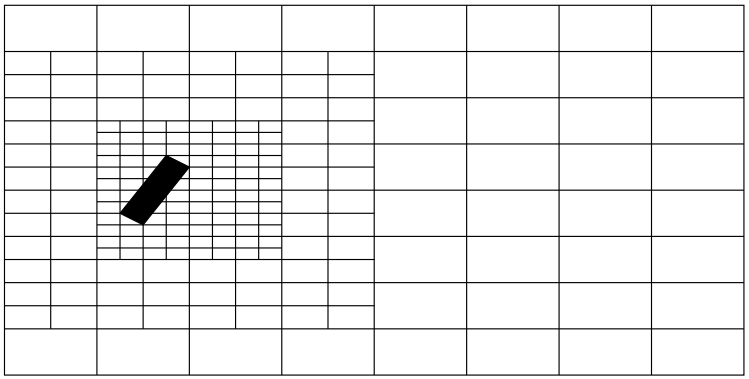
Recall our initial question:



(Re-)computation of $\widehat{A}^{(k)}$ required for each element?

Element Stiffness Matrices and Meshes

Recall our initial question:

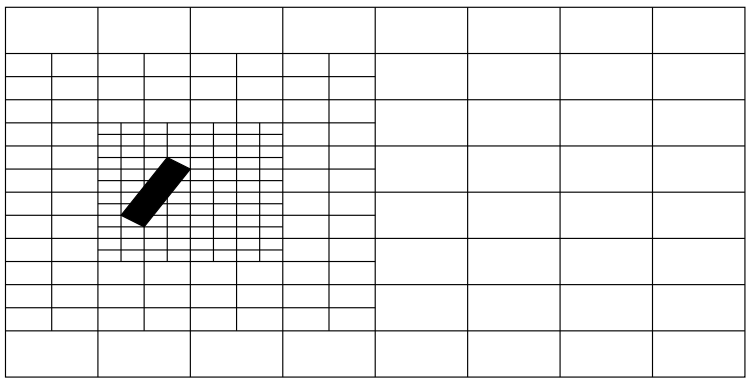


(Re-)computation of $\widehat{A}^{(k)}$ required for each element?

- left grid: no (up to scaling with mesh size)
- right grid: yes?

Element Stiffness Matrices and Meshes

Recall our initial question:



(Re-)computation of $\widehat{A}^{(k)}$ required for each element?

- left grid: no (up to scaling with mesh size)
- right grid: yes?

$\Rightarrow$ efficient computation of $\widehat{A}^{(k)}$ via reference elements

Simple Example: 1D Poisson

Consider FEM for 1D Poisson on an adaptive grid:

- grid points given as x_i, not necessarily equidistant; define elements $\Omega^{(k)} = [x_k, x_{k+1}]$
- piecewise linear basis functions ("hat functions") $\phi_i(x)$; $\phi_i(x)$ non-zero only on two elements: $\Omega^{(i-1)}$ and $\Omega^{(i)}$
- mapping local to global indices: $\iota^{(k)}(1) := k$ and $\iota^{(k)}(2) := k + 1$
- wanted: $\widehat{A}^{(k)}_{\mu,\nu} := \int_{\Omega^{(k)}} \frac{\partial}{\partial x}\phi^{(k)}_\nu(x)\frac{\partial}{\partial x}\phi^{(k)}_\mu(x)\,\mathrm{dx}$ for every element k

Simple Example: 1D Poisson

Consider FEM for 1D Poisson on an adaptive grid:

- grid points given as x_i, not necessarily equidistant; define elements $\Omega^{(k)} = [x_k, x_{k+1}]$
- piecewise linear basis functions ("hat functions") $\phi_i(x)$; $\phi_i(x)$ non-zero only on two elements: $\Omega^{(i-1)}$ and $\Omega^{(i)}$
- mapping local to global indices: $\iota^{(k)}(1) := k$ and $\iota^{(k)}(2) := k + 1$
- wanted: $\widehat{A}^{(k)}_{\mu,\nu} := \int_{\Omega^{(k)}} \frac{\partial}{\partial x}\phi^{(k)}_\nu(x)\frac{\partial}{\partial x}\phi^{(k)}_\mu(x)\,\mathrm{d}x$ for every element k

Introduce **reference element** Ω_{ref} with linear basis functions Φ_μ:

$$\Phi_1(\xi) := \xi \qquad \Phi_2(\xi) := 1 - \xi \qquad \xi \in \Omega_{\text{ref}} := [0, 1]$$

and a mapping $x = \chi^{(k)}(\xi)$ for each $\Omega^{(k)}$ defined as

$$\chi^{(k)}(\xi) := x_k + \xi(x_{k+1} - x_k)$$

such that $\phi^{(k)}_\mu(x) = \phi^{(k)}_\mu\left(\chi^{(k)}(\xi)\right) := \Phi_\mu(\xi)$ for $x \in \Omega^{(k)}$.

Simple Example: 1D Poisson (2)

We need to compute

$$\widehat{A}_{\mu,\nu}^{(k)} := \int_{x_k}^{x_{k+1}} \frac{\partial}{\partial x}\phi_\nu^{(k)}(x)\frac{\partial}{\partial x}\phi_\mu^{(k)}(x)\,\mathrm{dx} = \int_{\chi^{(k)}(0)}^{\chi^{(k)}(1)} \frac{\partial}{\partial x}\phi_\nu^{(k)}(x)\frac{\partial}{\partial x}\phi_\mu^{(k)}(x)\,\mathrm{dx}$$

Compare integration by substitution: $\int_{g(a)}^{g(b)} f(x)\,\mathrm{dx} = \int_a^b f(g(t))g'(t)\,\mathrm{dt}$

$$\widehat{A}_{\mu,\nu}^{(k)} = \int_0^1 \frac{\partial}{\partial x}\phi_\nu^{(k)}\left(\chi^{(k)}(\xi)\right)\frac{\partial}{\partial x}\phi_\mu^{(k)}\left(\chi^{(k)}(\xi)\right)(x_{k+1}-x_k)\,\mathrm{d}\xi.$$

Note that:

- $\frac{\partial}{\partial \xi}\chi_k(\xi) = (x_{k+1}-x_k)$ corresponds to the factor $g'(t)$ in the substitution rule
- we still have to deal with the partial derivatives in x

Simple Example: 1D Poisson (3)

We require the chain rule, $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial g}\frac{\partial g}{\partial x}$, to obtain:

$$\frac{\partial}{\partial x}\phi_\mu^{(k)}\left(\chi^{(k)}(\xi)\right) = \frac{\partial}{\partial x}\Phi_\mu(\xi) = \frac{\partial}{\partial \xi}\Phi_\mu(\xi)\frac{\partial \xi}{\partial x}$$

As $x = \chi^{(k)}(\xi) := x_k + \xi(x_{k+1} - x_k)$, we have $\xi = \dfrac{x - x_k}{x_{k+1} - x_k}$ and thus:

$$\frac{\partial}{\partial \xi}\Phi_\mu(\xi)\frac{\partial \xi}{\partial x} = \frac{\partial}{\partial \xi}\Phi_\mu(\xi)\frac{1}{x_{k+1} - x_k}$$

Finally, we get:

$$\begin{aligned}\widehat{A}_{\mu,\nu}^{(k)} &= \int_0^1 \frac{\partial}{\partial x}\phi_\nu^{(k)}\left(\chi^{(k)}(\xi)\right)\frac{\partial}{\partial x}\phi_\mu^{(k)}\left(\chi^{(k)}(\xi)\right)(x_{k+1} - x_k)\,\mathrm{d}\xi \\ &= \int_0^1 \frac{\partial}{\partial \xi}\Phi_\nu(\xi)\frac{\partial}{\partial \xi}\Phi_\mu(\xi)\frac{1}{(x_{k+1} - x_k)^2}(x_{k+1} - x_k)\,\mathrm{d}\xi.\end{aligned}$$

Simple Example: 1D Poisson (4)

Altogether, element stiffness matrices are computed as:

$$\widehat{A}^{(k)}_{\mu,\nu} = \frac{1}{x_{k+1} - x_k} \int_0^1 \frac{\partial}{\partial \xi} \Phi_\nu(\xi) \frac{\partial}{\partial \xi} \Phi_\mu(\xi) \, \mathrm{d}\xi$$

With $\Phi_1(\xi) = \xi$ and $\Phi_2(\xi) = 1 - \xi$, we obtain that

$$\widehat{A}^{(k)} = \frac{1}{x_{k+1} - x_k} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Simple Example: 1D Poisson (4)

Altogether, element stiffness matrices are computed as:

$$\widehat{A}^{(k)}_{\mu,\nu} = \frac{1}{x_{k+1} - x_k} \int_0^1 \frac{\partial}{\partial \xi} \Phi_\nu(\xi) \frac{\partial}{\partial \xi} \Phi_\mu(\xi) \, \mathrm{d}\xi$$

With $\Phi_1(\xi) = \xi$ and $\Phi_2(\xi) = 1 - \xi$, we obtain that

$$\widehat{A}^{(k)} = \frac{1}{x_{k+1} - x_k} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Observations:

- the integral values only depend on the reference element $\Omega_{\text{ref}} = [0, 1]$ and the reference basis functions Φ_μ, Φ_ν
- all other element matrices are obtained by simple scaling
- we did **not** use that Φ_μ, Φ_ν are linear functions; **same computation for more complicated basis functions!**

Outlook: Extension to 2D and 3D

For 2D, 3D, etc., we require the multidimensional substitution rule:

$$\int_{\Omega^{(k)}} f(x,y,z)\,\mathrm{d}\Omega = \int_{\Omega_{\text{ref}}} f\left(x(\xi,\eta,\zeta), y(\xi,\eta,\zeta), z(\xi,\eta,\zeta)\right) |\mathcal{J}|\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta$$

where $|\mathcal{J}|$ is the determinant of the Jacobian:

$$|\mathcal{J}| = \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \zeta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} & \frac{\partial y}{\partial \zeta} \\ \frac{\partial z}{\partial \xi} & \frac{\partial z}{\partial \eta} & \frac{\partial z}{\partial \zeta} \end{vmatrix}$$

and also the multidimensional chain rule.

Result:

- similar, element-wise computation of stiffness matrices via reference elements
- Jacobian instead of factor $\frac{1}{x_{k+1}-x_k}$; more complicated, in general

Outlook: Further Components and Aspects of FEM

Additional components and “features” of Finite Element Methods:

- **mesh generation:**
  structured/unstructured meshes consisting of triangles, tetrahedra, rectangles, cuboids, quadrilaterals, hexahedrals, prisms, . . .
- **higher-order basis functions:**
  polynomials of higher degree in various dimensions; “nodal” vs. “spectral” elements, . . .
- **quadrature rules for integration:**
  Gaussian quadrature, e.g., to compute integrals of weak forms; resp. choice of quadrature points for certain elements?
- **solving the system of equations:**
  e.g., set up a global system of equations explicitly, or work directly on element representation?

. . . and many more . . .