The Finite Element Method (FEM)

FEM is a systematic way of getting approximated solutions to PDEs.

Let us first look at a 1D problem (this is covered in §11.5 B&F)

\[
\begin{aligned}
(BVP) \quad \begin{cases}
-\frac{d^2 u}{dx^2} &= f \\

u(0) &= 0 \\

u(1) &= 0
\end{cases}
\end{aligned}
\]

Then, B.C. can be considered but we choose homog.-Dirichlet B.C. for simplicity.

Let \( \nu \) be a sufficiently smooth test function with \( \nu(0) = \nu(1) = 0 \).

Multiplying on both sides by \( \nu \) and integrating we get:

\[
(f, \nu) = \int_0^1 f \nu \, dx = -\int_0^1 \nu'' \nu \, dx = \left. \frac{-\nu'}{2} \right|_0^1 + \int_0^1 \nu' \nu' \, dx = 0
\]

We reformulate (BVP) in the variational or weak form:

Find \( u \in V \) s.t.

\[
\alpha(u, v) = (f, v) \quad \forall v \in V.
\]  

The space \( V \) of functions is:

\[
V = \left\{ v \mid \int_0^1 v^2 \, dx < \infty, \int_0^1 (v')^2 \, dx < \infty \text{ and } v(0) = v(1) = 0 \right\}
\]

Note we \( V \) satisfies B.C. automatically. Also, def of \( V \) needs to be adjusted to particular B.C.
Here we used the bilinear form or energy inner product:

$$\alpha(u,v) = \int u'v' \, dx$$

This definition of bilinear form needs to be adjusted to particular problem and could be something more complicated (see 2D).

**Question:** are weak formulation and BVP the same?

- Not necessarily: a solution to BVP must have a second derivative (f)
- Solving (WF) allows us to make sense of what is a "solution" to BVP when we only have one derivative.

Although there are solutions to (WF) that are not solutions to (BVP) one can show that:

**Theorem:** Let \( f \in C^0([0,1]) \) and \( u \in C^0([0,1]) \) solve (WF), then \( u \) solves (BVP).

So when solution is smooth enough: \((BVP) \Rightarrow (WF)\)

**Ritz Galerkin approximation:** This is the theoretical basis for FEM but can also be used in other settings.

Let \( SCV \) be a finite-dimensional subspace of \( V \); this is the "search space" that we seek the solution in and also the "test function space" that we will replace \( V \) in (WF).
The Ritz Galerkin approximation:

Find \( u_0 \in S \) s.t.
\[
\alpha(u_0, v) = (f, v) \quad \forall v \in S
\]

(RG)

Notice how close this is to (WF). The difference is that we now have a finite number of equations to enforce instead of an infinite number of eqs. as in (WF).

Let \( \{\phi_i\}_{i=1}^\infty \) be a basis of \( S \). This means that any function \( v \in S \) can be written as:

\[
v = \sum_{i=1}^{\infty} v_i \phi_i(x) \quad \text{and this decomposition is unique.}
\]

(Another way: \( S = \text{span} \{\phi_i\}_{i=1}^\infty \) and \( \dim S = n \))

We can solve (RG) by solving a linear system:

Let \( u_0 = \sum_{i=1}^{n} U_i \phi_i \)

unknown coeff to be found.

then:

\[
\alpha(u_0, v) = \sum_{i=1}^{n} U_i \alpha(\phi_i, v) = (f, v) \quad \forall v \in S
\]

by lim. of (***)

\[
\Rightarrow \alpha(u_0, \phi_j) = \sum_{i=1}^{n} U_i \alpha(\phi_i, \phi_j) = (f, \phi_j) \quad j = 1, \ldots, m
\]
This gives us a linear system:

\[ AU = F \]

where \( A \in \mathbb{R}^{n \times n} \) and \( A_{ij} = a(\phi_i, \phi_j), i, j = 1, \ldots, n \).

\( F \in \mathbb{R}^n \)

\( U \in \mathbb{R}^n \)

\[ U_i = \text{coeff of } u_i \text{ on basis } \{ \phi_i \} \]

**Question:** This is nice, but do we know for sure that this system \( AU = F \) is solvable?

Yes and the sol. is unique.

**Proof:** For contradiction assume \( F \neq 0 \) s.t. \( A \cdot V = 0 \)

let \( \sigma = \sum_{i=1}^{n} V_i \phi_i \) then

\[ a(\sigma, \phi_j) = 0 \quad j = 1, \ldots, n \]

\[ \sum_{i=1}^{n} V_i a(\sigma, \phi_i) = 0 \]

\[ \Rightarrow a(\sigma, \sigma) = 0 \]

\[ \Rightarrow \int_{\Omega} (\sigma')^2 \, dx \]

\[ \Rightarrow \sigma' = 0 \text{ on } [0,1] \quad \text{(we use continuity)} \]

but we know \( \sigma(0) = 0 \Rightarrow \sigma(x) = 0 \text{ in } [0,1] \)

\[ \Rightarrow V = 0, \text{ a contradiction. QED.} \]
The stiffness matrix $A$ has very nice properties:

- $A^T = A$ because $a(u,v) = a(v,u)$

- $A$ is positive definite:

$$V^T A V = \sum_{i,j} V_i V_j A_{ij} = -a(v,v) \geq 0$$

where $v = \sum_{i=1}^{n} V_i \phi_i$.

Moreover:

$$V^T A V = 0 \Rightarrow v = 0 \Rightarrow V = 0$$

using some argument as before.

Remark: One big advantage of FEM over Finite Differences is that the solution we get is a function and not just a collection of points. We can evaluate this function at any point (not only the grid points as in finite diff.) and we can differentiate it, iterate it etc.

Moreover there is a well-established machinery to get error estimates for the function & its derivatives.
The basic or fundamental error estimate:

**Galarkin orthogonality:**

\[
\alpha(u, w) = (f, w) \quad \forall w \in S \quad (u \text{ solves WF})
\]

\[
\alpha(\mu_s, w) = (f, w) \quad \forall w \in S \quad (\mu_s \text{ solves RG})
\]

\[
\Rightarrow \alpha(u - \mu_s, w) = 0 \quad \forall w \in S \Rightarrow \text{ Galarkin } \perp
\]

Remember picture (by the way this is same picture as for least square...)

Right angle (orthogonality) is measured with \(\alpha(\cdot, \cdot)\) inner product.

\(S = \text{approx. space,}\)

\(\Rightarrow \text{ Galarkin } \perp \text{ means error } u - \mu_s \in \perp \text{ in the sense of } \alpha(\cdot, \cdot) \)

to the approx. space.

Now let us introduce the so called energy norm:

\[
\|u\|_{\text{H}} = (\alpha(u, u))^{\frac{1}{2}} \quad \text{which satisfies the Cundy - Shwang inequality:}
\]

\[
|\alpha(u, w)| \leq \|u\|_{\text{H}} \cdot \|w\|_{\text{H}} \quad \text{for } u, w \in V.
\]
Then for all $v \in S$:

\[
\|u - v\|_E^2 = a(u - v, u - v) = a(u - v, u - v) + a(v - u, v - u) \\
= 0 \text{ by Galerkin}
\]

\[
\leq \|u - v\|_E \|u - v\|_E
\]

If $\|u - v\|_E \neq 0$ then:

\[
\|u - v\|_E \leq \|u - s\|_E \forall s \in S
\]

\[
\Rightarrow \|u - v\|_E \leq \inf_{s \in S} \|u - s\|_E
\]

But since $v \in S$:

\[
\inf_{s \in S} \|u - s\|_E \leq \|u - v\|_E \\
\Rightarrow \|u - v\|_E = \inf_{s \in S} \|u - s\|_E
\]

Since we have an element (namely $v$) in $S$ realizing the infimum we may write:

**Theorem:** $\|u - v\|_E = \min_{s \in S} \|u - s\|_E$

**Translation:** If $G$ approx gives best approx. to $u$ in $S$, if we use $E$ norm as measure then $v$ is norm.

Translation: $G$ approx gives best approx. to $u$ in $S$, if we use $E$ norm.
Before we see what happens with other norms (say $L^2$ norm) we need to look at a particular example of FEM:

**1D P1 elements**

Partition interval $[0,1]$:

$0 = x_0 < x_1 < x_2 \ldots x_n < x_{n+1} = 1$

Let $S$ be the linear space of functions $\phi$ s.t.:

i) $\phi \in C^0[0,1]$

ii) $\phi|_{[x_{i-1},x_i]}$ is a linear poly, for $i = 1, \ldots, n+1$.

iii) $\phi(0) = \phi(1) = 0$.

Recall $V = \{ \phi \mid \int_0^1 \phi' \phi dx < \infty, \int_0^1 (\phi')^2 dx < \infty \text{ and } \phi(0) = \phi(1) = 0 \}$ it is not hard to check that:

$S \subset V$ ($\phi'$ is defined almost everywhere for a piece linear func.)

For convenience we define the basis functions $\phi_i, i = 1, \ldots, n$ with requirement:

$$\phi_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} = "\text{Kronecker delta}"$$

$$\phi_i(x)$$

\[
\begin{array}{c}
\delta_{i-1} \\
\delta_{i} \\
\delta_{i+1}
\end{array}
\]
\[ \{ \phi_i \}_{i=1}^n \text{ is a basis for } S \]

\[ \{ x_i \}_{i=1}^n \text{ are nodes of discretization.} \]

\[ \{ \nu(x_i) \}_{i=1}^n \text{ are nodal values of a function } \nu \]

Any function in \( S \) can be written uniquely as:

\[ \nu(x) = \sum_{i=1}^n \nu_i \phi_i(x) \]

and the coefficients in expansion come right away from the \( S \) property of the \( \{ \phi_j \} \):

\[ \nu(x_j) = \sum_{i=1}^n \nu_i \phi_i(x_j) = \nu_j \]

\[ \sum_{i=1}^n \phi_i(x) \phi_j(x) = \delta_{ij} \]

\[ \nu(x) = \sum_{i=1}^n \nu(x_j) \phi_j(x) \]

\( \nu(x) \) is a function, so we can find its value anywhere on \([0,1]\).

For a function \( \nu \) not necessarily in \( S \) we define its interpolant:

\[ \nu_I(x) = \sum_{i=1}^n \nu(x_i) \phi_i(x) \]
Notice that in general \( v \neq v^h \). Equality holds iff \( v \in S \).

How well can we approximate a function with its interpolant?

**Theorem:** Let \( h = \max x_i - x_{i-1} \). Then for all \( u \in V \):

\[
\| u - u_h \|_E \leq Ch \| u \|_W \|
\]

Here \( C \) is a constant independent of \( h \) and \( u \).

(It is of common usage in the FEM literature to have \( C \) a generic positive constant that may be different from one expression to the next)

**Proof:** The proof of this approximation result consists in:

i) Showing **local error result on an element (interval)** \([x_{i-1}, x_i]\) (using polynomial approx theory we know from MATH 5610)

ii) Summing up approx errors over all elements.

[Note: this proof was not seen in class and I am including it just for your general numerical analysis culture]
The step in gives the poly's estimate:

\[(\ast) \quad \int_{x_{i-1}}^{x_i} [(u-u_x)^2 \, dx \leq C (x_i-x_{i-1})^2 \int_{x_{i-1}}^{x_i} [u''(x)]^2 \, dx \]

Then summing up over elements:

\[\|u-u_x\|_{L^2} = \int_{0}^{1} [(u-u_x)]^2 \, dx = \sum_{i=1}^{n+1} \int_{x_{i-1}}^{x_i} [(u-u_x)]^2 \, dx \]

\[\leq C \sum_{i=1}^{n+1} (x_i-x_{i-1})^2 \int_{x_{i-1}}^{x_i} [u''(x)]^2 \, dx \]

\[\leq C h^2 \sum_{i=1}^{n+1} \int_{x_{i-1}}^{x_i} [u''(x)]^2 \, dx \]

\[= C h^2 \|u''\|^2.\]

\[\Rightarrow \text{result: } \|u-u_x\|_{L^2} \leq C h \|u''\| \text{ by taking square root.}\]

To show the poly's estimate (\(\ast\)), we can rewrite it using \(e = u-u_x\):

\[\int_{x_{i-1}}^{x_i} [e(x)]^2 \, dx \leq C (x_i-x_{i-1})^2 \int_{x_{i-1}}^{x_i} [e''(x)]^2 \, dx \]

since \(u'' = e'' \) (as \(u_x = 0\)).

Then we change variables from \([x_{i-1}, x_i]\) to \([0,1]\):

\[x(\hat{x}) = x_{i-1} + \hat{x} (x_i-x_{i-1}), \quad \hat{x} \in [0,1].\]

\[\hat{e}(\hat{x}) = e(x(\hat{x})) = e(x_{i-1} + \hat{x} (x_i-x_{i-1}))\]
c.o.v. im integral: \[ dx = (x_i - x_{i-1}) \, d\hat{x} \]

\[ (x_i - x_{i-1}) \int_0^1 [e'(x(x))]^2 \, d\hat{x} \leq c \int_0^1 [e''(x(\hat{x}))]^2 \, d\hat{x} \]

Chain rule:

\[ \hat{e}'(\hat{x}) = (x_i - x_{i-1}) e'(x(\hat{x})) \]
\[ \hat{e}''(\hat{x}) = (x_i - x_{i-1})^2 e''(x(\hat{x})) \]

Thus:

\[ \frac{1}{(x_i - x_{i-1})^2} \int_0^1 (\hat{e}'(\hat{x}))^2 \, d\hat{x} \leq c \frac{1}{(x_i - x_{i-1})^4} \int_0^1 (\hat{e}''(\hat{x}))^2 \, d\hat{x} \]

So we need to check estimate:

\[ \int_0^1 (\hat{e}'(\hat{x}))^2 \, d\hat{x} \leq c \int_0^1 (\hat{e}''(\hat{x}))^2 \, d\hat{x} \]

For clarity let us show: \((\omega = \hat{e})\)

\[ \int_0^1 (\omega'(x))^2 \, dx \leq c \int_0^1 (\omega''(x))^2 \, dx \]

Since \(\omega(0) = \omega(1) = 0\) (by interp prop, error is zero at end points)

We can use Rolle's Theorem (or MVT):

\[ \exists \xi \in (0,1) \text{ s.t. } \omega'('\xi') = 0 \]

\[ \Rightarrow \omega'(y) - \omega'(\xi) = \int_0^\xi \omega''(z) \, dz \]
By Cauchy-Schwarz:

\[ |w'(y)| = \left| \int_{x_0}^{y} w''(x) \, dx \right| \leq \int_{x_0}^{y} w''(x)^2 \, dx \leq \left( \int_{x_0}^{y} w''(x) \, dx \right)^2 \]

\[ \leq \sqrt{1 - x_0^2} \cdot \sqrt{1 - x_0^2} \]

Squaring and integrating we get:

\[ \int_{y_0}^{y_1} w(x)^2 \, dy \leq \int_{y_0}^{y_1} |y - x_0| \, dy \cdot \int_{x_0}^{y_1} w''(x)^2 \, dx \]

\[ \leq \frac{1}{2} \text{ for } x_0 \in [0, 1] \]

\[ \Rightarrow \text{ we get result.} \quad \square \]
Computer implementation of FEM

Key idea: Compute \( A = [a(\phi_i, \phi_j)]_{ij} \) by summing contributions of each element.

Naïve FEM implementation

\[
\text{for } i = 1 \ldots n \\
\text{for } j = 1 \ldots n \\
[A_{ij}] = a(\phi_i, \phi_j)
\]

Element by element

\[
\text{for } e = 1, \ldots \# \text{ elements} \\
A = A + A^e
\]

\( A^e \) contribution from element \( e \) to stiffness matrix.

\[
a(\mu, \nu) = \int_0^1 \mu' \nu' \, dx = \sum_{e=1}^{n+1} \int_{x_{e-1}}^{x_e} \mu' \nu' \, dx
\]

\[
= a_e(\mu, \nu) = \text{local bilinear form}
\]

Local bilinear form is easy to evaluate because

\[
\phi_i = 0 \text{ on } K_e \Rightarrow a_e(\phi_i, \phi_j) = 0
\]

Moreover \( \phi_i |_{K_e} = \text{polynomial} \) and integrals in polynomials are easy to compute.

However, we need a \underline{local-to-global map} which relates the local degree of freedom from an element to its corresponding global d.o.f.
In our 1D example:

\[ x_0 = 0 \quad x_1 \quad x_2 \ldots \quad x_n \quad x_{n+1} = 1 \]

Elements \( K_e = [x_{e-1}, x_e] \), \( e = 1, \ldots, n+1 \).

Each element has 2 degrees of freedom (the endpoints).

\[ K_e \]

\[ j=0 \quad j=1 \]

In order to get the global index associated with node \( j \), we can construct the table (matrix):

<table>
<thead>
<tr>
<th>( e )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>\ldots</th>
<th>( e+1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( e )</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>\ldots</td>
<td>( e+1 )</td>
</tr>
</tbody>
</table>

or more succinctly: \( i(e,j) = e + j - 1 \)

Of course: local to global map depends on particular partition of comp domain.

Now rewrite interpolant of a function \( f \) as follows:

\[ f_i = \sum_{e=0}^{n+1} f(x_i) \phi_e = \sum_{e=0}^{n+1} \sum_{j=0}^{1} f(x_i(c_{e,j})) \phi_j \phi_e \]

Note: this expression is wrong at the nodes, however these mistakes at a few points do not affect result because we integrate.
Here we use notation: $\Phi_i^{E} \big|_{i=0}$ for the linear ($P_1$) basis functions on interval $K_e = [x_{e-1}, x_e]$. 

$$\phi_i(x) = \Phi_i \left( \frac{x-x_{e-1}}{x_e-x_{e-1}} \right)$$

Where: 

$$\phi_0(x) = \begin{cases} 1-x & x \in [0,1] \\ 0 & \text{otherwise} \end{cases}$$

$$\phi_1(x) = \begin{cases} x & x \in [0,1] \\ 0 & \text{otherwise} \end{cases}$$

So we have related the "local" basis functions $\Phi_i^{E}$ to the reference (parent) element $[0,1]$ with an affine mapping:

$$[0,1] \rightarrow [x_{e-1}, x_e]$$

Thus:

$$a(u,v) = \sum_{e} a_e(u,v)$$

where

$$a_e(u,v) = \int_{K_e} u v \, dx$$

$$= \sum_{i,j} \left( \sum_{i=0}^{1} \frac{\mu_i(e_{ij}) \phi_i^{E}}{\int_{K_e} \phi_i^{E} dx'} \right) \left( \sum_{j=0}^{1} \frac{\nu_j(e_{ij}) \phi_j^{E}}{\int_{K_e} \phi_j^{E} dx'} \right) dx$$

Chapeau = 

$$= \frac{1}{|K_e|^2} \int_{K_e} \left( \sum_{i=0}^{1} \frac{\mu_i(e_{ij}) \phi_i^{E}}{\int_{K_e} \phi_i^{E} dx'} \right) \left( \sum_{j=0}^{1} \frac{\nu_j(e_{ij}) \phi_j^{E}}{\int_{K_e} \phi_j^{E} dx'} \right) dx$$

C.o.V. 

$$= \frac{1}{|K_e|} \int_{0}^{1} \left( \sum_{i=0}^{1} \frac{\mu_i(e_{ij}) \phi_i^{E}}{\int_{K_e} \phi_i^{E} dx'} \right) \left( \sum_{j=0}^{1} \frac{\nu_j(e_{ij}) \phi_j^{E}}{\int_{K_e} \phi_j^{E} dx'} \right) dx$$
$$a_e(u, u') = \frac{1}{|k_e|} \left[ \mathbf{v}(x) \mathbf{v}(x') \right] A_{loc} \left[ \mathbf{u}(x) \mathbf{u}(x') \right]$$

where \((A^e_{loc})_{i,j} = \int_0^1 \phi_{i-1} \phi_{j-1}' \, dx, \quad i,j = 1, 2\) is the "local" stiffness matrix.

**Typical FEM code looks like this:**

- \(A = \text{all zeros} \quad \# \text{d.o.f \times d.o.f. matrix (sparse)}\)
- \(F = \text{all zeros} \quad \# \text{d.o.f. \times 1}\)

For \(e = 1, \ldots \# \text{elements}\)

\[
\begin{align*}
A(i(e,:), i(e,:)) &= A(i(e,:), i(e,:)) + A_{loc}^e \\
F(i(e,:)) &= F(i(e,:)) + F_{loc}^e
\end{align*}
\]

when:

\[
(A_{loc}^e)_{i,j} = a_e^e \left( \phi_i^e, \phi_j^e \right), \quad i,j = 1, \ldots \# \text{local d.o.f.}
\]

\[
(F_{loc}^e)_{i} = \left( f, \phi_i^e \right), \quad i = 1, \ldots \# \text{local d.o.f.}
\]
Error estimates revisited

We showed earlier the basic error estimate:

$$\| u - u_S \|_E = \min_{v \in S} \| u - v \|_E$$

(BE)

where $\| u \|_E = (a(u, u))^{\frac{1}{2}}$

and $u$ solves (WF)

$u_S$ solves (R6).

However, bound on derivatives may not be practical!

How to get an estimate of the $L^2$ error:

$$\| u - u_S \|_{L^2} = \left( \int_0^1 (u - u_S)^2 \, dx \right)^{\frac{1}{2}}$$

First using (BE) we obtain:

$$\left\| u - u_S \right\|_E \leq \left\| u - u_I \right\|_E \leq C R \| u \|_E$$

where $u_I \in S$

$$R = \max_{i=1, \ldots, n} \left| x_i - x_{i-1} \right|$$

To get $L^2$ error estimate we use the so-called duality or Aubin-Nitsche trick:

Let $w$ be a solution to:

$$\begin{cases}
-w'' = u - u_S \quad &\text{on } [0, 1] \\
w(0) = w(1) = 0
\end{cases}$$
Then:
\[ \|\mathbf{u} - \mathbf{u}_S\|_E^2 = (\mathbf{u} - \mathbf{u}_S, \mathbf{u} - \mathbf{u}_S) = (\mathbf{u} - \mathbf{u}_S, \omega) \]
\[ \text{implies} \quad (\mathbf{u} - \mathbf{u}_S, \omega) - (\mathbf{u}_S, \omega') \bigg|_{\theta = 1} = 0 \text{ by B.C.} \]
\[ = a(\mathbf{u} - \mathbf{u}_S, \omega - \omega') \quad \forall \omega \in S \]

This using Schwarz inequality:
\[ \|\mathbf{u} - \mathbf{u}_S\|_E^2 \leq \|\mathbf{u} - \mathbf{u}_S\| \|\omega - \omega'\|_E \]
\[ \Rightarrow \|\mathbf{u} - \mathbf{u}_S\| \leq \|\mathbf{u} - \mathbf{u}_S\|_E \frac{\|\omega - \omega'\|_E}{\|\omega'\|} \]

Taking inf. over \( S \):
\[ \|\mathbf{u} - \mathbf{u}_S\| \leq \|\mathbf{u} - \mathbf{u}_S\|_E \inf_{\omega \in S} \frac{\|\omega - \omega'\|_E}{\|\omega'\|} \]

Using inf. result:
\[ \inf_{\omega \in S} \|\omega - \omega'\|_E \leq C \|\omega - \omega'\|_E \leq C \|\omega\| \]
\[ \Rightarrow \|\mathbf{u} - \mathbf{u}_S\| \leq C \|\mathbf{u} - \mathbf{u}_S\|_E \]

Using inf. result again:
\[ \|\mathbf{u} - \mathbf{u}_S\| \leq C \|\mathbf{u}_S\| \leq C \|\mathbf{u}_S\|_E \]
Consider the problem:
\[
\begin{cases}
- \Delta u = f \text{ in } \Omega \\
\quad u = 0 \text{ on } \partial \Omega.
\end{cases}
\]

\(\partial \Omega = \text{boundary of } \Omega\)

\(n(x) = \text{outward unit normal to } \partial \Omega \text{ at } x\)

To find (WF) multiply PDE on both sides by some test function

\(\forall \sigma \in V = \left\{ \sigma \mid \int_\Omega \sigma^2 \, dx < \infty, \int_\Omega |\nabla \sigma|^2 \, dx < \infty, \sigma|_{\partial \Omega} = 0 \right\}\)

and integrate:

\[-\int_\Omega \Delta u \sigma \, dx = \int_\Omega f \sigma \, dx\]

Then use analogues of IBP, which is Green's identity:

\[
\int_\Omega -\Delta u \, \sigma \, dx = \int_\Omega \nabla u \cdot \nabla \sigma \, dx - \int_{\partial \Omega} \sigma (\Delta u) \nu \sigma \, ds
\]

\(= 0 \text{ since } \nabla u = 0\).

Thus we get (WF):

\[
\begin{align*}
\text{Find } & u \in V \text{ s.t.} \\
& a(u, v) = (f, v) \quad \forall v \in V
\end{align*}
\]

where:

\[
a(u, v) = \int_\Omega \nabla u \cdot \nabla v \, dx
\]

\[
(f, v) = \int_\Omega f \, v \, dx
\]
(WF) needs to be adjusted if B.C. are different.

Implementation of 2D P1 elements

First we need to partition Ω into elements, usually triangle or quadrilatera. Here we focus on triangle.

**Triangulation**

Given coordinates of points in 2D (and even n-dim) it is possible to find a triangulation that maximizes the minimal angle of triangle.

⇒ **Delaunay** triangulation.

Triangulation is exact i.e. \( \bigcup_{e} \Omega_e = \Omega \), when \( \Omega = \text{convex polygon} \).

If it is not, then we are introducing an error.

There is a well-known open source library for computing triangulations called **DHULL**, which is the engine behind Matlab's delaunay function.

\[
\text{tri} = \text{delaunay}(x,y); \quad x = \text{vec of x-coordinate}.
\]
\[
y = \overbrace{\text{vec of y-coordinate}}^{\text{vec of y}}
\]
\[
\text{tri} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 1 \end{bmatrix}
\]

\[
\text{tri}(e,:) = \text{index of vertices of triangle e.}
\]
Since error estimates for 2DFEM degrade when triangles are close to degenerate (when they have a small angle) it is possible to adjust position of nodes (or add/remove) to improve min angle. \( \rightarrow \) many mesh generators

- **triangle** - J. R. Siewchuck (industrial strength)
- **distmesh** - Strang and Pearson (simple to use and understand)

Since here we are using \( P_1 \) elements d.o.f. are values at vertices.

![Triangular Element](image)

So the local-to-global map is given automatically by the tri array from Delaunay triangulation:

\[
\text{tri}(e,j) = \text{global idx of } j\text{-th vertex of triangle } e.
\]

**Basis functions of \( S \):**

\[
\phi_j(x) = \text{polynomial linear on triangulation and s.t. }
\]

\[
\phi_j(x_i) = \delta_{ij}.
\]

\[
\rightarrow \text{pyramid shape.}
\]
P1 elements are an affine family of elements:

\[ F_k \]

\[ (0,1) = (\xi, \eta) \]

\[ (1,0) = (\xi_2, \eta_2) \]

\[ (0,0) = (\xi_1, \eta_1) \]

\[ (x_3, y_3) \]

\[ (x_2, y_2) \]

\[ (x_i, y_i) \]

\[ F_k (\xi, \eta) = B \begin{bmatrix} \xi \\ \eta \end{bmatrix} + b \]

\[ = \begin{bmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{bmatrix} \begin{bmatrix} \xi \\ \eta \end{bmatrix} + \begin{bmatrix} x_i \\ y_i \end{bmatrix} \]

Easy to check that:

\[ F_k (\xi_i, \eta_i) = (x_i, y_i) \quad i = 1, 2, 3. \]

Also basis functions restricted to element \( K \) can be expressed in terms of basis functions at parent element by means of \( F_k \) map:

\[ \phi_i (\xi, \eta) = \delta_{ij} \]

\[ \begin{align*}
\phi_1 (\xi, \eta) &= 1 - \xi - \eta \\
\phi_2 (\xi, \eta) &= \xi \\
\phi_3 (\xi, \eta) &= \eta
\end{align*} \]
Local basis function of element $K$ is then:

$$
\phi_i^K(x, y) = \phi_i(F_k^{-1}(x, y)) = (\phi_i \circ F_k^{-1})(x, y)
$$

**Stiffness matrix**

As in 10 we assemble the (global) stiffness matrix by adding the contribution of each element:

for $e = 1, \ldots, \# \text{ of triangles}$

$$
\mathbf{A}((\mathbf{t}_i, e_i), (\mathbf{t}_j, e_j)) = \mathbf{A}(\mathbf{t}_i, e_i), \mathbf{t}_j(e_j)) + \mathbf{A}_{\text{loc}}^{K_e} \mathbf{K}_e
$$

where local stiffness matrix is:

$$
(\mathbf{A}_{\text{loc}}^{K_e})_{ij} = a_{K_e}(\phi_i^{K_e}, \phi_j^{K_e}) = \sum_{K_e} \int_{K_e} \nabla \phi_i^{K_e} \cdot \nabla \phi_j^{K_e} \, dx
$$

for $i, j = 1, 2, 3$.

The above loop computes the right thing because:

$$
A_{ij} = a(\phi_i, \phi_j) = \sum_{e} \int_{K_e} \nabla \phi_i \cdot \nabla \phi_j \, dx = \sum_{e} \sum_{K_e} \nabla \phi_i^{K_e} \cdot \nabla \phi_j^{K_e} \, dx
$$

$$
= \sum_{K} (\phi_i^{K}, \phi_j^{K}) \text{ Kelement.}
$$
So that for \( n \in \mathcal{E} \):

\[
\alpha_k (\tau, \omega) = \alpha_k \left( \sum_{i=1}^{3} \omega \phi_i \phi_i^k, \sum_{j=1}^{3} \mu \phi_j \phi_j^k \right)
\]

\[
= \sum_{i,j=1}^{3} \omega \phi_i \phi_j \mu \phi_j^k \phi_i^k
\]

\[
= \sum_{i,j=1}^{3} \omega \phi_i \phi_j \mu \phi_j^k \phi_i^k
\]

Local stiffness matrix calculation

Idea: change of variables and chain rule

Recall:

- Change of variables (c.o.v.)

\[
\int_\Omega f(\xi) \, d\xi = \int_{\hat{\Omega}} f(F^{-1}(\xi)) \left| \det D F^{-1} \right| \, d\hat{\xi}
\]

(check with dilation)

- Chain rule (for gradients)

\[
\nabla (f \circ F)(\xi) = D F^T [\nabla \xi] (D f) \circ F(\xi).
\]

Thus:

\[
(A^K_{\text{loc}})_{ij} = \int_K \nabla \phi_i^k(x, y) \cdot \nabla \phi_j^k(x, y) \, d\xi d\eta
\]

\[
= \int_K \nabla (\phi_i \circ F_k^{-1})(x, y) \cdot \nabla (\phi_j \circ F_k^{-1})(x, y) \, d\xi d\eta
\]

Chain rule:

\[
= \int_K \left[ D F_k^{-T} \nabla \phi_i (F_k^{-1}(x, y)) \right]^T D F_k^{-T} \nabla \phi_j (\cdots) \, d\xi d\eta
\]

C.O.V.:

\[
= \int_K \nabla \phi_i (\xi, \eta)^T (D F_k^{-1} D F_k^{-T}) \nabla \phi_j (\xi, \eta) \, d\xi d\eta
\]

\[
\left| \det D F_k \right| \, d\xi d\eta
\]
(Floc)j = \sum_k (\sum_{i,j} \phi_k(x,y) \phi_j(x,y) \delta x \delta y) = \sum_k \delta x \delta y \int \int F(x',y') \phi_k(x') \phi_j(x') \delta x \delta y

\text{where}

F_{e} = \int \int F(x',y') \delta x \delta y

for e = 1, \ldots, \# triangles.

\text{Assume for simplicity that } \phi \text{ is } S \text{. Then, move} \text{ all RHS terms to the left side and integrate over each element.}

\text{RHS can be considered by doing more elaborate numerical integration.}

The right hand side can be assembled in a similar way.

\text{Notice:}

\text{new } G = \left[ \phi_1 \phi_2 \phi_3 \right] = \left[ \begin{array}{c} 1 \end{array} \right]
Thus,

\[ \mathbf{F}_{loc} = M_{loc}^{K} \left[ f(x_{i}(k,s), y_{i}(k,s)) \right] \]

where \( (M_{loc}^{K})_{ij} = \int_{K} \phi_{i}^{K}(x, y) \phi_{j}^{K}(x, y) \, dx \, dy \)

These integrals can be evaluated early on the ref element by c.o.v.

\[ (M_{loc}^{K})_{ij} = \int_{K} \phi_{i}^{K}(x, y) \phi_{j}^{K}(x, y) \left| \det \mathbf{D} \right| \, dx \, dy \]

\[ = \frac{1}{24} \left| \det \mathbf{D} \right| \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \]