the domain into a number of small, simple elements. Then a field quantity is interpolated by a polynomial over an element. Adjacent elements share the degrees-of-freedom at connecting nodes. This process results in a set of simultaneous algebraic equations. Finally we just put all the element equations together, solve the equations, and obtain unknown variables at nodes. The advantage of FEM is that it uses the concept of piecewise polynomial interpolation. By connecting elements together, the field quantity becomes interpolated over the entire structure in piecewise fashion. The disadvantage is that we get only approximate solutions.

In order to have a linear system, the integral algebraic equation of (1) with finite element is established. Consider that it is a 1D equation with the domain from a to b that we project on the basis (test) function b_i (Figure 7) :

$$\int_a^b dx(\vec{\nabla}\cdot\vec{j})b_i = \int_a^b dx q''' b_i \tag{4}$$

where $\vec{j} = -k\vec{\nabla}T$.

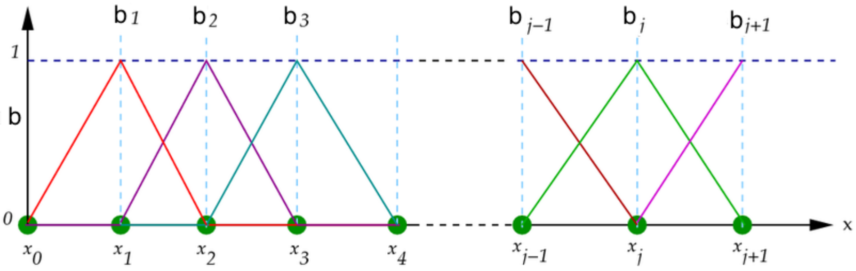

Figure 7: Finite element basis functions

Using the Green formula integration by parts :

$$\int_a^b dx\vec{\nabla}\cdot(b_i\vec{j}) - \int_a^b dx\vec{\nabla}b_i\cdot\vec{j} = \int_a^b dx q''' b_i \tag{5}$$

As $\vec{j} = -k\vec{\nabla}T$:

$$b_i\vec{j}\cdot\vec{n}|_a^b + \int_a^b dxk\vec{\nabla}b_i\cdot\vec{\nabla}T = \int_a^b dx q''' b_i \tag{6}$$

The boundary condition could be applied on the first term $\vec{j}\cdot\vec{n} = -k\vec{\nabla}T\cdot\vec{n}$. For example, $-k\vec{\nabla}T\cdot\vec{n} = g$ if Neumann applies with the given value g.

If we replace $T(x)$ by $\sum_{j=1}^{N} T_j b_j(x)$, which is piecewise linear. We obtain :

$$\sum_{j=1}^{N}(\int_a^b dxk\vec{\nabla}b_i\cdot\vec{\nabla}b_j)T_j = \int_a^b dx q''' b_i - b_i\vec{j}\cdot\vec{n} \tag{7}$$

with $1 \leqslant i \leqslant N$.

We have obtained N equations with N unknowns. Many engineering phenomena can be expressed by "governing equations" and "boundary conditions", so does the heat conduction equation.

Regarding the boundaries, we have :

$$\begin{cases} x = 0(left) \Rightarrow \frac{\partial T}{\partial x} = 0 \\ x = L(right) \Rightarrow hT_{mod} + k\frac{\partial T}{\partial L} = hT_{mod} = \omega \end{cases} \tag{8}$$

where L is the length of the domain and ω is a given value.These could help us deal with left and right boundary conditions imposed by the user (Dirichlet / Neumann / Robin).

As one basis function (e.g. b_1) has two linear sides, only one side (or part) will be used in an element interval. So we can write $b_i(x) = \sum_e b_i^e(x)$, where e stands for element. Each basis function consists of two element functions. So we can have :

$$\int_a^b dxk\vec{\nabla}b_i \cdot \vec{\nabla}b_j = \sum_e \int_e dxk\vec{\nabla}b_i^e \cdot \vec{\nabla}b_j^e \tag{9}$$

because b_i^e and b_j^e are zeros outside the element e.

Finally we have the weak form of the Eq. (1) :

$$\sum_e \sum_{\{j\}\in e} (\int_e dxk\vec{\nabla}b_i^e \cdot \vec{\nabla}b_j^e)T_j = \sum_e (\int_e dxq'''b_i^e - b_i^e\vec{j} \cdot \vec{n}) \tag{10}$$

We note A_{ij} for the content in the left parentheses and B_j for the content in the right parentheses. Thanks to the FEM, we can obtain a set of simultaneous algebraic equations at nodes : $A_{ij} \cdot T_j = B_j$. Every single local matrix AE_{ij} obtained by FEM is 2x2 because each element matrix is valid between two nodes and is supported by two adjacent basis functions. If we have the index i for the row and j for the column, an addition of two element matrixes example is shown in the Figure 8.

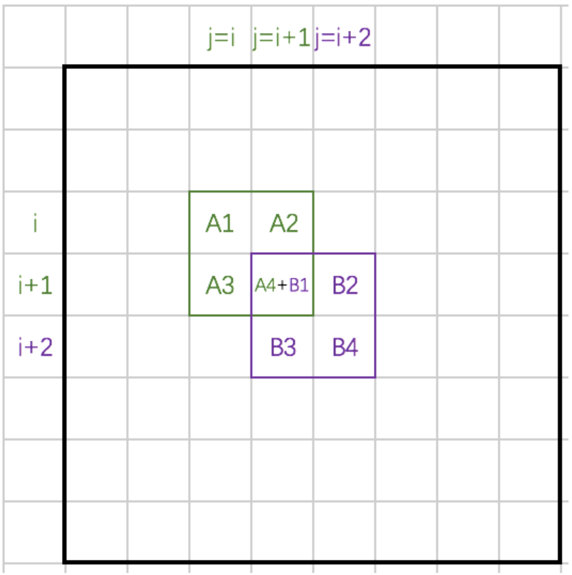

Figure 8: Element matrixes representation

We should assemble all the element matrixes into our big matrix A_{ij} (NxN) by addition in order to solve the system. Finally we can have a linear system in the Figure 9.

$$\begin{pmatrix} A_{11} & \cdots & A_{1j} & A_{1N} \\ \vdots & \ddots & & \vdots \\ A_{j1} & & A_{jj} & A_{jN} \\ A_{N1} & \cdots & A_{Nj} & A_{NN} \end{pmatrix} \begin{pmatrix} T_1 \\ \vdots \\ T_j \\ T_N \end{pmatrix} = \begin{pmatrix} B_1 \\ \vdots \\ B_j \\ B_N \end{pmatrix}$$

Figure 9: Matrix representation of linear system

4.2 Manipulations

We will first write a MATLAB code using FEM to solve numerically the heat conduction in steady state for a single TRIGA rod. The MATLAB code will serve to compare with PDT's heat conduction results when we verify PDT.

The 1D steady-state heat conduction equation in the Cartesian coordinate is as follows :

$$-k\frac{\partial^2 T}{\partial x^2} = q \tag{11}$$

The analytical solution should be a quadratic equation : $T(x) = -\frac{q}{2k}x^2 + c_1 x + c_2$. Only one constant material is used but all kinds of boundary conditions can be invoked. We can solve the two unknown coefficients c_1 and c_2 using the two boundary conditions (left and right) chosen by user (Dirichlet / Neumann / Robin).

Based on a version of a small 1D FEM code for the steady state, 1-group, neutron diffusion equation, I rewrite the code for the heat conduction equation. By applying the FEM for the numerical resolution, we obtain the same results when we compare them with the analytical equations implemented in the MATLAB code for constant conductivity and constant heat source (Figure 10, blue is for the numerical solution and red if for the analytical solution).

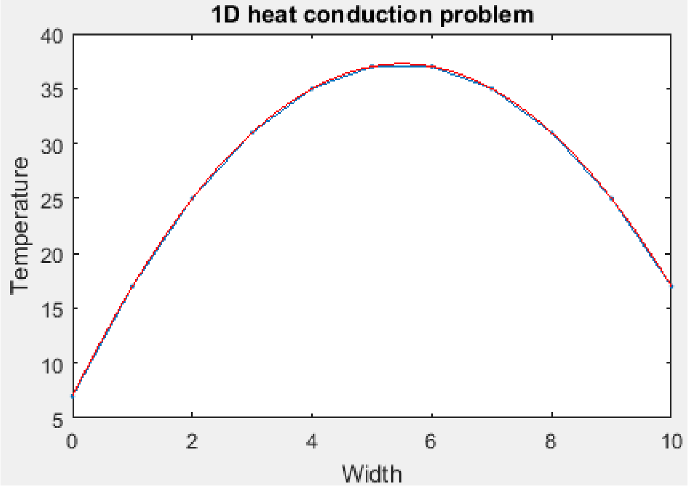

Figure 10: 1D heat conduction problem

The next step is to create several materials to deal with a fuel-like problem (Figure 5), always in the Cartesian coordinate. Each material will use its own conductivity and heat source term. In the FEM assembly process, depending in which material we are, we will need to use the correct conductivity function and heat source term. We should also pay attention to the FEM interval values in different material zones. We just do not want to have a FEM interval that occupies two zones. Now the temperature gap modification could be added. We add this functionality by changing the connectivity array to add 2 temperatures at the same node and by applying these internal boundary conditions after the matrix has been assembled.

In the theory, we give the interface a value even though the gap interface have two sides (left and right) in the connectivity table. In practice, we should give two values to the both sides of the interface for the connectivity. We just open a gap on the interface because there is no connectivity between the two different materials (Figure 11). Thus they become two independent matrixes and we will connect them ("x" in the Figure) by Newton's law of cooling $\varphi = -k\frac{\partial T}{\partial n}|_s = h_{gap}(T_s - T_v)$, where the index s represents surface and v represents volume. The 4 "x" values are simply the value of h_{gap} with a plus or minus sign depending on the directional derivative. The 4 "x" values should be added to the big matrix A.

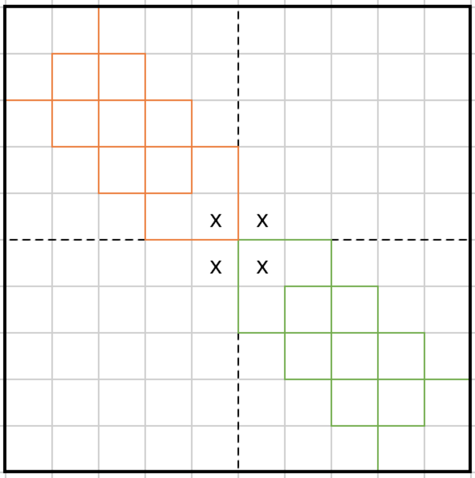

Figure 11: Matrix representation with gap modification

The analytical solution should be adapted to the case of 3 zones. The source exist only in the fuel zone (material 2). Each zone has a solution. In the Zirconium zone (material 1) and the clad zone (material 3), the solution is a simple linear equation with different coefficients : $T_1(x) = c_1x + c_2$ and $T_3(x) = c_5x + c_6$. In the fuel zone, the solution is quadratic but we can obtain the coefficient for the quadratic term : $T_2(x) = -\frac{q}{2k_2}x^2 + c_3x + c_4$. So we have 6 unknowns. We need 6 equations to solve these unknowns. At the left, we may have any boundary condition (Dirichlet, Neumann, or Robin). The same thing for the right side. This will give us 2 equations. Between materials 1 and 2, we have the continuity of the temperature and the heat flux. This will give us 2 more equations. But between materials 2 and 3, we have that special internal condition discussed previously. It specifically is about the temperature being not continuous any more, so we need 2 more conditions that will come from the interface between materials 2 and 3 using the Newton's law of cooling. In the end, we will have 6 equations for 6 unknowns. The 4 of them are fixed, the other two depend on what boundary condition the user will choose on the left