Kevin Wallimann wkevin@student.ethz.ch 27.01.2014

FEM Guideline

Note: This overview is on purpose coarse and leaves out a detailed derivation for the sake of simplicity. Left out "tools" that are needed for a complete derivation are marked in bold green.

Model equation

$$-div\left(\sigma(x) grad\, u(x)\right) = f(x)$$
$$u = 0 \; on \; \partial\Omega$$

Variational form aka weak form

Recipe: Multiply the PDE with an abstract test function $v(x)$ and integrate it over the whole domain Ω

$$\int_\Omega -div\big(\sigma(x) grad\, u(x)\big)\, v(x)\, dV = \int_\Omega f(x) v(x) \;\; \forall v \in V_0$$

Apply **Green's first theorem** to the LHS

$$\int_\Omega \sigma(x)\text{grad}\, u(x)\text{grad}\, v(x) dV - \int_{\partial\Omega} \sigma(x)\text{grad}\, u(x) \cdot v(x) \cdot n\, dS = \int_\Omega f(x) v(x)$$

By choosing $v(x) = 0$ on $\partial\Omega$, the second term vanishes. We are perfectly allowed to define v on-the-fly, since we haven't defined it more specifically up to now.

$$\int_\Omega \sigma(x) grad\, u(x)\; grad\, v(x)\; dx = \int_\Omega f(x) v(x) dx \; \forall v \in V$$

This is called the variational or weak form of the PDE.

Sidenote: Dirichlet boundary conditions

Choosing $v(x) = 0$ on the boundary is not as arbitrary as it might seem. Rather it means that Dirichlet boundary conditions apply, i.e. the values on the boundary are prescribed (pinning conditions). If that were not the case, i.e. $v(x) \neq 0$ on the boundary, then (homogeneous) Neumann boundary condition apply (free boundary).

Dirichlet boundary conditions are imposed on the basis functions, unlike Neumann boundary conditions.

Sidenote: Neumann boundary conditions

If Dirichlet BC do not apply, i.e. if $v(x) \neq 0$ on $\partial\Omega$ then the second term does not vanish. To get rid of it anyway, $\sigma(x)\text{grad}\, u(x)$ must be zero, implying the so called homogeneous Neumann boundary condition:

$$\sigma(x)\text{grad}\, u(x) = 0 \text{ on } \partial\Omega$$

If non-homogeneous Neumann BC are given explicitly, i.e.

$$\sigma(x)\text{grad}\, u(x) = h(x) \text{on } \partial\Omega$$

Then the variational form reads

$$\int_{\Omega} \sigma(x)\text{grad}\,u(x)\text{grad}\,v(x)dV - \int_{\partial\Omega} (\underbrace{\boldsymbol{h}(\boldsymbol{x})}_{\sigma(x)\text{grad}\,u(x)} \cdot \boldsymbol{n})v(x)dS = \int_{\Omega} f(x)v(x)$$

You may wonder how Dirichlet boundary conditions are incorporated. Dirichlet boundary conditions are handled later with the basis functions. Unlike Neumann boundary conditions, Dirichlet boundary conditions are not

Discretisation: Basis functions

First of all, the domain Ω is discretised to an unstructured mesh. An unstructured mesh is defined by its nodes and its edges. A common mesh is a triangulation, i.e. the domain consists of small triangles.

Ritz-Galerkin discretisation

$$u(x) \approx u_N(x) = u_0(x) + \sum_{i}^{N} \mu_i b_N^i(x)$$

$u_0(x)$ is the offset function holding boundary values, μ is the vector holding result coefficients for interior nodes and $b_N(x)$ denotes a basis function, where N is the number of nodes in the mesh. $b_N^i(x)$ is an instance of this basis function at node i.

Sidenote: Offset functions

In the above discretisation, $u_0(x)$ is called a "offset function". Offset functions enable the incorporation of Dirichlet BC. In fact, $u_0(x)$ is a vector which is nonzero only for the boundary nodes. For every boundary node, it holds its prescribed boundary value. Although the index 0 might be misleading, $u_0(x)$ is not an initial condition here.

Basis functions

Technically, a basis consists of several local shape functions, i.e. $\mathbb{B}_N = \{b_N^1, b_N^2, \dots, b_N^N\}$. There is a local shape function for every node. Therefore, every triangle supports three shape functions, while all other shape functions are zero on this triangle. Hence, we can just look at one triangle to understand all triangles (aka the whole domain). NB: If the finite elements were quadrilaterals, then every quadrilateral would support four shape functions.

Every basis function (aka shape functions) must satisfy certain criteria.

- $b_N^j(x) = \begin{cases} 1, & i = j \\ 0, & \text{else} \end{cases}$
- $b_N^j(x)$ should be at least once differentiable

There is a variety of basis functions available, the easiest being linear basis functions.

Tools: **Linear basis functions, barycentric coordinates**

Expert tool: **Polynomial basis functions**

Generalisation: Quadrilaterals, polygons

Solving variational form → Matrix equations

The variational form can be solved by inserting the discrete basis into it.

$$\int_\Omega \sigma(x)\text{grad}\, u(x)\text{grad}\, v(x)dx = \int_\Omega f(x)v(x)dx\ \forall v \in V$$

$$\int_\Omega \sigma(x)\text{grad}\left(u_0(x) + \sum_{i=1}^{N} \mu_i b_N^i(x)\right)\text{grad}\, v(x)dx = \int_\Omega f(x)v(x)dx\ \forall v \in V$$

$$\int_\Omega \sigma(x)\left[\text{grad}\left(\sum_{i=1}^{N} \mu_i b_N^i(x)\right) + \text{grad}\, u_0(x)\right]\text{grad}\, v(x)dx = \int_\Omega f(x)v(x)dx\ \forall v \in V$$

$$\sum_{i=1}^{N} \mu_i \int_\Omega \sigma(x)\text{grad}\left(b_N^i(x)\right)\text{grad}\, v(x)dx = \int_\Omega f(x)v(x)dx - \int_\Omega \sigma(x)\text{grad}\, u_0(x)\text{grad}\, v(x)\ \ \forall v \in V$$

Now discretise $v(x)$ in the same fashion as $u(x)$ except that we don't care about boundary terms, i.e. $v(x) = \sum_j^N v_j b_N^j$

$$\sum_{j=1}^{N} v_j \sum_{i=1}^{N} \mu_i \underbrace{\int_\Omega \sigma(x)\text{grad}\left(b_N^i(x)\right)\text{grad}\ \left(b_N^j(x)\right)dx}_{a\left(b_N^i, b_N^j\right)}$$

$$= \sum_{j=1}^{N} v_j \underbrace{\int_\Omega f(x) b_N^j(x)dx}_{l\left(b_N^j\right)} - \sum_{j=1}^{N} v_j \underbrace{\int_\Omega \sigma(x)\text{grad}\, u_0(x)\text{grad}\, b_N^j(x)}_{a\left(u_0, b_N^j\right)}\ \forall v_j \in \mathbb{R}$$

By reordering we can see that we don't need to care about v_j

$$\sum_{j=1}^{N} v_j \left(\underbrace{\sum_{i=1}^{N} \mu_i a\left(b_N^i, b_N^j\right) - l\left(b_N^j\right) + a\left(u_0, b_N^j\right)}_{=0}\right) = 0\ \forall v_j \in \mathbb{R}$$

The expression inside the bracket must be zero for all j, since the equation must hold for all v_j. Therefore, the variational is reduced to

$$\sum_{i=1}^{N} \mu_i a\left(b_N^i, b_N^j\right) = l\left(b_N^j\right) - a\left(u_0, b_N^j\right)\ \ for\ j = 1, \dots, N$$

u_0 denotes the boundary function. Using the nodal basis functions, it can be discretized as

$$u_0 = \sum_{p\in\partial\Omega} \omega_p b_N^p(x)$$

p denote the points on the boundary. The variational form can be rewritten as

$$\sum_{i=1}^{N} \mu_i a\left(b_N^i, b_N^j\right) = l\left(b_N^j\right) - \sum_{p\in\partial\Omega} \omega_p\, a\left(b_N^p, b_N^j\right)\ \ for\ j = 1, \dots, N$$

This is now nothing else than a **linear system of equations** $A\mu = \varphi$, where

$$A = \left(a\left(b_N^i, b_N^j\right)\right)_{i,j=1}^{N} \in \mathbb{R}^{N,N}$$

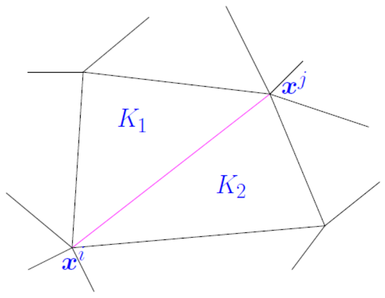
$$\varphi = \left(l(b_N^j) - \boldsymbol{\omega} \cdot a(b_N^p, b_N^j)_{p \in \partial\Omega} \right)_j^N \in \mathbb{R}^N$$

Note: The Dirichlet boundary values can be incorporated in φ where ω is the vector holding the boundary values for each boundary point. It is important to note that N is the number of points inside Ω, i.e. the boundary points are not included to N

Computation and assembly of Galerkin matrix (LHS)

$$A_{ij} = \int_\Omega \sigma(x) \mathrm{grad}\left(b_N^i(x)\right) \mathrm{grad}\ \left(b_N^j(x)\right) dx$$
$$= \int_{K_1} \sigma(x)\ grad\ b_{N|K_1}^i \cdot grad\ b_{N|K_1}^j dx + \int_{K_2} \sigma(x) grad\ b_{N|K_2}^i \cdot grad\ b_{N|K_2}^j dx$$

$A_{ij}$is nonzero only if the nodal basis functions of node I and node j overlap. This is only the case, if node I and node j are neighbours, i.e. connected by an edge. K_1 and K_2 are triangles on either side of the edge.



For i=j, A_{ii} is not composed of only two integrals, but of arbitrarily many, since the nodal function on I overlaps itself on all adjacent triangles. It turns out that we don't have to treat this case specially.

From the perspective of the matrix element A_{ij} we need to know which nodes are involved. This is trivial, since it's by definition node I and node j.

However we can view the situation from the perspective of a triangle. From this point of view, we need to know to which matrix elements a triangle contributes. It turns out that this perspective is computationally more efficient, since it allows to simply loop over the triangles.

Sidenote: A is symmetric positive definite. Symmetry follows from the fact that edge ij is the same as edge ji, positive definiteness follows from the positive definiteness of the integral. Furthermore, A is sparse if the basis functions are hat functions.

Computation of local stiffness matrices

The local stiffness matrices are 3x3 (for triangles) matrices. Like the Galerkin matrix A, it is symmetric positive definite and given as

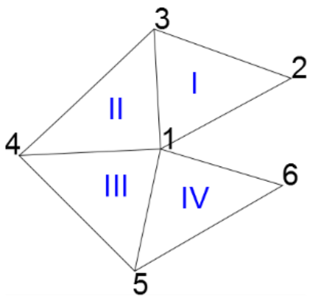
$$LSM = \left(\int_K \sigma(x) grad\ b_{N|K}^s \cdot grad\ b_{N|K}^t dx \right)_{s,t=1}^3 \in \mathbb{R}^{3,3}$$

If the grid was structured in quadrilaterals, the local stiffness matrices would be 4x4 matrices.

The integral can be evaluated using a **quadrature rule** for a unit triangle and **transformation rules** for general triangles. For triangles with one curved boundary, see **boundary approximation.** There is an **analytical solution for $\sigma(x) = 1$**. See tools for details.

Local indexing to global indexing

As already indicated above, we need to know for every element in LSM to which matrix element A_{ij} it must be added. Therefore, we need to know which index each node of a triangle has in the matrix A. This boils down to an index list in the following fashion.



Corresponding global indizes for triangle	**Local index 1**	**Local index 2**	**Local index 3**
I	1	2	3
II	1	3	4
III	1	4	5
IV	1	5	6

Each row denotes one triangle. In total there are 4 triangles and 6 nodes.

Computation and assembly of RHS

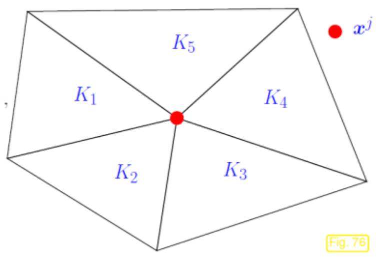
The right hand side consists of two parts

$$\varphi_j = l(b_N^j) - \boldsymbol{\omega} \cdot a(b_N^p, b_N^j)_{p\in\partial\Omega}$$

The computation of $\omega \cdot a(b_N^p, b_N^j)_{p\in\partial\Omega}$ boils down to the computation of the Galerkin matrix (see above, Computation of LHS) where only edges connecting to a boundary node are taken into account.

For the computation of $l(b_N^j)$ we use a similar argument as for the computation of the Galerkin matrix.

$$\int_\Omega f(x) b_N^j(x)dx = \sum_{l=1}^{N} \int_{K_l} f(x) b_{N|K_l}^j(x)dx$$



K_l are the triangles adjacent to node j. $\int_\Omega f(x) b_N^j(x)dx$ is only nonzero in the neighbourhood of node j. Therefore, it is again beneficial to think from the point of view of a triangle.

Again we need to find out to which global node each local node contributes.

Computation of element load vector

The element load vector evaluates $l(b_N^j)$ for each triangle. It is thus defined as

$$ELV = \left(\int_K f(x) b_{N|K}^j(x)dx\right)_{j=1}^3$$

The evaluation is possible with the **2d trapezoidal rule**. In this case, ELV reduces to

$$ELV \approx \frac{|K|}{3}\begin{pmatrix} f(a_1) \\ f(a_2) \\ f(a_3) \end{pmatrix}$$

Where a_1, a_2, a_3 denote the corners of the triangle.

As with the Galerkin matrix, the local indices 1, 2, 3 can be converted to the corresponding global indices by a table.

Boundary conditions

Dirichlet

In the above derivation, Dirichlet boundary conditions were imposed directly on the RHS using the term $\int_\Omega -\boldsymbol{\omega} \cdot a\left(b_N^p, b_N^j\right)_{p\in\partial\Omega, j=[1,N]}$. The Galerkin matrix was only computed for inner nodes by the integrals $\int_\Omega a\left(b_N^i, b_N^j\right)$. In fact, it can be easier to implement if both integrals and the emerging matrices were computed in the same step, into the same matrix, a priori without distinguishing between a boundary node and an inner node. Such a Galerkin matrix had the structure

$$\begin{pmatrix} A_{00} & A_{p0} \\ A_{0p} & A_{pp} \end{pmatrix} =: \begin{pmatrix} \left(\int_\Omega a\left(b_N^i, b_N^j\right)\right)_{i,j=1}^{N} & \left(\int_\Omega a\left(b_N^p, b_N^j\right)\right)_{p\in\partial\Omega, j=[1,N]} \\ \left(\int_\Omega a\left(b_N^i, b_N^p\right)\right)_{i=[1,N], p\in\partial\Omega} & \left(\int_\Omega a\left(b_N^p, b_N^q\right)\right)_{p,q\in\partial\Omega} \end{pmatrix}$$

The solution vector μ could also be split into two parts, i.e. $\mu = \left(\mu_0, \mu_p\right)^T$. The right hand side is split into an inner and a boundary part, too, i.e. $\varphi = \left(\varphi_0, \varphi_p\right)^T$. Now φ_p is nothing else than the prescribed boundary values, that is, $\varphi_p = \boldsymbol{\omega}$. Conversely, $\varphi_0 = l\left(b_N^j\right)$, the function influencing the inner nodes. With this formulation, we have now the following linear system of equations

$$\begin{pmatrix} A_{00} & A_{p0} \\ A_{0p} & A_{pp} \end{pmatrix} \begin{pmatrix} \mu_0 \\ \mu_p \end{pmatrix} = \begin{pmatrix} \varphi_0 \\ \omega \end{pmatrix}$$

Immediately it is clear that $\mu_p = \omega$ otherwise the Dirichlet boundary conditions would be broken. Therefore, we can rewrite

$$A_{00}\mu_0 = \varphi_0 - A_{p0} \cdot \boldsymbol{\omega}$$

This should now look quite familiar, since it corresponds to the derivation before this section. On the left hand side, only the inner nodes are present, while the boundary values are again imposed on the right hand side.

Neumann

Not quite sure about that stuff, but it makes sort of sense.

$$\int_\Omega \sigma(x) \operatorname{grad} u(x) \operatorname{grad} v(x) dV = \int_{\partial\Omega} (\underbrace{\boldsymbol{h}(\boldsymbol{x})}_{\sigma(\mathrm{x})\operatorname{grad} u(x)} \cdot \boldsymbol{n}) v(x)\, dS + \int_\Omega f(x) v(x)$$

This is the variational form if Neumann boundary conditions apply. As shown in the equation, the right hand side must be altered to impose Neumann boundary conditions.

Therefore we just need to know how to solve the integral

$$\int_{\partial\Omega} (\boldsymbol{h}(x) \cdot \boldsymbol{n}) v(x)\, dS$$

First, we need to discretise $v(x)$

$$\int_{\partial\Omega} \underbrace{(\boldsymbol{h}(x) \cdot \boldsymbol{n})}_{\eta(x)} b_N^i(x)\, dS\,, i = 1,2,3$$

Here we have already adopted the triangle-centric point of view with the three nodal basis functions. Since the integral is over $\partial\Omega$ this is a line integral over a boundary edge. Analogously to the element load vector, let us define a element Neumann vector

$$ENV = \left(\int_{\partial\Omega} \underbrace{(\boldsymbol{h}(x) \cdot \boldsymbol{n})}_{\eta(x)} b_N^i(x) dS \right)_{i=1}^{3}$$

The line integral is nonzero only along a boundary edge, therefore ENV looks differently if

- The boundary edge connects nodes 1 and 2
$$ENV = \frac{\|a_1 - a_2\|}{2} \begin{pmatrix} \eta(a_1) \\ \eta(a_2) \\ 0 \end{pmatrix}$$
- The boundary edge connects nodes 1 and 3
$$ENV = \frac{\|a_1 - a_3\|}{2} \begin{pmatrix} \eta(a_1) \\ 0 \\ \eta(a_3) \end{pmatrix}$$
- The boundary edge connects nodes 2 and 3
$$ENV = \frac{\|a_2 - a_3\|}{2} \begin{pmatrix} 0 \\ \eta(a_2) \\ \eta(a_3) \end{pmatrix}$$

For the quadrature, the 1d trapezoidal rule was used.

Sidenote: Of course, ENV looks again different if the triangle has more than one boundary edge.