## 4.1 The Finite Element Method

Consider the PDE

$$L[u(\vec{r})] = f(\vec{r}), \tag{4.1}$$

defined in a domain $\Omega$, where $L[\cdot]$ represents a linear differential operator, $u(\vec{r})$ is the unknown function to be determined, and $f(\vec{r})$ is a given source function. The finite element method consists in discretizing the continuum problem (4.1), so that an approximate solution can be found by solving an algebraic system of equations.

Two main approaches for obtaining the approximate solution are the Ritz method and Galerkin's method [152]. The Ritz method is based on a variational formulation of the PDE, which corresponds to a minimization problem of a functional [152]. Since Galerkin's method allows for more general variational formulations [152], Galerkin's approach is used throughout this work.

Subsections

## 4.1.1 Galerkin's Method

Multiplying (4.1) by a function $v(\vec{r})$, which is called test or trial function, and integrating over the simulation domain gives the variational formulation

$$\int_\Omega v(\vec{r})L[u(\vec{r})]d\Omega = \int_\Omega v(\vec{r})f(\vec{r})d\Omega. \tag{4.2}$$

Using the notation

$$(a,b) = \int_\Omega a(\vec{r})b(\vec{r})d\Omega, \tag{4.3}$$

(4.2) can be written as

$$(L[u],v) = (f,v)\,. \tag{4.4}$$

In order to obtain the corresponding discrete problem, the simulation domain, , is divided in a set of $m$ elements, $T_1, T_2, ..., T_m$, which do not overlap, i.e. $\forall i \neq j : T_i \cap T_j = 0$. The mesh obtained by such a domain discretization is represented by

$$T_h(\Omega) = \bigcup_{i=1}^{m} T_i. \tag{4.5}$$

Further, one defines a set $P$ of grid points, also called nodes, with each point $p_k \in P$ being described by a unique global index $k = 1, 2, ..., N$, where $N$ is the total number of grid points in the mesh.

The approximate solution, $u_h(\vec{r})$, for the unknown function, $u(\vec{r})$, is given by [152]

$$u_h(\vec{r}) = \sum_{i=1}^{N} u_i N_i(\vec{r}), \tag{4.6}$$

where $N_i(\vec{r})$ are the so-called basis (or shape) functions. The approximate solution of (4.4) is determined by the coefficients $u_i$, which represent the value of the unknown function at the node $i$. At the node $i$, where the point is given by the coordinates $\vec{r}_i$, the basis functions must satisfy the condition

$$N_j(\vec{r}_i) = \delta_{ij}, \qquad i, j = 1, ..., N. \tag{4.7}$$

Typically, the basis functions are chosen to be low order polynomials. Substituting (4.6) in (4.4), and choosing $v = N_j(\vec{r})$ one obtains

$$\left( L[\sum_{i=1}^{N} u_i N_i], N_j \right) = (f, N_j), \qquad j = 1, ..., N, \tag{4.8}$$

and since $L[\cdot]$ is a linear operator and the coefficients $u_i$ are constants one can write

$$\sum_{i=1}^{N} u_i (L[N_i], N_j) = (f, N_j), \qquad j = 1, ..., N. \tag{4.9}$$

Equation (4.9) is, in fact, a linear system of $N$ equations with $N$ unknowns, $u_1, u_2, ..., u_N$. Thus, it can be written in matrix notation as

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{4.10}$$

where $\mathbf{A} = (a_{ij})$ is called stiffness matrix, given by the elements

$$a_{ij} = (L[N_i], N_j) = \int_{\Omega} L[N_i(\vec{r})] N_j(\vec{r}) d\Omega, \qquad i, j = 1, ..., N, \tag{4.11}$$

$\mathbf{x} = (u_1, ..., u_N)^T$ is the vector of unknown coefficients, and $\mathbf{b} = (b_1, ..., b_N)^T$ is the load vector, given by

(4.12)

$$b_j = (f, N_j) = \int_\Omega f(\vec{r}) N_j(\vec{r}) d\Omega, \qquad j = 1, ..., N.$$

### 4.1.2 Assembly

Applying the finite element method to solve a given PDE leads to an algebraic system of equations. In order to solve this system of equations, the global stiffness matrix, $\mathbf{A}$, and the load vector, $\mathbf{b}$, have to be determined. However, instead of computing them using directly (4.11) and (4.12), in practice they are computed by summing the contributions from the different elements [152, 153, 154] according to

$$a_{ij} = \sum_{T\in \mathcal{T}_h(\Omega)} (L[N_i], N_j)_T = \sum_{T\in \mathcal{T}_h(\Omega)} \int_T L[N_i(\vec{r})] N_j(\vec{r}) d\Omega, \qquad i,j = 1,...,N \tag{4.13}$$

$$b_j = \sum_{T\in \mathcal{T}_h(\Omega)} (f, N_j)_T = \sum_{T\in \mathcal{T}_h(\Omega)} \int_T f(\vec{r}) N_j(\vec{r}) d\Omega, \qquad j = 1,...,N. \tag{4.14}$$

Note that $(L[N_i], N_j)_T = 0$ unless both $N_i$ and $N_j$ belong to the same element $T$. Thus, the calculations (4.13) and (4.14) can be limited to the nodes of the element $T$, so that $i,j = 1,...,N_V$, where $N_V$ is the number of vertices of the element. In this way, for each element $T \in \mathcal{T}_h(\Omega)$, a $N_V \times N_V$ matrix is obtained, which is called element stiffness or nucleus matrix. Thus, the general system matrix, $\mathbf{A}$, can be computed by first computing the nucleus matrices for each $T \in \mathcal{T}_h(\Omega)$ and then summing the contributions from each element according to (4.13) [152]. The right-hand side vector, $\mathbf{b}$, is computed in the same way. This process of constructing the general system matrix is called assembly [152]. The main advantage of this assembly process is that it greatly simplifies the computation of the system matrix and right-hand side vector, since (4.11) and (4.12) can be easily calculated for each element of the domain discretization.

## 4.1.3 Shape Function

The shape function is the function which interpolates the solution between the discrete values obtained at the mesh nodes. Therefore, appropriate functions have to be used and, as already mentioned, low order polynomials are typically chosen as shape functions. In this work linear shape functions are used.

For three-dimensional finite element simulations it is convenient to discretize the simulation domain using tetrahedrons, as depicted in Figure 4.1. Thus, linear shape functions must be defined for each tetrahedron of the mesh, in order to apply the Galerkin method described in Section 4.1.1.

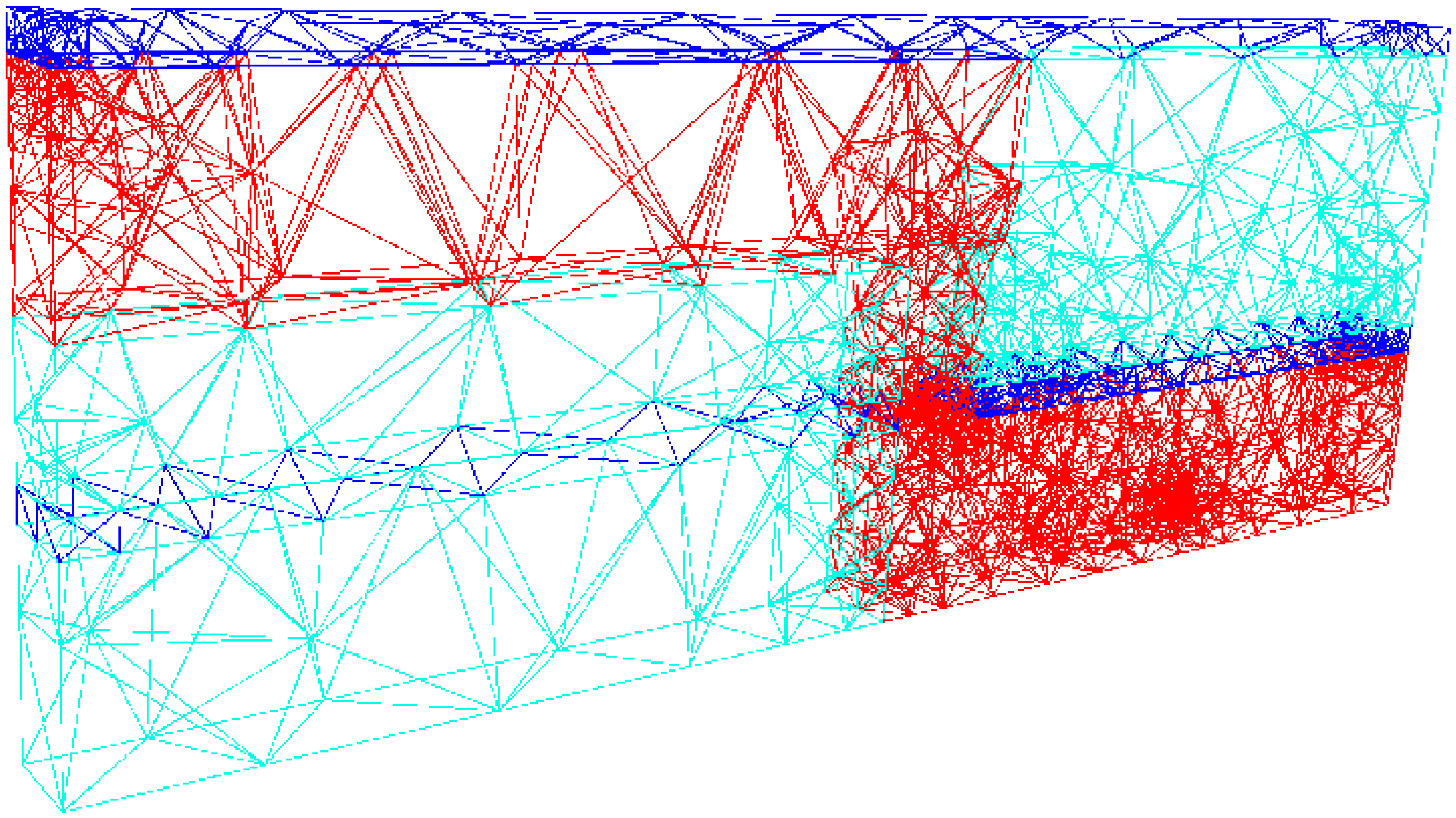


Figure 4.1: Finite element mesh of a three-dimensional interconnect structure discretized with tetrahedrons.

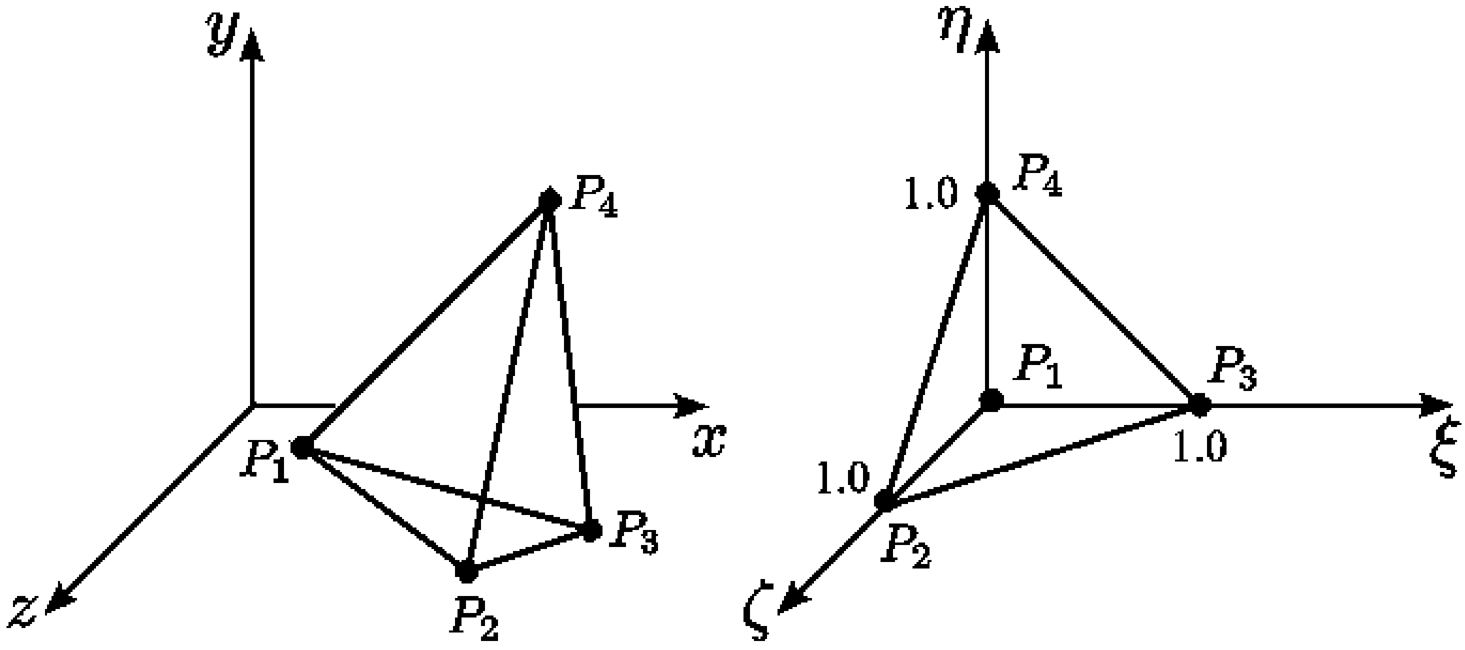
Consider a tetrahedron in a cartesian system as depicted in Figure 4.2(a). The linear shape function of the node $i$ has the form [153]

$$N_i(x,y,z) = a_i + b_i x + c_i y + d_i z, \tag{4.15}$$

where $i = 1,...,4$. The coefficients, $a_i, b_i, c_i,$ and $d_i$ for each nodal basis function of the tetrahedral element can be calculated considering the condition [152]

$$N_j(\vec{r}_i) = \delta_{ij}, \qquad i,j = 1,...,4. \tag{4.16}$$

As a result, a system of $4$ equations for the $4$ unknown coefficients is obtained. This procedure has to be repeated for all tetrahedrons of the mesh, so that the basis functions of all grid nodes are determined. Furthermore, in order to obtain the discrete system of equations (4.9), the shape functions have to be derived and integrated, as shown by (4.11) and (4.12).



(a) (b)

Figure: Tetrahedral finite element. (a) Original coordinate system. (b) Transformed coordinate system.

The calculations can be significantly simplified by carring out a coordinate transformation. A tetrahedron in a transformed coordinate system is shown in Figure 4.2(b). Each point $(x,y,z)$ of the tetrahedron in the original coordinate system can be mapped to a corresponding point $(\xi,\eta,\zeta)$ in the transformed coordinate system [155]

$$\begin{aligned} x &= x_1 + (x_2 - x_1)\xi + (x_3 - x_1)\eta + (x_4 - x_1)\zeta, \\ y &= y_1 + (y_2 - y_1)\xi + (y_3 - y_1)\eta + (y_4 - y_1)\zeta, \\ z &= z_1 + (z_2 - z_1)\xi + (z_3 - z_1)\eta + (z_4 - z_1)\zeta, \end{aligned} \tag{4.17}$$

which in matrix form leads to the Jacobian matrix

$$\mathbf{J} = \begin{bmatrix} x_2 - x_1 & x_3 - x_1 & x_4 - x_1 \\ y_2 - y_1 & y_3 - y_1 & y_4 - y_1 \\ z_2 - z_1 & z_3 - z_1 & z_4 - z_1 \end{bmatrix}. \tag{4.18}$$

In this way, the nodal basis functions for the tetrahedron in the transformed coordinate system are given by [155]

$$\begin{aligned} N_1^t(\xi,\eta,\zeta) &= 1 - \xi - \eta - \zeta, \\ N_2^t(\xi,\eta,\zeta) &= \xi, \\ N_3^t(\xi,\eta,\zeta) &= \eta, \\ N_4^t(\xi,\eta,\zeta) &= \zeta. \end{aligned}$$

These shape functions are rather simple, so that the derivatives and integrals required for the finite element formulation can be readily evaluated in the transformed coordinate system. Given a function $f(x,y,z)$, the gradient in the transformed coordinates is of the form

$$\nabla^t f = \begin{bmatrix} \frac{\partial f}{\partial \xi} & \frac{\partial f}{\partial \eta} & \frac{\partial f}{\partial \zeta} \end{bmatrix}^T, \tag{4.20}$$

where the derivatives are calculated via the chain rule by

$$\begin{aligned} \frac{\partial f}{\partial \xi} &= \frac{\partial f}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial \xi} + \frac{\partial f}{\partial z}\frac{\partial z}{\partial \xi}, \\ \frac{\partial f}{\partial \eta} &= \frac{\partial f}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial \eta} + \frac{\partial f}{\partial z}\frac{\partial z}{\partial \eta}, \\ \frac{\partial f}{\partial \zeta} &= \frac{\partial f}{\partial x}\frac{\partial x}{\partial \zeta} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial \zeta} + \frac{\partial f}{\partial z}\frac{\partial z}{\partial \zeta}. \end{aligned}$$

These equations can be expressed in matrix notation as

$$\begin{bmatrix} \frac{\partial f}{\partial \xi} \\ \frac{\partial f}{\partial \eta} \\ \frac{\partial f}{\partial \zeta} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{bmatrix}, \tag{4.22}$$

or

$$\nabla^t f = \mathbf{J}^T \nabla f, \tag{4.23}$$

where $\mathbf{J}^T$ is the transpose of the Jacobian matrix. Thus, the gradient in the original coordinate system can be calculated using the transformed coordinate gradient by

$$\nabla f = \left(\mathbf{J}^T\right)^{-1} \nabla^t f = \mathbf{\Lambda} \nabla^t f, \tag{4.24}$$

where $\mathbf{\Lambda} = \left(\mathbf{J}^T\right)^{-1}$.

Performing such a coordinate transformation significantly simplifies the practical implementation of the FEM. The nodal shape functions in the transformed coordinates are fixed and known in advance, thus, it is not necessary to solve the system of equations formed by (4.15) and (4.16) for each element of the mesh. Only the Jacobian matrix has to be determined, and the required calculations for the finite element formulation can be easily evaluated.

Next Up Previous Contents
Next: 4.2 Discretization of the Up: 4.1 The Finite Element Previous: 4.1.2 Assembly