Series 4

Computational Methods for
Engineering Applications
Last edited: November 9, 2020
Due date: November 18 at 23:59

Template codes are available on the course's webpage at `https://moodle-app2.let.ethz.ch/course/view.php?id=13412`.

Exercise 1 Linear Finite Elements in 1D

In class the finite element discretization of a 2-point boundary value problem by means of trial and test spaces of merely continuous piecewise linear functions was discussed. In this problem, we practise the crucial steps for the linear variational problem

$$u \in H_0^1([a,b]): \quad \int_a^b u'(x)\, v'(x) = \int_a^b f(x) v(x)\ dx, \quad \forall v \in H_0^1([a,b]), \tag{1}$$

where $-\infty < a < b < \infty$ and $f \in C^0([a,b])$. Please note that both trial and test functions vanish at the endpoints of the interval, as indicated by the subscript "0" in the symbol for the function space.

1a)

Derive the stiffness matrix for (1), when using the trial and test space of continuous, piecewise linear functions on an *equidistant* mesh $\mathcal{M}$ with $N \in \mathbb{N}$ interior nodes. The standard basis of hat functions is to be used.

1b)

(Core problem) In the template file `fem.cpp`, implement the function

```
void createStiffnessMatrix(SparseMatrix& A, int N, double dx)
```

(with `typedef Eigen::SparseMatrix<double> SparseMatrix`), that computes the matrix `A` as in subproblem **1a)**. The argument `N` denotes the number of interior grid points, and `dx` denotes the cell length.

1c)

(Core problem) To obtain the right-hand side vector of the linear system arising from the finite element discretization of (1) as described in section **1a)**, one relies on the composite trapezoidal rule on $\mathcal{M} = \{x_0 = a, x_1, \ldots, x_N, x_{N+1} = b\}$ for numerical quadrature:

$$\int_a^b f(x)\,dx \approx f(a)\frac{h}{2} + h\sum_{i=1}^{N} f(x_i) + f(b)\frac{h}{2}, \tag{2}$$

with h the mesh size.

In the template file `fem.cpp`, implement the function

```
void createRHS(Vector& rhs, FunctionPointer f, int N, double dx, const Vector& x)
```

(where `typedef Eigen::VectorXd Vector` and `typedef double(*FunctionPointer)(double)`), that computes the right-hand side for (1) when discretising with the standard basis of hat functions for linear finite elements, and when using the trapezoidal quadrature rule to compute the integrals. The argument `f` is a function pointer to the function f, the vector `x` contains the gridpoints, including the endpoints, and the other arguments are as in subproblem **1b)**.

1d)

(Core problem) In the template file `fem.cpp`, implement the function

```
void femSolve(Vector& u, Vector& x, FunctionPointer f, int N, double a, double b, double ua
    ↪ = 0.0, double ub = 0.0)
```

that computes and stores in `u` the values of the finite element solution u_N at the nodes of the mesh $\mathcal{M}$ and returns them in the row vector `u`. The arguments `a` and `b` supply the domain $\Omega = [a, b]$, whereas `f` is a function pointer to the source function f. The argument `N` passes the number of interior nodes of the equidistant mesh. In the vector `u`, include also the boundary values of u.

1e)

State and justify the asymptotic computational complexity of `femSolve` in terms of the problem size parameter N.

1f)

Plot the finite element solution u_N for $\Omega := [-\pi, \pi]$, $f(x) = \sin(x)$, and $N = 50, 100, 200$. To validate your code compare u_N with the exact analytic solution $u(x) = \sin(x)$.

1g)

(Core problem) Extend your above implementation of `femSolve` using the optional arguments `ua`, `ub` that specify *boundary values* for the solution u of (1), supposing now $u_a, u_b \neq 0$. This means that now we seek to solve (1) under the constraints $u(a) = u_a$, $u(b) = u_b$.
Hint: Use the offset function technique in section 9.1 in the lecture notes.

1h)

Plot the finite element solution u_N for $\Omega := [-\pi, \pi]$, $f(x) = \cos(x)$, $u_a = -1$, $u_b = \frac{1}{2}$ and $N = 50$. To validate your code compare u_N with the exact analytic solution $u(x) = \cos(x) + \frac{3}{4\pi}x + \frac{3}{4}$.

Tip: Make sure that your program exactly respects the boundary conditions by running it with a very low number of interior points (e.g. $N = 3$).

Exercise 2 Green's Formula

To derive the variational formulation from a PDE, we needed multi-dimensional integration by parts as expressed through Green's first formula. In this problem we study the derivation of Green's formula, thus practicing elementary vector analysis.

2a)

Prove Green's formula for $\Omega \subset \mathbb{R}^2$

$$-\int_{\Omega} (\nabla \cdot \mathbf{j})\, v \, d\mathbf{x} = -\int_{\partial\Omega} \mathbf{j} \cdot \mathbf{n}\, v \, dS + \int_{\Omega} \mathbf{j} \cdot \nabla v \, d\mathbf{x},$$

where $\mathbf{j} \in \mathcal{C}^1(\overline{\Omega})^2$ and $v \in \mathcal{C}^1(\overline{\Omega})$.

Hint: Apply Gauss' theorem.

$$\int_\Omega \nabla \cdot \mathbf{F} \, d\mathbf{x} = \int_{\partial\Omega} \mathbf{F} \cdot \mathbf{n} \; dS,$$

where $\mathbf{F} \in \mathcal{C}^1(\overline{\Omega})^2$.

Remember that a function $f : \Omega \to \mathbb{R}$ is said to be of **class one**, written $f \in \mathcal{C}^1(\Omega)$, if f is differentiable, and f' is continuous.

Exercise 3 Integration over the reference element

This exercise deals with the idea of the **reference element**. The first time one encounters this approach, a rather natural reaction is wondering "why do we need this?". Here, we attempt to show how it makes all but the simplest problems easier to deal with.

The best way of convincing oneself that using a reference element is worth the effort is simple – let's try to live without it!

Consider an arbitrary triangle K with vertices $A = (a_1, a_2)$, $B = (b_1, b_2)$ and $C = (c_1, c_2)$ as in Fig. 1.

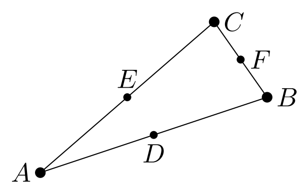

Figure 1: Arbitrary triangle K

3a)

Let φ_A be the **linear** basis function $\varphi_A : K \to \mathbb{R}$, with $\varphi_A(A) = 1$, $\varphi_A(B) = \varphi_A(C) = 0$. Linear basis functions are of the form:

$$\varphi(x, y) = \alpha + \beta x + \gamma y$$

Write the linear system of equations that α, β and γ verify for φ_A.

3b)

Compute one diagonal entry in the local assembly matrix for K for Poisson's equation. That is, compute

$$\int_K \langle \nabla\varphi_A(\mathbf{x}), \nabla\varphi_A(\mathbf{x}) \rangle d\mathbf{x}$$

3c)

Let ϕ_A be the **quadratic** basis function corresponding to A, $\phi_A : K \to \mathbb{R}$. For quadratic FEM on triangular meshes, we need nodes on the midpoints of the sides as well; denote them

$$D = \frac{A+B}{2}, \quad E = \frac{A+C}{2}, \quad F = \frac{B+C}{2}$$

It must hold that $\phi_A(A) = 1$, and $\phi_A(B) = \phi_A(C) = \phi_A(D) = \phi_A(E) = \phi_A(F) = 0$. The generic expression for a quadratic basis function on a triangle is:

$$\phi(x, y) = a + bx + cy + dx^2 + exy + fy^2$$

Write the linear system of equations that a, b, c, d, e, f verify for ϕ_A.

3d)

Compute one diagonal entry in the local assembly matrix for K for Poisson's equation, **without using the reference element**. That is, compute

$$\int_K \langle \nabla\phi_A(\mathbf{x}), \nabla\phi_A(\mathbf{x})\rangle d\mathbf{x}$$

Hint: This task involves easy, but *many*, computations! You are not expected to compute the full answer, just follow them until the point where all that's left to do is repeating operations.

3e)

Using the reference element technique, compute

$$\int_K \langle \nabla\phi_A(\mathbf{x}), \nabla\phi_A(\mathbf{x})\rangle d\mathbf{x}$$

Exercise 4 Integration over the reference element

In this exercise, we will derive the expression for the coefficients in the stiffness matrix as an integral over the reference domain.

Let $T = \{K_i\}_{i=1}^M$ be a triangulation of $\Omega \subset \mathbb{R}^2$, $\{\mathcal{N}_i\}_{i=1}^N$ the set of interior nodes of T, and $\{\varphi_i^N : \Omega \to \mathbb{R}\}_{i=1}^N$ the hat functions in the vector space V:

$$V = \{v : \Omega \to \mathbb{R} : v \text{ continuous}, \ v|_K \text{ linear } \forall K \in T, \ v = 0 \text{ on } \partial\Omega\}$$

Consider the variational formulation of the Poisson equation in Ω for $f : \Omega \to \mathbb{R}$:

$$\int_\Omega \langle \nabla u(x), \nabla v(x) \rangle \, dx = \int_\Omega f(x) v(x) dx, \tag{3}$$

for v a test function in V. As we have seen in the lectures, to find an approximation $\tilde{u} \in V$ to u, we test eq. (3) against all hat functions. Separating the domain into triangles $K \in T$, we obtain the (element-wise) stiffness matrix A^K with coefficients:

$$(A^K)_{i,j} = \int_K \left\langle \nabla \varphi_i^N(x), \nabla \varphi_j^N(x) \right\rangle dx$$

However, when implementing, it is usually advantageous to integrate over a set domain. Thus we introduced the concept of a "reference element" $\hat{K} \subset \mathbb{R}^2$, the triangle defined by points $\{\hat{\mathcal{N}}_a = (0,0), \hat{\mathcal{N}}_b = (1,0), \hat{\mathcal{N}}_c = (0,1)\}$; i.e.

$$\hat{K} = \{(x,y) \in \mathbb{R}^2 : \; x \geq 0, \; y \geq 0, \; x + y \leq 1\}.$$

Fix $K \in T$. We denote $\Phi_K : \hat{K} \to K$ the affine map:

$$x = \Phi_K(\hat{x}) = (\mathcal{N}_b - \mathcal{N}_a, \; \mathcal{N}_c - \mathcal{N}_a)\, \hat{x} + \mathcal{N}_a =: J_K \hat{x} + \mathcal{N}_a$$

where $\mathcal{N}_a$, $\mathcal{N}_b$, $\mathcal{N}_c$ are the three vertices of K, and $\hat{x}$ denotes points in $\hat{K}$. Thus, $\Phi_K(\hat{\mathcal{N}}_\alpha) = \mathcal{N}_\alpha$, for $\alpha \in \{a,b,c\}$. Denote $\hat{\varphi}_\alpha : \hat{K} \to \mathbb{R}$, $\alpha \in \{a,b,c\}$ the hat functions in the reference domain; linear functions with $\hat{\varphi}_\alpha(\hat{\mathcal{N}}_\beta) = \delta_{\alpha,\beta}$, $\alpha, \beta \in \{a,b,c\}$.

4a)

Show that $\forall \alpha \in \{a,b,c\}$,

$$\varphi_\alpha \circ \Phi_K \equiv \hat{\varphi}_\alpha.$$

4b)

We denote $\nabla = (\partial_x, \partial_y)^\intercal$, $\hat{\nabla} = (\partial_{\hat{x}}, \partial_{\hat{y}})^\intercal$. Show that

$$\nabla \varphi_\alpha(x) = \left(J_K^\intercal\right)^{-1} \hat{\nabla} \hat{\varphi}_\alpha(\hat{x})$$

Hint: Use task **4a)** and apply the (multidimensional) chain rule.

4c)

Conclude that

$$(A^K)_{i,j} = \int_{\hat{K}} \left\langle \left(J_K^{-1}\right)^\intercal \hat{\nabla} \hat{\varphi}_i(\hat{x}), \left(J_K^{-1}\right)^\intercal \hat{\nabla} \hat{\varphi}_j(\hat{x}) \right\rangle |\det J_K| \, d\hat{x}$$

Hint: Apply a change of variables and task **4b)**