Computer Implementation 1.4 *(Matlab)* *Assembly procedure (p. 25)*

The assembly process is conceptually straight-forward but is clearly tedious and thus error prone when performing computations by hand. Fortunately it is fairly easy to establish a *Mathematica* or *Matlab* based procedure to assemble equations for any finite element model. The process in *Matlab* is illustrated here. The global system is 10×10. We start the process by defining a 10×10 matrix $\boldsymbol{K}$ and a 10×1 vector $\boldsymbol{R}$ using the Table command as follows.

MatlabFiles\Chap1\AssemblyEx.m

The global system is 10×10. We start the process by defining a 10×10 matrix $\boldsymbol{K}$ and a 10×1 vector $\boldsymbol{R}$ using the Table command as follows.

```
K=zeros(10); R = zeros(10,1);
```

Consider assembly of element 1 whose $\boldsymbol{k}$ matrix and $\boldsymbol{r}$ vector are as follows.

```
k = [111,201,301; 201,222,232; 301,232,333]
r = [11; 12; 13]
```

This element contributes to 1, 2 and 5 degrees of freedom. We define a vector lm (element assembly location vector) to indicate the degrees of freedom to which this element contributes.

```
lm = [1, 2, 5]
```

For assembling r into the global R vector, the appropriate locations are those given in the lm vector. Thus we must extracts elements 1, 2 and 5 of the R vector and add the **r** vector to them. The *Matlab* syntax R(lm) accomplishes the task of extracting the required elements. Thus we can assemble r into global R as follows.

```
R(lm) = R(lm) + r
>>
R =

    11
    12
     0
     0
    13
     0
     0
     0
     0
     0
```

For assembling k into the global K, the appropriate locations are combinations of entries given in the lm vector. *Matlab* generates these combinations automatically if arguments for extracting elements are two lists. Thus we can assemble k into global K as follows.

```
K(lm, lm) = K(lm, lm) + k
>>
K =

   111   201     0     0   301     0     0     0     0     0
   201   222     0     0   232     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
   301   232     0     0   333     0     0     0     0     0
```

```
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
```

Now consider the assembly of element 2 whose ***k*** matrix and ***r*** vector are as follows.

```
k = [77,80,90; 80,88,100; 90,100,99]
r = [21; 22; 23]
```

This element contributes to 2, 6 and 5 degrees of freedom and thus we define the lm vector for this element as follows.

```
lm = [2, 6, 5]
```

The assembly of r into global R is as follows.

```
R(lm) = R(lm) + r
>>
R =

    11
    33
     0
     0
    36
    22
     0
     0
     0
     0
```

The assembly of k into global K is as follows.

```
K(lm, lm) = K(lm, lm) + k
>>

K =

   111   201     0     0   301     0     0     0     0     0
   201   299     0     0   322    80     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
   301   322     0     0   432   100     0     0     0     0
     0    80     0     0   100    88     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0     0     0
```

Clearly all elements can easily be assembled using this procedure. As will be illustrated in later examples, the process can be streamlined even further by using the *for* loop function in *Matlab*.