Approximation of functions by piecewise defined trial functions

In a previous chapter ϕ was approximted by

a continuous function defined over the whole domain Ω

$\Omega \in \Re^n$ with $\Gamma \in \Re^{n-1}$ its boundary

$$\phi \cong \hat{\phi} = \psi(x) + \sum_m a_m N_m(x) \quad \text{(completeness requirement)}$$

$\{ N_m ; m = 1,2,3,\cdots \}$ such that $N_m\big|_\Gamma = 0$

$\psi\big|_\Gamma = \phi\big|_\Gamma$

N_m and ψ defined globally over Ω

Main drawback :

treatment of arbitrary geometries

boundary conditions to be imposed

Approximation of functions by piecewise defined trial functions

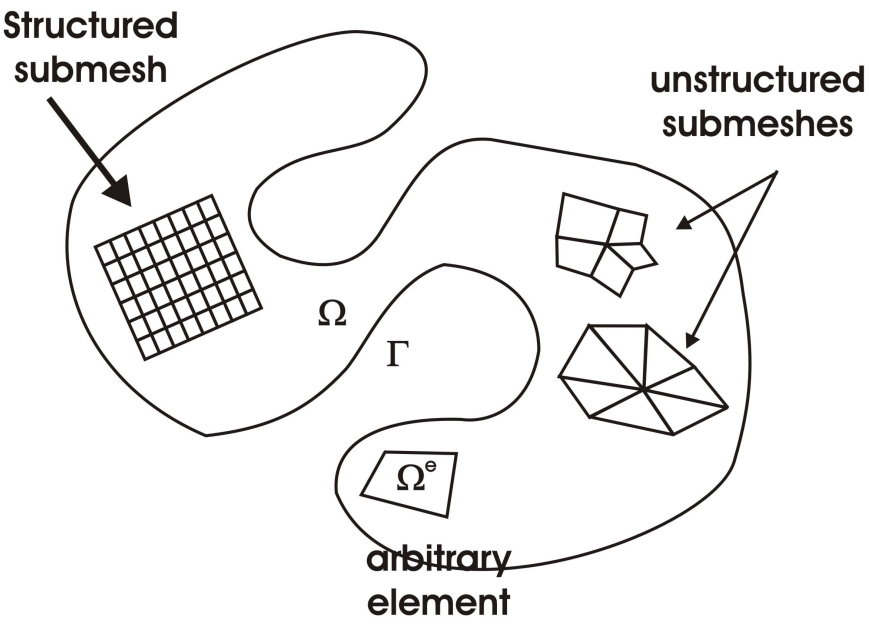
Now $\Omega = \bigcup_e \Omega^e \quad ; \quad \bigcap_e \Omega^e = \{\ \}$ empty set

ϕ is approximated piecewise over each subdomain (locally)

$$\int_\Omega W_l R_\Omega dx = \sum_{e=1}^{E} \int_{\Omega^e} W_l \; R_\Omega \; d\Omega$$

$$\int_\Gamma \overline{W_l} R_\Gamma dx = \sum_{e=1}^{E} \int_{\Gamma^e} \overline{W_l} \; R_\Gamma \; d\Gamma$$

$$\sum_{e=1}^{E} \Omega^e = \Omega \qquad ; \qquad \sum_{e=1}^{E} \Gamma^e = \Gamma$$



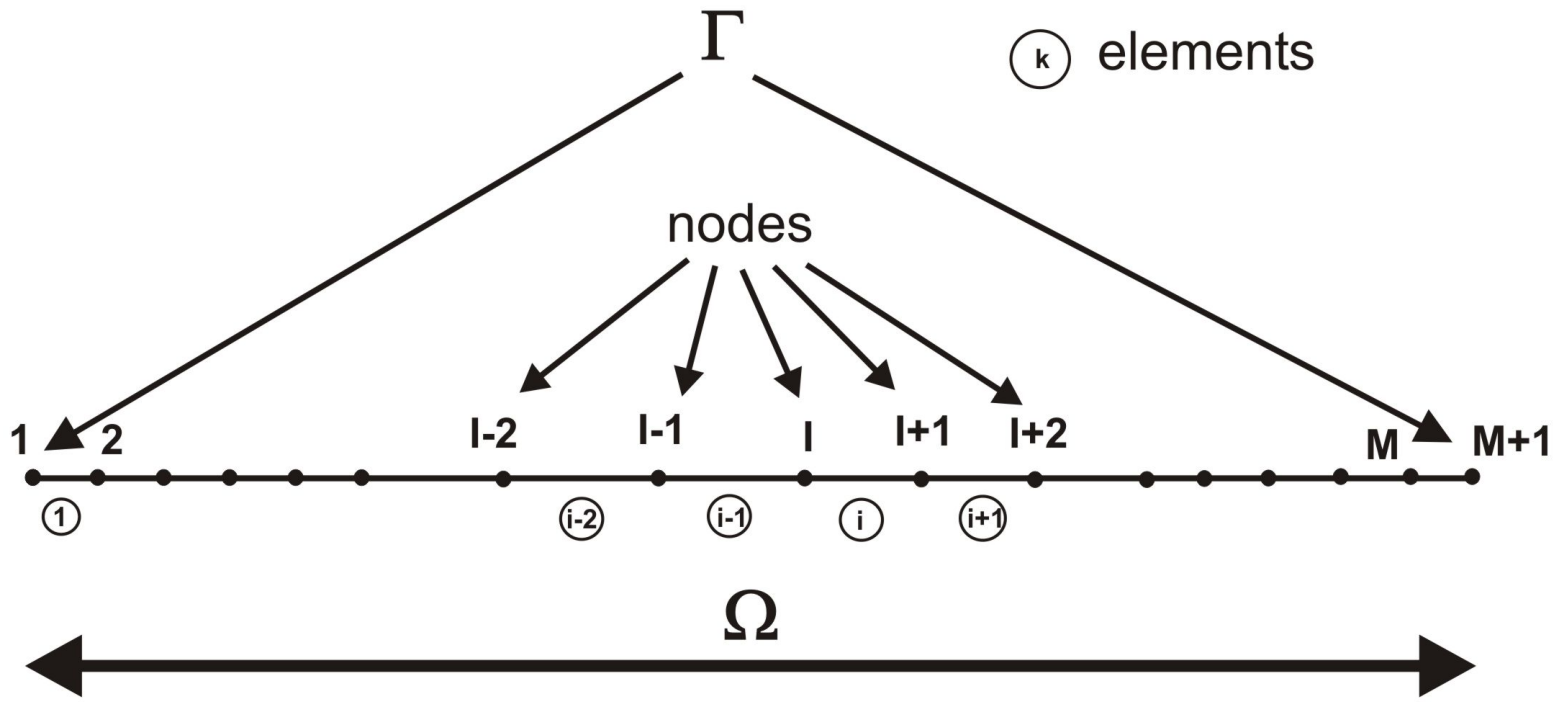
Γ^e boundary of each Ω^e

Ω^e arbitrary, but may be simple enough

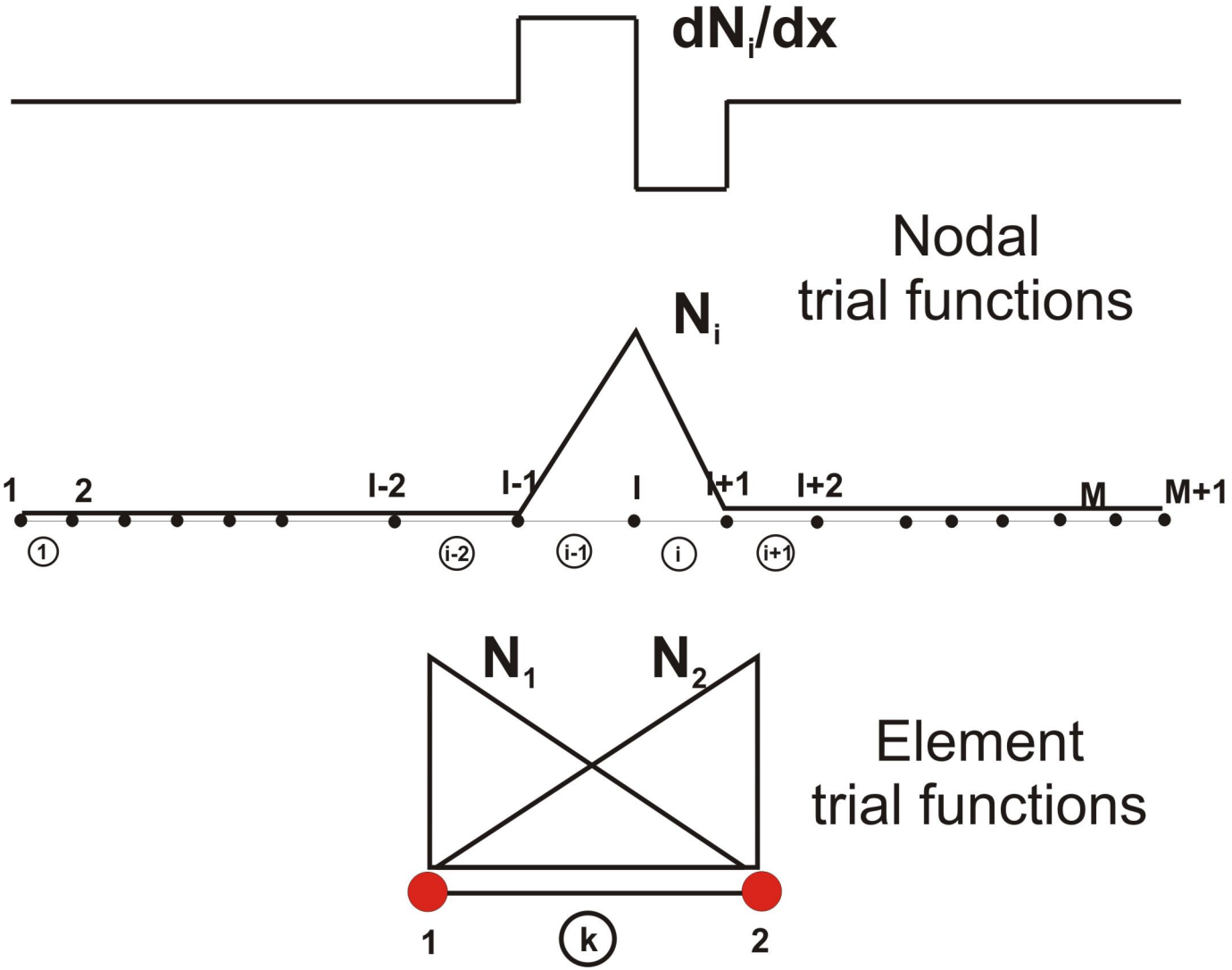
Approximation of functions by piecewise defined trial functions

However piecewise approximation of trial function produces a loss of regularity for the globally assembled functions.

In 1D the elements are intervals



Approximation of functions by piecewise defined trial functions



Some typical locally defined narrow-base shape functions

Approximate a continuous function ϕ defined over $\Omega = \left[0, L_x\right]$ discretized by a set of points $\left\{ x_l ; l = 1,2,\ldots M_n \right\}$ with

$$x_1 = 0; x_{M_n} = L_x$$

$$\Omega = \bigcup_e \Omega^e = \bigcup_{e=1}^{M_n - 1} \left[x_{e-1}, x_e \right]$$

$$\phi \cong \hat{\phi} = \psi + \sum_{m=1}^{M_n - 1} \phi_m N_m(x) \text{ with } \hat{\phi} \text{ piecewise constant}$$

Approximating a given function in 1D

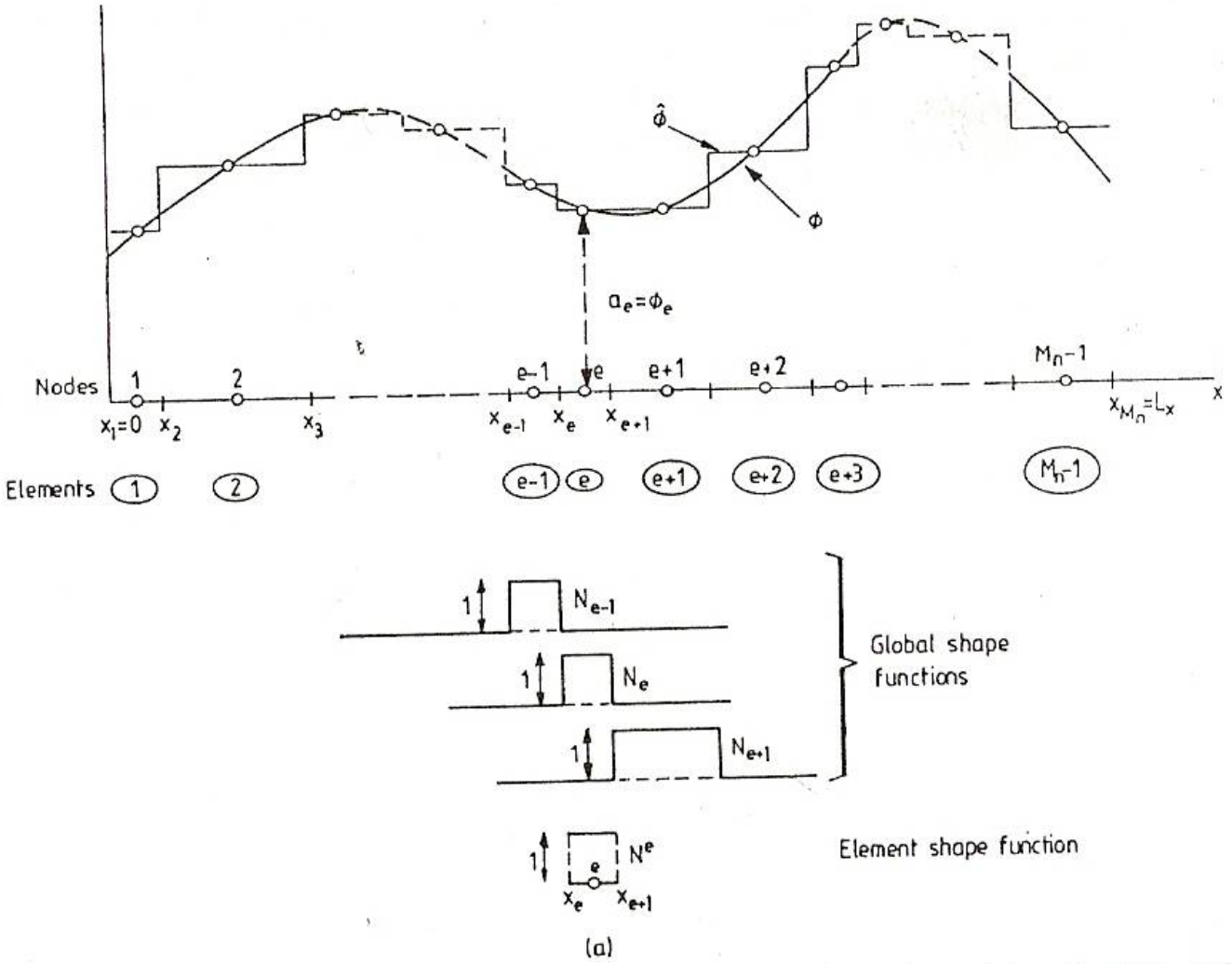


FIGURE 3.1. Approximating a given function in one dimension using point collocation and (*a*) piecewise constant elements and (*b*) piecewise linear elements.

Approximating a given function in 1D

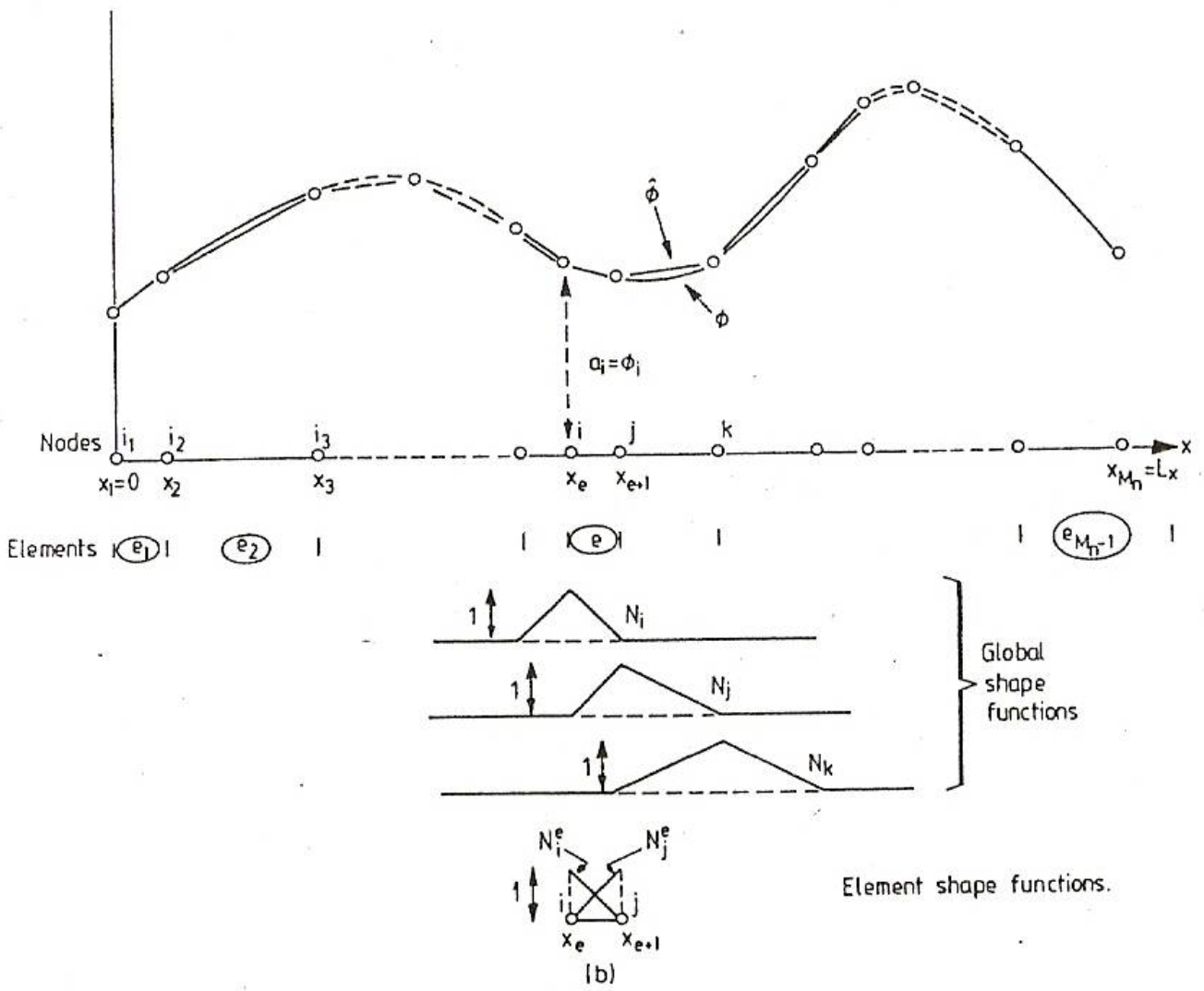


FIGURE 3.1. (*continued*).

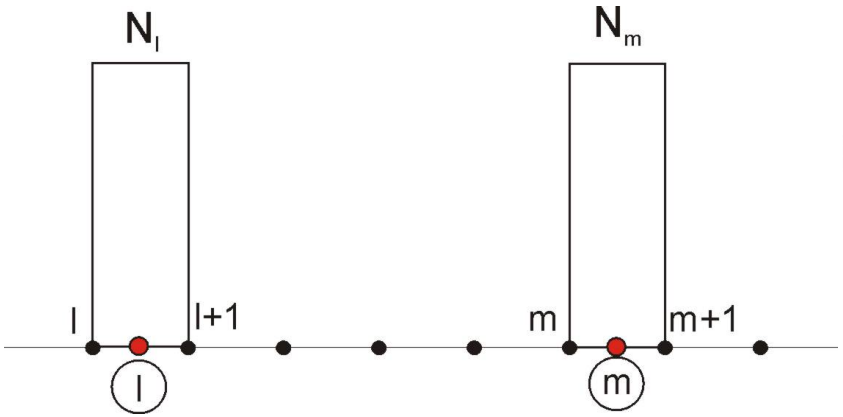
Point collocation & piecewise constant approximation of 1D functions (see Ej_3_1 routine)

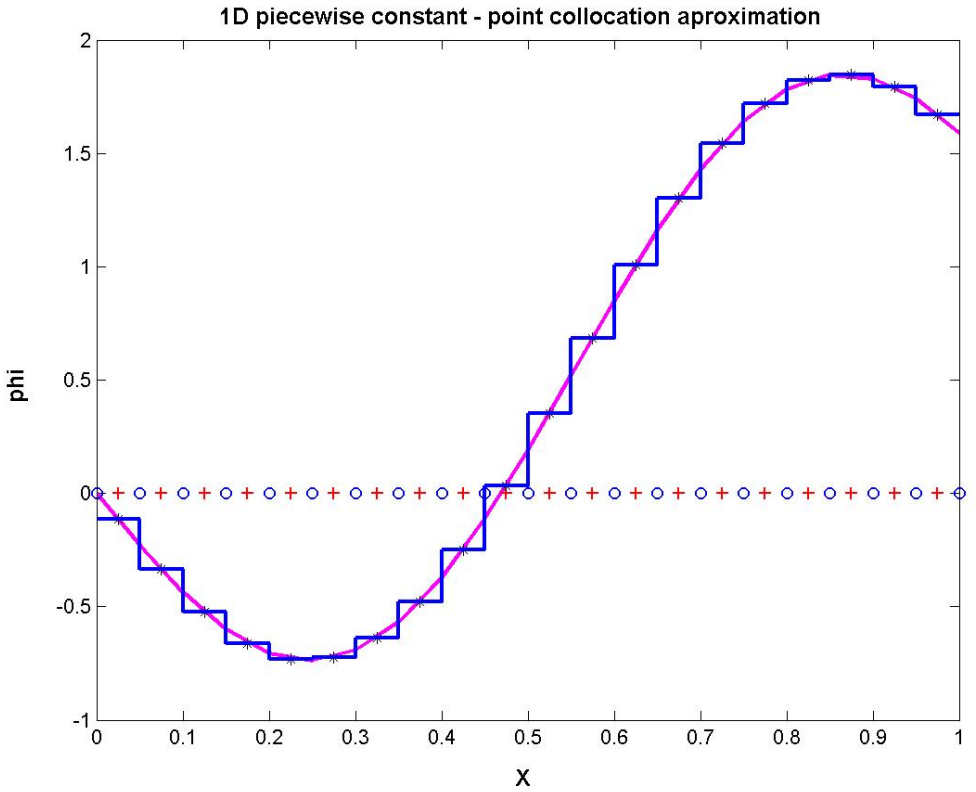
Using Weighted Residual Method (WRM) with $\psi = 0$

$$\int_{\Omega} W_l \, R_{\Omega} \, d\Omega = \int_{\Omega} \delta(x - x_l)\left(\phi - \hat{\phi}\right) d\Omega = \int_{\Omega} \delta(x - x_l)\left(\phi - \sum_{m=1}^{M_n - 1} \phi_m N_m(x)\right) d\Omega = 0$$

$$K_{lm} = \int_{\Omega} \delta(x - x_l) N_m(x) \, d\Omega = N_m(x_l) = \delta_{lm} \quad \text{(by orthonormality of trial functions)}$$

$$f_l = \int_{\Omega} \delta(x - x_l) \, \phi(x) \, d\Omega = \phi(x_l)$$





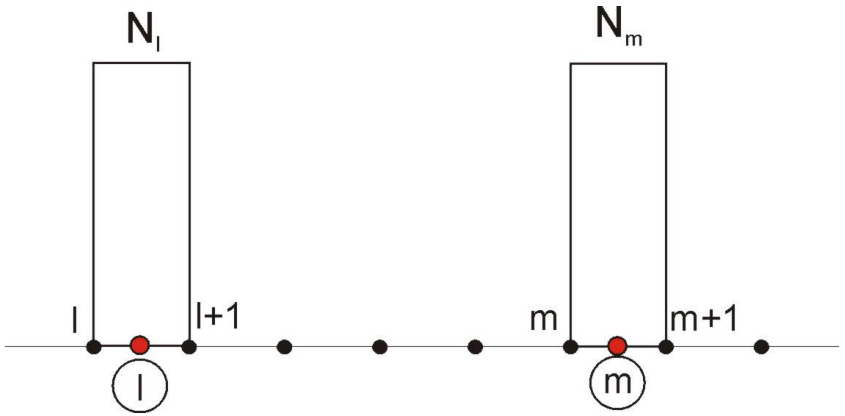
Galerkin & piecewise constant approximation of 1D functions (see Ej_3_1_2 routine)

Using Weighted Residual Method (WRM)

$$\int_{\Omega} W_l \, R_{\Omega} \, d\Omega = \int_{\Omega} W_l \, \left(\phi - \hat{\phi}\right) d\Omega = \int_{\Omega} N_l \left(\phi - \sum_{m=1}^{M_n - 1} \phi_m N_m(x) \right) d\Omega = 0$$

$$K_{lm} = \int_{\Omega} N_l \, N_m \, d\Omega = \delta_{lm} \left| \Omega^m \right| \quad \text{(by orthogonality of trial functions)}$$

$$f_l = \int_{\Omega} N_l \, \phi(x) \, d\Omega = \int_{\Omega^l} \phi(x) \, d\Omega$$



Point collocation

a = [-0.1159 -0.3365 -0.5244 -0.6608 -0.7308 -0.7249 -0.6396 -0.4776 -0.2480 0.0351 0.3531 0.6847 1.0077 1.3002 1.5431 1.7211 1.8239 1.8474 1.7936 1.6709]

Galerkin solution

a = [-0.1154 -0.3351 -0.5223 -0.6580 -0.7276 -0.7215 -0.6363 -0.4748 -0.2458 0.0365 0.3536 0.6844 1.0064 1.2982 1.5404 1.7179 1.8205 1.8441 1.7907 1.6686]

Point collocation & piecewise linear approximation of 1D functions (see Ej_3_1_3 routine)

Using Weighted Residual Method (WRM) with $\psi = 0$

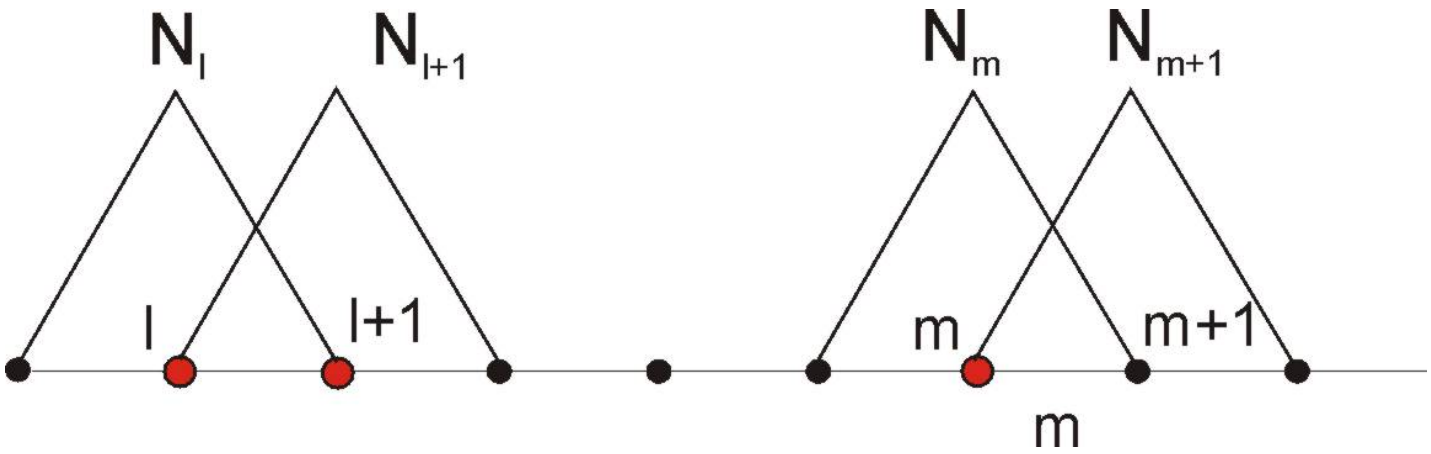
$$\int_{\Omega} W_l \, R_{\Omega} \, d\Omega = \int_{\Omega} \delta(x - x_l)\left(\phi - \hat{\phi}\right) d\Omega = \int_{\Omega} \delta(x - x_l)\left(\phi - \sum_{m=1}^{M_n} \phi_m N_m(x)\right) d\Omega = 0$$

the unknowns are now nodal values instead of element-wise values

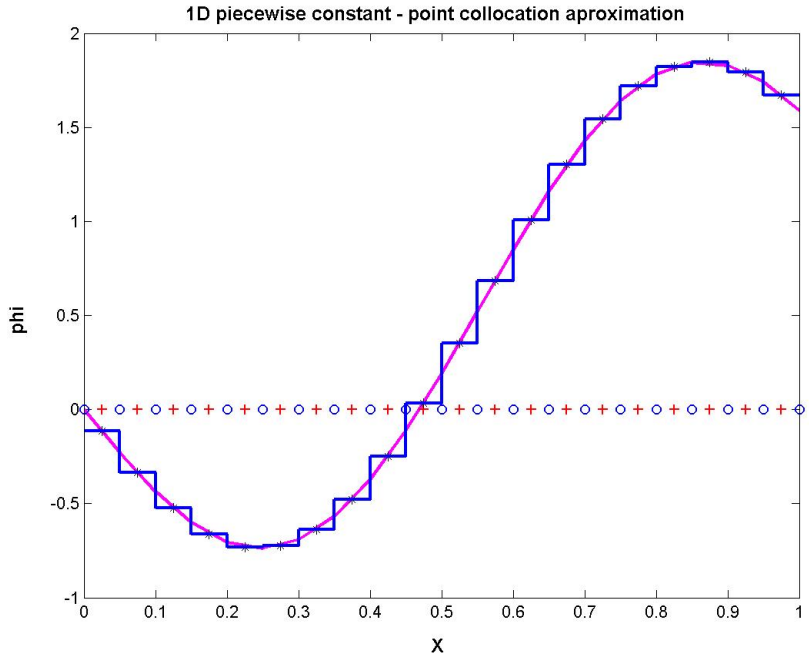
$$K_{lm} = \int_{\Omega} \delta(x - x_l) N_m(x) \, d\Omega = N_m(x_l) = \delta_{lm} \quad \text{(by orthonormality of trial functions)}$$

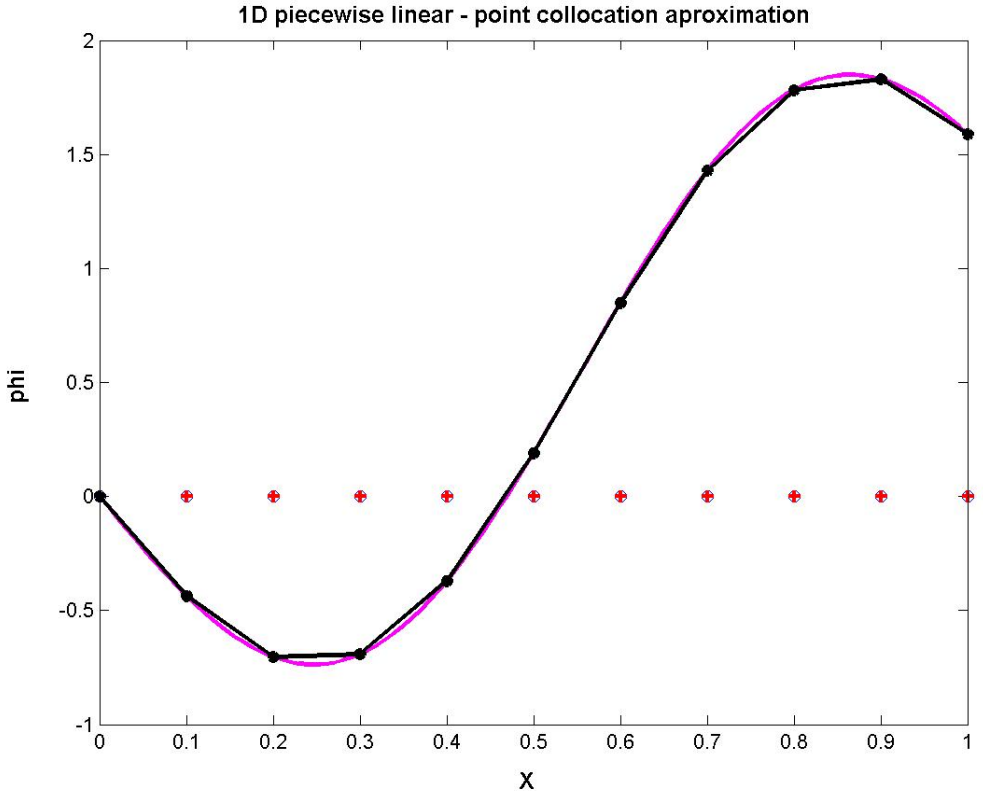
$$f_l = \int_{\Omega} \delta(x - x_l) \, \phi(x) \, d\Omega = \phi(x_l)$$

Ka = f is the same as using point collocation with piecewise constant !



piecewise constant vs linear





Galerkin & piecewise linear approximation of 1D functions

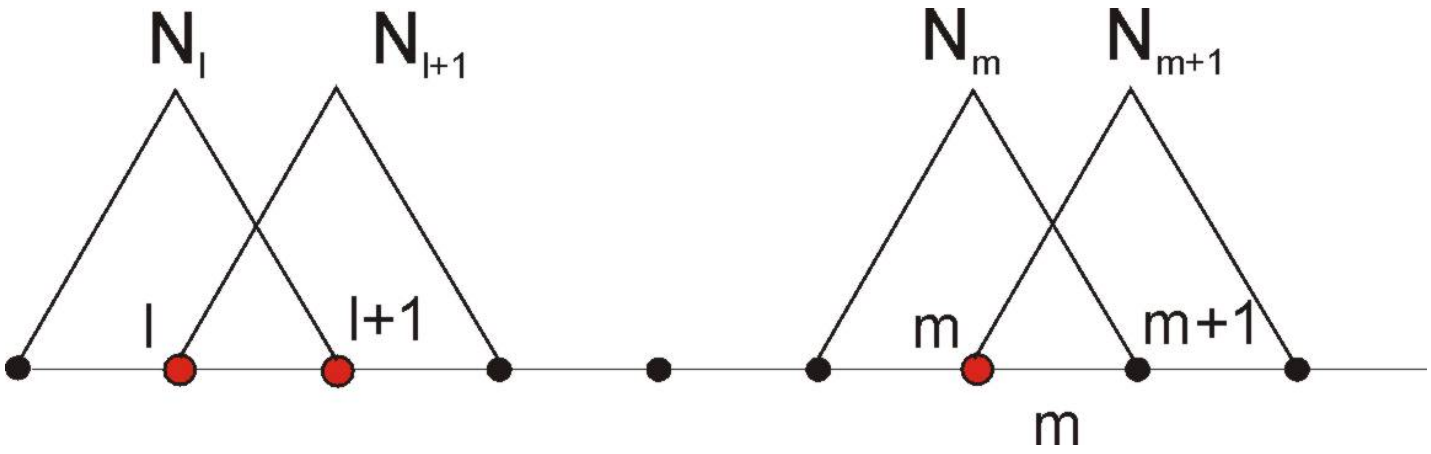
Using Weighted Residual Method (WRM)

$$\int_\Omega W_l \, R_\Omega \, d\Omega = \int_\Omega W_l \, \left(\phi - \hat{\phi}\right) d\Omega = \int_\Omega N_l \, \left(\phi - \sum_{m=1}^{M_n} \phi_m N_m(x) \right) d\Omega = 0$$

$$K_{lm} = \int_\Omega N_l \, N_m \, d\Omega \quad \neq 0 \quad \text{si} \quad |l-m| \leq 1 \quad \text{(for linear (*))}$$

$$f_l = \int_\Omega N_l \, \phi(x) \, d\Omega$$

(*) assuming a natural numbering of nodes in 1D

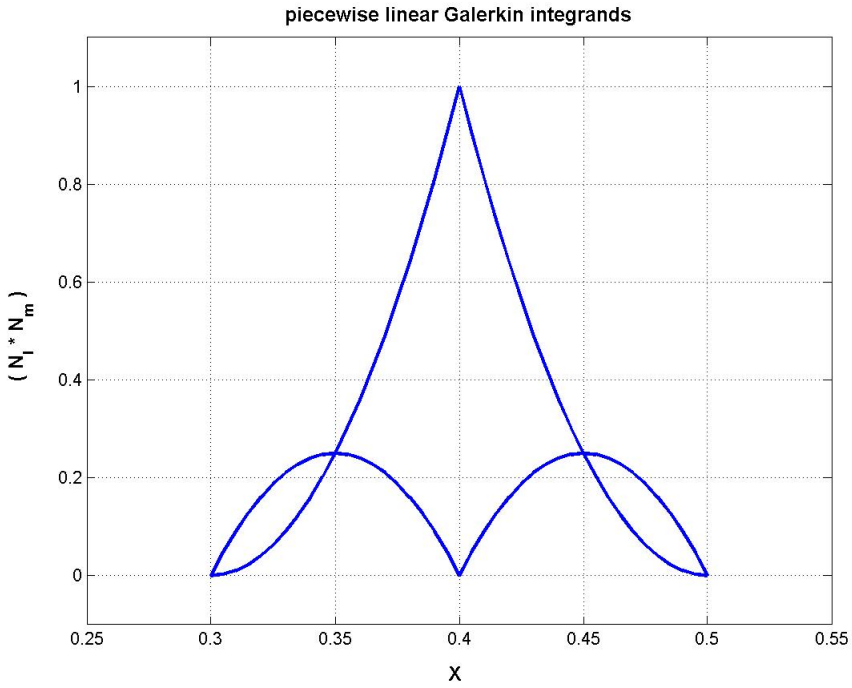


Galerkin & piecewise linear integral kernel for Galerkin

```
K = zeros(Nx+1,Nx+1);
f = zeros(Nx+1,1);

for l=2:Nx
   psi = (0:2)'/2;
   xx  = psi*(x(l)-x(l-1))+x(l-1);
   Na = shape_Ej_3_1_4(xx,x(l-1),x(l));
   Nb = shape_Ej_3_1_4(xx,x(l),x(l-1));
   K(l,l-1) = K(l,l-1) + trapz(xx,Nb.*Na);
   K(l,l) = K(l,l) + trapz(xx,Nb.*Nb);
   xx  = psi*(x(l+1)-x(l))+x(l);
   Na = shape_Ej_3_1_4(xx,x(l+1),x(l));
   Nb = shape_Ej_3_1_4(xx,x(l),x(l+1));
   K(l,l) = K(l,l) + trapz(xx,Na.*Na);
   K(l,l+1) = K(l,l+1) + trapz(xx,Na.*Nb);

   psi = (0:20)'/20;
   xx  = psi*(x(l)-x(l-1))+x(l-1);
   f(l,1) = f(l,1) + trapz(xx,shape_Ej_3_1_4(xx,x(l-1),x(l)).*ffun_Ej_3_1_4(xx));
   xx  = psi*(x(l+1)-x(l))+x(l);
   f(l,1) = f(l,1) + trapz(xx,shape_Ej_3_1_4(xx,x(l+1),x(l)).*ffun_Ej_3_1_4(xx));
end
```



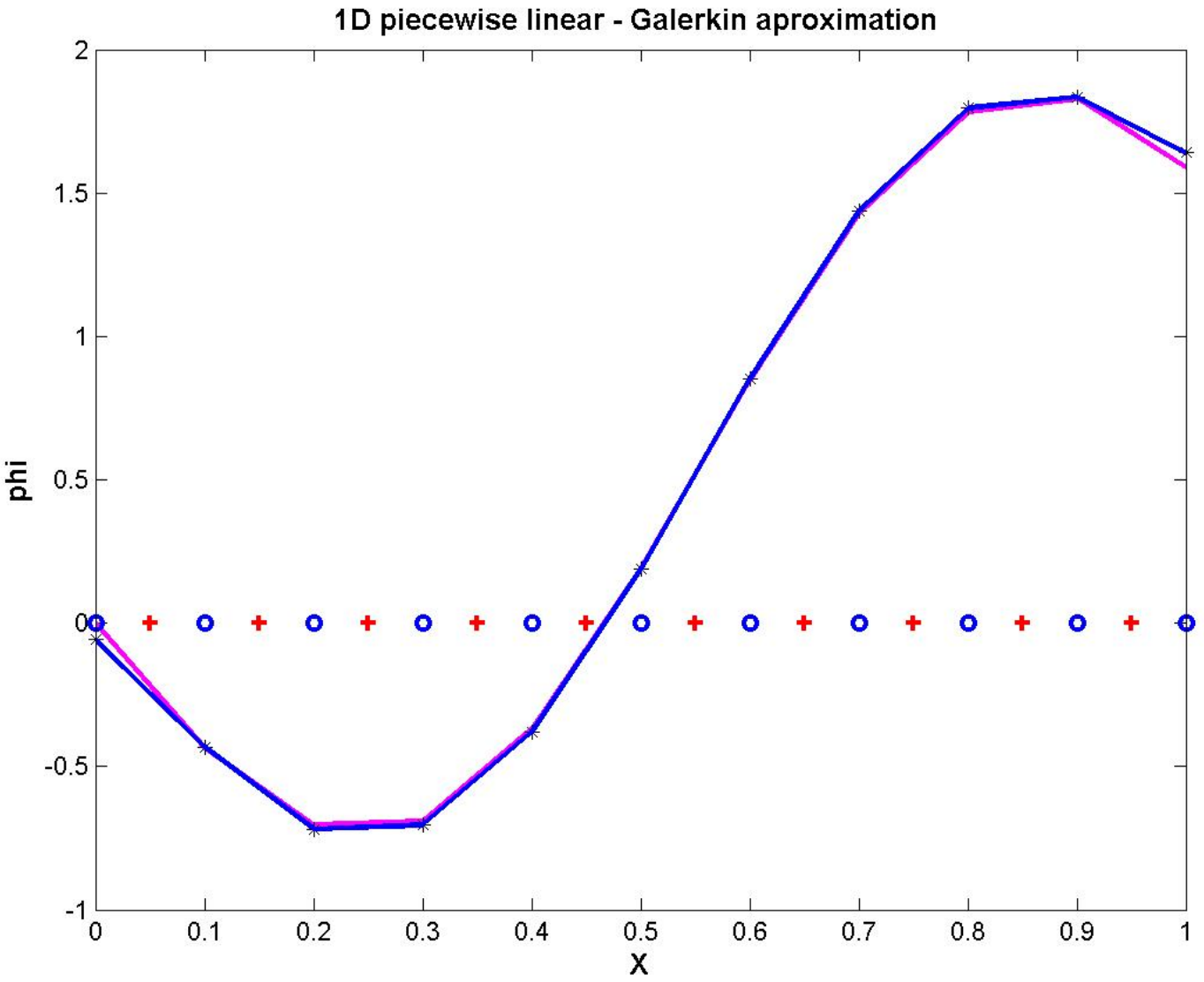
Galerkin & piecewise linear boundary terms

```
l=Nx+1;
psi = (0:2)'/2;
xx  = psi*(x(l)-x(l-1))+x(l-1);
Na = shape_Ej_3_1_4(xx,x(l-1),x(l));
Nb = shape_Ej_3_1_4(xx,x(l),x(l-1));
K(l,l-1) = K(l,l-1) + trapz(xx,Nb.*Na);
K(l,l) = K(l,l) + trapz(xx,Nb.*Nb);

l=1;
xx  = psi*(x(l+1)-x(l))+x(l);
Na = shape_Ej_3_1_4(xx,x(l+1),x(l));
Nb = shape_Ej_3_1_4(xx,x(l),x(l+1));
K(l,l) = K(l,l) + trapz(xx,Na.*Na);
K(l,l+1) = K(l,l+1) + trapz(xx,Na.*Nb);

l=Nx+1;
psi = (0:20)'/20;
xx  = psi*(x(l)-x(l-1))+x(l-1);
f(l,1) = f(l,1) + trapz(xx,shape_Ej_3_1_4(xx,x(l-1),x(l)).*ffun_Ej_3_1_4(xx));
l=1;
xx  = psi*(x(l+1)-x(l))+x(l);
f(l,1) = f(l,1) + trapz(xx,shape_Ej_3_1_4(xx,x(l+1),x(l)).*ffun_Ej_3_1_4(xx));
```

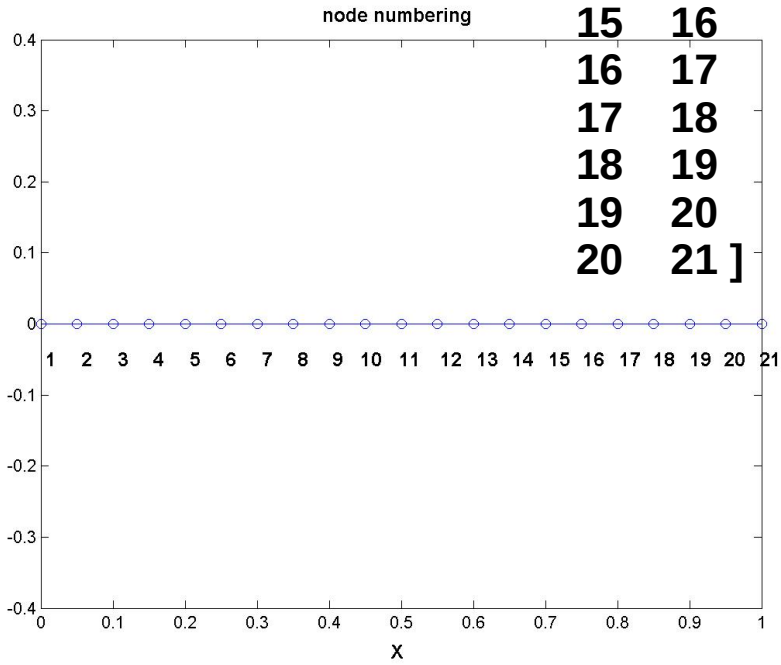
piecewise linear Galerkin results



Galerkin & piecewise linear
Element-wise assembling

```
for k=1:numel
    psi = (0:10)'/10;
    node1 = icone(k,1);
    node2 = icone(k,2);
    xx  = psi*(xnod(node2,1)-xnod(node1,1))+xnod(node1,1);
    Na = shape_Ej_3_1_4(xx,xnod(node2,1),xnod(node1,1));
    Nb = shape_Ej_3_1_4(xx,xnod(node1,1),xnod(node2,1));
    Ke(k,1,1) = Ke(k,1,1) + trapz(xx,Na.*Na);
    Ke(k,1,2) = Ke(k,1,2) + trapz(xx,Na.*Nb);
    Ke(k,2,1) = Ke(k,2,1) + trapz(xx,Nb.*Na);
    Ke(k,2,2) = Ke(k,2,2) + trapz(xx,Nb.*Nb);
    fe(k,1) = fe(k,1) + trapz(xx,Na.*ffun_Ej_3_1_4(xx));
    fe(k,2) = fe(k,2) + trapz(xx,Nb.*ffun_Ej_3_1_4(xx));
end
% gather Ke and fe in Kg and fg
Kg = zeros(Nx+1,Nx+1); fg = zeros(Nx+1,1);
for k=1:numel
    node1 = icone(k,1);
    node2 = icone(k,2);
    Kg(node1,node1)=Kg(node1,node1)+Ke(k,1,1);
    Kg(node1,node2)=Kg(node1,node2)+Ke(k,1,2);
    Kg(node2,node1)=Kg(node2,node1)+Ke(k,2,1);
    Kg(node2,node2)=Kg(node2,node2)+Ke(k,2,2);
    fg(node1,1) = fg(node1,1) + fe(k,1);
    fg(node2,1) = fg(node2,1) + fe(k,2);
end
% solver
a = Kg\fg;
```

```
icone = [ ...
     1     2
     2     3
     3     4
     4     5
     5     6
     6     7
     7     8
     8     9
     9    10
    10    11
    11    12
    12    13
    13    14
    14    15
    15    16
    16    17
    17    18
    18    19
    19    20
    20    21 ]
```



Approximating a given function in 2D

Using piecewise constant on triangles (Fig 3.2 (a))

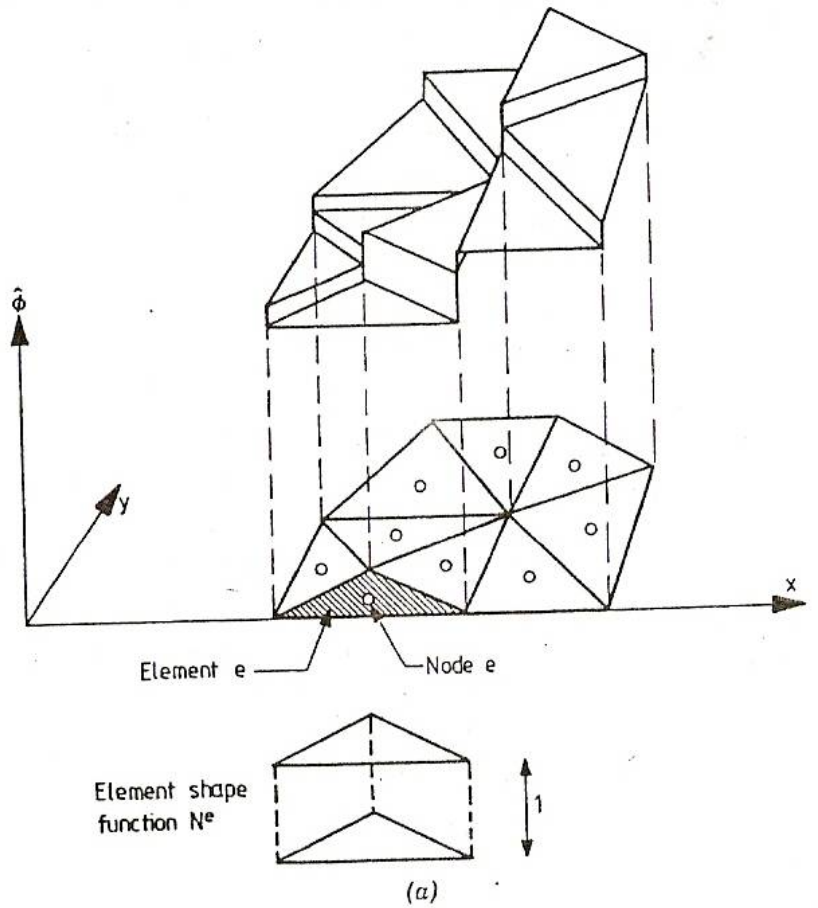
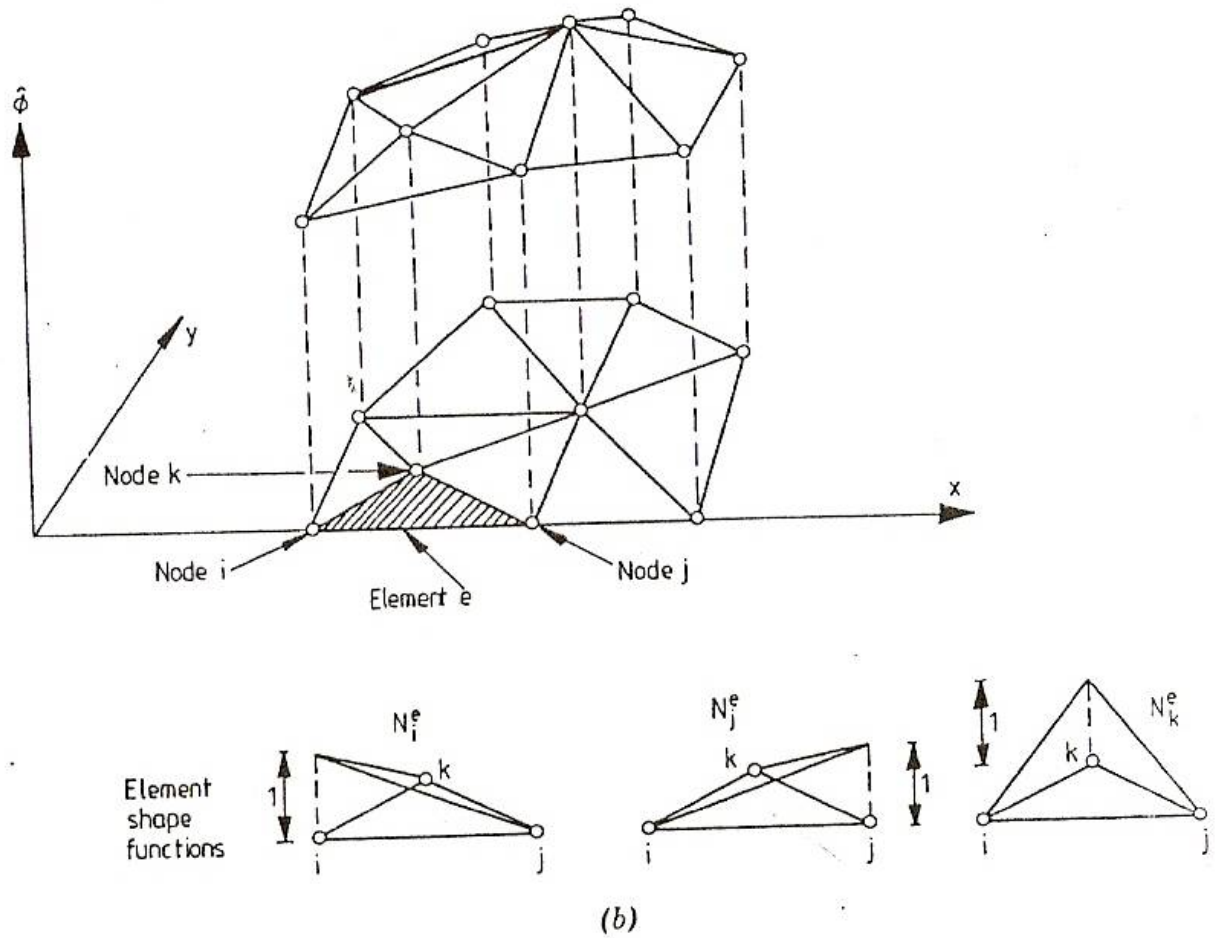


FIGURE 3.2. Approximating a given function in two dimensions using point collocation and (*a*) piecewise constant triangular elements and (*b*) piecewise linear triangular elements.

Approximating a given function in 2D

Using piecewise linear on triangles (Fig 3.2 (b))



(b)

FIGURE 3.2. (*continued*).