Computer Assignment EE4550: Block 2

Finite Element Solver of a Poisson Equation in Two Dimensions

The objective of this assignment is to guide the student to the development of a finite element solver for a 2D Poisson equation given a geometry and a triangular mesh on this geometry. We will assume the mesh to be constructed using the PDE Toolbox in Matlab. This toolbox can be controlled through both a graphical user's interface (GUI) as well as through a command language interface. For educational purpose we will use the latter in this assignment. As before, this assignment consists of a compulsory and an elective part. The former in turn consists of both pen-and-paper and implementation exercises.

In this assignment we aim at solving the Poisson equation on the unit square $(x, y) \in \Omega = (0, 1) \times (0, 1)$. We denote the boundary $\partial\Omega$ of Ω by Γ. We assume as source function $g(x, y)$ with domain Ω to be given and set out to find the function $u(x, y)$ that is solution of the partial differential equation (PDE)

$$-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = g(x, y) \text{ for } (x, y) \in \Omega \tag{1}$$

as well as the homogeneous Dirichlet boundary conditions

$$u = 0 \text{ on } \Gamma = \partial\Omega. \tag{2}$$

The *Laplacian* of u is the function

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \nabla \cdot \nabla u = \operatorname{div} \operatorname{grad} u\,. \tag{3}$$

With this notation the PDE can be written as

$$-\Delta u = g(x, y) \text{ for } (x, y) \in \Omega\,. \tag{4}$$

This equation is called the Poisson equation. In case that $g(x, y) = 0$ it is called the Laplace equation. For particular choices of the source function $g(x, y)$ this PDE supplied with boundary conditions can be solved analytically using separation of variables for instances. In this assignment we will choose $g(x, y)$ such that a given function $u(x, y)$ is the solution of the problem.

Compulsory Theoretical Part

Assignment 1 Choose the function $g(x, y)$ such that the function $u_{ex}(x, y) = x\,(x - 1)\,y\,(1 - y)$ is the exact solution of the above problem. Do so by computing $-\Delta u_{ex}$. Use the `meshgrid` command in Matlab to plot the function $u(x, y)$ for $0 \leq x \leq 1$ and $0 \leq y \leq 1$ for future reference. Other possible choices for u_{ex} include functions of the form $u_{ex}(x, y) = x\,(x - 1)\,y\,(1 - y)\,\widehat{u}_{ex}(x, y)$ where $\widehat{u}_{ex}(x, y)$ is a twice differentiable (i.e. sufficiently) function of x and y.

Assignment 2 Cast the above Poisson equation supplied with the Dirichlet boundary conditions in its continuous weak or variational form. Pay in particular attention to the order of derivatives used and to the choice of the vector space of test functions to resolve the boundary conditions. Please refer to the lecture notes for completing this assignment.

Assume that Ω is discretized by a mesh consisting of $nelem$ elements e_k where k runs from 1 to $nelem$ such that $\Omega = \cup_{k=1}^{nelem} e_k$. Assume the mesh to consist of $nnodes$ vertices (or nodes) $\mathbf{x}_{ij} = (x_{ij}, y_{ij})$. Assume that the nodes of the triangle e_k can be labeled in both a *local* enumeration $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ and in a *global enumeration*. Assume furthermore that on each element e_i the solution to the previously introduced Poisson problem is approximated by linear (first order) Lagrangian shape functions. On each element e_k the weak form is discretized. This requires the computation of an element 3×3 matrix S_{e_k} and the element 3×1 vector f_{e_k} on each element e_k. This matrix and vector are a representation of the second derivative operator $-\Delta$ and the source function $g(x)$ locally on the element e_k.

Assignment 3 The mesh on Ω allows to construct a finite dimensional subspace in which the finite element approximation $u^h(x, y)$ to $u(x, y)$ can be found. Write down the discrete variational form for the approximation $u^h(x, y)$.

Assume the triangle e_k with nodes $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ to be an element of the mesh on Ω. Assume $\phi_1(x, y)$, $\phi_2(x, y)$ and $\phi_3(x, y)$ to be the representation on e_k of the linear shape function centered on the nodes $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$. Then $\phi_1(x, y)$, $\phi_2(x, y)$ and $\phi_3(x, y)$ can be expressed as

$$\phi_1(x, y) = a_1\, x + b_1\, y + c_1 \tag{5}$$

$$\phi_2(x, y) = a_2\, x + b_2\, y + c_2 \tag{6}$$

$$\phi_3(x, y) = a_3\, x + b_3\, y + c_3\,, \tag{7}$$

where the nine coefficients a_i, b_i and c_i for $1 \leq i \leq 3$ should be chosen such that the nine conditions $\phi_1(\mathbf{x}_1) = 1$, $\phi_1(\mathbf{x}_2) = 0 = \phi_1(\mathbf{x}_3)$ and similarly for $\phi_2(x, y)$ and $\phi_3(x, y)$ hold. Given the coordinates of the triangle, the coefficients of the basis functions on the element can thus be found by solving a linear system. The gradient of the shape function $\phi_i(x, y)$ can thus be expressed as

$$\nabla \phi_i(x, y) = \left(\frac{\partial \phi_i}{\partial x}, \frac{\partial \phi_i}{\partial y} \right) = (a_i, b_i) \tag{8}$$

and the inner product of the gradient of the shape function $\phi_i(x, y)$ and the gradient of the shape function $\phi_j(x, y)$ as

$$\nabla \phi_i(x, y) \cdot \nabla \phi_j(x, y) = \frac{\partial \phi_i}{\partial x} \frac{\partial \phi_j}{\partial x} + \frac{\partial \phi_i}{\partial y} \frac{\partial \phi_j}{\partial y} = a_i\, a_j + b_i\, b_j \tag{9}$$

Given a function $f(x, y)$ with domain Ω and given a triangle e_k with nodes $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ and area area(e_k), the surface integral of $f(x, y)$ over the triangle e_k can be approximated by the following trapezoidal rule

$$\int_{e_k} f(x, y)\, dx\, dy \approx \frac{\text{area}(e_k)}{3} \left[f(\mathbf{x}_1) + f(\mathbf{x}_2) + f(\mathbf{x}_3) \right]. \tag{10}$$

Assignment 4 Given the coordinates of the three nodes $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ of a triangular element e_k of the mesh on Ω, derive the coeffient matrix and the right-hand side vector of the linear system that allows to compute the coefficients a_i, b_i and c_i for $1 \leq i \leq 3$. Derive also an expression for the area of e_k.

Assignment 5 Give the expression for the element 3×3 matrix S_{e_i} and the element 3×1 vector f_{e_k} on the element e_k. Use the trapezoidal rule to approximate the integrals on e_k that appear in these expressions.

The contribution S_{e_k} and f_{e_k} on element e_k need to be assembled to the global matrix S and f. This assembly requires the mapping from the local enumeration of the nodes on element e_k to the global enumeration of nodes on the mesh of Ω. The local and global enumeration run from 1 to 3 and from 1 to $nnodes$, respectively. This mapping gives a unique global index to each left and right node of the element e_k and is defined by the connectivity matrix or topology of the mesh.

Assignment 6 Explain why the global matrix S and the global vector f are of size $(nnodes) \times (nnodes)$ and $(nnodes) \times 1$, respectively.

Having the matrix S and the vector f the discrete finite element solution $u^h(x)$ can be computed by a linear system solve.

Compulsory Implementation Part

This part guides the student to the development of a finite element code for solving the above Poisson equation by guiding in the construction of a mesh, the assembly of the linear system on the mesh and solution of the linear system.

Assignment 7 (Constructing the geometry) The unit square domain Ω can be plotted using Matlab's PDE Toolbox in at least two ways. One can either use `pdegplot('squareg')` or `pderect([0 1 0 1])`. The latter starts the GUI of the PDE Toolbox, while the former does not. Use any function to plot the geometry Ω.

Assignment 8 (Constructing the mesh) A mesh on the domain Ω can be constructed using the function `initmesh`. This function uses as input the geometry. To construct a mesh on the unit square using `initmesh`, use the input argument `'squareg'` as in the previous assignment. The size of the elements that `initmesh` generates can be controlled using the input argument `Hmax` to `initmesh`. Other input arguments that control the functioning of `initmesh` can be found using `help initmesh`. The function call `[p,e,t] = initmesh('squareg')` captures the mesh in the output argument `[p,e,t]` where

- `p` is the list of points (both interior and boundary) of the mesh: more precisely, `p` is a matrix of size 2 by $nnodes$ such that the k-th column of `p` contains the x and y-coordinate of the k-th node of the mesh;
- `e` is the list of edges (both interior and boundary) of the mesh;
- `t` is the list of triangles of the mesh: more precisely, `t` is a matrix of size 4 by $nelems$ such that the k-th column of `t` is a integer vector of 4 components. The first three components of this vector are the labels in the global enumeration of the first, second and third node of the k-th element of the mesh. The fourth index is a label of the subdomain to which the element belongs.
  IMPORTANT NOTICE: It is important to observe that the matrix `t` thus contains the information of the topology of the mesh. The matrix `t` is thus the equivalent of the matrix `elemat` in the 1D assignment.

Capturing the output argument `[p,e,t]` allows to plot the mesh using the function call `pdemesh(p,e,t)`. Generate various meshes on Ω using `initmesh` using various values of `Hmax` and plot the meshes generated using `pdemesh`.

Assignment 9 (Separating global indices of interior and boundary nodes) (This part of the assigment is technical, but not hard) In the treatment of the boundary conditions, it will be important to be able to distinguish between the interior and the boundary nodes. Our objective is thus to separate the list if nodes indices running from 1 to $nnodes$ in two parts in which the first and second part correspond to the boundary and interior nodes. We will do so finding the matrix `B1` such that `B1` acting as a filter on all indices returns the indices corresponding to the nodes on the interior. Let more precisely I denote the integer vector of size $nnodes$ such that $I(i) = i$ for $1 \leq i \leq nnodes$. Then our goal in to find the rectangluar matrix `B1` with entries 0 and 1 only such that `B1`$*I$ is an shorter integer vector I_{int} that contains the indices of the interior nodes only. It appear that the function `assempde` is able to generate this matrix `B1`.

Proceed as follows:

- given the previously generated varaibles `[p,e,t]`, call the function `assempde` using the following argument list (see `help assempde` for details)
  ```
  [K, F, B1, UD] = assempde('squareb1',p,e,t,1,0,1);
  ```
- generate the vector I using
  ```
  [discard,nnodes] = size(p);
  I = [1:nnodes]';
  ```
- find the indices corresponding to the interior nodes I_{int} using
  ```
  Iint = B1 * I;
  ```
- find the indices corresponding to the boundary I_{bnd} by taking the complement of I_{int} in I
  ```
  Ibnd = setdiff(I, Iint);
  ```

Assignment 10 Write the routines *SourceFct.m* and *ExactSolu.m* that given the input $(x, y) \in \Omega$ return as output the source function $g(x, y)$ and exact solution $u_{ex}(x, y)$, respectively.

Assignment 11 Write a routine, called *GenerateElementMatrix.m*, that given the coordinates of the nodes $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ of element e_k

- computes the area of the element e_k;
- computes the coefficients a_i, b_i and c_i for $1 \leq i \leq 3$;
- generates the 3×3 element matrix S_{e_k} such that

$$S_{e_k} = \text{area}(e_k) \left(a_i\, a_j + b_i\, b_j\right)_{1\leq i,j\leq 3} . \tag{11}$$

Assignment 12 Write a routine, called *GenerateElementVector.m*, that given the coordinates of the nodes $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ of element e_k

- computes the area of the element e_k;
- generates the 3×1 element vector f_{e_k} such that

$$f_{e_k} = \frac{\text{area}(e_k)}{3} \begin{pmatrix} g(\mathbf{x}_1) \\ g(\mathbf{x}_2) \\ g(\mathbf{x}_3) \end{pmatrix} . \tag{12}$$

Assignment 13 Write a routine, called *AssembleMatrix.m*, that assembles the element matrices S_{e_k} on each element into the global matrix $nnodes \times nnodes$ matrix S. To so by first initializing S to be an empty $nnodes \times nnodes$ matrix and subsequently performing a loop over the elements. In this loop the element matrices are generated and added to the global matrix. In this addition the connectivity of the mesh defined by the matrix *elmat* needs to be taken into account. Write therefore a triple for-loop in which

- the outermost loop indexed by $k = 1, \ldots, nelem$ traverses the elements;
- on each element e_k the element matrix S_{e_k} on the element e_k is computed;
- the innermost two loops indexed by $i, j = 1, 2, 3$ traverse the nodes on kth element;
- the following statement is placed in the innermost loop

$$S(t(i, k), t(j, k)) = S(t(i, k), t(j, k)) + S_{e_k}(i, j) . \tag{13}$$

Assignment 14 Write a routine, called *AssembleVector.m*, that assembles the element vectors f_{e_k} on each element into the global matrix $nnodes \times 1$ vector f. To so by first initializing f to be an empty $nnodes \times 1$ vector and subsequently performing a loop over the elements. In this loop the element vectors are generated and added to the global vector. In this addition the connectivity of the mesh defined by the matrix *elmat* needs to be taken into account. Write therefore a double for-loop in which

- the outermost loop indexed by $k = 1, \ldots, nelem$ traverses the elements;
- on each element e_k the element matrix f_{e_k} on the element e_k is computed;
- the innermost loop indexed by $i = 1, 2, 3$ traverse the nodes on kth element;
- the following statement is placed in the innermost loop

$$f(t(i, k)) = f(t(i, k)) + f_{e_k}(i) . \tag{14}$$

Assignment 15 Modify the first equation of the linear system $S\,u^h = f$ in such a way that the finite element solution $u^h(x)$ satisfied the Dirichlet boundary conditions on the boundary of Ω. This can be accomplished by modifying the first equation of the linear system in the following way. Modify the first row of the matrix S by setting $S(I_{bnd},:) = 0$ and $S(I_{bnd}, I_{bnd}) = I_d$ where I_d is the identify matrix of appropriate size. Modify the first element of f by setting $f(I_{bnd}) = 0$.

Assignment 16 Run the assembly routines to get the matrix S and vector f for $n = 100$. Visualize the matrix S using the command *spy* in matlab. Can you give an interpretation of the picture you obtain?

Assignment 17 Compute the finite element solution u^h using $u^h = S\backslash f$ in Matlab. Compare this solution with the analytical solution found in the first assignment. Use either the function `pdemesh` with the additional argument `U` or the function `pdesurf` to do so.

Elective Part

- look into the details on how the mesh is generated. Report on the construct of the mesh of the boundary and the advancement of the mesh from the boundary to the interior of the domain;
- try other shapes for the domain of computation Ω;
- implement second order Lagrangian elements;