Top Math Summer School on

Adaptive Finite Elements: Analysis and Implementation
Organized by: Kunibert G. Siebert

Pedro Morin

Instituto de Matemática Aplicada del Litoral
Universidad Nacional del Litoral

Santa Fe - Argentina

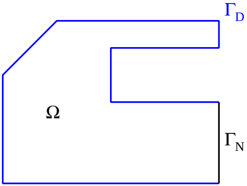
July 28 – August 2, 2008
Frauenchiemsee, Germany

Outline

Weak Formulation

Let us consider the problem

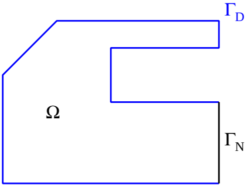
$$
\begin{aligned}
-\operatorname{div}(a\nabla u) + b\cdot\nabla u + c\,u = f \qquad & \text{in } \Omega,\\
u = g_D \qquad & \text{on } \Gamma_D,\\
a\frac{\partial u}{\partial n} = g_N \qquad & \text{on } \Gamma_N,
\end{aligned}
$$



Weak Formulation

Let us consider the problem

$$
\begin{aligned}
-\operatorname{div}(a\nabla u) + b\cdot\nabla u + c\,u &= f \qquad \text{in } \Omega,\\
u &= g_D \qquad \text{on } \Gamma_D,\\
a\frac{\partial u}{\partial n} &= g_N \qquad \text{on } \Gamma_N,
\end{aligned}
$$



Its weak form reads: Find $u \in H^1_{\Gamma_D, g_D}(\Omega)$ such that

$$
\int_\Omega a\nabla u\cdot\nabla v + b\cdot\nabla u\, v + c\, u\, v\, dx = \int_\Omega f\, v\, dx + \int_{\Gamma_N} g_N\, v\, ds, \quad \forall v \in H^1_{\Gamma_D,0}(\Omega).
$$

Here $H^1_{\Gamma_D,g_D}(\Omega) = \{v \in H^1(\Omega) \mid v_{|\Gamma_D} = g_D\}$.

Weak Formulation

Defining

$$B : H^1(\Omega) \times H^1(\Omega) \to \mathbb{R}$$

$$B[v, w] = \int_\Omega a\nabla v \cdot \nabla w + b \cdot \nabla v\, w + c\, v\, w\, dx$$

and

$$F : H^1(\Omega) \to \mathbb{R}$$

$$F(v) = \int_\Omega f\, v\, dx + \int_{\Gamma_N} g_N\, v\, ds$$

Weak Formulation

Defining

$$B : H^1(\Omega) \times H^1(\Omega) \to \mathbb{R}$$
$$B[v,w] = \int_\Omega a\nabla v \cdot \nabla w + b \cdot \nabla v\, w + c\, v\, w\, dx$$

and

$$F : H^1(\Omega) \to \mathbb{R}$$
$$F(v) = \int_\Omega f\, v\, dx + \int_{\Gamma_N} g_N\, v\, ds$$

The weak form reads:

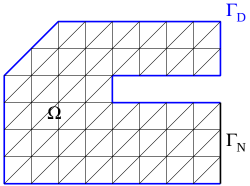
Find $u \in H^1_{\Gamma_D, g_D}(\Omega)$ such that $B[u,v] = F(v), \quad \forall v \in H^1_{\Gamma_D,0}(\Omega).$

Finite Element Formulation:

Consider a triangulation $\mathcal{T}$ of Omega, as for example in the figure, and let

$$\mathbb{V}^{\mathcal{T}} = \{v \in C(\overline{\Omega}) : v_{|T} \text{ is linear}, \forall T \in \mathcal{T}\}$$

Thus each function $v \in \mathbb{V}^{\mathcal{T}}$ is determined by its value at all the *vertices*.



Finite Element Formulation:

Consider a triangulation $\mathcal{T}$ of Omega, as for example in the figure, and let

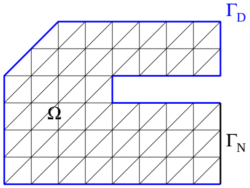
$$\mathbb{V}^{\mathcal{T}} = \{v \in C(\overline{\Omega}) : v_{|T} \text{ is linear}, \forall T \in \mathcal{T}\}$$

Thus each function $v \in \mathbb{V}^{\mathcal{T}}$ is determined by its value at all the *vertices*.

The finite element formulation is thus

Find $u_T \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, g_D}$

$$B[u_{\mathcal{T}}, v_{\mathcal{T}}] = F(v_{\mathcal{T}}), \quad \forall v_{\mathcal{T}} \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, 0}.$$



Finite Element Formulation:

Consider a triangulation $\mathcal{T}$ of Omega, as for example in the figure, and let

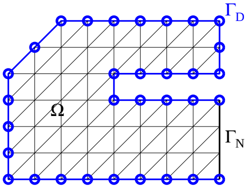
$$\mathbb{V}^{\mathcal{T}} = \{v \in C(\overline{\Omega}) : v_{|T} \text{ is linear}, \forall T \in \mathcal{T}\}$$

Thus each function $v \in \mathbb{V}^{\mathcal{T}}$ is determined by its value at all the *vertices*.

The finite element formulation is thus

Find $u_T \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, g_D}$

$$B[u_{\mathcal{T}}, v_{\mathcal{T}}] = F(v_{\mathcal{T}}), \quad \forall v_{\mathcal{T}} \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, 0}.$$



Here

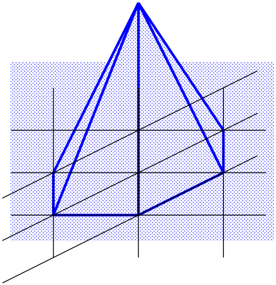
$$\mathbb{V}^{\mathcal{T}}_{\Gamma_D, g_D} = \{v \in \mathbb{V}^{\mathcal{T}} : v(x) = g_D(x) \text{ for every } \textit{vertex } x \text{ on } \Gamma_D\}$$

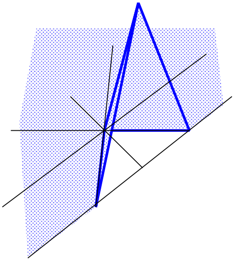
Towards the implementation:

Consider the *nodal basis* $\{\phi_j\}_{j=1}^{N_{\mathcal{T}}}$ of $\mathbb{V}^{\mathcal{T}}$ of functions

$$\phi_j \in \mathbb{V}^{\mathcal{T}} : \qquad \phi_j(x_i) = \delta_{ij}, \qquad i, j = 1, 2, \ldots, N$$

where x_i, $i = 1, 2, \ldots, N$ denote the vertices of the triangulation.





Then, if $v \in \mathbb{V}^{\mathcal{T}}$,

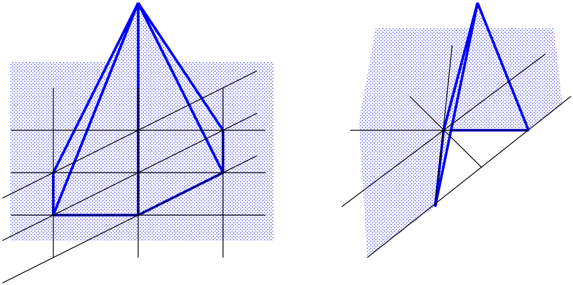
$$v(x) = \sum_{i=1}^{N} v_i \phi_i(x) = \sum_{i=1}^{N} v(x_i) \phi_i(x)$$

Towards the implementation:

Consider the *nodal basis* $\{\phi_j\}_{j=1}^{N_{\mathcal{T}}}$ of $\mathbb{V}^{\mathcal{T}}$ of functions

$$\phi_j \in \mathbb{V}^{\mathcal{T}} : \qquad \phi_j(x_i) = \delta_{ij}, \qquad i,j = 1,2,\dots,N$$

where x_i, $i = 1,2,\dots,N$ denote the vertices of the triangulation.



Then, if $v \in \mathbb{V}^{\mathcal{T}}$,

$$v(x) = \sum_{i=1}^{N} v_i \phi_i(x) = \sum_{i=1}^{N} v(x_i)\phi_i(x)$$

only three terms are added at each x

Implementable FE formulation:

The finite element formulation:

Find $u_{\mathcal{T}} \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, g_D}$

$$B[u_{\mathcal{T}}, v_{\mathcal{T}}] = F(v_{\mathcal{T}}), \quad \forall v_{\mathcal{T}} \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, 0}.$$

Writing $u_{\mathcal{T}} = \sum_{j=1}^{N} \mathbf{u}_j \phi_j$.

Implementable FE formulation:

The finite element formulation:

Find $u_{\mathcal{T}} \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, g_D}$

$$B[u_{\mathcal{T}}, v_{\mathcal{T}}] = F(v_{\mathcal{T}}), \quad \forall v_{\mathcal{T}} \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, 0}.$$

Writing $u_{\mathcal{T}} = \sum_{j=1}^{N} \mathbf{u}_j \phi_j$.

Equivalent formulation: Find $\mathbf{u}_j$, $j = 1, 2, \ldots, N$, such that

$$B\left[\sum_{j=1}^{N} \mathbf{u}_j \phi_j, \phi_i\right] = F(\phi_i), \qquad i = 1, 2, \ldots, N \text{ and } x_i \notin \Gamma_D$$

$$\mathbf{u}_i = g_D(x_i), \qquad i = 1, 2, \ldots, N \text{ and } x_i \in \Gamma_D$$

Implementable FE formulation:

The finite element formulation:

Find $u_{\mathcal{T}} \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, g_D}$

$$B[u_{\mathcal{T}}, v_{\mathcal{T}}] = F(v_{\mathcal{T}}), \quad \forall v_{\mathcal{T}} \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, 0}.$$

Writing $u_{\mathcal{T}} = \sum_{j=1}^{N} \mathbf{u}_j \phi_j$.

Equivalent formulation: Find $\mathbf{u}_j$, $j = 1, 2, \dots, N$, such that

$$\sum_{j=1}^{N} \mathbf{u}_j B\left[\phi_j, \phi_i\right] = F(\phi_i), \qquad i = 1, 2, \dots, N \text{ and } x_i \notin \Gamma_D$$

$$\mathbf{u}_i = g_D(x_i), \qquad i = 1, 2, \dots, N \text{ and } x_i \in \Gamma_D$$

Implementable FE formulation:

The finite element formulation:

Find $u_{\mathcal{T}} \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, g_D}$

$$B[u_{\mathcal{T}}, v_{\mathcal{T}}] = F(v_{\mathcal{T}}), \quad \forall v_{\mathcal{T}} \in \mathbb{V}^{\mathcal{T}}_{\Gamma_D, 0}.$$

Writing $u_{\mathcal{T}} = \sum_{j=1}^{N} \mathbf{u}_j \phi_j$.

Equivalent formulation: Find $\mathbf{u}_j$, $j = 1, 2, \ldots, N$, such that

$$\sum_{j=1}^{N} \mathbf{u}_j B[\phi_j, \phi_i] = F(\phi_i), \qquad i = 1, 2, \ldots, N \text{ and } x_i \notin \Gamma_D$$

$$\mathbf{u}_i = g_D(x_i), \qquad i = 1, 2, \ldots, N \text{ and } x_i \in \Gamma_D$$

$\rightsquigarrow$ a square $N \times N$ linear system

Implementable FE formulation:

Find $\mathbf{u}_j$, $j = 1, 2, \ldots, N$, such that

$$\sum_{j=1}^{N} \mathbf{u}_j B[\phi_j, \phi_i] = F(\phi_i), \qquad i = 1, 2, \ldots, N \text{ and } x_i \notin \Gamma_D$$

$$\mathbf{u}_i = g_D(x_i), \qquad i = 1, 2, \ldots, N \text{ and } x_i \in \Gamma_D$$

$\rightsquigarrow$ a square $N \times N$ linear system

$$A\mathbf{u} = \mathbf{f}$$

with

$$\begin{aligned} A_{ij} &= B[\phi_j, \phi_i] && \text{if } x_i \notin \Gamma_D \\ A_{ij} &= \delta_{ij} && \text{if } x_i \in \Gamma_D \\ \mathbf{f}_i &= F(\phi_i) && \text{if } x_i \notin \Gamma_D \\ \mathbf{f}_i &= g_D(x_i), && \text{if } x_i \in \Gamma_D \end{aligned}$$

Implementable FE formulation:

Find $\mathbf{u}_j$, $j = 1, 2, \ldots, N$, such that

$$\sum_{j=1}^{N} \mathbf{u}_j B[\phi_j, \phi_i] = F(\phi_i), \qquad i = 1, 2, \ldots, N \text{ and } x_i \notin \Gamma_D$$

$$\mathbf{u}_i = g_D(x_i), \qquad i = 1, 2, \ldots, N \text{ and } x_i \in \Gamma_D$$

$\rightsquigarrow$ a square $N \times N$ linear system

$$A\mathbf{u} = \mathbf{f}$$

with

$$A_{ij} = B[\phi_j, \phi_i] = \int_\Omega a \nabla\phi_j \cdot \nabla\phi_i + b \cdot \nabla\phi_j \phi_i + c\,\phi_j \phi_i \, dx \qquad \text{if } x_i \notin \Gamma_D$$

$$A_{ij} = \delta_{ij} \qquad \text{if } x_i \in \Gamma_D$$

$$\mathbf{f}_i = F(\phi_i) = \int_\Omega f \phi_i \, dx + \int_{\Gamma_N} g_N \phi_i \, ds \qquad \text{if } x_i \notin \Gamma_D$$

$$\mathbf{f}_i = g_D(x_i), \qquad \text{if } x_i \in \Gamma_D$$

Implementation:

We want a computer program that:

- Reads a triangulation defining the domain and the boundary regions
- Sets the equation parameters a, b, c, and data f, g_D, g_N
- Assembles the system matrix and right-hand side
- Solves the system and outputs the solution

Implementation:

Observe that:

$$\int_\Omega f\phi_i\,dx = \sum_{T\in\mathcal{T}} \int_T f\phi_i\,dx$$

$$\int_{\Gamma_N} g_N\phi_i\,ds = \sum_{T\in\mathcal{T}} \int_{\partial T\cap\Gamma_N} g_N\phi_i\,ds = \sum_{S\subset\Gamma_N} \int_S g_N\phi_i\,ds$$

$$\int_\Omega a\,\nabla\phi_j\cdot\nabla\phi_i\,dx = \sum_{T\in\mathcal{T}} \int_T a\,\nabla\phi_j\cdot\nabla\phi_i$$

$$\int_\Omega b\cdot\nabla\phi_j\,\phi_i = \sum_{T\in\mathcal{T}} \int_T b\cdot\nabla\phi_j\,\phi_i$$

$$\int_\Omega c\,\phi_j\phi_i\,dx = \sum_{T\in\mathcal{T}} \int_T c\,\phi_j\phi_i\,dx$$

Implementation:

Observe that:

$$\int_\Omega f\phi_i\,dx = \sum_{\substack{T\in\mathcal{T}\\ T\subset\mathrm{supp}(\phi_i)}} \int_T f\phi_i\,dx$$

$$\int_{\Gamma_N} g_N\phi_i\,ds = \sum_{\substack{T\in\mathcal{T}\\ T\subset\mathrm{supp}(\phi_i)}} \int_{\partial T\cap\Gamma_N} g_N\phi_i\,ds = \sum_{\substack{S\subset\Gamma_N\\ S\subset\mathrm{supp}(\phi_i)}} \int_S g_N\phi_i\,ds$$

$$\int_\Omega a\,\nabla\phi_j\cdot\nabla\phi_i\,dx = \sum_{\substack{T\in\mathcal{T}\\ T\subset\mathrm{supp}(\phi_i)\cap\mathrm{supp}(\phi_j)}} \int_T a\,\nabla\phi_j\cdot\nabla\phi_i$$

$$\int_\Omega b\cdot\nabla\phi_j\,\phi_i = \sum_{\substack{T\in\mathcal{T}\\ T\subset\mathrm{supp}(\phi_i)\cap\mathrm{supp}(\phi_j)}} \int_T b\cdot\nabla\phi_j\,\phi_i$$

$$\int_\Omega c\,\phi_j\phi_i\,dx = \sum_{\substack{T\in\mathcal{T}\\ T\subset\mathrm{supp}(\phi_i)\cap\mathrm{supp}(\phi_j)}} \int_T c\,\phi_j\phi_i\,dx$$

just a few!

Assembly of the right-hand side

Consider the part $$\int_\Omega f\phi_i\,dx = \sum_{\substack{T\in\mathcal{T}\\ T\subset\operatorname{supp}(\phi_i)}} \int_T f\phi_i\,dx$$

Assembly of the right-hand side

Consider the part $$\int_\Omega f\phi_i\,dx = \sum_{\substack{T\in\mathcal{T}\\ T\subset \mathrm{supp}(\phi_i)}} \int_T f\phi_i\,dx$$

Idea: Loop over the elements, for each element do:

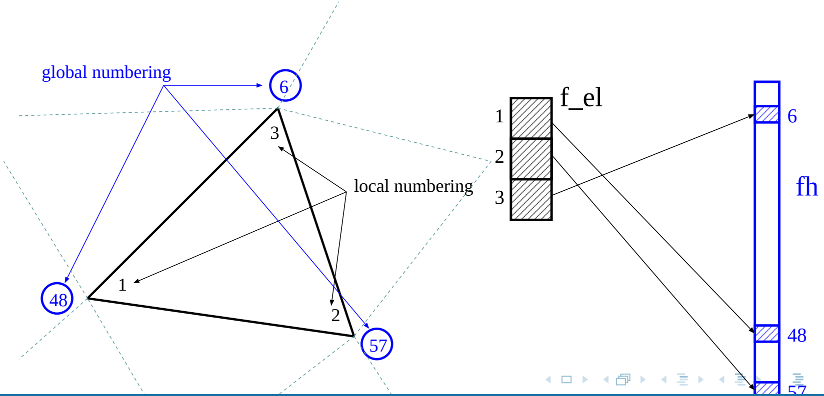
- compute all the integrals that are nonzero at the element;
- add the computed integrals at the proper positions of the right-hand side vector $\mathbf{f}$.

Assembly of the right-hand side

Consider the part $$\int_\Omega f\phi_i\,dx = \sum_{\substack{T\in\mathcal{T}\\ T\subset \mathrm{supp}(\phi_i)}} \int_T f\phi_i\,dx$$

Idea: Loop over the elements, for each element do:

- compute all the integrals that are nonzero at the element;
- add the computed integrals at the proper positions of the right-hand side vector $\mathbf{f}$.

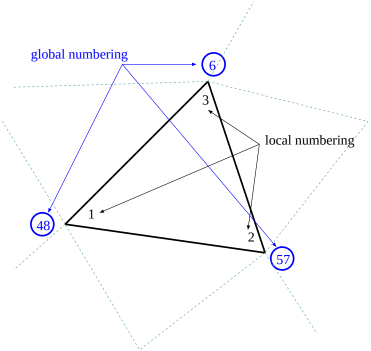


Local basis functions

Observe that

$$\phi_{48|T} = \varphi_T^1$$
$$\phi_{57|T} = \varphi_T^2$$
$$\phi_{6|T} = \varphi_T^3$$

Where φ_T^j is the linear function on T that equals one at the j-th local vertex and zero at the others.

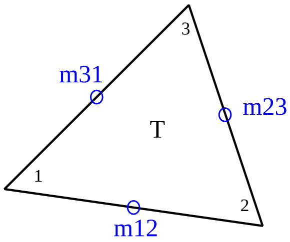


Quadrature

We will use the quadrature formula

$$\int_T g\,dx \approx \frac{|T|}{3}\Big[g(m_{12}) + g(m_{23}) + g(m_{31})\Big] \qquad \longleftarrow \qquad \text{midpoint rule}$$

which is exact for quadratic polynomials.



Quadrature

We will use the quadrature formula

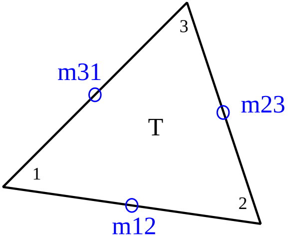
$$\int_T g\,dx \approx \frac{|T|}{3}\Big[g(m_{12}) + g(m_{23}) + g(m_{31})\Big] \quad \longleftarrow \quad \text{midpoint rule}$$

which is exact for quadratic polynomials.

$$f_{el}(1) = \frac{|T|}{3}\left[f(m_{12})\frac{1}{2} + f(m_{23})\,0 + f(m_{31})\frac{1}{2}\right]$$

$$f_{el}(2) = \frac{|T|}{3}\left[f(m_{12})\frac{1}{2} + f(m_{23})\frac{1}{2} + f(m_{31})\,0\right]$$

$$f_{el}(3) = \frac{|T|}{3}\left[f(m_{12})\,0 + f(m_{23})\frac{1}{2} + f(m_{31})\frac{1}{2}\right]$$



Mesh representation

We will assume the existence of four files:

- `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
- `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
- `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
- `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment.

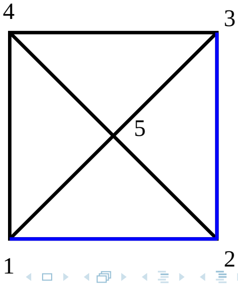
Mesh representation

We will assume the existence of four files:

- vertex_coordinates.txt: containing the coordinates of the vertices of the mesh; one line per vertex.
- elem_vertices.txt: containing the numbers (indices) of the three vertices of each element; one line per element.
- dirichlet.txt: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
- neumann.txt: containing a list of segments that lie on Γ_N, one line per segment.

vertex_coordinates.txt

```
0.0  0.0
1.0  0.0
1.0  1.0
0.0  1.0
0.5  0.5
```



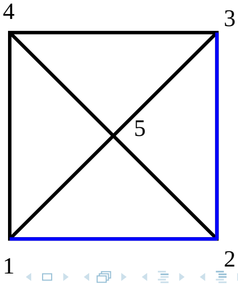
Mesh representation

We will assume the existence of four files:

- `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
- `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
- `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
- `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment.

`elem_vertices.txt`

```
1 2 5
2 3 5
3 4 5
4 1 5
```



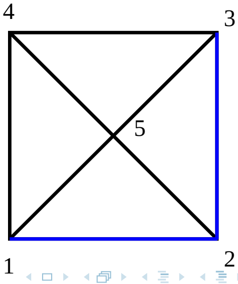
Mesh representation

We will assume the existence of four files:

- `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
- `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
- `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
- `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment.

`dirichlet.txt`

1
2
3



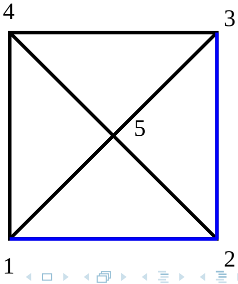
Mesh representation

We will assume the existence of four files:

- vertex_coordinates.txt: containing the coordinates of the vertices of the mesh; one line per vertex.
- elem_vertices.txt: containing the numbers (indices) of the three vertices of each element; one line per element.
- dirichlet.txt: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
- neumann.txt: containing a list of segments that lie on Γ_N, one line per segment.

neumann.txt

```
3 4
4 1
```



Mesh representation

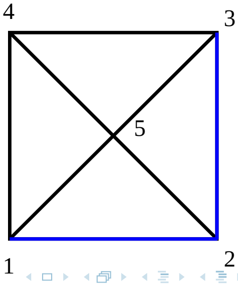
We will assume the existence of four files:

- `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
- `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
- `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
- `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment.

Mesh Generation

In the folder `fixed_mesh` there are two OCTAVE functions that generate meshes:

- `gen_mesh_rectangle.m`
- `gen_mesh_L_shape.m`



Back to the assembly

Let us see some parts of the code fem.m

Initialization

```
coef_a = 1.0;
coef_c = 1.0;

fc_f = inline('sin(pi*x(1))*sin(pi*x(2))', 'x');
fc_gD = inline('1', 'x');
fc_gN = inline('0', 'x');

vertex_coordinates = load('vertex_coordinates.txt');
elem_vertices      = load('elem_vertices.txt');
dirichlet          = load('dirichlet.txt');
neumann            = load('neumann.txt');

n_vertices = size(vertex_coordinates, 1);
n_elem     = size(elem_vertices, 1);
```

Back to the assembly

Let us see some parts of the code `fem.m`

Loop over elements

```
fh = zeros(n_vertices, 1);

for el = 1 : n_elem
    v_elem = elem_vertices( el, : );

    v1 = vertex_coordinates( v_elem(1), :)' ; % coords. of 1st vertex o
    v2 = vertex_coordinates( v_elem(2), :)' ; % coords. of 2nd vertex o
    v3 = vertex_coordinates( v_elem(3), :)' ; % coords. of 3rd vertex o

    m12 = (v1 + v2) / 2; % midpoint of side 1-2
    m23 = (v2 + v3) / 2; % midpoint of side 2-3
    m31 = (v3 + v1) / 2; % midpoint of side 3-1

    % evaluation of f at the quadrature points
    f12 = fc_f(m12);  f23 = fc_f(m23);  f31 = fc_f(m31);
```

Back to the assembly

Let us see some parts of the code fem.m

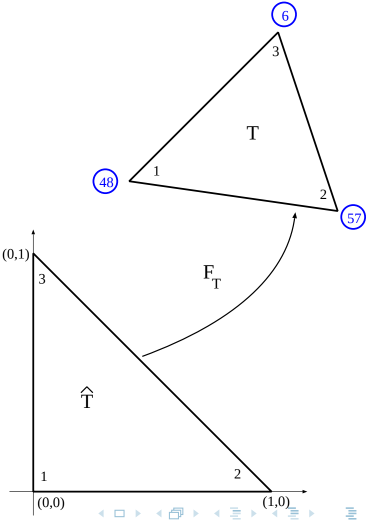
Loop over elements

```
fh = zeros(n_vertices, 1);

for el = 1 : n_elem

    v_elem = elem_vertices( el, : );

    [...]

    % computation of the element load vector
    f_el = [ (f12+f31)*0.5 ; (f12+f23)*0.5 ; (f23+f31)*0.5 ] ...
           * (el_area/3);

    % contributions added to the global load vector
    fh( v_elem ) = fh( v_elem ) + f_el;
end
```

Reference element and mapping

We let v_T^i, $i = 1, 2, 3$ denote the vertex coordinates of the element T.
In our example

$$v_T^1 = x_{48}$$
$$v_T^2 = x_{57}$$
$$v_T^3 = x_6.$$



Reference element and mapping

We let v_T^i, $i = 1, 2, 3$ denote the vertex coordinates of the element T.

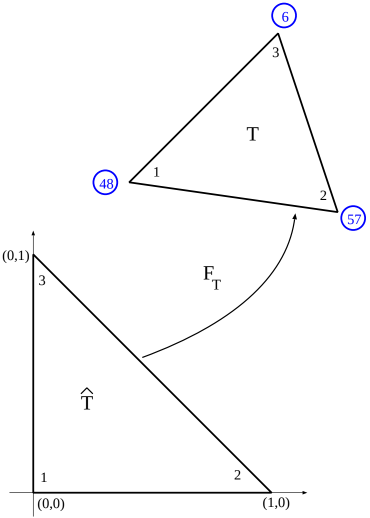
Then $F_T : \hat{T} \to T$

$$F_T(\hat{x}) = v_T^1 + \hat{x}_1(v_T^2 - v_T^1) + \hat{x}_2(v_T^3 - v_T^1)$$
$$= v_T^1 + \underbrace{\left[v_T^2 - v_T^1 \mid v_T^3 - v_T^1\right]}_{B} \begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \end{bmatrix}$$

maps $\hat{T}$ onto T, and

$$B = DF_T$$

$$\frac{|T|}{|\hat{T}|} = 2\,|T| = \det(B).$$

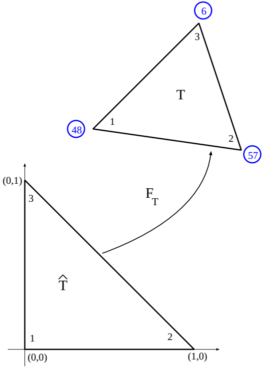


Reference element and mapping

If we define the basis functions on the reference element $\hat{\phi}_i : \hat{T} \to \mathbb{R}$ as

$$\hat{\phi}_1(\hat{x}_1, \hat{x}_2) = 1 - \hat{x}_1 - \hat{x}_2$$
$$\hat{\phi}_2(\hat{x}_1, \hat{x}_2) = \hat{x}_1$$
$$\hat{\phi}_3(\hat{x}_1, \hat{x}_2) = \hat{x}_2$$

Then $\varphi_T^i = \hat{\phi}_i \circ F_T^{-1}$ and $\hat{\phi}_i = \varphi_i \circ F_T.$



Reference element and mapping

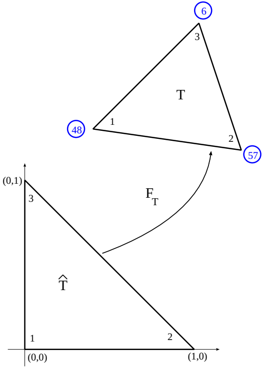
If we define the basis functions on the reference element $\hat{\phi}_i : \hat{T} \to \mathbb{R}$ as

$$\hat{\phi}_1(\hat{x}_1, \hat{x}_2) = 1 - \hat{x}_1 - \hat{x}_2$$
$$\hat{\phi}_2(\hat{x}_1, \hat{x}_2) = \hat{x}_1$$
$$\hat{\phi}_3(\hat{x}_1, \hat{x}_2) = \hat{x}_2$$

Then $\varphi_T^i = \hat{\phi}_i \circ F_T^{-1}$ and $\hat{\phi}_i = \varphi_i \circ F_T$.

And by the chain rule

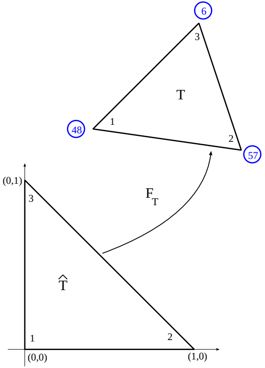
$$\frac{\partial \hat{\phi}_i}{\partial \hat{x}_k} = \sum_\ell \frac{\partial \varphi_i}{\partial x_\ell} \frac{\partial F_{T,\ell}}{\partial x_k} = \sum_\ell \frac{\partial \varphi_i}{\partial x_\ell} B_{\ell k} = \operatorname*{col}_k(B) \cdot \nabla \varphi_i$$



Reference element and mapping

If we define the basis functions on the reference element $\hat{\phi}_i : \hat{T} \to \mathbb{R}$ as

$$\hat{\phi}_1(\hat{x}_1, \hat{x}_2) = 1 - \hat{x}_1 - \hat{x}_2$$
$$\hat{\phi}_2(\hat{x}_1, \hat{x}_2) = \hat{x}_1$$
$$\hat{\phi}_3(\hat{x}_1, \hat{x}_2) = \hat{x}_2$$



Then $\varphi_T^i = \hat{\phi}_i \circ F_T^{-1}$ and $\hat{\phi}_i = \varphi_i \circ F_T$.

And by the chain rule

$$\frac{\partial \hat{\phi}_i}{\partial \hat{x}_k} = \sum_\ell \frac{\partial \varphi_i}{\partial x_\ell} \frac{\partial F_{T,\ell}}{\partial x_k} = \sum_\ell \frac{\partial \varphi_i}{\partial x_\ell} B_{\ell k} = \underset{k}{\mathrm{col}}(B) \cdot \nabla \varphi_i$$

Thus (if gradients are columns)

$$\hat{\nabla} \hat{\phi}_i = B^T \nabla \varphi_i \qquad \text{or} \qquad \nabla \varphi_i = B^{-T} \hat{\nabla} \hat{\phi}_i$$

Computing $\int_T \nabla\varphi_j \cdot \nabla\varphi_i \, dx$

Recall

$$\hat{\nabla}\hat{\phi}_i = B^T \nabla\varphi_i \qquad \text{or} \qquad \nabla\varphi_i = B^{-T}\hat{\nabla}\hat{\phi}_i$$

Computing $\int_T \nabla\varphi_j \cdot \nabla\varphi_i \, dx$

Recall

$$\hat{\nabla}\hat{\phi}_i = B^T \nabla\varphi_i \qquad \text{or} \qquad \nabla\varphi_i = B^{-T}\hat{\nabla}\hat{\phi}_i$$

Then

$$\nabla\varphi_j \cdot \nabla\varphi_i = \nabla\varphi_j^T \nabla\varphi_i = \hat{\nabla}\hat{\phi}_j^T B^{-1} B^{-T} \hat{\nabla}\hat{\phi}_i$$

Computing $\int_T \nabla\varphi_j \cdot \nabla\varphi_i \, dx$

Recall

$$\hat\nabla\hat\phi_i = B^T \nabla\varphi_i \qquad \text{or} \qquad \nabla\varphi_i = B^{-T}\hat\nabla\hat\phi_i$$

Then

$$\nabla\varphi_j \cdot \nabla\varphi_i = \nabla\varphi_j^T \nabla\varphi_i = \hat\nabla\hat\phi_j^T B^{-1} B^{-T} \hat\nabla\hat\phi_i$$

On the other hand, B is constant, and thus

$$\int_T \nabla\varphi_j \cdot \nabla\varphi_i \, dx = \int_{\hat T} \hat\nabla\hat\phi_j^T B^{-1} B^{-T} \hat\nabla\hat\phi_i |\det(B)| \, d\hat x = \frac{|\det(B)|}{2} \hat\nabla\hat\phi_j^T B^{-1} B^{-T} \hat\nabla\hat\phi_i$$

Computing $\int_T \nabla\varphi_j \cdot \nabla\varphi_i \, dx$

On the other hand, B is constant, and thus

$$\int_T \nabla\varphi_j \cdot \nabla\varphi_i \, dx = \int_{\hat{T}} \hat{\nabla}\hat{\phi}_j^T B^{-1} B^{-T} \hat{\nabla}\hat{\phi}_i |\det(B)| \, d\hat{x} = \frac{|\det(B)|}{2} \hat{\nabla}\hat{\phi}_j^T B^{-1} B^{-T} \hat{\nabla}\hat{\phi}_i$$

Also

$$\begin{aligned} \hat{\phi}_1 &= 1 - \hat{x}_1 - \hat{x}_2 \\ \hat{\phi}_2 &= \hat{x}_1 \\ \hat{\phi}_3 &= \hat{x}_2 \end{aligned} \qquad \Longrightarrow \qquad \hat{\nabla}\hat{\phi}_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \quad \hat{\nabla}\hat{\phi}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \hat{\nabla}\hat{\phi}_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Computing $\int_T \nabla\varphi_j \cdot \nabla\varphi_i \, dx$

On the other hand, B is constant, and thus

$$\int_T \nabla\varphi_j \cdot \nabla\varphi_i \, dx = \int_{\hat{T}} \hat{\nabla}\hat{\phi}_j^T B^{-1} B^{-T} \hat{\nabla}\hat{\phi}_i |\det(B)| \, d\hat{x} = \frac{|\det(B)|}{2} \hat{\nabla}\hat{\phi}_j^T B^{-1} B^{-T} \hat{\nabla}\hat{\phi}_i$$

Also

$$\begin{aligned} \hat{\phi}_1 &= 1 - \hat{x}_1 - \hat{x}_2 \\ \hat{\phi}_2 &= \hat{x}_1 \\ \hat{\phi}_3 &= \hat{x}_2 \end{aligned} \qquad \Longrightarrow \qquad \hat{\nabla}\hat{\phi}_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \quad \hat{\nabla}\hat{\phi}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \hat{\nabla}\hat{\phi}_3 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Defining $\texttt{grd\_bas\_fcts} = \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}$ we have that

$$\texttt{el\_mat\_a} = \frac{|\det(B)|}{2} \texttt{grd\_bas\_fcts}^T (B^{-1} B^{-T}) \, \texttt{grd\_bas\_fcts}$$

is a 3×3 matrix satisfying

$$\texttt{mat\_el}_{ij} = \int_T \nabla\varphi_j \cdot \nabla\varphi_i$$

Back to the assembly

Let us see some parts of the code fem.m

Loop over elements

```
A = sparse(n_vertices, n_vertices);

for el = 1 : n_elem
    v_elem = elem_vertices( el, : );
    v1 = vertex_coordinates( v_elem(1), :)' ; % coords. of 1st vertex o
    v2 = vertex_coordinates( v_elem(2), :)' ; % coords. of 2nd vertex o
    v3 = vertex_coordinates( v_elem(3), :)' ; % coords. of 3rd vertex o

    % derivative of the affine transformation from the reference
    % element onto the current element
    B = [ v2-v1  v3-v1 ];
    % element area
    el_area = abs(det(B)) * 0.5;

    % gradients of the basis functions in the reference element
    grd_bas_fcts = [ -1 -1 ; 1 0 ; 0 1 ]' ;
```

Back to the assembly

Let us see some parts of the code fem.m

Loop over elements

```
A = sparse(n_vertices, n_vertices);
for el = 1 : n_elem
    v_elem = elem_vertices( el, : );

    [...]

    % gradients of the basis functions in the reference element
    grd_bas_fcts = [ -1 -1 ; 1 0 ; 0 1 ]' ;

    el_mat = coef_a * grd_bas_fcts' * (Binv*Binv') * grd_bas_fcts ...
             * el_area;

    % contributions added to the global matrix
    A( v_elem, v_elem ) = A( v_elem, v_elem ) + el_mat;
end
```

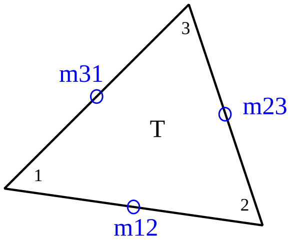
Computing $\int_T \varphi_j \varphi_i \, dx$

Here we just use the midpoint rule:

$$\int_T \varphi_i \varphi_i \, dx = \frac{|T|}{3} \left(\frac{1}{2}\frac{1}{2} + \frac{1}{2}\frac{1}{2} + 0\,0 \right) = |T| \frac{1}{6}$$

and if $i \neq j$

$$\int_T \varphi_i \varphi_j \, dx = \frac{|T|}{3} \left(\frac{1}{2}\frac{1}{2} + \frac{1}{2}\,0 + 0\,\frac{1}{2} \right) = |T| \frac{1}{12}$$



Computing $\int_T \varphi_j \varphi_i \, dx$

Here we just use the midpoint rule:

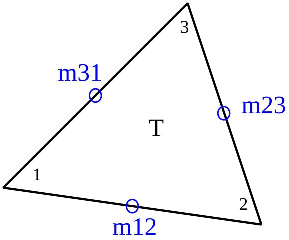
$$\int_T \varphi_i \varphi_i \, dx = \frac{|T|}{3} \left(\frac{1}{2}\frac{1}{2} + \frac{1}{2}\frac{1}{2} + 0\,0 \right) = |T|\frac{1}{6}$$

and if $i \neq j$

$$\int_T \varphi_i \varphi_j \, dx = \frac{|T|}{3} \left(\frac{1}{2}\frac{1}{2} + \frac{1}{2}\,0 + 0\,\frac{1}{2} \right) = |T|\frac{1}{12}$$

Therefore

$$\texttt{el\_mat\_c} = \texttt{el\_area} \begin{bmatrix} \frac{1}{6} & \frac{1}{12} & \frac{1}{12} \\ \frac{1}{12} & \frac{1}{6} & \frac{1}{12} \\ \frac{1}{12} & \frac{1}{12} & \frac{1}{6} \end{bmatrix}$$



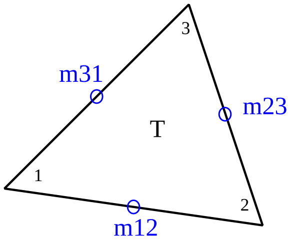
Computing $\int_T \varphi_j \varphi_i \, dx$

Here we just use the midpoint rule:

$$\int_T \varphi_i \varphi_i \, dx = \frac{|T|}{3}\left(\frac{1}{2}\frac{1}{2} + \frac{1}{2}\frac{1}{2} + 0\,0\right) = |T|\frac{1}{6}$$

and if $i \neq j$

$$\int_T \varphi_i \varphi_j \, dx = \frac{|T|}{3}\left(\frac{1}{2}\frac{1}{2} + \frac{1}{2}\,0 + 0\,\frac{1}{2}\right) = |T|\frac{1}{12}$$



Therefore

$$\texttt{el\_mat\_c} = \texttt{el\_area} \begin{bmatrix} \frac{1}{6} & \frac{1}{12} & \frac{1}{12} \\ \frac{1}{12} & \frac{1}{6} & \frac{1}{12} \\ \frac{1}{12} & \frac{1}{12} & \frac{1}{6} \end{bmatrix}$$

In the code

```
el_mat = coef_a*el_area * grd_bas_fcts'*(Binv*Binv')*grd_bas_fcts ...
       + coef_c*el_area * [1/6 1/12 1/12; 1/12 1/6 1/12; 1/12 1/12 1/6]
```

Computing $\int_T b \cdot \nabla\varphi_j \, \varphi_i \, dx$

The computation of

$$\int_T b \cdot \nabla\varphi_j \, \varphi_i \, dx$$

is left as exercise.

Do the computations and include the corresponding modifications into the code.

Outline

Boundary conditions

Recall that we need to solve

$$A\mathbf{u} = \mathbf{f}$$

Boundary conditions

Recall that we need to solve

$$A\mathbf{u} = \mathbf{f}$$

with

$$A_{ij} = B[\phi_j, \phi_i] = \int_\Omega a\,\nabla\phi_j \cdot \nabla\phi_i + b \cdot \nabla\phi_j \phi_i + c\,\phi_j \phi_i\,dx \qquad \text{if } x_i \notin \Gamma_D$$

$$A_{ij} = \delta_{ij} \qquad \text{if } x_i \in \Gamma_D$$

$$\mathbf{f}_i = F(\phi_i) = \int_\Omega f\phi_i\,dx + \int_{\Gamma_N} g_N \phi_i\,ds \qquad \text{if } x_i \notin \Gamma_D$$

$$\mathbf{f}_i = g_D(x_i), \qquad \text{if } x_i \in \Gamma_D$$

Boundary conditions

Recall that we need to solve

$$A\mathbf{u} = \mathbf{f}$$

with

$$A_{ij} = B[\phi_j, \phi_i] = \int_\Omega a\,\nabla\phi_j \cdot \nabla\phi_i + b\cdot\nabla\phi_j\phi_i + c\,\phi_j\phi_i\,dx \qquad \text{if } x_i \notin \Gamma_D$$

$$A_{ij} = \delta_{ij} \qquad \text{if } x_i \in \Gamma_D$$

$$\mathbf{f}_i = F(\phi_i) = \int_\Omega f\phi_i\,dx + \int_{\Gamma_N} g_N\phi_i\,ds \qquad \text{if } x_i \notin \Gamma_D$$

$$\mathbf{f}_i = g_D(x_i), \qquad \text{if } x_i \in \Gamma_D$$

But so far, for $i, j = 1, 2, \ldots, N$

$$\mathtt{A}_{ij} = B[\phi_j, \phi_i], \qquad \text{and} \qquad \mathbf{f}_i = \int_\Omega f\phi_i\,dx.$$

This is ok if x_i is not on Γ_D, and the Neumann contribution is missing.

Neumann boundary conditions

We now loop over the *Neumann edges* and add the contributions

$$\int_S g_N \, \varphi_i \, ds$$

to the corresponding entries on the right-hand side vector.
The integrals on the edges are approximated with Simpson's rule

$$\int_a^b g \, dx \approx \frac{b-a}{6}\Big[g(a) + 4g(\frac{a+b}{2}) + g(b)\Big]$$

which is exact for cubic polynomials.

Neumann boundary conditions

We now loop over the *Neumann edges* and add the contributions

$$\int_S g_N \, \varphi_i \, ds$$

to the corresponding entries on the right-hand side vector.
The integrals on the edges are approximated with Simpson's rule

$$\int_a^b g \, dx \approx \frac{b-a}{6} \Big[g(a) + 4g(\frac{a+b}{2}) + g(b) \Big]$$

which is exact for cubic polynomials.
Then

$$\int_S g_N \, \varphi_1 \, ds = \frac{|S|}{6} \Big[g_N(v_S^1) \, 1 + 4 g_N(m) \, \frac{1}{2} + g_N(v_S^2) \, 0 \Big]$$

$$\int_S g_N \, \varphi_2 \, ds = \frac{|S|}{6} \Big[g_N(v_S^1) \, 0 + 4 g_N(m) \, \frac{1}{2} + g_N(v_S^2) \, 1 \Big]$$

Neumann boundary conditions (code)

$$\int_S g_N \, \varphi_1 \, ds = \frac{|S|}{6} \Big[g_N(v_S^1) + 2 g_N(m) \Big]$$

$$\int_S g_N \, \varphi_2 \, ds = \frac{|S|}{6} \Big[2 g_N(m) + g_N(v_S^2) \Big]$$

```
n_neumann_segments = size(neumann, 1);
for i = 1:n_neuman_segments
  v_seg = neumann(i, :);
  v1 = vertex_coordinates( v_seg(1) , : );   % coords. of 1st vertex
  v2 = vertex_coordinates( v_seg(2) , : );   % coords. of 2nd vertex
  segment_length = norm(v2-v1);

  m = (v1 + v2) / 2;
  g1 = fc_gN(v1);   g2 = fc_gN(v2);   gm = fc_gN(m);

  f_seg = [ g1 + 2 * gm ;  2 * gm + g2 ] * segment_length / 6;

  fh( v_seg ) = fh( v_seg ) + f_seg;
end
```

Dirichlet boundary conditions

If $x_i \in \Gamma_D$ we have to change the i-th equation of the system:

- we have to set the i-th row of A to e_i^T,
- the right-hand side $\mathbf{f}_i$ should be $g(x_i)$.

Dirichlet boundary conditions

If $x_i \in \Gamma_D$ we have to change the i-th equation of the system:

- we have to set the i-th row of A to e_i^T,
- the right-hand side $\mathbf{f}_i$ should be $g(x_i)$.

This is done as follows in the code

```
for i = 1:length(dirichlet)
  diri = dirichlet(i);
  A(diri,:)     = zeros(1, n_vertices);
  A(diri,diri) = 1;
  fh(diri)      = fc_gD( vertex_coordinates(diri, :) );
end
```

Use of provided scripts and functions

To solve a problem using `fem`, we must:

- Generate the files describing the mesh:
  - `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
  - `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
  - `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
  - `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment. This file should not exist if all the boundary is Dirichlet.
- Set the following parameters and data inside `fem.m`
  - Equation coefficients `coef_a`, `coef_b` and `coef_c` corresponding to a, b, c, respectively. They are assumed constant in this version, but feel free to generalize the code. (remember that the convective term $b \cdot \nabla u$ is not implemented)
  - Functions `fc_f`, `fc_gD` and `fc_gN`, corresponding to f, g_D, g_N, respectively.

Use of provided scripts and functions

To solve a problem using `fem`, we must:

- Generate the files describing the mesh:
  - `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
  - `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
  - `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
  - `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment. This file should not exist if all the boundary is Dirichlet.
- Set the following parameters and data inside `fem.m`
  - Equation coefficients `coef_a`, `coef_b` and `coef_c` corresponding to a, b, c, respectively. They are assumed constant in this version, but feel free to generalize the code. (remember that the convective term $b \cdot \nabla u$ is not implemented)
  - Functions `fc_f`, `fc_gD` and `fc_gN`, corresponding to f, g_D, g_N, respectively.

Use of provided scripts and functions

To solve a problem using `fem`, we must:

- Generate the files describing the mesh:
  - `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
  - `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
  - `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
  - `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment. This file should not exist if all the boundary is Dirichlet.
- Set the following parameters and data inside `fem.m`
  - Equation coefficients `coef_a`, `coef_b` and `coef_c` corresponding to a, b, c, respectively. They are assumed constant in this version, but feel free to generalize the code. (remember that the convective term $b \cdot \nabla u$ is not implemented)
  - Functions `fc_f`, `fc_gD` and `fc_gN`, corresponding to f, g_D, g_N, respectively.

Use of provided scripts and functions

To solve a problem using `fem`, we must:

- Generate the files describing the mesh:
  - `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
  - `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
  - `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
  - `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment. This file should not exist if all the boundary is Dirichlet.
- Set the following parameters and data inside `fem.m`
  - Equation coefficients `coef_a`, `coef_b` and `coef_c` corresponding to a, b, c, respectively. They are assumed constant in this version, but feel free to generalize the code. (remember that the convective term $b \cdot \nabla u$ is not implemented)
  - Functions `fc_f`, `fc_gD` and `fc_gN`, corresponding to f, g_D, g_N, respectively.

Use of provided scripts and functions

To solve a problem using `fem`, we must:

- Generate the files describing the mesh:
    - `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
    - `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
    - `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
    - `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment. This file should not exist if all the boundary is Dirichlet.
- Set the following parameters and data inside `fem.m`
    - Equation coefficients `coef_a`, `coef_b` and `coef_c` corresponding to a, b, c, respectively. They are assumed constant in this version, but feel free to generalize the code. (remember that the convective term $b \cdot \nabla u$ is not implemented)
    - Functions `fc_f`, `fc_gD` and `fc_gN`, corresponding to f, g_D, g_N, respectively.

Use of provided scripts and functions

To solve a problem using `fem`, we must:

- Generate the files describing the mesh:
  - `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
  - `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
  - `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
  - `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment. This file should not exist if all the boundary is Dirichlet.
- Set the following parameters and data inside `fem.m`
  - Equation coefficients `coef_a`, `coef_b` and `coef_c` corresponding to a, b, c, respectively. They are assumed constant in this version, but feel free to generalize the code. (remember that the convective term $b \cdot \nabla u$ is not implemented)
  - Functions `fc_f`, `fc_gD` and `fc_gN`, corresponding to f, g_D, g_N, respectively.

Use of provided scripts and functions

To solve a problem using `fem`, we must:

- Generate the files describing the mesh:
  - `vertex_coordinates.txt`: containing the coordinates of the vertices of the mesh; one line per vertex.
  - `elem_vertices.txt`: containing the numbers (indices) of the three vertices of each element; one line per element.
  - `dirichlet.txt`: containing a list with the numbers of the vertices that lie on the Dirichtlet part of the boundary Γ_D; one line per vertex.
  - `neumann.txt`: containing a list of segments that lie on Γ_N, one line per segment. This file should not exist if all the boundary is Dirichlet.
- Set the following parameters and data inside `fem.m`
  - Equation coefficients `coef_a`, `coef_b` and `coef_c` corresponding to a, b, c, respectively. They are assumed constant in this version, but feel free to generalize the code. (remember that the convective term $b \cdot \nabla u$ is not implemented)
  - Functions `fc_f`, `fc_gD` and `fc_gN`, corresponding to f, g_D, g_N, respectively.

Exercises:

1. Solve Poisson equation with pure Dirichlet boundary conditions:

$$\begin{aligned} -\Delta u &= f \qquad \text{in } \Omega = (-1,1)\times(-1,1) \\ u &= g_D \qquad \text{on } \Gamma = \partial\Omega \end{aligned}$$

   Choose f and g_D so that the exact solution $u(x) = e^{-10|x|^2}$.
   - Create the meshes using `gen_mesh_rectangle` with $N = M = 4, 8, 16, 32, 64$.
   - Solve the equations and compute the L_2 and H^1 errors using the provided functions `L2_err` and `H1_err`. Compute the experimental orders of convergence for both norms.

2. Repeat the previous exercise with Ω the L-shaped domain $(-1,1)\times(-1,1)\setminus[0,1]\times[0,1]$, and the exact solution given in polar coordinates by

$$u(r,\theta) = r^{2/3}\sin\left(\frac{2\theta}{3}\right).$$

   Hints: $f \equiv 0$ and the meshes can be generated with `gen_mesh_L_shape`.