Theory behind the project in Software and Computing for Applied Physics at UNIBO

Mats Seglem

May 2023

1 Introduction

This is the more in-depth theory behind the code in the repository Software and Computing for Appplied Physics at GitHub. The theory is almost exclusively based on the book Numerical models for differential problems (1), and what I learned in the course TMA4220 Numerical solution of partial differential equations using the finite element method at NTNU in the fall of 2022.

2 2D Poisson problem

The problem I am going to solve is the two-dimensional Posson problem on the unit disc with homogeneous Dirichlet boundary conditions. This problem will be solved using the finite element method. In mathematical formulations, this problem is given by

$$\begin{cases} \nabla^2 u(x,y) = -f(x,y), \ (x,y) \in \Omega \\ u(x,y) = 0, \ (x,y) \in \partial\Omega \end{cases} \tag{1}$$

on the the domain Ω, the unit disk, $\Omega = \{(x,y) : x^2 + y^2 \leq 1\}$. Then $\partial\Omega = \{(x,y) : x^2 + y^2 = 1\}$.

To be able to solve this problem using the finite element method, we need the weak formulation. Assume u satisfies problem 1. Then, multiplying with an arbitrary test function v and integrating over Ω, we get

$$\iint\limits_{\Omega} \nabla^2 uv \mathrm{d}\Omega = -\iint\limits_{\Omega} fv \mathrm{d}\Omega. \tag{2}$$

Greens formula for the divergence operator is

$$\int_{\Omega} \varphi \mathrm{div}(\mathbf{b}) \mathrm{d}\Omega = -\int_{\Omega} \mathbf{b} \nabla\varphi \mathrm{d}\Omega + \int_{\partial\Omega} \mathbf{b} \cdot \mathbf{n} \varphi \mathrm{d}s.$$

Applying this formula on 2 with $\varphi = v$ and $\mathbf{b} = \nabla u$ yields

$$\iint\limits_{\Omega} \nabla^2 uv \mathrm{d}\Omega = -\iint\limits_{\Omega} \nabla u \nabla v \mathrm{d}\Omega + \int\limits_{\partial\Omega} \frac{\partial u}{\partial n} v \mathrm{d}s. \tag{3}$$

From 1, we can see that $u = 0$ on $\partial\Omega$, thus $\frac{\partial u}{\partial n} = 0$ on $\partial\Omega$. Hence,

$$\iint\limits_{\Omega} \nabla^2 uv \mathrm{d}\Omega = -\iint\limits_{\Omega} \nabla u \nabla v \mathrm{d}\Omega$$

and thus the problem can be rewritten as

$$\iint\limits_{\Omega} \nabla u \nabla v \mathrm{d}\Omega = \iint\limits_{\Omega} f v \mathrm{d}\Omega \iff a(u, v) = l(v) \ \forall v \in X. \tag{4}$$

For these integrals to make sense, we need $\nabla u \nabla v \in L^1(\Omega)$, and for this we need ∇u, $\nabla v \in [L^2(\Omega)]^2$. Including that u, v vanishes at the boundary, this gives us the space

$$X = H_0^1(\Omega) = \{v \in H^1(\Omega) : v = 0 \text{ on } \partial\Omega\}$$
$$H^1(\Omega) = \{v : \Omega \to R \text{ s.t } v \in L^2(\Omega), \frac{\partial v}{\partial x_i} \in L^2(\Omega), i = 1, 2\}.$$

Bilinearity of a and linearity of l follows trivially from linearity of weak derivatives and integrals.

In conclusion, the weak formulation is

$$\begin{aligned} &\text{find } u \in H_0^1(\Omega) : \\ &a(u, v) = l(v) \ \forall v \in H_0^1(\Omega) \\ &a(u, v) = \iint\limits_{\Omega} \nabla u \nabla v \mathrm{d}\Omega \\ &l(v) = \iint\limits_{\Omega} f v \mathrm{d}\Omega \end{aligned}$$

It is still hard to find a solution to this problem, so we look for a solution in a smaller space $X_h \subset X$. Let our domain Ω be discretized into M triangles, such that $\Omega = \cup_{k=1}^{M} K_k$. Each triangle K_k is then defined by its three corner nodes (x_i, y_i), and there is a basis function corresponding to each node. The space X_h is defined by

$$X_h = \left\{ v \in X = H_0^1 : v|_{K_k} \in P_1(K_k), 1 \le k \le M \right\}.$$

and the basis functions $\{\varphi_i\}_{i=1}^n$ satisfy

$$X_h = \text{span}\{\phi_i\}_{i=1}^n, \quad \varphi_j(\mathbf{x_i}) = \delta_{ij}.$$

We search to find $u_h \in X_h$, $\forall v \in X_h$. We write u_h and v_h as weighted sums of the basis functions

$$\begin{aligned} u_h &= \sum_{i=1}^{n} u_h^i \varphi_i(x,y) \\ v_h &= \sum_{j=1}^{n} v_h^j \varphi_j(x,y) \end{aligned}$$

From the weak formulation, we then get

$$\begin{aligned} a(u_h, v_h) &= l(v_h) \\ \iint\limits_{\Omega} \nabla u_h \nabla v_h \mathrm{d}\Omega &= \iint\limits_{\Omega} f v_h \mathrm{d}\Omega \\ \iint\limits_{\Omega} \sum_{i=1}^{n} u_h^i \nabla\varphi_i \sum_{j=1}^{n} v_h^j \nabla\varphi_j \mathrm{d}\Omega &= \iint\limits_{\Omega} f \sum_{j=1}^{n} v_h^j \varphi_j \mathrm{d}\Omega \\ \sum_{i=1}^{n}\sum_{j=1}^{n} u_h^i v_h^j \iint\limits_{\Omega} \nabla\varphi_i \nabla\varphi_j \mathrm{d}\Omega &= \sum_{j=1}^{n} v_h^j \iint\limits_{\Omega} f\varphi_j \\ \sum_{i=1}^{n}\sum_{j=1}^{n} u_h^i v_h^j a(\varphi_i, \varphi_j) &= \sum_{j=1}^{n} v_h^j l(\varphi_j) \qquad (5) \\ \mathbf{v}^T A\mathbf{u} &= \mathbf{v}^T \mathbf{f} \;\forall v \in X_h \\ A\mathbf{u} &= \mathbf{f} \end{aligned}$$

with use of the linearity of a() and l(). The Galerkin formulation; Find $u_h \in X_h$ such that $a(u_h, v) = l(v) \forall v \in X_h$, is then equivalent with solving the linear system

$$A\mathbf{u} = \mathbf{f}, \qquad (6)$$

with

$$\begin{aligned} A &= [A_{ij}] = [a(\varphi_i, \varphi_j)] \\ \mathbf{u} &= [u_h^i] \\ \mathbf{f} &= [f_j] = [l(\varphi_j)] \end{aligned}$$

as we can see from equation 5.

3 Meshing

Following the discussion of the galerkin formulation, we have to divide our domain (the unit disc) into triangles. Each triangle defined by its corner nodes is called an element.

4 Assembling Stiffness Matrix and Load Vector

The matrix A in 6 is called the stiffness matrix. We assemble this matrix based on this pseudocode in figure 1

4.4.2 Procedure for $\underline{\bar{A}}_h$

To form $\underline{\widetilde{A}}_h$:

$$
\begin{aligned}
&\text{zero } \underline{\widetilde{A}}_h;\\
&\Big\{\text{for} \quad k = 1, \ldots, K\\
&\qquad \{\text{for} \quad \alpha = 1, 2, 3\\
&\qquad\qquad i = \theta(k, \alpha)\ ;\\
&\qquad \{\text{for} \quad \beta = 1, 2, 3\\
&\qquad\qquad j = \theta(k, \beta)\ ;\\
&\qquad \widetilde{A}_{h\,i\,j} = \widetilde{A}_{h\,i\,j} + A^k_{\alpha\beta}\ ;\}\ \}\ \Big\}
\end{aligned}
$$

Figure 1: Pseudocode for assembly of the matrix A.

The global stiffness matrix is built up from many local 2x2 elemental matrices. To calculate the elemental matrices we need the local basis functions

$$\mathcal{H}^k_\alpha = c^k_\alpha + c^k_{x\alpha} x + c^k_{y\alpha} y, \tag{7}$$

where we find the coefficients c^k_α, $c^k_{x\alpha}$ and $c^k_{y\alpha}$ from

$$\begin{bmatrix} 1 & x^k_1 & y^k_1 \\ 1 & x^k_2 & y^k_2 \\ 1 & x^k_3 & y^k_3 \end{bmatrix} \begin{bmatrix} c^k_1 \\ c^k_{x_1} \\ c^k_{y_1} \end{bmatrix} \text{ or } \begin{bmatrix} c^k_2 \\ c^k_{x_2} \\ c^k_{y_2} \end{bmatrix} \text{ or } \begin{bmatrix} c^k_3 \\ c^k_{x_3} \\ c^k_{y_3} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \text{ or } \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \text{ or } \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

The elemental matrices is then

$$
\begin{aligned}
(A^k)_{\alpha,\beta} &= \int_{T^k_h} \frac{\partial \mathcal{H}^k_{1,2or3}}{\partial x} \frac{\partial \mathcal{H}^k_{1,2or3}}{\partial x} + \frac{\partial \mathcal{H}^k_{1,2or3}}{\partial y} \frac{\partial \mathcal{H}^k_{1,2or3}}{\partial y} d\Omega \\
&= Area(T^k_h) \cdot (c^k_{x\alpha} c^k_{x\beta} + c^k_{y\alpha} c^k_{y\beta}).
\end{aligned}
$$

The right hand side of equation (6) is called the load vector. The load vector can be assembled in a similar way, following the pseudocode in figure 2.

4.4.3 Procedure for $\underline{\tilde{F}}_h$

To form $\underline{\widetilde{F}}_h$:

zero $\underline{\widetilde{F}}_h$;

{for $k = 1, \ldots, K$

{for $\alpha = 1, 2, 3$

$i = \theta(k, \alpha)$;

$\widetilde{F}_{h\ i} = \widetilde{F}_{h\ i} + F_\alpha^k$; } }

Figure 2: Pseudocode for assembly of F

As for the stiffness matrix, the global load vector is assebled from elemental loads. The elemental loads are

$$F_\alpha^k = \int_\Omega f \cdot \mathcal{H}_{1,2or3}^k,$$

where $\mathcal{H}_\alpha^k$ is calculated as in equation 7.

5 Numerical integration

To solve the integrals for assembling both the elemental matrices and elemental loads, we need a way to do numerical integration. For this project, I implemented a gaussian quadrature scheme. In one dimension the Gauss quadrature takes the form

$$\int_{-1}^{1} g(z)dz \approx \sum_{i=1}^{N_q} \rho_q g(z_q),$$

where N_q is the number of integration points, z_q are the Gaussian quadrature points and ρ_q are the associated Gaussian weights. This extends to higher dimensions by

$$\int_{\hat{\Omega}} g(x)dz \approx \sum_{i=1}^{N_q} \rho_q g(z_q).$$

specifying the vector quadrature points zq as well as integrating over a suitable reference domain $\hat{\Omega}$, which are triangles in our case. In higher dimensional case, we first map the integrand to barycentric coordinates, and then the gauss

N_q	(z_1, z_2, z_3)	ρ
1	(1/3, 1/3, 1/3)	1
3	(1/2, 1/2, 0)	1/3
	(1/2, 0, 1/2)	1/3
	(0, 1/2, 1/2)	1/3
4	(1/3, 1/3, 1/3)	-9/16
	(3/5, 1/5, 1/5)	25/48
	(1/5, 3/5, 1/5)	25/48
	(1/5, 1/5, 3/5)	25/48

Table 1: Caption

points are given as triplets in this coordinate system. The barycentric coordinates (z_1, z_2, z_3) are such that

$$z_1 + z_2 + z_3 = 1$$

and it gives out the point of interest from

$$P = z_1 p_1 + z_2 p_2 + z_3 p_3$$

where p_i are the triangle vertices. For 2D gaussian quadrature, the gaussian quadrature nodes and weights are given in table 1.

6 Imposing boundary conditions

After the assembly of the stiffness matrix and load vector, we are in the situation that the stiffness matrix A is singular, and thus we have no solution to the linear system 6. So, we have to impose the boundary conditions!

As we only consider homogeneous dirichlet boundary condtions in this case, the simplest way to impose them, is to remove the rows and columns in A corresponding to function values on the boundary, and remove the same elements from the load vector.

After this, the only thing remaining is to solve the linear system.

References

[1] Quarteroni, Alfio (2019). *Numerical Models for Differential Problems.* Springer. 3rd ed.