2D Heat Conduction: Computer Implementation:

Now that we have discussed various finite element approximations for steady two-dimensional finite element approximations, we turn to a discussion of efficient computer implementation. Just as in the one-dimensional setting, we turn to element formation and assembly. However, as opposed to the one-dimensional setting, we also form and assemble boundary element stiffness matrices and force vectors for each of the boundary edges in our finite element mesh.

For all of the finite element approximations discussed over the past several lectures, the degrees of freedom of the Galerkin solution:

$$T^h(\vec{x}) = \sum_{A \in \eta - \eta_D} T^h(\vec{x}_A) N_A(\vec{x}) + \sum_{A \in \eta_D} g(\vec{x}_A) N_A(\vec{x})$$

$T^h(\vec{x}_A)$: Degree of Freedom

$g^h(\vec{x})$

may be attained via the solution of a linear system:

$$\underline{\underline{K}}\,\underline{d} = \underline{F}$$

where:

$$K_{PQ} = b^h(N_B, N_A) \qquad \text{where} \quad ID(A) = P,\ ID(B) = Q$$

$$F_P = \ell^h(N_A) - b^h(g^h, N_A) \qquad \text{where} \quad ID(A) = P$$

$$d_Q = T^h(\vec{x}_B) \qquad \text{where} \quad ID(B) = Q$$

To construct the stiffness matrix $\underline{\underline{K}}$ and $\underline{F}$, we decompose each into sums of integrals over interior elements and boundary edges:

$$K_{AB} = \sum_{\Omega_h^e \in \mu^h} \int_{\Omega_h^e} K \vec{\nabla} N_A \cdot \vec{\nabla} N_B \, d\Omega_h + \sum_{\Gamma_h^e \in \mathcal{E}_R^h} \int_{\Gamma_h^e} \beta \, N_A \, N_B \, d\Gamma$$

$$F_A = \sum_{\Omega_h^e \in \mu^h} \int_{\Omega_h^e} f \, N_A \, d\Omega_h - \sum_{\Omega_h^e \in \mu^h} \int_{\Omega_h^e} K \vec{\nabla} g^h \cdot \vec{\nabla} N_A \, d\Omega$$

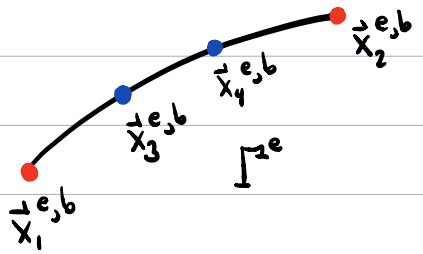
$$+ \sum_{\Gamma_h^e \in \mathcal{E}_N^h} \int_{\Gamma_h^e} h \, N_A \, d\Gamma + \sum_{\Gamma_h^e \in \mathcal{E}_R^h} \int_{\Gamma_h^e} \beta \, T_R \, N_A \, d\Gamma - \sum_{\Gamma_h^e \in \mathcal{E}_R^h} \int_{\Gamma_h^e} \beta \, g^h \, N_A \, d\Gamma$$

As discussed in the last several lectures, we have that:

$$N_A(\vec{x}) = \begin{cases} (\hat{N}_a \circ \vec{\xi}^e)(\vec{x}) & \text{if there is an } a \text{ such that } A = ID(a,e) \\ 0 & \text{otherwise} \end{cases}$$

for each interior element Ω_h^e where $\vec{\xi}^e : \Omega_h^e \to \hat{\Omega}$ is the inverse of the element map $\vec{x}^e : \hat{\Omega} \to \Omega_h^e$. In similar fashion, we can write basis functions over each boundary edge in terms of boundary shape functions, a boundary element connectivity, and a boundary element map. Let $\Gamma_h^e \in \mathcal{E}_{\Gamma}^h$ be a boundary edge:

$k = 3$



● = Vertex Node

● = Edge Node

There are two vertex nodes and k-1 edge nodes attached to the edge. We locally renumber these nodes as indicated above according to the following rules:

Rule #1: The first two nodes $\{\vec{x}_a^{e,b}\}_{a=1}^{2}$ are the vertex nodes ordered such that the domain is to our left as we walk from $\vec{x}_1^{e,b}$ to $\vec{x}_2^{e,b}$.

Rule #2: The remaining nodes $\{\vec{x}_a^{e,b}\}_{a=3}^{nenb}$ are the edge nodes ordered in increasing fashion as we walk from $\vec{x}_1^{e,b}$ to $\vec{x}_2^{e,b}$.

Note that $nenb$ is the number of local nodes. We relate the global node numbering to this local node numbering using a boundary element connectivity:

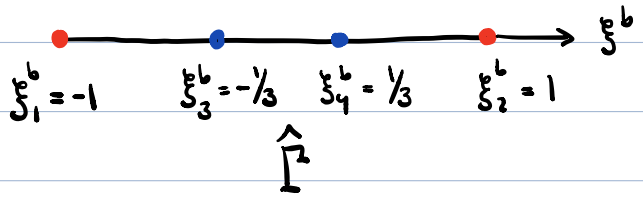
$$A = IENB(a, e)$$

- A: Global Node Number
- a: Local Node Number
- e: Boundary Edge Number

Now, let $\hat{\Gamma}^e := (-1, 1)$, and let $\{\xi_a^b\}_{a=1}^{nenb}$ be $nenb$ equispaced nodes

over $\hat{\Gamma}^2$ ordered such that $\xi_1^b = -1$, $\xi_2^b = 1$, and the rest of the nodes are non-decreasing:

$k=3$



We call $\hat{\Gamma}^2$ the parent or reference boundary element. Now let $\{\hat{N}_a^b\}_{a=1}^{n_{enb}}$ be the boundary shape functions satisfying:

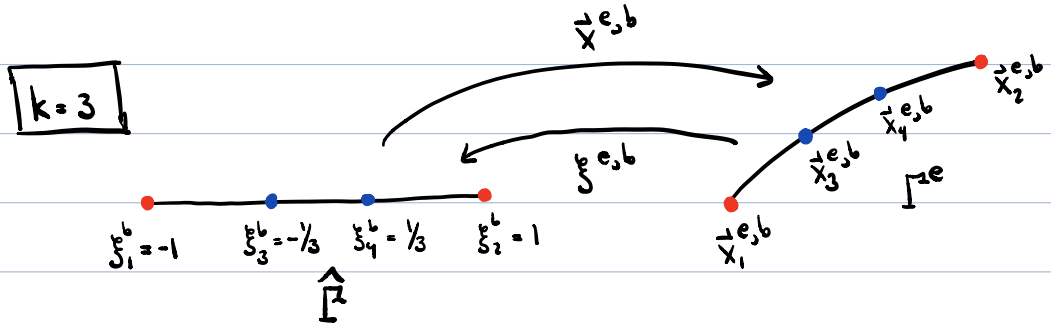
$$\hat{N}_a^b(\xi_b^b) = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{if } a \neq b \end{cases}$$

These are precisely the one-dimensional shape functions considered for the earlier one-dimensional model problem. Then we can define a boundary element map $\vec{x}^{e,b} : \hat{\Gamma}^2 \to \Gamma^e$:

$$\vec{x}^{e,b}(\xi^b) = \sum_{a=1}^{n_{enb}} \vec{x}_a^{e,b} \hat{N}_a^b(\xi^b)$$

With inverse $\xi^{e,b} : \Gamma^e \to \hat{\Gamma}^2$:

$k=3$



Moreover, it holds that:

$$N_A(\vec{x}) = \begin{cases} (\hat{N}_a^b \circ \xi^{e,b})(\vec{x}) & \text{if there is an } a \text{ such that } A = \mathrm{IENB}(a,e) \\ 0 & \text{otherwise} \end{cases}$$

Thus, as earlier claimed, we can write basis functions over each boundary edge in terms of boundary shape functions, a boundary element connectivity, and a boundary element map.

Now, for each interior element Ω^e, define:

$$N_a^e(\vec{x}) = (\hat{N}_a \circ \vec{\xi}^e)(\vec{x}) \qquad \text{for } a = 1, \ldots, n_{en}$$

and for each boundary edge Γ^e, define:

$$N_a^{e,b}(\vec{x}) = (\hat{N}_a^b \circ \xi^{e,b})(\vec{x}) \qquad \text{for } a = 1, \ldots, n_{enb}$$

We can then define for each interior element Ω^e an element stiffness matrix:

$$k_{ab}^e := \int_{\Omega^e} K \, \vec{\nabla} N_a^e \cdot \vec{\nabla} N_b^e \, d\Omega \qquad \text{for } a,b = 1, \ldots, n_{en}$$

and an element force vector:

$$f_a^e := \int_{\Omega^e} f \, N_a^e \qquad \text{for } a = 1, \ldots, n_{en}$$

and we can define for each boundary edge Γ^e a boundary element stiffness matrix:

$$k_{ab}^{e,b} := \int_{\Gamma^e \cap \Gamma_R^h} \beta N_a^{e,b} N_b^{e,b} \, d\Gamma \qquad \text{for } a,b = 1,\dots,n_{enb}$$

and a boundary element force vector:

$$f_a^{e,b} := \int_{\Gamma^e \cap \Gamma_N^h} h N_a^{e,b} \, d\Gamma + \int_{\Gamma^e \cap \Gamma_R^h} \beta T_R N_a^{e,b} \, d\Gamma \qquad \text{for } a = 1,\dots,n_{enb}$$

Moreover, we can assemble the global stiffness matrix and force vector from these local quantities:

$$K_{PQ} = \sum_{e=1}^{n_{el}} \sum_{a=1}^{n_{en}} \sum_{b=1}^{n_{en}} k_{ab}^{e} \qquad \text{Element Assembly}$$

$$P = ID(IEN(a,e)) \quad Q = ID(IEN(b,e))$$

$$+ \sum_{e=1}^{n_{elb}} \sum_{a=1}^{n_{enb}} \sum_{b=1}^{n_{enb}} k_{ab}^{e,b} \qquad \text{Boundary Element Assembly}$$

$$P = ID(IENB(a,e)) \quad Q = ID(IENB(b,e))$$

and:

$$F_p = \sum_{e=1}^{n_{el}} \sum_{a=1}^{n_{en}} \left(f_a^e - \sum_{b=1}^{n_{en}} k_{ab}^e \, g(\vec{x}_b^e) \right)$$

Element Assembly

$P = ID(IEN(a,e))$ $\quad ID(IEN(b,e)) = 0$

$$+ \sum_{e=1}^{n_{el}} \sum_{a=1}^{n_{enb}} \left(f_a^{e,b} - \sum_{b=1}^{n_{enb}} k_{ab}^{e,b} \, g(\vec{x}_b^{e,b}) \right)$$

Boundary Element Assembly

$P = ID(IENB(a,e))$ $\quad ID(IENB(b,e)) = 0$

In pseudo code:

Assembly Pseudo code

Set $\underline{\underline{K}} = \underline{\underline{0}}$ and $\underline{F} = \underline{0}$

```
for e = 1 : nel
    for a = 1 : nen
        Set P = ID(IEN(a,e))
        if P ≠ 0
            for b = 1 : nen
                Set Q = ID(IEN(b,e))
                if Q ≠ 0
                    K_PQ = K_PQ + k^e_ab
                else
```

$F_p = F_p - k_{ab}^{e} g(\vec{x}_b^{e})$

endif

endfor

$F_p = F_p + f_a^{e}$

endif

endfor

endfor

for $e = 1 : n_{elb}$

for $a = 1 : n_{enb}$

Set $P = ID(IENB(a,e))$

if $P \neq 0$

for $b = 1 : n_{enb}$

Set $Q = ID(IENB(b,e))$

if $Q \neq 0$

$K_{PQ} = K_{PQ} + k_{ab}^{e,b}$

else

$F_p = F_p - k_{ab}^{e,b} g(\vec{x}_b^{e,b})$

endif

endfor

endif

$F_p = F_p - f_a^{e,b}$

endfor

endfor

To compute the entries of the element stiffness matrices and force vectors and boundary element stiffness matrices and force vectors, we employ the same two tricks as we did in the one-dimensional setting:

Trick #1: We pull integrals back to the parent element.

Trick #2: We use quadrature to approximate parent element integrals.

Pulling back element integrals appearing in the element stiffness matrices and force vectors yields the expressions:

$$k_{ab}^e = \int_{\hat{\Omega}_h} K(\vec{x}^e(\vec{\xi})) \; \vec{\nabla} N_a^e(\vec{x}^e(\vec{\xi})) \cdot \vec{\nabla} N_b^e(\vec{x}^e(\vec{\xi})) \, j^e(\vec{\xi}) \, d\hat{\Omega}$$

$$f_a^e = \int_{\hat{\Omega}} f(\vec{x}^e(\vec{\xi})) \; N_a^e(\vec{x}^e(\vec{\xi})) \, j^e(\vec{\xi}) \, d\hat{\Omega}$$

where $j^e : \hat{\Omega}_h \to \mathbb{R}$ is the Jacobian determinant:

$$j^e(\vec{\xi}) := \det\left(\underline{\underline{J}}^e(\vec{\xi})\right)$$

where:

$$\underline{\underline{J}}^e(\vec{\xi}) := \begin{bmatrix} \frac{\partial x_1^e}{\partial \xi_1}(\vec{\xi}) & \frac{\partial x_1^e}{\partial \xi_2}(\vec{\xi}) \\ \frac{\partial x_2^e}{\partial \xi_1}(\vec{\xi}) & \frac{\partial x_2^e}{\partial \xi_2}(\vec{\xi}) \end{bmatrix}$$

is the Jacobian. Note that:

$$N_a^e\left(\vec{x}^e\left(\vec{\xi}\right)\right) = \hat{N}_a\left(\vec{\xi}\right)$$

and by the chain rule:

$$\frac{\partial N_a^e}{\partial x_1}\left(\vec{x}^e\left(\vec{\xi}\right)\right) = \frac{\partial \hat{N}_a}{\partial \xi_1}\left(\vec{\xi}\right)\frac{\partial \xi_1^e}{\partial x_1}\left(\vec{x}^e\left(\vec{\xi}\right)\right) + \frac{\partial \hat{N}_a}{\partial \xi_2}\left(\vec{\xi}\right)\frac{\partial \xi_2^e}{\partial x_1}\left(\vec{x}^e\left(\vec{\xi}\right)\right)$$

$$\frac{\partial N_a^e}{\partial x_2}\left(\vec{x}^e\left(\vec{\xi}\right)\right) = \frac{\partial \hat{N}_a}{\partial \xi_1}\left(\vec{\xi}\right)\frac{\partial \xi_1^e}{\partial x_2}\left(\vec{x}^e\left(\vec{\xi}\right)\right) + \frac{\partial \hat{N}_a}{\partial \xi_2}\left(\vec{\xi}\right)\frac{\partial \xi_2^e}{\partial x_2}\left(\vec{x}^e\left(\vec{\xi}\right)\right)$$

where, by the inverse function theorem:

$$\begin{bmatrix} \frac{\partial \xi_1^e}{\partial x_1}\left(\vec{x}^e\left(\vec{\xi}\right)\right) & \frac{\partial \xi_1^e}{\partial x_2}\left(\vec{x}^e\left(\vec{\xi}\right)\right) \\ \frac{\partial \xi_2^e}{\partial x_1}\left(\vec{x}^e\left(\vec{\xi}\right)\right) & \frac{\partial \xi_2^e}{\partial x_2}\left(\vec{x}^e\left(\vec{\xi}\right)\right) \end{bmatrix} = \left(\underline{\underline{J}}^e\left(\vec{\xi}\right)\right)^{-1}$$

Thus the above integrals can be expressed entirely in terms of shape functions over the parent element and the element mapping and its Jacobian. To numerically compute the integrals, we turn to a quadrature rule over the parent element with points $\{\hat{\xi}_\ell\}_{\ell=1}^{n_q}$ and weights $\{\hat{w}_\ell\}_{\ell=1}^{n_q}$:

$$k_{ab}^e \approx \sum_{\ell=1}^{n_q} K\left(\vec{x}^e\left(\hat{\xi}_\ell\right)\right) \vec{\nabla} N_a^e\left(\vec{x}^e\left(\hat{\xi}_\ell\right)\right) \cdot \vec{\nabla} N_b^e\left(\vec{x}^e\left(\hat{\xi}_\ell\right)\right) j^e\left(\hat{\xi}_\ell\right) \hat{w}_\ell$$

$$f_a^e \approx \sum_{\ell=1}^{n_q} f\left(\vec{x}^e\left(\hat{\xi}_\ell\right)\right) N_a^e\left(\vec{x}^e\left(\hat{\xi}_\ell\right)\right) j\left(\hat{\xi}_\ell\right) \hat{w}_\ell$$

In pseudocode:

Element Formation Pseudocode

Set $\underline{\underline{k}}^e = \underline{\underline{0}}$ and $\underline{f}^e = \underline{0}$

```
for ℓ = 1 : n_q
    for a = 1 : n_en
        for b = 1 : n_en
            k^e_ab = k^e_ab + K(x⃗^e(ξ̂_ℓ)) ∇⃗N^e_a(x⃗^e(ξ̂_ℓ)) ·
                              ∇⃗N^e_b(x⃗^e(ξ̂_ℓ)) j^e(ξ̂_ℓ) ŵ_ℓ
        endfor
        f^e_a = f^e_a + f(x⃗^e(ξ̂_ℓ)) N^e_a(x⃗^e(ξ̂_ℓ)) j(ξ̂_ℓ) ŵ_ℓ
    endfor
endfor
```

Pulling back boundary element integrals appearing in the boundary element stiffness matrices and force vectors yields the expressions:

$$k_{ab}^{e,b} = \begin{cases} \displaystyle\int_{\hat{\Gamma}^2} \mathcal{B}\left(\vec{x}^{e,b}(\xi^b)\right) N_a^{e,b}\left(\vec{x}^{e,b}(\xi^b)\right) N_b^{e,b}\left(\vec{x}^{e,b}(\xi^b)\right) j_{\Gamma^2}^e(\xi^b)\, d\hat{\Gamma}^2 \\ \qquad \text{if } \Gamma^{2\,e} \in \mathcal{E}_R^h, \\ 0 \\ \qquad \text{otherwise} \end{cases}$$

and:

$$f_a^{e,b} = \begin{cases} \displaystyle\int_{\hat{\Gamma}^2} h\left(\vec{x}^{e,b}\left(\xi^b\right)\right) N_a^{e,b}\left(\vec{x}^{e,b}\left(\xi^b\right)\right) j_{\Gamma^2}^e\left(\xi^b\right) d\hat{\Gamma}^2 \\ \qquad \text{if } \Gamma^{2e} \in \mathcal{E}_N^h, \\ \displaystyle\int_{\hat{\Gamma}^2} \beta\left(\vec{x}^{e,b}\left(\xi^b\right)\right) T_R\left(\vec{x}^{e,b}\left(\xi^b\right)\right) N_a^{e,b}\left(\vec{x}^{e,b}\left(\xi^b\right)\right) j_{\Gamma^2}^e\left(\xi^b\right) d\hat{\Gamma}^2 \\ \qquad \text{if } \Gamma^{2e} \in \mathcal{E}_R^h, \\ 0 \\ \qquad \text{otherwise} \end{cases}$$

where $j_{\Gamma^2}^e : \hat{\Gamma}^2 \longrightarrow \mathbb{R}$ is the surface determinant:

$$j_{\Gamma^2}^e\left(\xi^b\right) := \left| \frac{\partial \vec{x}^{e,b}}{\partial \xi^b}\left(\xi^b\right) \right|$$

Note that:

$$N_a^{e,b}\left(\vec{x}^{e,b}\left(\xi^b\right)\right) = \hat{N}_a^b\left(\xi^b\right)$$

So the above integrals can be expressed entirely in terms of boundary shape functions, the boundary element mapping, and the surface determinant. To numerically compute the integrals, we turn again to quadrature, this time using quadrature points $\left\{ \xi_l^b \right\}_{l=1}^{n_q^b}$ and $\left\{ w_l^b \right\}_{l=1}^{n_q^b}$ defined over the boundary parent element. In pseudocode:

Boundary Element Formation Pseudo code

Set $\underline{\underline{k}}^{e,b} = \underline{\underline{0}}$ and $\underline{f}^{e,b} = \underline{0}$

for $l = 1 : n_{qb}$

if $\Gamma^e \in \mathcal{E}_N^h$

for $a = 1 : n_{enb}$

$$f_a^{e,b} = f_a^{e,b} + h\left(\vec{x}^{e,b}\left(\hat{\xi}_l^b\right)\right) N_a^{e,b}\left(\vec{x}^{e,b}\left(\hat{\xi}_l^b\right)\right) j_{\Gamma^e}^e\left(\hat{\xi}_l^b\right) \hat{w}_l$$

endfor

else if $\Gamma^e \in \mathcal{E}_R^h$

for $a = 1 : n_{enb}$

for $b = 1 : n_{enb}$

$$k_{ab}^{e,b} = k_{ab}^{e,b} + \beta\left(\vec{x}^{e,b}\left(\hat{\xi}_l^b\right)\right) N_a^{e,b}\left(\vec{x}^{e,b}\left(\hat{\xi}_l^b\right)\right) N_b^{e,b}\left(\vec{x}^{e,b}\left(\hat{\xi}_l^b\right)\right) j_{\Gamma^e}^e\left(\hat{\xi}_l^b\right) \hat{w}_l$$

endfor

$$f_a^{e,b} = f_a^{e,b} + \beta\left(\vec{x}^{e,b}\left(\hat{\xi}_l^b\right)\right) T_R\left(\vec{x}^{e,b}\left(\hat{\xi}_l^b\right)\right) N_a^{e,b}\left(\vec{x}^{e,b}\left(\hat{\xi}_l^b\right)\right) j_{\Gamma^e}^e\left(\hat{\xi}_l^b\right) \hat{w}_l$$

endfor

endif

endfor

It remains finally to specify the quadrature points and weights for both the parent element and the parent boundary element. For the parent

boundary element, one can choose Gaussian quadrature points and weights:

Number of Points, n_{qb}	Quadrature Points, $\hat{\xi}^b_l$	Quadrature Weights, $\hat{w}^b_l$
1	0	2
2	$\pm \frac{1}{\sqrt{3}}$	1
3	0	$\frac{8}{9}$
	$\pm \sqrt{\frac{3}{5}}$	$\frac{5}{9}$
4	$\pm \sqrt{\frac{3}{7} - \frac{2}{7}\sqrt{\frac{6}{5}}}$	$\frac{18+\sqrt{30}}{36}$
	$\pm \sqrt{\frac{3}{7} + \frac{2}{7}\sqrt{\frac{6}{5}}}$	$\frac{18-\sqrt{30}}{36}$

For a degree k finite element approximation, it usually suffices to choose $n_{qb} = k+1$. If the parent element is the biunit square, one can tensor product the parent boundary element quadrature rule:

$$\hat{\xi}_l = \left(\hat{\xi}^b_{l_1}, \hat{\xi}^b_{l_2}\right)$$

$$\hat{w}_l = \hat{w}^b_{l_1} * \hat{w}^b_{l_2}$$

$$l = (l_2 - 1) * n_{qb} + l_1$$

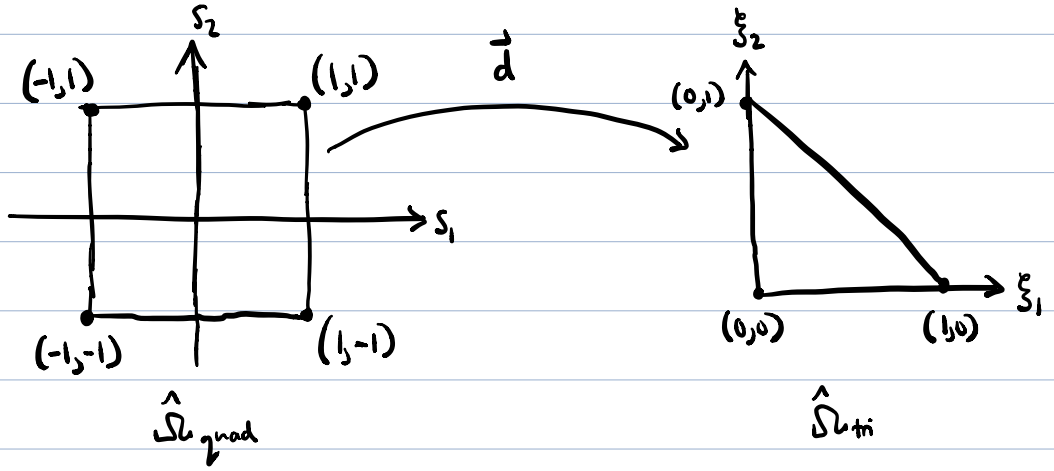
for $l_1, l_2 = 1, \ldots, n_{qb}$, yielding $n_q = n_{qb} * n_{qb}$ quadrature points and weights over the parent element. This quadrature rule follows from

the iterated approximation:

$$\int_{-1}^{1}\int_{-1}^{1} \hat{f}(\xi_1,\xi_2)\,d\xi_1 d\xi_2 \approx \int_{-1}^{1}\left(\sum_{l_1=1}^{nqb} \hat{f}(\overset{\sim}{\xi}^{b}_{l_1},\xi_2)\,\tilde{w}^{b}_{l_1}\right) d\xi_2$$

$$\approx \sum_{l_2=1}^{nqb}\sum_{l_1=1}^{nqb} \hat{f}(\overset{\sim}{\xi}^{b}_{l_1},\overset{\sim}{\xi}^{b}_{l_2})\,\tilde{w}^{b}_{l_1}\,\tilde{w}^{b}_{l_2}$$

If the parent element is the unit right triangle, building a suitable quadrature rule is not as straight-forward. One option is to map every point in the biunit square to the unit right triangle using the so-called Duffy transform:



where:

$$\vec{d}(\vec{s}) = \begin{bmatrix} \frac{1}{2}(1+s_1) \\ \frac{1}{4}(1-s_1)(1+s_2) \end{bmatrix}$$

Then we can pull back any integral over $\hat{\Omega}_{tri}$ to an integral over $\hat{\Omega}_{quad}$:

$$\int_{\hat{\Omega}_{h,tri}} \hat{f}\, d\hat{\Omega} = \int_{\hat{\Omega}_{quad}} \hat{f}(\vec{d}(\vec{s}))\, j^d(\vec{s})\, d\hat{\Omega}$$

where:

$$j^d(\vec{s}) = \det \begin{bmatrix} \frac{\partial d_1}{\partial s_1}(\vec{s}) & \frac{\partial d_1}{\partial s_2}(\vec{s}) \\ \frac{\partial d_2}{\partial s_1}(\vec{s}) & \frac{\partial d_2}{\partial s_2}(\vec{s}) \end{bmatrix}$$

We can then approximate the integral over $\hat{\Omega}_{quad}$ using tensor product quadrature, leading to:

$$\int_{\hat{\Omega}_{h,tri}} \hat{f}\, d\hat{\Omega} \approx \sum_{l_2=1}^{n_{qb}} \sum_{l_1=1}^{n_{qb}} \hat{f}\left(\vec{d}\left(\tilde{\xi}^{qb}_{l_1}, \tilde{\xi}^{qb}_{l_2}\right)\right) \tilde{w}^{qb}_{l_1} \tilde{w}^{qb}_{l_2}\, j^d\left(\tilde{\xi}^{qb}_{l_1}, \tilde{\xi}^{qb}_{l_2}\right)$$

or equivalently:

$$\int_{\hat{\Omega}_{h,tri}} \hat{f}\, d\hat{\Omega} \approx \sum_{l=1}^{n_q} \hat{f}\left(\tilde{\xi}_l\right) \tilde{w}_l$$

where:

$$\tilde{\xi}_l = \vec{d}\left(\tilde{\xi}^{qb}_{l_1}, \tilde{\xi}^{qb}_{l_2}\right)$$

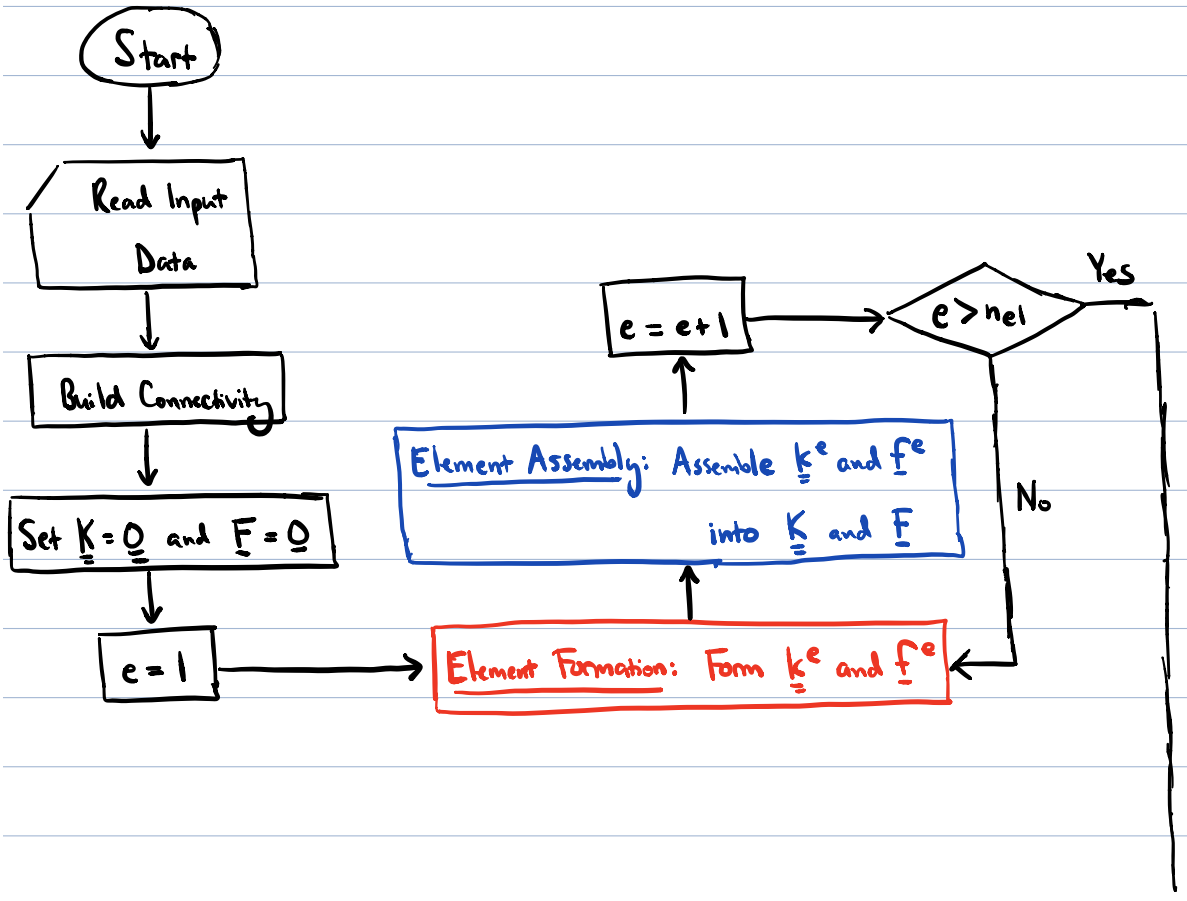
$$\tilde{w}_l = \tilde{w}^{qb}_{l_1} * \tilde{w}^{qb}_{l_2} * j^d\left(\tilde{\xi}^{qb}_{l_1}, \tilde{\xi}^{qb}_{l_2}\right)$$

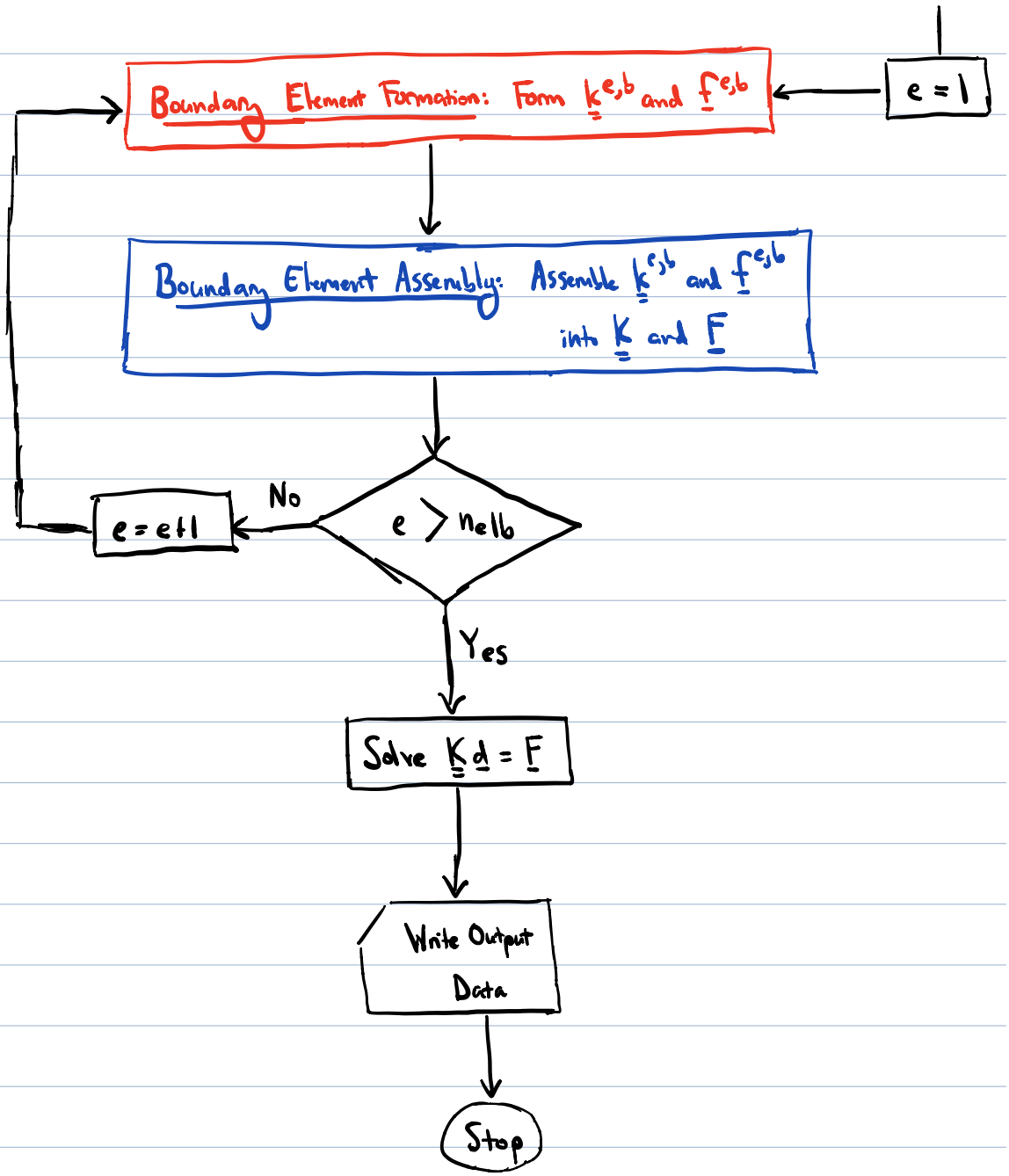
$$l = (l_2 - 1) * n_{qb} + l_1$$

Unfortunately, this results in a larger number of quadrature points than

required for both stability and accuracy. In practice, one often employs quadrature schemes specifically tailored for numerical integration over triangles (and the finite element code provided to the class employs such a scheme). See, for instance, Cowper's classical paper "Gaussian quadrature formulas for triangles" or Dunavant's paper "High-degree efficient symmetrical Gaussian quadrature rules for the triangle".

With all the above in hand, a finite element code for steady two-dimensional heat conduction has the following control flow:




e = 1
Boundary Element Formation: Form $\underline{\underline{k}}^{e,b}$ and $\underline{f}^{e,b}$
Boundary Element Assembly: Assemble $\underline{\underline{k}}^{e,b}$ and $\underline{f}^{e,b}$ into $\underline{\underline{K}}$ and $\underline{F}$
$e > n_{elb}$
No
e = e+1
Yes
Solve $\underline{\underline{K}}\,\underline{d} = \underline{F}$
Write Output Data
Stop