TMA4220: Programming project - part 1

by Kjetil André Johannessen
TMA4220 - Numerical solution of partial differential equations using the finite element method

The programming project will be split into two parts. This is the first part and is mandatory for all students. It is an introduction into the finite element method and is designed to build up a solid code base for part 2, where you will actually solve larger, real life problems. For the second part you will have to choose one of several tasks to implement.

1 Gaussian quadrature

At the heart of every finite element code, lies the evaluation of an integral. This integral may be of varying complexity depending on the problem at hand, and many of these integrals does not even have a known analytical solution. Some integrals are *possible* to solve analytically, but of such computational complexity that it is impractical to do so. As such, one often refers to numerical integration schemes to do the core integration. One popular integration scheme is the *Gaussian quadrature*.

In one dimension the gauss quadrature takes the form

$$\int_{-1}^{1} g(x)dx \approx \sum_{q=1}^{N_q} \rho_q g(z_q),$$

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

$$\int_{\hat{\Omega}} g(\boldsymbol{x})d\boldsymbol{x} \approx \sum_{q=1}^{N_q} \rho_q g(\boldsymbol{z}_q),$$

and specifying the vector quadrature points $\boldsymbol{z}_q$ as well as integrating over a suitable reference domain $\hat{\Omega}$ (i.e. squares or triangles in 2D, tetrahedra or cubes in 3D).

a) 1D quadrature

Write a matlab function `I = quadrature1D(a,b,Nq,g)`. With the following arguments:

`I`	$\in \mathbb{R}$	value of the integral
`a`	$\in \mathbb{R}$	integration start
`b`	$\in \mathbb{R}$	integration end
`Nq`	$\in [1, 4]$	number of integration points
`g`	$: \mathbb{R} \to \mathbb{R}$	function pointer*

verify that the function evaluates correctly by comparing with the analytical solution of the integral

$$\int_1^2 e^x \, dx$$

N_q	z_q	ρ_q
1-point-rule	0	2
2-point-rule	$-\sqrt{1/3}$	1
	$\sqrt{1/3}$	1
3-point-rule	$-\sqrt{3/5}$	5/9
	0	8/9
	$\sqrt{3/5}$	5/9
4-point-rule	$-\sqrt{\frac{3+2\sqrt{6/5}}{7}}$	$\frac{18-\sqrt{30}}{36}$
	$-\sqrt{\frac{3-2\sqrt{6/5}}{7}}$	$\frac{18+\sqrt{30}}{36}$
	$\sqrt{\frac{3-2\sqrt{6/5}}{7}}$	$\frac{18+\sqrt{30}}{36}$
	$\sqrt{\frac{3+2\sqrt{6/5}}{7}}$	$\frac{18-\sqrt{30}}{36}$

Table 1: 1D gauss quadrature

b) 2D quadrature

Using all numerical quadratures, it is important to first map the function to the referance domain. In one dimension, this is the interval $\hat{x} \in [-1, 1]$. In higher dimensions, we often map to barycentric coordinates (or area coordinates as they are known in 2D). The gauss points are then given as triplets in this coordinate system. The area coordinates are defined by

$$\begin{aligned} \lambda_1 &= \frac{A_1}{A} \\ \lambda_2 &= \frac{A_2}{A} \\ \lambda_3 &= \frac{A_3}{A} \end{aligned}$$

where A_1, A_2 and A_3 are the area of the triangles depicted in figure 1 and A is the total area of the triangle. Note that these do not form a linear independent basis as $\lambda_1 + \lambda_2 + \lambda_3 = 1$.

N_q	$(\lambda_1, \lambda_2, \lambda_3)$	ρ
1-point rule	$(1/3, 1/3, 1/3)$	1
3-point rule	$(1/2, 1/2, 0)$	1/3
	$(1/2, 0, 1/2)$	1/3
	$(0, 1/2, 1/2)$	1/3
4-point rule	$(1/3, 1/3, 1/3)$	-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 2: 2D gauss quadrature

Write a matlab function `I = quadrature2D(p1,p2,p3,Nq,g)`. With the following arguments:

Hint: An easy way of mapping barycentric coordinates $\boldsymbol{\lambda}$ to physical coordinates $\boldsymbol{x}$ is by $\boldsymbol{x} = \lambda_1 \boldsymbol{p}_1 + \lambda_2 \boldsymbol{p}_2 + \lambda_3 \boldsymbol{p}_3$, where $\boldsymbol{p}_i, i = 1, 2, 3$ is the corner points of the triangle.

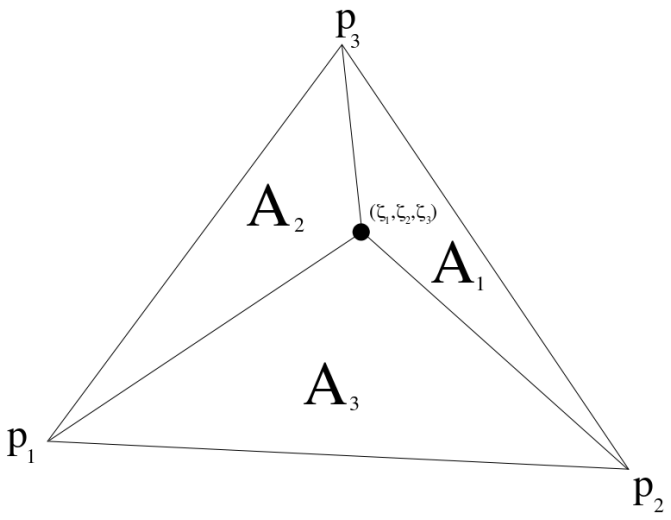

Figure 1: Barycentric coordinates in two dimensions

`I`	$\in \mathbb{R}$	value of the integral
`p1`	$\in \mathbb{R}^2$	first corner point of the triangle
`p2`	$\in \mathbb{R}^2$	second corner point of the triangle
`p3`	$\in \mathbb{R}^2$	third corner point of the triangle
`Nq`	$\in \{1, 3, 4\}$	number of integration points
`g`	$: \mathbb{R}^2 \to \mathbb{R}$	function pointer*

verify that the function evaluates correctly by comparing with the analytical solution of the integral

$$\iint_{\Omega} \log(x + y)\, dx\, dy,$$

where Ω is the triangle defined by the corner points $(1, 0)$, $(3, 1)$ and $(3, 2)$.

c) 3D quadrature

The extension of the barycentric coordinates to 3 dimensions and tetrahedral elements, should be straightforward. The integration schemes can be found in the following table

Write a matlab function `I = quadrature3D(p1,p2,p3,p4,Nq,g)`. With the following arguments:

N_q	$(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$	ρ
1-point rule	$(1/4, 1/4, 1/4, 1/4)$	1
4-point rule	(0.5854102, 0.1381966, 0.1381966, 0.1381966)	0.25
	(0.1381966, 0.5854102, 0.1381966, 0.1381966)	0.25
	(0.1381966, 0.1381966, 0.5854102, 0.1381966)	0.25
	(0.1381966, 0.1381966, 0.1381966, 0.5854102)	0.25
5-point rule	$(1/4, 1/4, 1/4, 1/4)$	-4/5
	$(1/2, 1/6, 1/6, 1/6)$	9/20
	$(1/6, 1/2, 1/6, 1/6)$	9/20
	$(1/6, 1/6, 1/2, 1/6)$	9/20
	$(1/6, 1/6, 1/6, 1/2)$	9/20

Table 3: 3D gauss quadrature

`I`	$\in \mathbb{R}$	value of the integral
`p1`	$\in \mathbb{R}^3$	first corner point of the tetrahedron
`p2`	$\in \mathbb{R}^3$	second corner point of the tetrahedron
`p3`	$\in \mathbb{R}^3$	third corner point of the tetrahedron
`p4`	$\in \mathbb{R}^3$	fourth corner point of the tetrahedron
`Nq`	$\in \{1, 4, 5\}$	number of integration points
`g`	$: \mathbb{R}^3 \to \mathbb{R}$	function pointer*

verify that the function evaluates correctly by comparing with the analytical solution of the integral

$$\iiint_{\Omega} e^x \, dx \, dy \, dz,$$

where Ω is the tetrahedron defined by the corner points $(0, 0, 0)$, $(0, 2, 0)$, $(0, 0, 2)$ and $(2, 0, 0)$.

(*) A function pointer in `matlab` is a variable which represents a function instead of the usual numerical values. In its simplest form it is declared as

```
f = @(x) x^2 + 1
```

which would cause the *variable* `f` to contain a *pointer* to the function $f(x) = x^2 + 1$. The function can then be evaluated using one of two methods

```
y = f(4);
y = feval(f,4);
```

both of which should yield the same result `y = 17`. A function may take in several arguments, i.e. $f(x, y) = x^2 + y^2$ may be declared as

```
f = @(x,y) x^2 + y^2
```

again the evaluation of the function is straightforward

```
y = f(2,2);
y = feval(f,2,2);
```

Provided that the actual function body is capable of vector or matrix operations, then the input arguments may be of vector or matrix form. The syntax remains unchanged by this. You may also use variables in the function declaration, i.e.

```
a = 2;
f = @(x) x*a
```

will result in a function `f` which is doubling its input argument (even if `a` is changed at a later point).

2 Poisson in 2 dimensions

We are going to solve the two-dimensional Poisson problem, given by

$$\begin{aligned} \nabla^2 u(x,y) &= -f(x,y) \\ u(x,y)|_{\partial\Omega} &= 0, \end{aligned} \tag{1}$$

with f given as

$$f(x,y) = 16\pi^2 xy(x^2+y^2)\sin\left(2\pi(x^2+y^2)\right) - 24xy\pi\cos\left(2\pi(x^2+y^2)\right)$$

on the domain Ω given by a 3 quarter slice of the unit disc, i.e. in polar coordinates: $\Omega = \{(r,\theta) \,:\, r \leq 1 \text{ and } \theta \in [0, 3\pi/2]\}$, see figure below

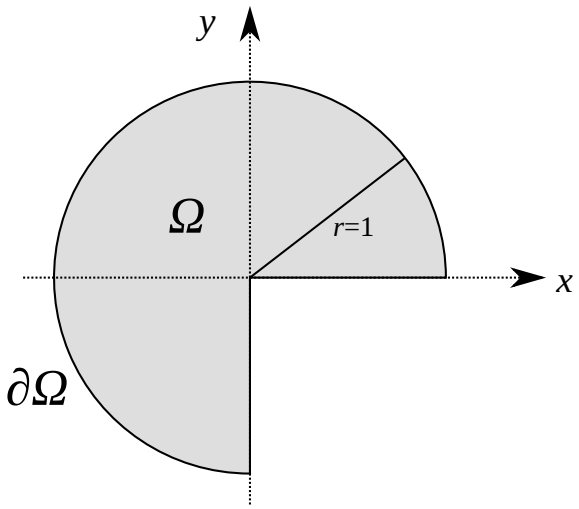

Figure 2: Domain Ω

a) Analytical solution

Verify that the following expression is in fact a solution to the problem (3)

$$u(x,y) = xy\sin\left(2\pi(x^2+y^2)\right). \tag{2}$$

b) Weak formulation

Show that the problem can be rewritten as

$$a(u,v) = F(v), \quad \forall v \in V.$$

with the bilinear functional a and the linear functional l given by

$$\begin{aligned} a(u,v) &= \iint_\Omega \nabla u \cdot \nabla v \, dx\, dy, \\ F(v) &= \iint_\Omega f v \, dx\, dy. \end{aligned}$$

What is the definition of the space V?

c) Galerkin projection

Instead of searching for a solution u in the entire space V we are going to be looking for a solution in a much smaller space $V_h \subset V$. Let Ω be discretized into M triangles such that our computational domain is the union of all of these $\Omega = \cup_{k=1}^{M} K_k$. Each triangle K_k is then defined by its three corner *nodes* $\boldsymbol{x}_i$. For each of these nodes there corresponds one basis function. The space V_h is then defined by

$$V_h = \{v \in V \,:\, v|_{K_k} \in \mathbb{P}_1(K_k), 1 \leq k \leq M\}$$

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

$$V_h = \text{span}\{\varphi_i\}_{i=1}^{n} \qquad \varphi_j(\boldsymbol{x}_i) = \delta_{ij}$$

and δ_{ij} is the Kronecker delta. By searching for a solution $u_h \in V_h$, it is then possible to write this as a weighted sum of the basis functions, i.e. $u_h = \sum_{i=1}^{n} u_h^i \varphi_i(x, y)$.

Show that the problem "Find $u_h \in V_h$ such that $a(u_h, v) = F(v) \quad \forall v \in V_h$" is equivalent to the following problem

Find $\boldsymbol{u}$ such that

$$\boldsymbol{A}\boldsymbol{u} = \boldsymbol{f} \tag{3}$$

with

$$\begin{aligned} \boldsymbol{A} &= [A_{ij}] = [a(\varphi_i, \varphi_j)] \\ \boldsymbol{u} &= [u_h^i] \\ \boldsymbol{f} &= [f_i] = [F(\varphi_i)]. \end{aligned}$$

d) Implementation

We are now going to actually solve the system (3). First we are going to take a look at the triangulation $\{K_k\}$. From the webpage `http://wiki.math.ntnu.no/tma4220/2016h/start` you may download the mesh generators. For organization purposes you might want to keep them in a separate directory and see the matlab `addpath` command.

The function `getSlice` is generating the domain Ω. Plot at least three meshes of different sizes using the mesh generated from this function. You may want to check the matlab function `trimesh` or `triplot`.

e) Stiffness matrix

Build the stiffness matrix $\boldsymbol{A}$. You may choose if you perform the integration analytically or by Gaussian quadrature.

The matrix $\boldsymbol{A}$ should now be singular. Verify this in your code and explain why this is the case.

f) Right hand side

Build the right hand side vector $\boldsymbol{f}$ in the same manner as $\boldsymbol{A}$. Here you might need to resort to Gaussian quadrature.

g) Boundary conditions

Implement the homogeneous dirichlet boundary conditions. Describe what method you used for this and how you did it.

h) Verification

Solve the system (3) and verify that you are getting (approximately) the same result as the analytical solution (2).

3 Neumann boundary conditions

We are going to change to boundary conditions of our problem to

$$\begin{aligned} \nabla^2 u(x,y) &= -f(x,y) \\ u(x,y)|_{\partial\Omega_D} &= 0, \\ \left.\frac{\partial u(x,y)}{\partial n}\right|_{\partial\Omega_N} &= g(x,y), \end{aligned} \tag{4}$$

with the source term f and exact solution u given as above, and g as

$$g(x,y) = \begin{cases} -x\sin(2\pi x^2), & y = 0 \text{ and } (x,y) \in \partial\Omega_N \\ +y\sin(2\pi y^2), & x = 0 \text{ and } (x,y) \in \partial\Omega_N \end{cases} \tag{5}$$

The dirichlet boundary condition is defined on $\partial\Omega_D = \{r = 1, \theta \in [0, 3pi/2]\}$, and the neumann boundary condition as $\partial\Omega_N = \{\theta = \{0, 3\pi/2\}\}$ shown in figure 3.

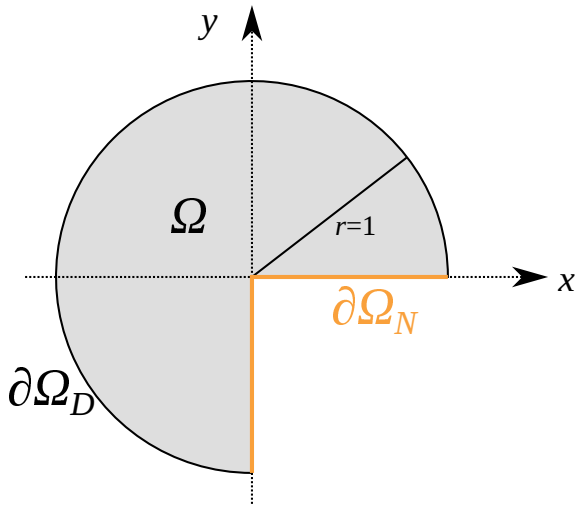

Figure 3: Dirichlet and Neumann boundary conditions

a) Boundary condition

Verify that (2) satisfies (5) on the boundary

b) Variational formulation

How does $a(\cdot,\cdot)$ and $F(\cdot)$ change with the introduction of Neumann boundary conditions?

c) Gauss quadrature

The neumann boundary condition is given as an integral and should be evaluated using Gaussian quadrature. Modify your quadrature methods from task 1 to solve line integrals in two dimensions, i.e. the method signature in `I = quadrature1D(a,b,Nq,g)` should change to `a` $\in \mathbb{R}^2$ and `b` $\in \mathbb{R}^2$.

d) Implementation

Change your code from task 2 to solve this new boundary value problem. How does your solution in the interior compare to the one you got in task 2? How does your solution at the boundary compare?

4 Moving into 3 dimensions

a) The Poisson in 3d

We are now going to solve the problem

$$\begin{aligned} \nabla^2 u &= -f \\ u|_{\partial\Omega} &= 0 \end{aligned}$$

in three dimensions, meaning that we are looking for a solution $u(x, y, z)$.

Generate a mesh, using the function `getBox` from the downloaded mesh generators. This will give you three variables which will describe the nodal points, the tetrahedral elements and the index of the boundary nodes. These should be familiar from task 2 as the only difference is that spatial coordinates have one more component, as well as the elements require one more index to describe.

Modify your code from task 2 to deal with tetrahedral elements in three dimensions. Use the following f

$$f(x, y, z) = -12\pi^2 \sin(2\pi x) \sin(2\pi y) \sin(2\pi z)$$

and homogeneous Dirichlet boundary conditions ($u^D = 0$). This corresponds to the exact solution

$$u(x, y, z) = \sin(2\pi x) \sin(2\pi y) \sin(2\pi z) \tag{6}$$

b) Inhomogeneous dirichlet boundary conditions

Modify your equation to solve the problem

$$\begin{aligned} \nabla^2 u &= -f \\ u|_{\partial\Omega} &= g(x, y, z) \end{aligned}$$

on the domain Ω given by the unit ball. Use the same f as above, but modify your code to use the inhomogeneous dirichlet conditions $g(x, y, z) = u(x, y, z)$ from (6). Use the function `getBall` as your grid generator.

c) Volume visualization

Plot the domain Ω (i.e. the unit sphere). Note that you will not be required to plot every element, as most will be hidden on the inside of the domain. See the matlab function `TriRep` or `tetramesh` for functionality relating to this.

Plot your solution using isosurfaces. Note that the matlab function `isosurface` requires your data to be structured, which it currently is not. You will have to post process the data to get it on the desired form. Read up on `TriScatteredInterp` for this.

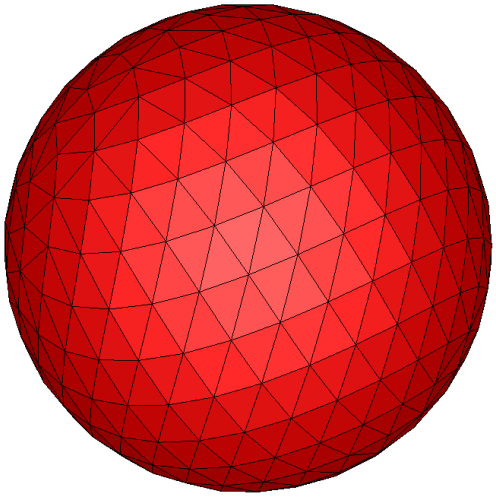

Figure 4: Sample 3d shape

c) GLview (optional)

GLview is a dedicated commercial visualising program to display the results from finite element analysis. It is available to all NTNU students through progdist. While it is possible to display 2D-results in GLview, the advantage of using a dedicated visualiser really is apparent when trying to draw 3D volumetric results.

Download and install GLview inova from progdist and write a `.vtf` output file to view your results. See the attached `writeVTF` function in the `grids.zip` file.

d) Neumann boundary condition (optional)

Change the problem definition to use Neumann boundary conditions for the top half of the sphere (i.e. $z > 0$) and Dirichlet for the bottom part. The neumann condition is given by

$$\begin{aligned} \frac{\partial u}{\partial n} \quad = \quad 2\pi(\quad & x \quad \cos(2\pi x) \quad \sin(2\pi y) \quad \sin(2\pi z) + \\ & y \quad \sin(2\pi x) \quad \cos(2\pi y) \quad \sin(2\pi z) + \\ & z \quad \sin(2\pi x) \quad \sin(2\pi y) \quad \cos(2\pi z)) \end{aligned}$$