FEniCS Course

Lecture 0: Introduction to FEM

Contributors
Anders Logg

What is FEM?

The finite element method is a framework and a recipe for discretization of differential equations

- Ordinary differential equations
- Partial differential equations
- Integral equations

- A recipe for discretization of PDE
- PDE $\rightarrow Ax = b$
- Different bases, stabilization, error control, adaptivity

The FEM cookbook

(i) $\mathcal{A}u = f$ *Partial differential equation*

Multiply by v

(ii) $a(u, v) = L(v)$ *Continuous variational problem*

Take $V_h \subset V$

(iii) $a(u_h, v) = L(v)$ *Discrete variational problem*

Let $u_h = \sum_j U_j \phi_j$

(iv) $AU = b$ *System of discrete equations*

The PDE (i)

Consider Poisson's equation, the Hello World of partial differential equations:

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega \\ u &= u_0 \quad \text{on } \partial\Omega \end{aligned}$$

Poisson's equation arises in numerous applications:

- heat conduction, electrostatics, diffusion of substances, twisting of elastic rods, inviscid fluid flow, water waves, magnetostatics, ...
- as part of numerical splitting strategies for more complicated systems of PDEs, in particular the Navier–Stokes equations

From PDE (i) to variational problem (ii)

The simple recipe is: multiply the PDE by a test function v and integrate over Ω:

$$-\int_\Omega (\Delta u) v \,\mathrm{d}x = \int_\Omega f v \,\mathrm{d}x$$

Then integrate by parts and set $v = 0$ on the Dirichlet boundary:

$$-\int_\Omega (\Delta u) v \,\mathrm{d}x = \int_\Omega \nabla u \cdot \nabla v \,\mathrm{d}x - \underbrace{\int_{\partial\Omega} \frac{\partial u}{\partial n} v \,\mathrm{d}s}_{=0}$$

We find that:

$$\int_\Omega \nabla u \cdot \nabla v \,\mathrm{d}x = \int_\Omega f v \,\mathrm{d}x$$

The variational problem (ii)

Find $u \in V$ such that

$$\int_{\Omega} \nabla u \cdot \nabla v \,\mathrm{d}x = \int_{\Omega} f v \,\mathrm{d}x$$

for all $v \in \hat{V}$

The trial space V and the test space $\hat{V}$ are (here) given by

$$\begin{aligned} V &= \{v \in H^1(\Omega) : v = u_0 \text{ on } \partial\Omega\} \\ \hat{V} &= \{v \in H^1(\Omega) : v = 0 \text{ on } \partial\Omega\} \end{aligned}$$

From continuous (ii) to discrete (iii) problem

We approximate the continuous variational problem with a discrete variational problem posed on finite dimensional subspaces of V and $\hat{V}$:

$$V_h \subset V$$
$$\hat{V}_h \subset \hat{V}$$

Find $u_h \in V_h \subset V$ such that

$$\int_\Omega \nabla u_h \cdot \nabla v \,\mathrm{d}x = \int_\Omega f v \,\mathrm{d}x$$

for all $v \in \hat{V}_h \subset \hat{V}$

From discrete variational problem (iii) to discrete system of equations (iv)

Choose a basis for the discrete function space:

$$V_h = \operatorname{span}\{\phi_j\}_{j=1}^N$$

Make an ansatz for the discrete solution:

$$u_h = \sum_{j=1}^N U_j \phi_j$$

Test against the basis functions:

$$\int_\Omega \nabla(\underbrace{\sum_{j=1}^N U_j \phi_j}_{u_h}) \cdot \nabla \phi_i \,\mathrm{d}x = \int_\Omega f \phi_i \,\mathrm{d}x$$

From discrete variational problem (iii) to discrete system of equations (iv), cont'd.

Rearrange to get:

$$\sum_{j=1}^{N} U_j \underbrace{\int_\Omega \nabla\phi_j \cdot \nabla\phi_i \,\mathrm{d}x}_{A_{ij}} = \underbrace{\int_\Omega f\phi_i \,\mathrm{d}x}_{b_i}$$

A linear system of equations:

$$AU = b$$

where

$$A_{ij} = \int_\Omega \nabla\phi_j \cdot \nabla\phi_i \,\mathrm{d}x \tag{1}$$

$$b_i = \int_\Omega f\phi_i \,\mathrm{d}x \tag{2}$$

The canonical abstract problem

(i) Partial differential equation:

$$\mathcal{A}u = f \quad \text{in } \Omega$$

(ii) Continuous variational problem: find $u \in V$ such that

$$a(u, v) = L(v) \quad \text{for all } v \in \hat{V}$$

(iii) Discrete variational problem: find $u_h \in V_h \subset V$ such that

$$a(u_h, v) = L(v) \quad \text{for all } v \in \hat{V}_h$$

(iv) Discrete system of equations for $u_h = \sum_{j=1}^{N} U_j \phi_j$:

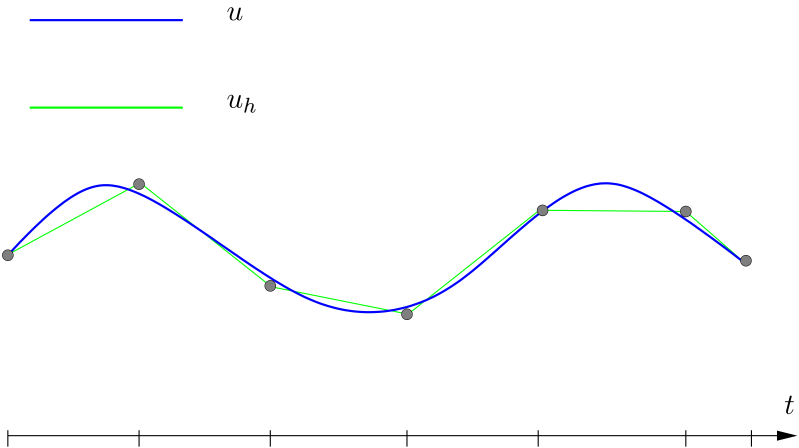
$$\begin{aligned} AU &= b \\ A_{ij} &= a(\phi_j, \phi_i) \\ b_i &= L(\phi_i) \end{aligned}$$

Important topics

- *How to choose V_h?*
- *How to compute A and b*
- *How to solve $AU = b$?*
- *Can we quantify/control How large the error $e = u - u_h$ is?*
- *Can we assess the cost of solving the system?*
- Extensions to nonlinear, time-dependent, complicated problems

How to choose V_h

Finite element function spaces



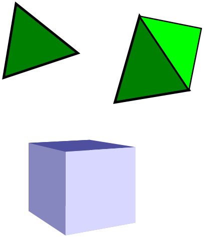
The finite element definition (Ciarlet 1975)

A finite element is a triple $(T, \mathcal{V}, \mathcal{L})$, where

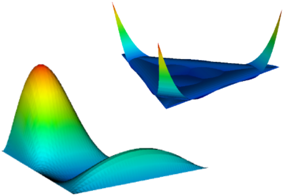
- the domain T is a bounded, closed subset of $\mathbb{R}^d$ (for $d = 1, 2, 3, \dots$) with nonempty interior and piecewise smooth boundary
- the space $\mathcal{V} = \mathcal{V}(T)$ is a finite dimensional function space on T of dimension n
- the set of degrees of freedom (nodes) $\mathcal{L} = \{\ell_1, \ell_2, \dots, \ell_n\}$ is a basis for the dual space $\mathcal{V}'$; that is, the space of bounded linear functionals on $\mathcal{V}$

The finite element definition (Ciarlet 1975)

T



$\mathcal{V}$



$\mathcal{L}$

$v(\bar{x})$

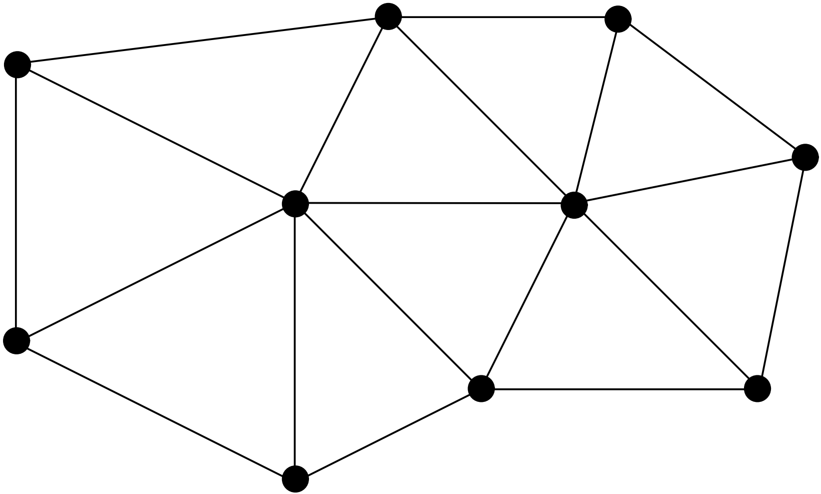
$v(\bar{x}) \cdot n$

$\int_T v(x) w(x) \, \mathrm{d}x$

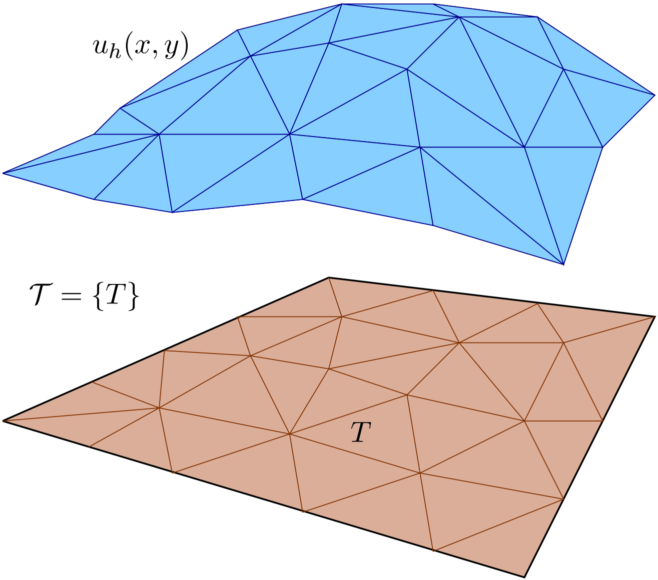
The linear Lagrange element: $(T, \mathcal{V}, \mathcal{L})$

- T is a line, triangle or tetrahedron
- $\mathcal{V}$ is the first-degree polynomials on T
- $\mathcal{L}$ is point evaluation at the vertices

The linear Lagrange element: $\mathcal{L}$



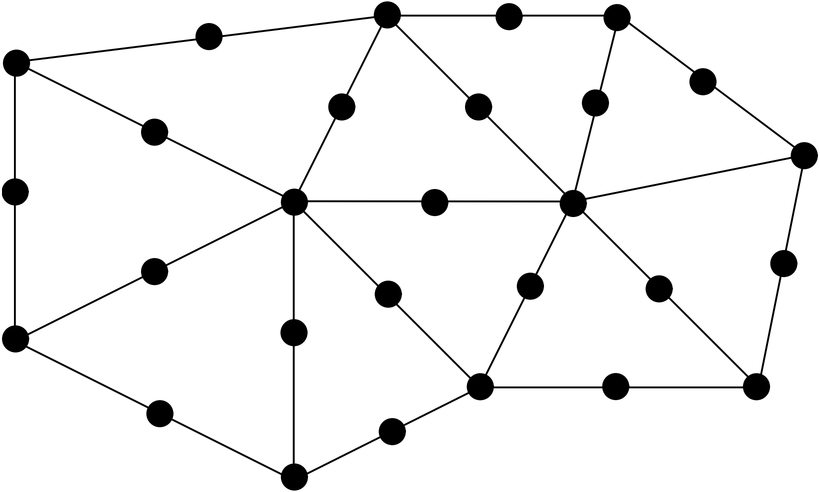
The linear Lagrange element: V_h



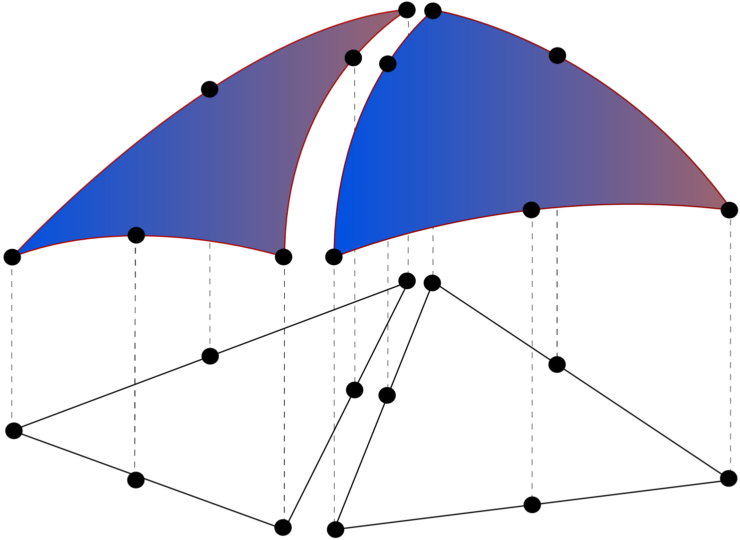
The quadratic Lagrange element: $(T, \mathcal{V}, \mathcal{L})$

- T is a line, triangle or tetrahedron
- $\mathcal{V}$ is the second-degree polynomials on T
- $\mathcal{L}$ is point evaluation at the vertices and edge midpoints

The quadratic Lagrange element: $\mathcal{L}$

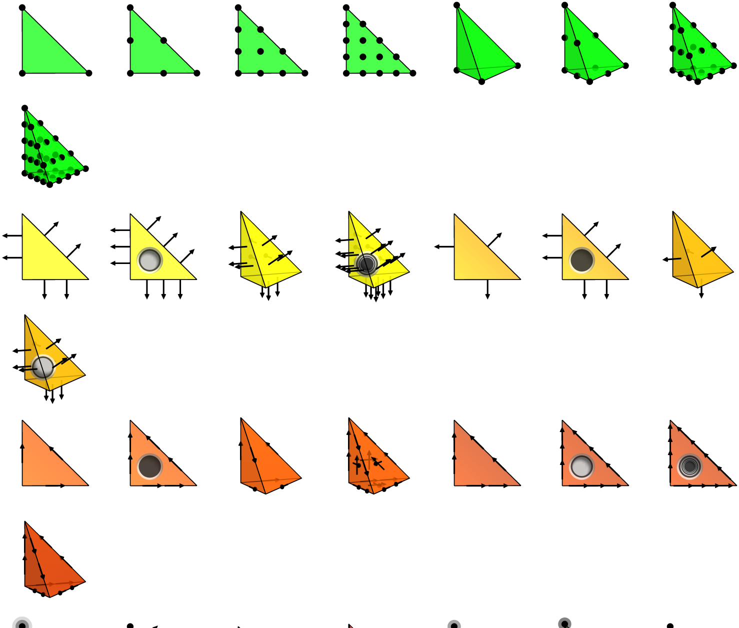


The quadratic Lagrange element: V_h



Families of elements

Families of elements



Computing the sparse matrix A

Naive assembly algorithm

$A = 0$

for $i = 1, \ldots, N$

    for $j = 1, \ldots, N$

        $A_{ij} = a(\phi_j, \phi_i)$

    end for

end for

The element matrix

The global matrix A is defined by

$$A_{ij} = a(\phi_j, \phi_i)$$

The *element matrix* A_T is defined by

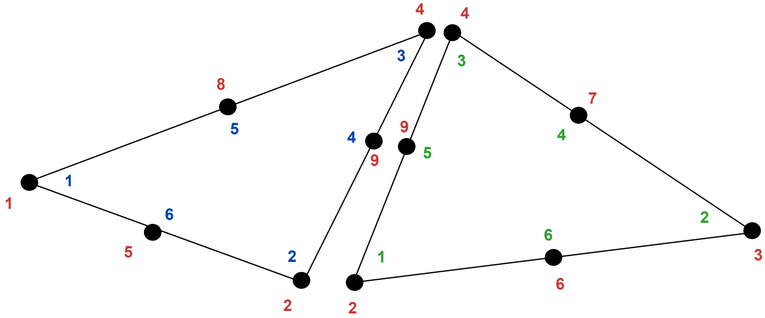
$$A_{T,ij} = a_T(\phi_j^T, \phi_i^T)$$

The local-to-global mapping

The global matrix ι_T is defined by

$$I = \iota_T(i)$$

where I is the *global index* corresponding to the *local index* i



The assembly algorithm

$A = 0$

for $T \in \mathcal{T}$

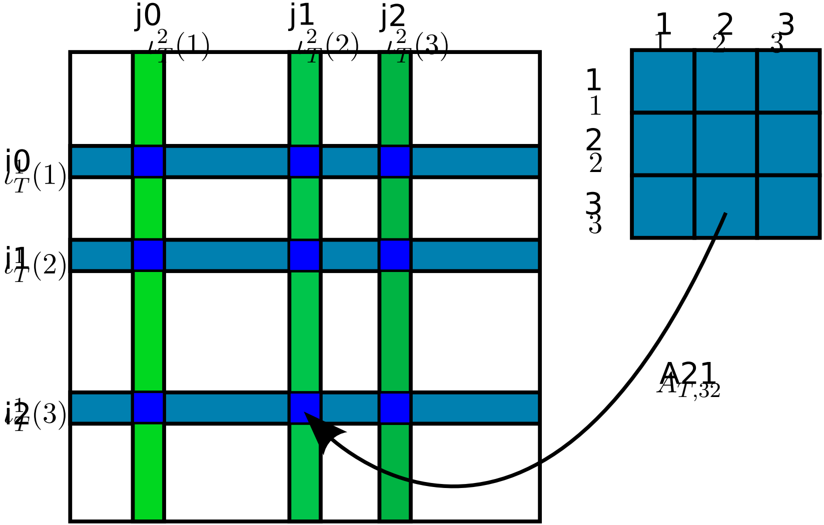
  Compute the element matrix A_T

  Compute the local-to-global mapping ι_T

  Add A_T to A according to ι_T

end for

Adding the element matrix A_T



Solving $AU = b$

Direct methods

- Gaussian elimination
  - Requires $\sim \frac{2}{3}N^3$ operations
- LU factorization: $A = LU$
  - Solve requires $\sim \frac{2}{3}N^3$ operations
  - Reuse L and U for repeated solves
- Cholesky factorization: $A = LL^\top$
  - Works if A is symmetric and positive definite
  - Solve requires $\sim \frac{1}{3}N^3$ operations
  - Reuse L for repeated solves

Iterative methods

Krylov subspace methods

- GMRES (Generalized Minimal RESidual method)
- CG (Conjugate Gradient method)
  - Works if A is symmetric and positive definite
- BiCGSTAB, MINRES, TFQMR, ...

Multigrid methods

- GMG (Geometric MultiGrid)
- AMG (Algebraic MultiGrid)

Preconditioners

- ILU, ICC, SOR, AMG, Jacobi, block-Jacobi, additive Schwarz, ...

Which method should I use?

Rules of thumb

- Direct methods for small systems
- Iterative methods for large systems
- Break-even at ca 100–1000 degrees of freedom
- Use a symmetric method for a symmetric system
  - Cholesky factorization (direct)
  - CG (iterative)
- Use a multigrid preconditioner for Poisson-like systems
- GMRES with ILU preconditioning is a good default choice