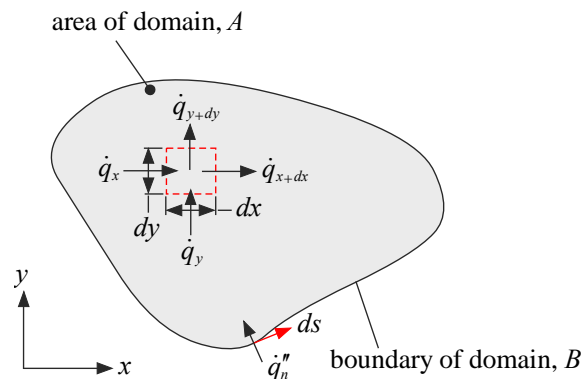


## 2.7.2 The Galerkin Weighted Residual Method

### Introduction

Sections 2.5 and 2.6 discuss the numerical solution to steady-state conduction problems using the finite difference technique. The theory behind finite difference solutions is intuitive and these solutions are easy to understand and straightforward to implement. However, the finite difference technique is most appropriate for problems where the computational domain is regularly shaped and the nodes are positioned in a regular pattern. The finite element technique is an alternative numerical method. The finite element technique is much less intuitive than the finite difference technique to most engineers and the underlying theory is more involved. However, once the theory has been developed the finite element technique can be applied as easily to irregular geometries with an arbitrary arrangement of nodes as it can to a regular geometry with a regular pattern of nodes. There are several books available that discuss the development and application of the finite element method in depth. Interested readers are referred to Myers (1987) and Moaveni (2003). In this section, the finite element solution technique is presented at a level that is sufficient to understand the underlying theory and allow the technique to be applied to relatively simple problems. The derivation follows from the explanation provided by Myers (1987) and the solution methodology described here has been used to develop the FEHT software that is presented in Section 2.7.1 and Appendix A.4.

Figure E6-1 illustrates a two-dimensional region that is defined by an irregular boundary. A heat transfer problem involving this type of geometry can most easily be addressed using the finite element technique.



**Figure E6-1: Two-dimensional region.**

The governing differential equation is derived using an energy balance on a differential control volume, shown in Figure E6-1. Assuming that the problem is steady state and there is no generation of thermal energy, the energy balance is:

$$\dot{q}_x + \dot{q}_y = \dot{q}_{x+dx} + \dot{q}_{y+dy} \quad (\text{E6-1})$$

The  $x+dx$  and  $y+dy$  terms are expanded as usual:

$$\dot{q}_x + \dot{q}_y = \dot{q}_x + \frac{\partial \dot{q}_x}{\partial x} dx + \dot{q}_y + \frac{\partial \dot{q}_y}{\partial y} dy \quad (\text{E6-2})$$

and simplified:

$$\frac{\partial \dot{q}_x}{\partial x} dx + \frac{\partial \dot{q}_y}{\partial y} dy = 0 \quad (\text{E6-3})$$

Fourier's law is substituted into Eq. (E6-3):

$$\frac{\partial}{\partial x} \left( -k th dy \frac{\partial T}{\partial x} \right) dx + \frac{\partial}{\partial y} \left( -k th dx \frac{\partial T}{\partial y} \right) dy = 0 \quad (\text{E6-4})$$

where  $th$  is the thickness of the domain. Equation (E6-4) is simplified to:

$$\frac{\partial}{\partial x} \left( -k \frac{\partial T}{\partial x} \right) + \frac{\partial}{\partial y} \left( -k \frac{\partial T}{\partial y} \right) = 0 \quad (\text{E6-5})$$

### ***The Weighted Average Residual Equation***

The finite element technique relies on identifying functions that represent, approximately, the exact solution. If the exact solution for temperature is substituted into Eq. (E6-5) then the right hand side must be exactly equal to zero anywhere in the computational domain. If an approximate solution for temperature,  $\hat{T}$ , is substituted into Eq. (E6-5) then the right hand side will not exactly equal zero. In this case, Eq. (E6-5) defines the residual:

$$\frac{\partial}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) + \frac{\partial}{\partial y} \left( -k \frac{\partial \hat{T}}{\partial y} \right) = \text{residual} \quad (\text{E6-6})$$

The finite element technique does not attempt to make this residual equal to zero. Rather, the finite element technique tries to make the weighted, average residual equal to zero. Equation (E6-5) is multiplied by a weighting function,  $f(x,y)$ , in order to define the weighted residual:

$$f \left[ \frac{\partial}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) + \frac{\partial}{\partial y} \left( -k \frac{\partial \hat{T}}{\partial y} \right) \right] = \text{weighted residual} \quad (\text{E6-7})$$

The weighted residual is integrated over the area of the computational domain:

$$\iint_A f \left[ \frac{\partial}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) + \frac{\partial}{\partial y} \left( -k \frac{\partial \hat{T}}{\partial y} \right) \right] dx dy = \text{weighted average residual} \quad (\text{E6-8})$$

Equation (E6-8) is split into two integrals:

E6: Section 2.7.2 *The Galerkin Weighted Residual Method*

$$\underbrace{\iint_A f \frac{\partial}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) dx dy}_{\text{Integral 1}} + \underbrace{\iint_A f \frac{\partial}{\partial y} \left( -k \frac{\partial \hat{T}}{\partial y} \right) dx dy}_{\text{Integral 2}} = \text{weighted average residual} \quad (\text{E6-9})$$

The first integral in Eq. (E6-9) is:

$$\text{Integral 1} = \int_{y_{\min}}^{y_{\max}} \underbrace{\left[ \int_{x_{\min}}^{x_{\max}} f \frac{\partial}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) dx \right]}_{x \text{ integral}} dy \quad (\text{E6-10})$$

where  $x_{\min}$  and  $x_{\max}$  are the lower and upper bounds, respectively, on the  $x$ -coordinate and  $y_{\min}$  and  $y_{\max}$  are the lower and upper bounds, respectively, on the  $y$ -coordinate. The  $x$  integral in Eq. (E6-10) can be integrated by parts. According to the chain rule:

$$\frac{\partial}{\partial x} \left[ f \left( -k \frac{\partial \hat{T}}{\partial x} \right) \right] = \frac{\partial f}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) + f \frac{\partial}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) \quad (\text{E6-11})$$

Integrating Eq. (E6-11) from  $x_{\min}$  to  $x_{\max}$  leads to:

$$\underbrace{\int_{x_{\min}}^{x_{\max}} \frac{\partial}{\partial x} \left[ f \left( -k \frac{\partial \hat{T}}{\partial x} \right) \right] dx}_{\text{integral of a derivative}} = \int_{x_{\min}}^{x_{\max}} \frac{\partial f}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) dx + \underbrace{\int_{x_{\min}}^{x_{\max}} f \frac{\partial}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) dx}_{x\text{-integral}} \quad (\text{E6-12})$$

The left side of Eq. (E6-12) is the integral of a derivative and the last term in Eq. (E6-12) is the  $x$ -integral that is required by Eq. (E6-10):

$$\underbrace{\int_{x_{\min}}^{x_{\max}} f \frac{\partial}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) dx}_{x \text{ integral}} = \left[ f \left( -k \frac{\partial \hat{T}}{\partial x} \right) \right]_{x_{\min}}^{x_{\max}} - \int_{x_{\min}}^{x_{\max}} \frac{\partial f}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) dx \quad (\text{E6-13})$$

Substituting Eq. (E6-13) into Eq. (E6-10) leads to:

$$\text{Integral 1} = \int_{y_{\min}}^{y_{\max}} \left\{ \left[ f \left( -k \frac{\partial \hat{T}}{\partial x} \right) \right]_{x_{\min}}^{x_{\max}} - \int_{x_{\min}}^{x_{\max}} \frac{\partial f}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) dx \right\} dy \quad (\text{E6-14})$$

Applying the limits of integration leads to:

E6: Section 2.7.2 *The Galerkin Weighted Residual Method*

$$\text{Integral 1} = \int_{y_{\min}}^{y_{\max}} \left[ f \left( -k \frac{\partial \hat{T}}{\partial x} \right) \right]_{x=x_{\max}} dy - \int_{y_{\min}}^{y_{\max}} \left[ f \left( -k \frac{\partial \hat{T}}{\partial x} \right) \right]_{x=x_{\min}} dy - \int_{y_{\min}}^{y_{\max}} \int_{x_{\min}}^{x_{\max}} \frac{\partial f}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) dx dy \quad (\text{E6-15})$$

The heat flux in the  $x$ -direction is given by Fourier's law:

$$\dot{q}_x'' = -k \frac{\partial \hat{T}}{\partial x} \quad (\text{E6-16})$$

Substituting Eq. (E6-16) into Eq. (E6-14) leads to:

$$\text{Integral 1} = \int_{y_{\min}}^{y_{\max}} (f \dot{q}_x'')_{x=x_{\max}} dy - \int_{y_{\min}}^{y_{\max}} (f \dot{q}_x'')_{x=x_{\min}} dy - \iint_A \frac{\partial f}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) dA \quad (\text{E6-17})$$

A similar process carried out on Integral 2 in Eq. (E6-9) leads to:

$$\text{Integral 2} = \int_{x_{\min}}^{x_{\max}} (f \dot{q}_y'')_{y=y_{\max}} dx - \int_{x_{\min}}^{x_{\max}} (f \dot{q}_y'')_{y=y_{\min}} dx - \iint_A \frac{\partial f}{\partial y} \left( -k \frac{\partial \hat{T}}{\partial y} \right) dA \quad (\text{E6-18})$$

where  $\dot{q}_y''$  is the heat flux in the  $y$ -direction. Substituting Eqs. (E6-17) and (E6-18) into Eq. (E6-9) leads to:

$$\begin{aligned} & \int_{y_{\min}}^{y_{\max}} (f \dot{q}_x'')_{x=x_{\max}} dy - \int_{y_{\min}}^{y_{\max}} (f \dot{q}_x'')_{x=x_{\min}} dy - \iint_A \frac{\partial f}{\partial x} \left( -k \frac{\partial \hat{T}}{\partial x} \right) dA + \\ & \int_{x_{\min}}^{x_{\max}} (f \dot{q}_y'')_{y=y_{\max}} dx - \int_{x_{\min}}^{x_{\max}} (f \dot{q}_y'')_{y=y_{\min}} dx - \iint_A \frac{\partial f}{\partial y} \left( -k \frac{\partial \hat{T}}{\partial y} \right) dA \quad (\text{E6-19}) \\ & = \text{weighted average residual} \end{aligned}$$

The temperature distribution,  $\hat{T}$ , is adjusted numerically by the finite element solution method so that the weighted average residual is zero. Therefore, Eq. (E6-19) can be rearranged:

$$\begin{aligned} & \underbrace{\iint_A \left[ k \frac{\partial f}{\partial x} \frac{\partial \hat{T}}{\partial x} + k \frac{\partial f}{\partial y} \frac{\partial \hat{T}}{\partial y} \right] dA}_{\text{weak form of the differential equation}} = \\ & \underbrace{- \int_{y_{\min}}^{y_{\max}} (f \dot{q}_x'')_{x=x_{\max}} dy + \int_{y_{\min}}^{y_{\max}} (f \dot{q}_x'')_{x=x_{\min}} dy - \int_{x_{\min}}^{x_{\max}} (f \dot{q}_y'')_{y=y_{\max}} dx + \int_{x_{\min}}^{x_{\max}} (f \dot{q}_y'')_{y=y_{\min}} dx}_{\text{weighted value of the net heat transfer into the region along the boundary}} \quad (\text{E6-20}) \end{aligned}$$

The left side of Eq. (E6-20) is referred to as the weak form of the partial differential equation and the right side of Eq. (E6-20) is the weighted value of the net rate of heat transfer into the boundary of the computational domain. The first two terms on the right side of Eq. (E6-20) are the  $x$ -projection of the conduction heat transfer into the left and right sides of the boundary, respectively, and the last two terms are the  $y$ -projection of the conduction heat transfer into the bottom and top sides of the boundary, respectively. Equation (E6-20) can be written more concisely as:

$$\iint_A \left[ k \frac{\partial f}{\partial x} \frac{\partial \hat{T}}{\partial x} + k \frac{\partial f}{\partial y} \frac{\partial \hat{T}}{\partial y} \right] dA = \int_B f \dot{q}_n'' ds \quad (\text{E6-21})$$

where  $\dot{q}_n''$  is the heat flux into the computational domain normal to the boundary and  $ds$  is the differential length along the boundary (see Figure E6-1 **Error! Reference source not found.**).

### ***The Approximate Temperature Function***

The approximate temperature distribution,  $\hat{T}(x, y)$ , that is substituted into Eq. (E6-21) and manipulated by the finite element technique is actually the sum of  $N$  functions:

$$\hat{T}(x, y) = w_1(x, y)\hat{T}_1 + w_2(x, y)\hat{T}_2 + \dots + w_N(x, y)\hat{T}_N \quad (\text{E6-22})$$

Equation (E6-22) can be written as the multiplication of two vectors:

$$\hat{T}(x, y) = \underbrace{\left[ w_1(x, y) \quad w_2(x, y) \quad \dots \quad w_N(x, y) \right]}_{\underline{w}^T} \underbrace{\begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \dots \\ \hat{T}_N \end{bmatrix}}_{\hat{\underline{T}}} \quad (\text{E6-23})$$

or

$$\hat{T}(x, y) = \underline{w}^T \hat{\underline{T}} \quad (\text{E6-24})$$

where the superscript  $T$  indicates the transpose of the vector  $\underline{w}$ . The vector  $\underline{w}$  contains dimensionless functions of  $x$  and  $y$ :

$$\underline{w} = \begin{bmatrix} w_1(x, y) \\ w_2(x, y) \\ \dots \\ w_N(x, y) \end{bmatrix} \quad (\text{E6-25})$$

and  $\hat{\underline{T}}$  is a vector of unknown temperatures (these are, eventually, the temperatures at each of the nodes placed in the computational domain):

$$\hat{\underline{T}} = \begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \dots \\ \hat{T}_N \end{bmatrix} \quad (\text{E6-26})$$

Substituting Eq. (E6-24) into Eq. (E6-21) leads to:

$$\iint_A \left[ k \frac{\partial f}{\partial x} \frac{\partial}{\partial x} (\underline{w}^T \hat{\underline{T}}) + k \frac{\partial f}{\partial y} \frac{\partial}{\partial y} (\underline{w}^T \hat{\underline{T}}) \right] dA = \int_B f \dot{q}_n'' ds \quad (\text{E6-27})$$

The partial derivative of the approximate temperature distribution with respect to  $x$  can be evaluated according to:

$$\frac{\partial}{\partial x} (\underline{w}^T \hat{\underline{T}}) = \frac{\partial}{\partial x} \left( \begin{bmatrix} w_1 & w_2 & \dots & w_N \end{bmatrix} \begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \dots \\ \hat{T}_N \end{bmatrix} \right) = \begin{bmatrix} \frac{\partial w_1}{\partial x} & \frac{\partial w_2}{\partial x} & \dots & \frac{\partial w_N}{\partial x} \end{bmatrix} \begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \dots \\ \hat{T}_N \end{bmatrix} = \frac{\partial \underline{w}^T}{\partial x} \hat{\underline{T}} \quad (\text{E6-28})$$

A similar process leads to the partial derivative with respect to  $y$ :

$$\frac{\partial}{\partial y} (\underline{w}^T \hat{\underline{T}}) = \frac{\partial \underline{w}^T}{\partial y} \hat{\underline{T}} \quad (\text{E6-29})$$

Substituting Eqs. (E6-28) and (E6-29) into Eq. (E6-27) leads to:

$$\iint_A \left[ k \frac{\partial f}{\partial x} \frac{\partial \underline{w}^T}{\partial x} \hat{\underline{T}} + k \frac{\partial f}{\partial y} \frac{\partial \underline{w}^T}{\partial y} \hat{\underline{T}} \right] dA = \int_B f \dot{q}_n'' ds \quad (\text{E6-30})$$

Note that the vector  $\hat{\underline{T}}$  is not a function of  $x$  or  $y$  since the temperatures in this vector are for specified positions (see Eq. (E6-26)), and therefore  $\hat{\underline{T}}$  may be removed from the integrand of Eq. (E6-30):

$$\left[ \iint_A \left( k \frac{\partial f}{\partial x} \frac{\partial \underline{w}^T}{\partial x} + k \frac{\partial f}{\partial y} \frac{\partial \underline{w}^T}{\partial y} \right) dA \right] \hat{\underline{T}} = \int_B f \dot{q}_n'' ds \quad (\text{E6-31})$$

Given a single weighting function  $f$ , Eq. (E6-31) represents a single equation in the  $N$  unknown temperatures  $\hat{T}_1, \hat{T}_2, \dots, \hat{T}_N$ . To see this clearly, expand the vectors in Eq. (E6-31):

$$\left[ \iint_A \left( k \frac{\partial f}{\partial x} \left[ \frac{\partial w_1}{\partial x} \frac{\partial w_2}{\partial x} \dots \frac{\partial w_N}{\partial x} \right] + k \frac{\partial f}{\partial y} \left[ \frac{\partial w_1}{\partial y} \frac{\partial w_2}{\partial y} \dots \frac{\partial w_N}{\partial y} \right] \right) dA \right] \begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \dots \\ \hat{T}_N \end{bmatrix} = \int_B f \dot{q}_n'' ds \quad (\text{E6-32})$$

and carry out the vector multiplication:

$$\begin{aligned} & \left[ \iint_A \left( k \frac{\partial f}{\partial x} \frac{\partial w_1}{\partial x} + k \frac{\partial f}{\partial y} \frac{\partial w_1}{\partial y} \right) dA \right] \hat{T}_1 + \left[ \iint_A \left( k \frac{\partial f}{\partial x} \frac{\partial w_2}{\partial x} + k \frac{\partial f}{\partial y} \frac{\partial w_2}{\partial y} \right) dA \right] \hat{T}_2 + \dots \\ & + \left[ \iint_A \left( k \frac{\partial f}{\partial x} \frac{\partial w_N}{\partial x} + k \frac{\partial f}{\partial y} \frac{\partial w_N}{\partial y} \right) dA \right] \hat{T}_N = \int_B f \dot{q}_n'' ds \end{aligned} \quad (\text{E6-33})$$

The result of each of the integrations in square brackets in Eq. (E6-33) is a scalar and therefore Eq. (E6-33) is a single, linear equation in  $N$  unknown temperatures  $\hat{T}_1$  through  $\hat{T}_N$ .

### ***Galerkin's Method***

Equation (E6-33) is underspecified. In order to obtain  $N$  equations of the form of Eq. (E6-33), we will write the equation for  $N$  different weighting functions,  $f_1, f_2, \dots, f_N$ :

$$\begin{aligned} & \left[ \iint_A \left( k \frac{\partial f_1}{\partial x} \frac{\partial w_1}{\partial x} + k \frac{\partial f_1}{\partial y} \frac{\partial w_1}{\partial y} \right) dA \right] \hat{T}_1 + \left[ \iint_A \left( k \frac{\partial f_1}{\partial x} \frac{\partial w_2}{\partial x} + k \frac{\partial f_1}{\partial y} \frac{\partial w_2}{\partial y} \right) dA \right] \hat{T}_2 + \dots \\ & + \left[ \iint_A \left( k \frac{\partial f_1}{\partial x} \frac{\partial w_N}{\partial x} + k \frac{\partial f_1}{\partial y} \frac{\partial w_N}{\partial y} \right) dA \right] \hat{T}_N = \int_B f_1 \dot{q}_n'' ds \\ & \left[ \iint_A \left( k \frac{\partial f_2}{\partial x} \frac{\partial w_1}{\partial x} + k \frac{\partial f_2}{\partial y} \frac{\partial w_1}{\partial y} \right) dA \right] \hat{T}_1 + \left[ \iint_A \left( k \frac{\partial f_2}{\partial x} \frac{\partial w_2}{\partial x} + k \frac{\partial f_2}{\partial y} \frac{\partial w_2}{\partial y} \right) dA \right] \hat{T}_2 + \dots \\ & + \left[ \iint_A \left( k \frac{\partial f_2}{\partial x} \frac{\partial w_N}{\partial x} + k \frac{\partial f_2}{\partial y} \frac{\partial w_N}{\partial y} \right) dA \right] \hat{T}_N = \int_B f_2 \dot{q}_n'' ds \end{aligned} \quad (\text{E6-34})$$

...

E6: Section 2.7.2 *The Galerkin Weighted Residual Method*

$$\left[ \iint_A \left( k \frac{\partial f_N}{\partial x} \frac{\partial w_1}{\partial x} + k \frac{\partial f_N}{\partial y} \frac{\partial w_1}{\partial y} \right) dA \right] \hat{T}_1 + \left[ \iint_A \left( k \frac{\partial f_N}{\partial x} \frac{\partial w_2}{\partial x} + k \frac{\partial f_N}{\partial y} \frac{\partial w_2}{\partial y} \right) dA \right] \hat{T}_2 + \dots$$

$$+ \left[ \iint_A \left( k \frac{\partial f_N}{\partial x} \frac{\partial w_N}{\partial x} + k \frac{\partial f_N}{\partial y} \frac{\partial w_N}{\partial y} \right) dA \right] \hat{T}_N = \int_B f_N \dot{q}_n'' ds$$

Galerkin's method takes the weighting functions,  $f_i$ , to be equal to the dimensionless functions that are used to specify the approximate temperature distribution,  $w_i$ :

$$f_i(x, y) = w_i(x, y) \quad \text{for } i = 1..N \quad (\text{E6-35})$$

Substituting Eq. (E6-35) into Eq. (E6-34) leads to:

$$\left[ \iint_A \left( k \frac{\partial w_1}{\partial x} \frac{\partial w_1}{\partial x} + k \frac{\partial w_1}{\partial y} \frac{\partial w_1}{\partial y} \right) dA \right] \hat{T}_1 + \left[ \iint_A \left( k \frac{\partial w_1}{\partial x} \frac{\partial w_2}{\partial x} + k \frac{\partial w_1}{\partial y} \frac{\partial w_2}{\partial y} \right) dA \right] \hat{T}_2 + \dots$$

$$+ \left[ \iint_A \left( k \frac{\partial w_1}{\partial x} \frac{\partial w_N}{\partial x} + k \frac{\partial w_1}{\partial y} \frac{\partial w_N}{\partial y} \right) dA \right] \hat{T}_N = \int_B w_1 \dot{q}_n'' ds$$
  

$$\left[ \iint_A \left( k \frac{\partial w_2}{\partial x} \frac{\partial w_1}{\partial x} + k \frac{\partial w_2}{\partial y} \frac{\partial w_1}{\partial y} \right) dA \right] \hat{T}_1 + \left[ \iint_A \left( k \frac{\partial w_2}{\partial x} \frac{\partial w_2}{\partial x} + k \frac{\partial w_2}{\partial y} \frac{\partial w_2}{\partial y} \right) dA \right] \hat{T}_2 + \dots$$

$$+ \left[ \iint_A \left( k \frac{\partial w_2}{\partial x} \frac{\partial w_N}{\partial x} + k \frac{\partial w_2}{\partial y} \frac{\partial w_N}{\partial y} \right) dA \right] \hat{T}_N = \int_B w_2 \dot{q}_n'' ds \quad (\text{E6-36})$$

...

$$\left[ \iint_A \left( k \frac{\partial w_N}{\partial x} \frac{\partial w_1}{\partial x} + k \frac{\partial w_N}{\partial y} \frac{\partial w_1}{\partial y} \right) dA \right] \hat{T}_1 + \left[ \iint_A \left( k \frac{\partial w_N}{\partial x} \frac{\partial w_2}{\partial x} + k \frac{\partial w_N}{\partial y} \frac{\partial w_2}{\partial y} \right) dA \right] \hat{T}_2 + \dots$$

$$+ \left[ \iint_A \left( k \frac{\partial w_N}{\partial x} \frac{\partial w_N}{\partial x} + k \frac{\partial w_N}{\partial y} \frac{\partial w_N}{\partial y} \right) dA \right] \hat{T}_N = \int_B w_N \dot{q}_n'' ds$$

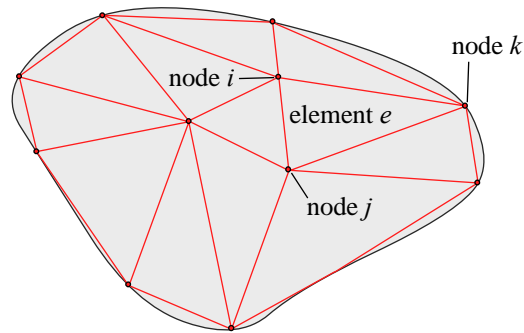
Given the functions  $w_1, w_2, \dots, w_N$ , the area integrals in Eq. (E6-36) can be evaluated explicitly in order to provide  $N$  linear equations in the unknown temperatures  $\hat{T}_1, \hat{T}_2, \dots, \hat{T}_N$ . Equation (E6-36) can be written more concisely in vector notation:

$$\underbrace{\left[ \iint_A k \left( \frac{\partial \underline{w}}{\partial x} \frac{\partial \underline{w}^T}{\partial x} + \frac{\partial \underline{w}}{\partial y} \frac{\partial \underline{w}^T}{\partial y} \right) dA \right]}_{\underline{K}} \hat{\underline{T}} = \underbrace{\int_B \underline{w} \dot{q}_n'' ds}_{\underline{q}} \quad (\text{E6-37})$$

The left side of Eq. (E6-37) is referred to as the conduction matrix  $\underline{\underline{K}}$ . The conduction matrix has size  $N \times N$  and multiplies a vector  $\hat{T}$  of unknown temperatures. The right side of Eq. (E6-37) is a column vector,  $\underline{q}$ . Therefore, Eq. (E6-37) is analogous to the matrix equation that resulted from the finite difference solution to a steady-state two-dimensional conduction problem,  $\underline{A}\underline{X} = \underline{b}$ , although the structure of  $\underline{\underline{K}}$  and  $\underline{A}$  are substantially different.

### ***Interpolating Functions***

In order to complete the finite element solution, it is necessary to specify the functions  $w_i(x, y)$ ; this process is guided by the discretization of the computational domain. **In Error! Reference source not found.**, a set of  $N$  nodes are distributed across the computational domain that was shown in **Error! Reference source not found.**. The positioning of the nodes need not be uniform, but nodes are placed in order to provide an approximation of the boundary. Lines drawn between nodes form triangular elements. The element  $e$  in Figure E6-2 is defined by the three nodes  $i, j$ , and  $k$ .

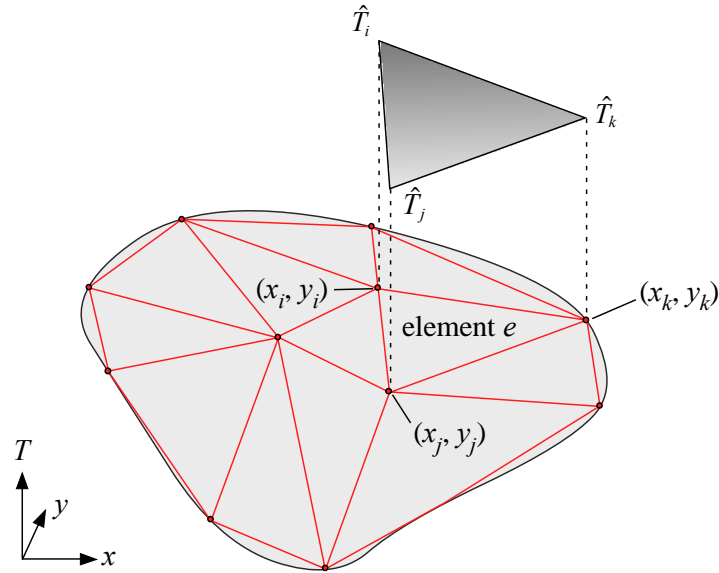


**Figure E6-2: Discretization of the computational domain.**

The temperatures  $\hat{T}_1, \hat{T}_2, \dots, \hat{T}_N$  are the numerical prediction of the temperature at each of the nodes in the domain and the functions  $w_1, w_2, \dots, w_N$  are interpolating functions. Recall the form of our assumed temperature function, Eq. (E6-22):

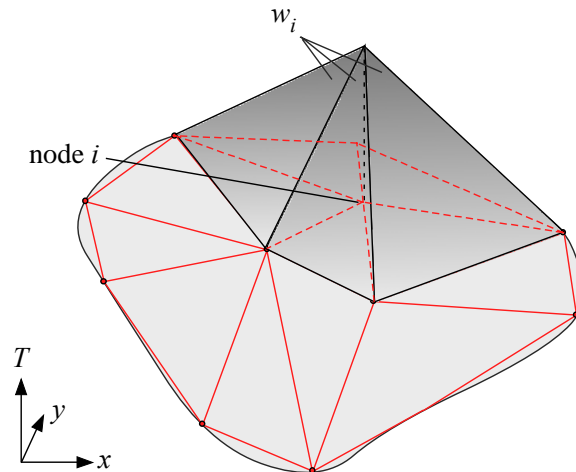
$$\hat{T}(x, y) = w_1(x, y)\hat{T}_1 + w_2(x, y)\hat{T}_2 + \dots + w_N(x, y)\hat{T}_N \quad (\text{E6-22})$$

The dimensionless functions  $w_i$  are interpolating functions; these functions are used to predict the temperature anywhere in the computational domain based the influence of the nodal temperatures. The temperature within any element is assumed to be a linear function of the three nodal temperatures that define that element. For example, the temperature at any location in element  $e$  shown in Figure E6-2 is a linear function of the nodal temperatures  $\hat{T}_i, \hat{T}_j$ , and  $\hat{T}_k$ . This is equivalent to assuming that the temperature within element  $e$  lies on a plane that is defined by the three points  $\hat{T}_i$  at  $(x_i, y_i)$ ,  $\hat{T}_j$  at  $(x_j, y_j)$ , and  $\hat{T}_k$  at  $(x_k, y_k)$  as shown in Figure E6-3.



**Figure E6-3: Temperature distribution in element  $e$  is a plane passing through nodal temperatures.**

The temperature at any point in element  $e$  is obtained by interpolating between the temperatures  $\hat{T}_i$ ,  $\hat{T}_j$ , and  $\hat{T}_k$ . Clearly then, the interpolating function  $w_i$  should be 1 at  $(x_i, y_i)$  and drop linearly to 0 at  $(x_j, y_j)$  and  $(x_k, y_k)$ . The value of  $w_i$  should be between 0 and 1 everywhere inside element  $e$  (or within any element that is adjacent to node  $i$ ). Figure E6-4 illustrates the interpolating function  $w_i$ ; the function is a pyramid with height 1 that is centered on node  $i$ . The interpolating function  $w_i$  is zero in any element that is not adjacent to node  $i$  because the value of  $\hat{T}_i$  does not influence the calculation of the temperature outside of those elements that are adjacent to node  $i$ .



**Figure E6-4: Interpolating function  $w_i$ .**

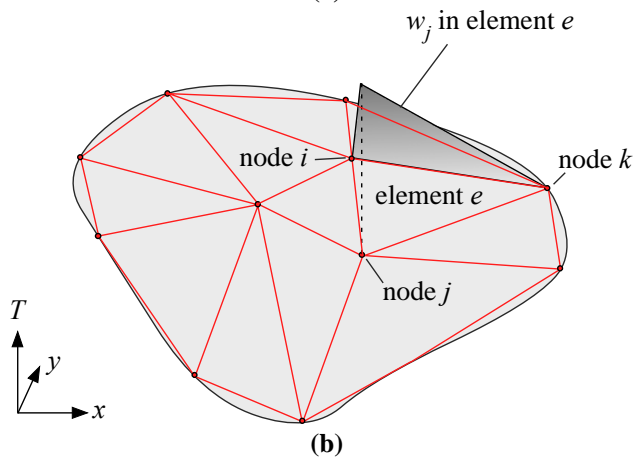
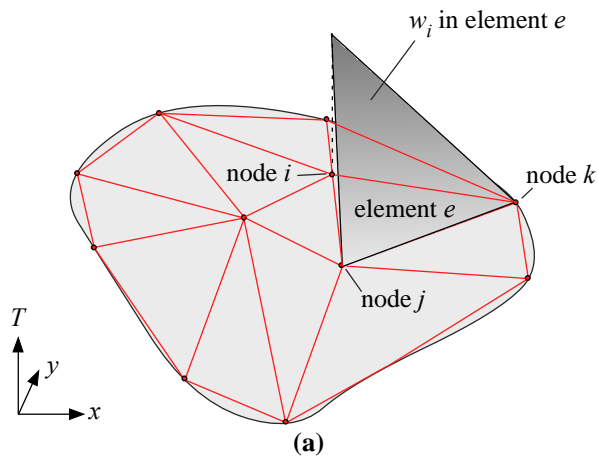
Figure E6-5 illustrates the interpolating functions  $w_i$ ,  $w_j$ , and  $w_k$  within element  $e$ . Note that the value of all of the other interpolating functions must be zero within element  $e$ . Therefore, the temperature anywhere within element  $e$  ( $\hat{T}_e$ ) is given by:

E6: Section 2.7.2 *The Galerkin Weighted Residual Method*

$$\hat{T}_e(x, y) = w_i(x, y)\hat{T}_i + w_j(x, y)\hat{T}_j + w_k(x, y)\hat{T}_k \quad (\text{E6-38})$$

or

$$\hat{T}_e = \begin{bmatrix} 0 & \dots & 0 & w_i & 0 & \dots & 0 & w_j & 0 & \dots & 0 & w_k & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \dots \\ \hat{T}_N \end{bmatrix} \quad (\text{E6-39})$$



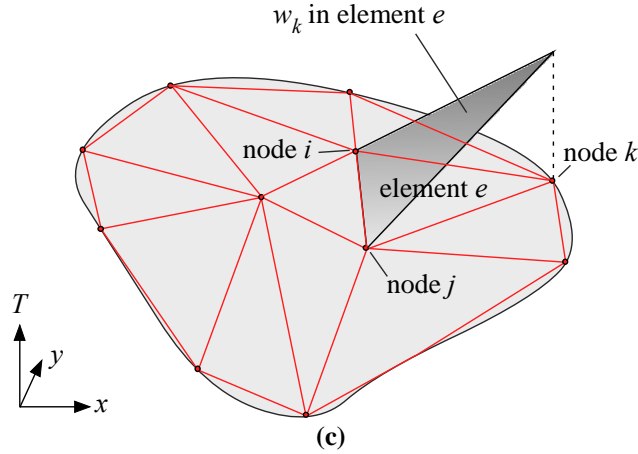


Figure E6-5: Interpolating functions (a)  $w_i$ , (b)  $w_j$ , and (c)  $w_k$  within element  $e$ .

For the triangular elements considered here, the interpolating functions are referred to as a triangular coordinates. The mathematical rules that govern triangular coordinates have been worked out and are presented in various texts, including Myers (1987). The coordinates of nodes  $i$ ,  $j$ , and  $k$  that define element  $e$  are  $(x_i, y_i)$ ,  $(x_j, y_j)$ , and  $(x_k, y_k)$ . The distance between these nodes in the  $x$  and  $y$  directions are abbreviated according to:

$$x_{ij} = x_j - x_i \quad (\text{E6-40})$$

$$x_{ik} = x_k - x_i \quad (\text{E6-41})$$

$$x_{jk} = x_k - x_j \quad (\text{E6-42})$$

$$y_{ij} = y_j - y_i \quad (\text{E6-43})$$

$$y_{ik} = y_k - y_i \quad (\text{E6-44})$$

$$y_{jk} = y_k - y_j \quad (\text{E6-45})$$

The area of element  $e$  is then:

$$A_e = \frac{|b_{ijk}|}{2} \quad (\text{E6-46})$$

where  $b_{ijk}$  is:

$$b_{ijk} = x_{ij} y_{jk} - x_{jk} y_{ij} \quad (\text{E6-47})$$

Provided that nodes  $i$ ,  $j$ , and  $k$  are defined in a counterclockwise fashion (as shown in Figure E6-5), the value of  $b_{ijk}$  will be positive and therefore:

$$A_e = \frac{b_{ijk}}{2} \quad (\text{E6-48})$$

The partial derivative of the interpolating functions with respect to  $x$  and  $y$  are:

$$\frac{\partial w_i}{\partial x} = -\frac{y_{jk}}{b_{ijk}} \quad (\text{E6-49})$$

$$\frac{\partial w_j}{\partial x} = \frac{y_{ik}}{b_{ijk}} \quad (\text{E6-50})$$

$$\frac{\partial w_k}{\partial x} = -\frac{y_{ij}}{b_{ijk}} \quad (\text{E6-51})$$

$$\frac{\partial w_i}{\partial y} = -\frac{x_{jk}}{b_{ijk}} \quad (\text{E6-52})$$

$$\frac{\partial w_j}{\partial y} = \frac{x_{ik}}{b_{ijk}} \quad (\text{E6-53})$$

$$\frac{\partial w_k}{\partial y} = -\frac{x_{ij}}{b_{ijk}} \quad (\text{E6-54})$$

The general formula that provides the integral of the product of triangular coordinates to arbitrary powers  $a$ ,  $b$ , and  $c$  is:

$$\iint_{A_e} w_i^a w_j^b w_k^c dA = \frac{a!b!c!}{(a+b+c+2)!} 2 A_e \quad (\text{E6-55})$$

### ***Assembling the Global Conduction Matrix***

The area integral in Eq. (E6-37) is evaluated by summing the area integral obtained over each of the elements:

$$\underline{\underline{K}} = \iint_A k \left( \frac{\partial w}{\partial x} \frac{\partial w^T}{\partial x} + \frac{\partial w}{\partial y} \frac{\partial w^T}{\partial y} \right) dA = \sum_{e=1}^{N_e} k_e \underbrace{\iint_{A_e} \left( \frac{\partial w}{\partial x} \frac{\partial w^T}{\partial x} + \frac{\partial w}{\partial y} \frac{\partial w^T}{\partial y} \right) dA}_{K_e = \text{area integral evaluated over element } e} = \sum_{i=1}^{N_e} \underline{\underline{K}}_e \quad (\text{E6-56})$$

E6: Section 2.7.2 *The Galerkin Weighted Residual Method*

where  $N_e$  is the number of elements and  $k_e$  is the thermal conductivity within element  $e$ . If we can evaluate the integral over any element  $e$  then it is possible to calculate the element conduction matrix  $\underline{\underline{K}}_e$  and sum these element conduction matrices in order to obtain the global conduction matrix  $\underline{\underline{K}}$ . The element conduction matrix is defined by:

$$\underline{\underline{K}}_e = k_e \iint_{A_e} \left( \frac{\partial w}{\partial x} \frac{\partial w^T}{\partial x} + \frac{\partial w}{\partial y} \frac{\partial w^T}{\partial y} \right) dA \quad (\text{E6-57})$$

Within element  $e$ , only weighting functions  $w_i$ ,  $w_j$ , and  $w_k$  are nonzero in the vector  $\underline{w}$  (see Figure E6-5):

$$\underline{w}^T = [0 \dots 0 w_i \ 0 \dots 0 w_j \ 0 \dots 0 w_k \ 0 \dots 0] \quad (\text{E6-58})$$

Equation (E6-57) is split into two area integrals:

$$\underline{\underline{K}}_e = k_e \underbrace{\iint_{A_e} \frac{\partial w}{\partial x} \frac{\partial w^T}{\partial x} dA}_{\text{Integral 1}} + k_e \underbrace{\iint_{A_e} \frac{\partial w}{\partial y} \frac{\partial w^T}{\partial y} dA}_{\text{Integral 2}} \quad (\text{E6-59})$$

Equation (E6-58) is substituted into the first integral in Eq.(E6-59):

$$\text{Integral 1} = k_e \iint_{A_e} \frac{\partial w}{\partial x} \frac{\partial w^T}{\partial x} dA = k_e \iint_{A_e} \frac{\partial}{\partial x} \begin{bmatrix} 0 \\ \dots \\ 0 \\ w_i \\ 0 \\ \dots \\ 0 \\ w_j \\ 0 \\ \dots \\ 0 \\ w_k \\ 0 \\ \dots \\ 0 \end{bmatrix} \frac{\partial}{\partial x} [0 \dots 0 w_i \ 0 \dots 0 w_j \ 0 \dots 0 w_k \ 0 \dots 0] dA \quad (\text{E6-60})$$

Substituting Eqs. (E6-49) through (E6-51) into Eq. (E6-60) leads to:

$$\text{Integral 1} = \frac{k_e}{b_{ijk}^2} \iint_{A_e} \begin{bmatrix} 0 \\ \dots \\ 0 \\ -y_{jk} \\ 0 \\ \dots \\ 0 \\ y_{ik} \\ 0 \\ \dots \\ 0 \\ -y_{ij} \\ 0 \\ \dots \\ 0 \end{bmatrix} \begin{bmatrix} 0 & \dots & 0 & -y_{jk} & 0 & \dots & 0 & y_{ik} & 0 & \dots & 0 & -y_{ij} & 0 & \dots & 0 \end{bmatrix} dA \quad (\text{E6-61})$$

Carrying out the vector multiplication in Eq. (E6-61) leads to:

$$\text{Integral 1} = \frac{k_e}{b_{ijk}^2} \iint_{A_e} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & y_{jk}^2 & \cdot & -y_{jk} y_{ik} & \cdot & y_{jk} y_{ik} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & -y_{ik} y_{jk} & \cdot & y_{ik}^2 & \cdot & -y_{ik} y_{ij} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & y_{ij} y_{jk} & \cdot & -y_{ij} y_{ik} & \cdot & y_{ij}^2 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} dA \quad (\text{E6-62})$$

Note that the matrix in the integrand of Eq. (E6-62) is independent of  $x$  and  $y$  and can be removed from the integral:

$$\text{Integral 1} = \frac{k_e}{b_{ijk}^2} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & y_{jk}^2 & \cdot & -y_{jk} y_{ik} & \cdot & y_{jk} y_{ik} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & -y_{ik} y_{jk} & \cdot & y_{ik}^2 & \cdot & -y_{ik} y_{ij} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & y_{ij} y_{jk} & \cdot & -y_{ij} y_{ik} & \cdot & y_{ij}^2 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \underbrace{\iint_{A_e} dA}_{A_e} \quad (\text{E6-63})$$

Therefore:

$$\text{Integral 1} = \frac{A_e k_e}{b_{ijk}^2} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & y_{jk}^2 & \cdot & -y_{jk} y_{ik} & \cdot & y_{jk} y_{ik} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & -y_{ik} y_{jk} & \cdot & y_{ik}^2 & \cdot & -y_{ik} y_{ij} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & y_{ij} y_{jk} & \cdot & -y_{ij} y_{ik} & \cdot & y_{ij}^2 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \quad (\text{E6-64})$$

A similar sequence of operations provides the second integral in Eq. (E6-59):

$$\text{Integral 2} = \frac{A_e k_e}{b_{ijk}^2} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & x_{jk}^2 & \cdot & -x_{jk} x_{ik} & \cdot & x_{jk} x_{ik} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & -x_{ik} x_{jk} & \cdot & x_{ik}^2 & \cdot & -x_{ik} x_{ij} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & x_{ij} x_{jk} & \cdot & -x_{ij} x_{ik} & \cdot & x_{ij}^2 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \quad (\text{E6-65})$$

Adding Eqs. (E6-64) and (E6-65) provides the element conduction matrix:

$$\underline{\underline{K_e}} = \frac{A_e k_e}{b_{ijk}^2} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & (x_{jk}^2 + y_{jk}^2) & \cdot & -(x_{jk} x_{ik} + y_{jk} y_{ik}) & \cdot & (x_{jk} x_{ij} + y_{jk} y_{ij}) & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & -(x_{ik} x_{jk} + y_{ik} y_{jk}) & \cdot & (x_{ik}^2 + y_{ik}^2) & \cdot & -(x_{ik} x_{ij} + y_{ik} y_{ij}) & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & (x_{ij} x_{jk} + y_{ij} y_{jk}) & \cdot & -(x_{ij} x_{ik} + y_{ij} y_{ik}) & \cdot & (x_{ij}^2 + y_{ij}^2) & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \begin{matrix} \text{row } i \\ \text{row } j \\ \text{row } k \end{matrix}$$

column *i*
column *j*
column *k*

(E6-66)

Given the coordinates of the three nodes that define the element, Eq. (E6-66) can be used to calculate the element conduction matrix. These element conduction matrices are summed according to Eq. (E6-56) in order to assemble the global conduction matrix.

**Assembling the Global Boundary Vector and Convection Matrix**

The right side of Eq. (E6-37) is the global boundary vector:

$$\underline{q} = \int_B \underline{w} \dot{q}_n'' ds \quad (\text{E6-67})$$

where  $\dot{q}_n''$  is the heat flux normal to the boundary, into the domain as shown in **Error! Reference source not found.** The heat flux may be a combination of a specified heat flux and a convective heat flux:

$$\dot{q}_n'' = \dot{q}_s'' + h(T_\infty - \hat{T}) \quad (\text{E6-68})$$

where  $\dot{q}_s''$  is the specified heat flux,  $h$  is the local heat transfer coefficient, and  $T_\infty$  is the local ambient temperature. Substituting Eq. (E6-68) into Eq. (E6-67) leads to:

$$\underline{q} = \int_B \underline{w} \left[ \dot{q}_s'' + h(T_\infty - \hat{T}) \right] ds \quad (\text{E6-69})$$

Just as the area integral over the computational domain was divided up by elements, the boundary integral in Eq. (E6-69) is divided up by boundary segment:

$$\underline{q} = \int_B \underline{w} \left[ \dot{q}_s'' + h(T_\infty - \hat{T}) \right] ds = \sum_{b=1}^{N_b} \underbrace{\int_{B_b} \underline{w} \left[ \dot{q}_{s,b}'' + h_b(T_{\infty,b} - \hat{T}) \right] ds}_{q_b} = \sum_{b=1}^{N_b} q_b \quad (\text{E6-70})$$

where  $N_b$  is the number of boundary segments,  $\dot{q}_{s,b}''$ ,  $h_b$ , and  $T_{\infty,b}$  are the specified heat flux, heat transfer coefficient and ambient temperature, respectively, that exist for boundary segment  $b$ . Boundary segment  $b$  is defined by two nodes,  $i$  and  $j$ , as shown in Figure E6-6.

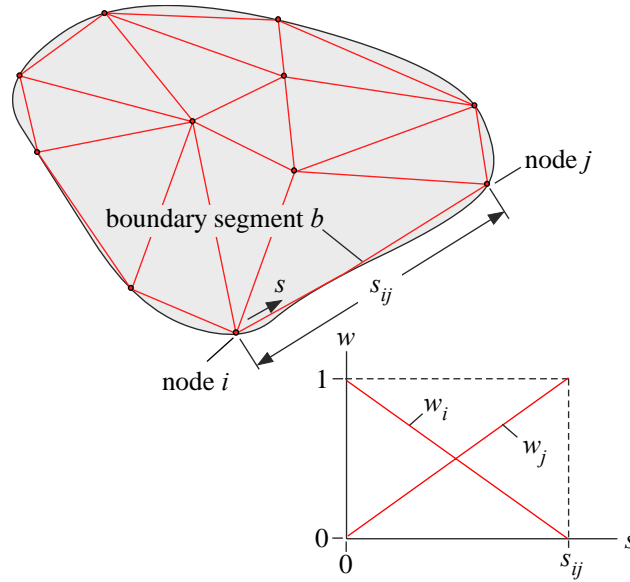


Figure E6-6: Boundary segment  $b$ .

Note that on boundary segment  $b$  between nodes  $i$  and  $j$ , the only weighting functions that are not zero are  $w_i$  and  $w_j$ . Therefore, the temperature on boundary segment  $b$  ( $\hat{T}_b$ ) is given by:

$$\hat{T}_b = w_i(x, y)\hat{T}_i + w_j(x, y)\hat{T}_j \quad (\text{E6-71})$$

or

$$\hat{T}_b = \begin{bmatrix} 0 & \dots & 0 & w_i & 0 & \dots & 0 & w_j & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \dots \\ \hat{T}_N \end{bmatrix} \quad (\text{E6-72})$$

As you move along the boundary segment in the  $s$ -direction, see Figure E6-6, the value of interpolating function  $w_i$  goes from 1 (at node  $i$ ) to 0 (at node  $j$ ) and  $w_j$  goes from 0 (at node  $i$ ) to 1 (at node  $j$ ). The segment boundary vector,  $\underline{q}_b$ , is defined according to:

$$\underline{q}_b = \int_{B_b} \underline{w} \left[ \dot{q}_{s,b}'' + h_b (T_{\infty,b} - \hat{T}) \right] ds \quad (\text{E6-73})$$

which can be divided into three separate integrals:

$$\underline{q}_b = \underbrace{\dot{q}_{s,b}'' \int_{B_b} \underline{w} ds}_{\text{Integral 1}} + \underbrace{h_b T_{\infty,b} \int_{B_b} \underline{w} ds}_{\text{Integral 2}} - \underbrace{h_b \int_{B_b} \underline{w} \hat{T} ds}_{\text{Integral 3}} \quad (\text{E6-74})$$

The first integral in Eq. (E6-74) is evaluated according to:

$$\text{Integral 1} = \dot{q}_{s,b}'' \int_{B_b} w ds = \dot{q}_{s,b}'' \int_{B_b} \begin{bmatrix} 0 \\ \dots \\ 0 \\ w_i \\ 0 \\ \dots \\ 0 \\ w_j \\ 0 \\ \dots \\ 0 \end{bmatrix} ds \quad (\text{E6-75})$$

The integral of  $w_i$  and  $w_j$  over the boundary segment is equal to the area under the curves shown in Figure E6-6:

$$\dot{q}_{s,b}'' \int_{B_b} w_i ds = \dot{q}_{s,b}'' \frac{s_{ij}}{2} \quad (\text{E6-76})$$

where  $s_{ij}$  is the length of the boundary segment. Therefore, Eq. (E6-75) can be written as:

$$\text{Integral 1} = \frac{\dot{q}_{s,b}'' s_{ij}}{2} \begin{bmatrix} 0 \\ \dots \\ 0 \\ \boxed{1} \text{--- row } i \\ 0 \\ \dots \\ 0 \\ \boxed{1} \text{--- row } j \\ 0 \\ \dots \\ 0 \end{bmatrix} \quad (\text{E6-77})$$

The second integral in Eq. (E6-74) can be evaluated in a similar manner:

$$\text{Integral 2} = \frac{h_b T_{\infty,b} s_{ij}}{2} \begin{bmatrix} 0 \\ \dots \\ 0 \\ \boxed{1} \text{--- row } i \\ 0 \\ \dots \\ 0 \\ \boxed{1} \text{--- row } j \\ 0 \\ \dots \\ 0 \end{bmatrix} \quad (\text{E6-78})$$

Equation (E6-72) is substituted into the third integral in Eq. (E6-74):

$$\text{Integral 3} = h_b \int_{B_b} \underline{w} \hat{T} ds = h_b B_b \int_{B^b} \begin{bmatrix} 0 \\ \dots \\ 0 \\ w_i \\ 0 \\ \dots \\ 0 \\ w_j \\ 0 \\ \dots \\ 0 \end{bmatrix} \begin{bmatrix} 0 & \dots & 0 & w_i & 0 & \dots & 0 & w_j & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \dots \\ \hat{T}_N \end{bmatrix} ds \quad (\text{E6-79})$$

Carrying out the matrix multiplication in Eq. (E6-79) leads to:

$$\text{Integral 3} = h_b \int_{B_b} \left( \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & w_i^2 & \cdot & w_i w_j & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & w_j w_i & \cdot & w_j^2 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \right) \begin{bmatrix} \hat{T}_1 \\ \hat{T}_2 \\ \dots \\ \hat{T}_N \end{bmatrix} ds \quad (\text{E6-80})$$

The weighting functions  $w_i$  and  $w_j$  along the boundary element  $b$  can be written in terms of  $s$  as (see Figure E6-6):

E6: Section 2.7.2 *The Galerkin Weighted Residual Method*

$$w_i = 1 - \frac{S}{S_{ij}} \tag{E6-81}$$

$$w_j = \frac{S}{S_{ij}} \tag{E6-82}$$

Equations (E6-81) and (E6-82) allow each of the integrals in Eq. (E6-80) to be evaluated. For example, the integral required by the entry in row  $j$  and column  $j$  in Eq. (E6-80) leads to:

$$h_b \int_{B_b} w_j^2 ds = h_b \int_0^{s_{ij}} \frac{S^2}{S_{ij}^2} ds = h_b \left[ \frac{S^3}{3S_{ij}^2} \right]_0^{s_{ij}} = \frac{h_b S_{ij}}{3} \tag{E6-83}$$

The other integrals in Eq. (E6-80) are carried out in a similar manner, leading to:

$$\text{Integral 3} = h_b S_{ij} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \frac{1}{3} & \cdot & \frac{1}{6} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \frac{1}{6} & \cdot & \frac{1}{3} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \begin{matrix} \text{row } i \\ \hat{T} \\ \text{row } j \\ \text{column } i \\ \text{column } j \end{matrix} \tag{E6-84}$$

Adding Eqs. (E6-77), (E6-78) and (E6-84) provides the segment boundary vector:

$$\underline{q}_b = \frac{\dot{q}_{s,b}'' S_{ij}}{2} \begin{bmatrix} \cdot \\ 1 \\ \cdot \\ 1 \\ \cdot \end{bmatrix} + \frac{h_b T_{\infty,b} S_{ij}}{2} \begin{bmatrix} \cdot \\ 1 \\ \cdot \\ 1 \\ \cdot \end{bmatrix} - h_b S_{ij} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \frac{1}{3} & \cdot & \frac{1}{6} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \frac{1}{6} & \cdot & \frac{1}{3} & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \begin{matrix} \text{row } i \\ \hat{T} \\ \text{row } j \\ \text{column } i \\ \text{column } j \end{matrix} \tag{E6-85}$$

Note that the segment boundary vector is composed of a column vector,  $\underline{Q}_b$ , and a matrix,  $\underline{H}_b$ , that multiplies the vector of unknown temperatures. Therefore, the global boundary vector is given by:

$$\underline{q} = \underline{Q} - \underline{H}\hat{T} \quad (\text{E6-86})$$

where  $\underline{Q}$  and  $\underline{H}$  are assembled from  $\underline{Q}_b$  and  $\underline{H}_b$  according to:

$$\underline{Q} = \sum_{b=1}^{N_b} \underline{Q}_b \quad (\text{E6-87})$$

and

$$\underline{H} = \sum_{b=1}^{N_b} \underline{H}_b \quad (\text{E6-88})$$

Substituting Eq. (E6-86) into Eq. (E6-37) leads to the matrix equation that must be solved to obtain the solution to the finite element problem:

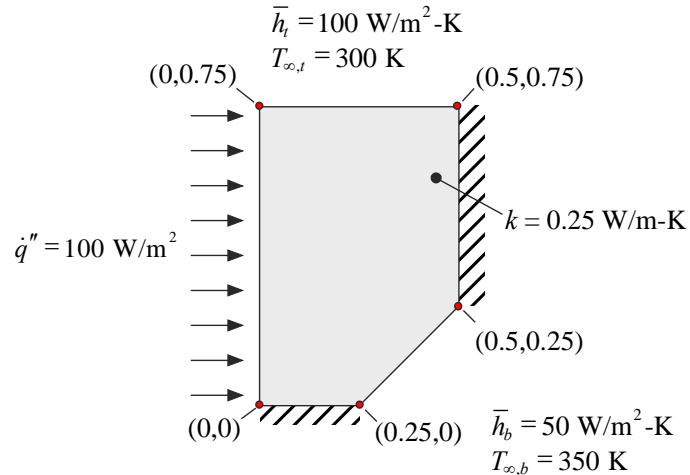
$$\underline{K}\hat{T} = \underline{Q} - \underline{H}\hat{T} \quad (\text{E6-89})$$

or

$$(\underline{K} + \underline{H})\hat{T} = \underline{Q} \quad (\text{E6-90})$$

### ***Implementing the Solution***

The finite element problem with linear triangular elements has been reduced to a bookkeeping problem. The information about the problem (e.g., the conductivity and boundary conditions) and the mesh (e.g., the location of the nodal coordinates, elements, and boundary segments) must be used to construct  $\underline{K}$ ,  $\underline{H}$  and  $\underline{Q}$ . Once  $\underline{K}$ ,  $\underline{H}$  and  $\underline{Q}$  are formed, the matrix equation given by Eq. (E6-90) is solved to obtain  $\hat{T}$ , the temperature at each node. The process is illustrated in the context of the problem shown in Figure E6-7.



**Figure E6-7:** Two-dimensional conduction problem used to illustrate the finite element solution. The coordinates of points are shown in m.

The first step is to define the sub-domains and boundaries that specify the problem. The information about the sub-domains is contained in matrix  $\underline{\underline{sd}}$ . The number of columns in the matrix  $\underline{\underline{sd}}$  is equal to the number of sub-domains in the computational domain. The information about each of the sub-domains is contained in the columns. For a more complex problem, the volumetric generation and other parameters might be required. For the simple problem shown in **Error! Reference source not found.** only the conductivity is required. There is only one sub-domain for the problem, as shown in Figure E6-8, and therefore the matrix  $\underline{\underline{sd}}$  is setup in the MATLAB script S2p7p2\_script according to:

```
clear all;
% setup details of computational domain
sdm=[0.25]'; % conductivity of each sub-domain
```

The information about the boundary conditions is contained in matrix  $\underline{\underline{bc}}$ . The number of columns in the matrix  $\underline{\underline{bc}}$  is equal to the number of boundaries (not boundary elements) in the computational domain. The information about the boundaries is contained in each column: the first row holds the heat transfer coefficient, the second row holds the adjacent fluid temperature, and the third row holds the specified heat flux into the computational domain. There are five boundaries for the problem shown in Figure E6-8 and therefore the matrix  $\underline{\underline{bc}}$  is setup according to:

```
bcm=[0,0,0; 50,350,0; 0,0,0; 100,300,0; 0,0,100]'; % boundary condition for each boundary
```

which can be examined from the MATLAB command window:

```
>> bcm
bcm =
```

E6: Section 2.7.2 *The Galerkin Weighted Residual Method*

0	50	0	100	0
0	350	0	300	0
0	0	0	0	100

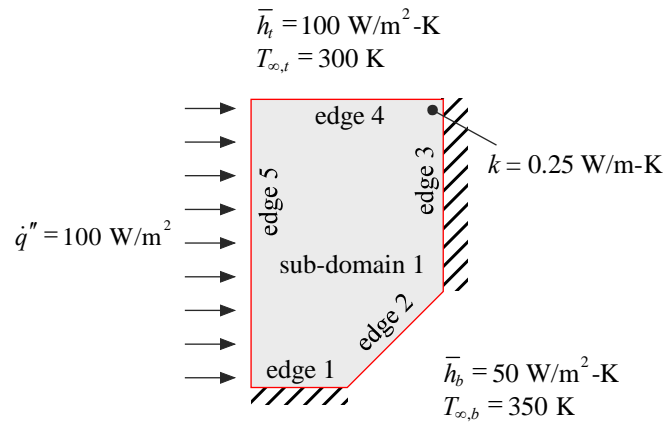


Figure E6-8: Sub-domains and boundaries that define the problem.

The next step is to define a mesh of triangular elements. A very crude mesh is shown in Figure E6-9.

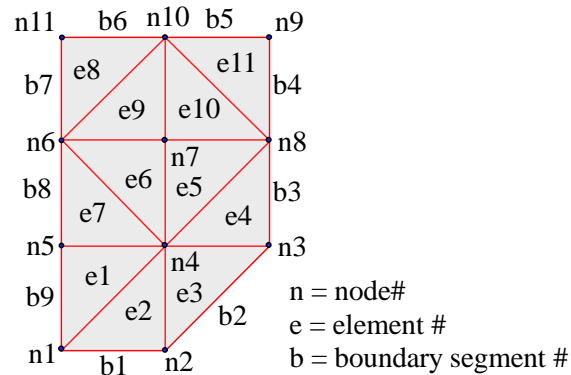


Figure E6-9: A crude mesh.

The mesh shown in Figure E6-9 is defined by  $N_n = 11$  nodes. Information about the nodes is stored in the point matrix,  $\underline{p}$ . There are  $N_n$  columns in the point matrix. The first row of each column contains the  $x$ -coordinate of the corresponding node and the second row contains the  $y$ -coordinate. The point matrix is setup for the grid shown in Figure E6-9 (refer to Figure E6-7 for the dimensions of the problem):

```
% setup details of the grid
N_n=11; % number of nodes
% setup point matrix - coordinates of each node
pm=[0,0; 0.25,0; 0.5,0.25; 0.25,0.25; 0,0.25; 0,0.5; 0.25,0.5; 0.5,0.5; 0.5,0.75; 0.25,0.75; 0,0.75]';
```

which leads to:

```
>> pm
```

```
pm =
    0 0.2500 0.5000 0.2500    0    0    0.2500 0.5000 0.5000 0.2500    0
    0    0 0.2500 0.2500 0.2500 0.5000 0.5000 0.5000 0.7500 0.7500 0.7500
```

The mesh is defined by  $N_b = 9$  boundary segments. Information about the boundary segments is stored in the edge matrix,  $\underline{e}$ . There are  $N_b$  columns in the edge matrix. The first two rows of each column contain the indices of the nodes that define the boundary segment. The third row contains the index of the boundary that the boundary segment lies on (as seen in Figure E6-8). The edge matrix is setup for the mesh shown in Figure E6-9:

```
N_b=9; % number of boundary segments
% setup edge matrix - nodes that define each segment and boundary
em=[1,2,1; 2,3,2; 3,8,3; 8,9,3; 9,10,4; 10,11,4; 11,6,5; 6,5,5; 5,1,5];
```

which leads to:

```
>> em
em =
    1    2    3    8    9   10   11    6    5
    2    3    8    9   10   11    6    5    1
    1    2    3    3    4    4    5    5    5
```

The mesh is defined by  $N_e = 11$  elements. Information about the edge is stored in the triangle matrix,  $\underline{t}$ . There are  $N_e$  columns in the triangle matrix. The first three rows contain the indices of the three nodes that define the elements, given in clockwise order. The fourth row contains the number of the sub-domain that the element lies in. The triangle matrix is setup for the mesh shown in **Error! Reference source not found.**:

```
N_e=11; % number of elements
% setup triangle matrix - nodes that define each element and subdomain
tm=[5,1,4,1; 1,2,4,1; 4,2,3,1; 4,3,8,1; 7,4,8,1; 6,4,7,1; 6,5,4,1; 11,6,10,1; 6,7,10,1; 10,7,8,1; 10,8,9,1];
```

which leads to:

```
>> tm
tm =
    5    1    4    4    7    6    6   11    6   10   10
    1    2    2    3    4    4    5    6    7    7    8
    4    4    3    8    8    7    4   10   10    8    9
    1    1    1    1    1    1    1    1    1    1    1
```

The element conduction matrix is derived in Eq. (E6-66), which is repeated below:



The parameter  $b_{ijk}$  and the area of the element are computed using Eqs. (E6-47) and (E6-46), respectively:

```
bijk=xij*yjk-xjk*yij;
Ae=abs(bijk)/2;           % area of element e
```

The conductivity of the element is obtained from the matrix  $\underline{sd}$  and the elements of the element conduction matrix are added to the global conduction matrix according to Eq. (E6-66).

```
% add element conduction matrix to global conduction matrix
ke=sdm(1,d);           % conductivity of element
K(i,i)=K(i,i)+Ae*ke*(xjk^2+yjk^2)/bijk^2;
K(i,j)=K(i,j)-Ae*ke*(xjk*xik+yjk*yik)/bijk^2;
K(i,k)=K(i,k)+Ae*ke*(xjk*xij+yjk*yij)/bijk^2;
K(j,i)=K(j,i)-Ae*ke*(xik*xjk+yik*yjk)/bijk^2;
K(j,j)=K(j,j)+Ae*ke*(xik^2+yik^2)/bijk^2;
K(j,k)=K(j,k)-Ae*ke*(xik*xij+yik*yij)/bijk^2;
K(k,i)=K(k,i)+Ae*ke*(xij*xjk+yij*yjk)/bijk^2;
K(k,j)=K(k,j)-Ae*ke*(xij*xik+yij*yik)/bijk^2;
K(k,k)=K(k,k)+Ae*ke*(xij^2+yij^2)/bijk^2;
```

end

The vector  $\underline{Q}_b$  and matrix  $\underline{H}_b$  are derived in Eq. (E6-85), which is repeated below:

$$\underline{q}_b = \frac{\hat{q}_{s,b}'' s_{ij}}{2} + \frac{h_b T_{\infty,b} s_{ij}}{2} - h_b s_{ij} \hat{T}$$

(E6-84)

These variables are initialized:

```
Q=zeros(N_n,1);           % global boundary vector initialization
H=zeros(N_n,N_n);        % global convection matrix initialization
```

and constructed boundary segment-by-segment using a for loop:

```
for b=1:N_b
```

## E6: Section 2.7.2 *The Galerkin Weighted Residual Method*

The indices of the two nodes that define the boundary segment and the boundary that the segment belongs to are obtained from the edge matrix:

```
i=em(1,b);           % index of node i
j=em(2,b);           % index of node j
bndry=em(3,b);       % boundary of boundary segment
```

The  $x$ - and  $y$ -distance separating the nodes and the linear distance between the nodes,  $s_{ij}$ , are computed:

```
xij=pm(1,j)-pm(1,i); % x-distance between nodes j and i on boundary segment b
yij=pm(2,j)-pm(2,i); % y-distance between nodes j and i on boundary segment b
sij=sqrt(xij^2+yij^2); % length of boundary segment b
```

The specified heat flux, heat transfer coefficient, and ambient temperature associated with the boundary segment is obtained from the matrix  $\underline{bc}$  and the elements of  $\underline{Q}_b$  and  $\underline{H}_b$  are added to the global vector  $\underline{Q}$  and matrix  $\underline{H}$  according to Eq. (E6-85).

```
qfsb=bcm(3,bndry);   % specified heat flux (W/m^2)
hb=bcm(1,bndry);     % heat transfer coefficient (W/m^2-K)
Tinfb=bcm(2,bndry);  % ambient temperature (K)

% add segment boundary vector to global boundary vector
Q(i,1)=Q(i,1)+qfsb*sij/2+hb*Tinfb*sij/2;
Q(j,1)=Q(j,1)+qfsb*sij/2+hb*Tinfb*sij/2;

% add segment convection matrix to global convective matrix
H(i,i)=H(i,i)+hb*sij/3;
H(i,j)=H(i,j)+hb*sij/6;
H(j,i)=H(j,i)+hb*sij/6;
H(j,j)=H(j,j)+hb*sij/3;
end
```

The temperatures at each node are obtained from Eq. (E6-90):

```
T=(K+H)\Q;           % obtain temperatures at each node
```

which leads to:

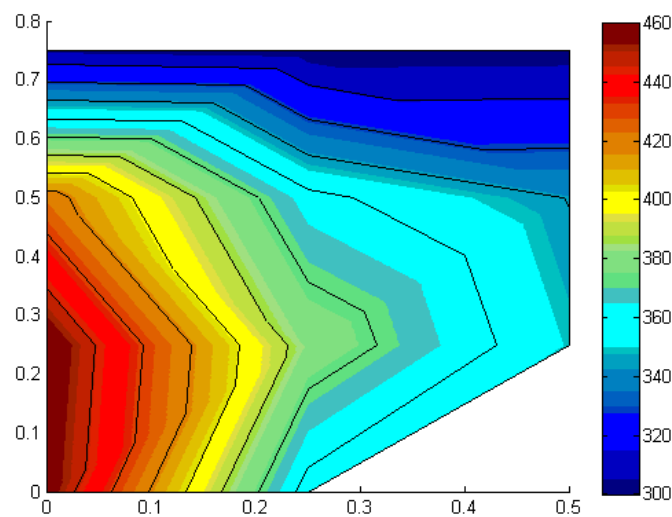
```
>> T
T =
456.9587
353.6660
349.3872
381.2069
460.2514
421.6331
361.5230
343.2677
```

```
300.6376
299.9842
303.2351
```

The organization of the mesh information is nearly the same as is used by the PDE toolbox in MATLAB. Therefore, the tools and commands that are available in the PDE toolbox can be used to manipulate and investigate this solution. If your version of MATLAB includes the PDE toolbox, then the solution can be plotted using the `pdeplot` command:

```
pdeplot(pm,em,tm,'xydata',T,'contour','on');
```

which leads to Figure E6-10.



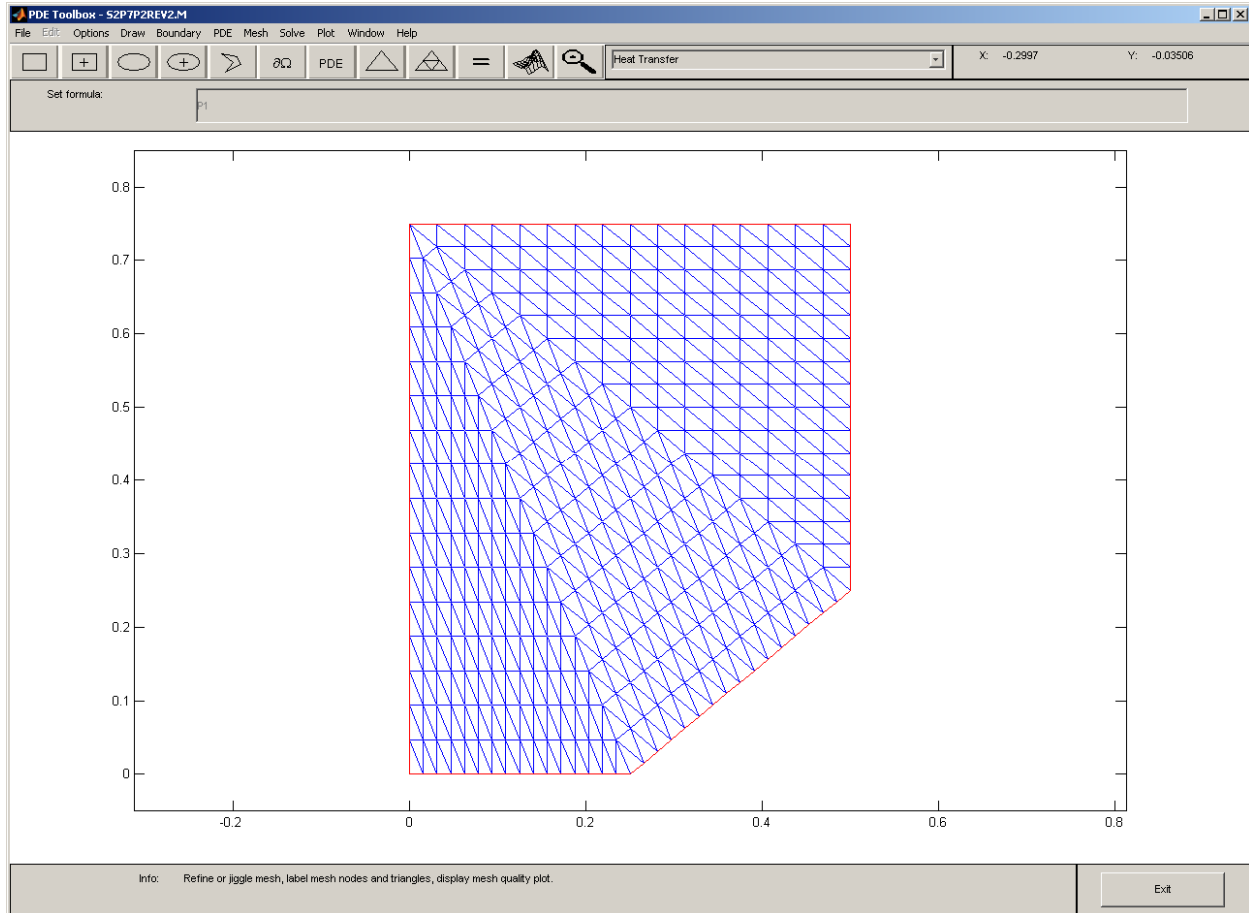
**Figure E6-10: Contour plot of the finite element solution.**

The PDE toolbox has a graphical user interface that can be used to quickly setup a computational domain and generate a mesh. From the command window, type:

```
>> pdetool
```

to open the graphical user interface. Specify the computational domain shown in Figure E6-7 using the polygon tool. Set the application to Heat Transfer from the Options menu. Select Initialize Mesh and Refine Mesh from the Mesh menu (Figure E6-11).

## E6: Section 2.7.2 *The Galerkin Weighted Residual Method*



**Figure E6-11: Mesh generated by the PDE toolbox.**

Select Export Mesh from the Mesh menu and the point matrix, edge matrix, and triangle matrix associated with the mesh generated by the PDE toolbox are exported to the command space. The refined mesh can be used in place of the coarse mesh that was generated manually by commenting out the specified values of  $\underline{p}$ ,  $\underline{e}$ , and  $\underline{t}$ .

```
%clear all;

% setup details of computational domain
sdm=[0.25]'; % conductivity of each sub-domain
bcm=[0,0,0; 50,350,0; 0,0,0; 100,300,0; 0,0,100]'; % boundary condition for each boundary

%% %% setup details of the grid
% N_n=11; % number of nodes
%% %% setup point matrix - coordinates of each node
% pm=[0,0; 0.25,0; 0.5,0.25; 0.25,0.25; 0,0.25; 0,0.5; 0.25,0.5; 0.5,0.5; 0.5,0.75; 0.25,0.75; 0,0.75]';
%
% N_b=9; % number of boundary segments
%% %% setup edge matrix - nodes that define each segment and boundary
% em=[1,2,1; 2,3,2; 3,8,3; 8,9,3; 9,10,4; 10,11,4; 11,6,5; 6,5,5; 5,1,5]';
%
% N_e=11; % number of elements
%% %% setup triangle matrix - nodes that define each element and subdomain
% tm=[5,1,4,1; 1,2,4,1; 4,2,3,1; 4,3,8,1; 7,4,8,1; 6,4,7,1; 6,5,4,1; 11,6,10,1; 6,7,10,1; 10,7,8,1; 10,8,9,1]';
```

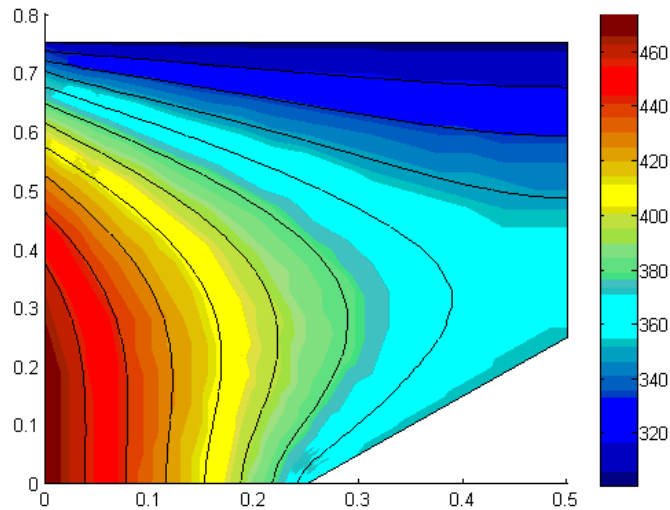
The structure of the point matrix and triangle matrix is identical to the ones that were generated manually:

```
[g,N_n]=size(p);           % number of nodes
pm=p;                     % point matrix is equal to the one produced by pdetool
[g,N_e]=size(t);         % number of elements
tm=t;                    % triangle matrix is equal to the one produced by pdetool
```

The first two rows of the edge matrix generated by MATLAB's PDE toolbox are identical to ours. The index of the boundary is contained in row five of the edge matrix generated by the PDE toolbox:

```
[g,N_b]=size(e);         % number of boundary segments
em=zeros(3,N_b);        % initialize edge matrix
em(1,:)=e(1,:);        % index of node i
em(2,:)=e(2,:);        % index of node j
em(3,:)=e(5,:);        % boundary
```

Otherwise, the script S2p7p2\_script remains unchanged. The solution generated with the more refined mesh is shown in Figure E6-12.



**Figure E6-12:** Contour plot of the finite element solution with the refined mesh generated by the PDE toolbox.