Finite Element Method for 1D second order elliptic equation

Xiaoping Zhang

School of Mathematics and Statistics

March 12, 2018

Outline

Target problem

- Solve

$$-\frac{d}{dx}\left(c(x)\frac{du(x)}{dx}\right) = f(x), \quad a < x < b,$$
$$u(a) = g_a, \quad u(b) = g_b.$$

  for $u(x)$.

- Why do we start from this problem?

  An easy look at the basic idea of the finite element method.

Numerical methods for partial differential equations:

- finite element method
- finite difference method
- finite volume method
- boundary element method
- ...

which use different techniques to discretize partial differential equations.

Weak formulation

- First, multiply a function $v(x)$ on both sides of the original equation,

$$-\frac{d}{dx}\left(c(x)\frac{du(x)}{dx}\right) = f(x), \ \ a < x < b$$

$$\Rightarrow \quad -\frac{d}{dx}\left(c(x)\frac{du(x)}{dx}\right) v(x) = f(x)v(x), \ \ a < x < b$$

$$\Rightarrow \quad -\int_a^b \frac{d}{dx}\left(c(x)\frac{du(x)}{dx}\right) v(x)\, dx = \int_a^b f(x)v(x)\, dx.$$

- $u(x)$ is called a trial function and $v(x)$ is called a test function.

Weak formulation

- Second, using integration by parts, we obtain

$$\int_a^b \frac{d}{dx}\left(c(x)\frac{du(x)}{dx}\right) v(x)\, dx$$

$$= \int_a^b \left(cu'\right)' v\, dx$$

$$= \int_a^b v\, d\left(cu'\right)$$

$$= cu'v\Big|_a^b - \int_a^b cu'\, dv$$

$$= c(b)u'(b)v(b) - c(a)u'(a)v(a) - \int_a^b cu'v'\, dx.$$

- Then

$$-c(b)u'(b)v(b) + c(a)u'(a)v(a) + \int_a^b cu'v'\,dx = \int_a^b fv\,dx.$$

- Since the solution at $x = a$ and $x = b$ are given by $u(a) = g_a$, $u(b) = g_b$, then we can choose the test function $v(x)$ such that $v(a) = v(b) = 0$.
- Hence

$$\int_a^b cu'v'\,dx = \int_a^b fv\,dx.$$

- What spaces should u and v belong to? Sobolev spaces!

1D Sobolev spaces

Definition (Support)

If u is a function, then its support $supp(u)$ is the closure of the set on which u is nonzero.

Definition (Compactly supported)

If u is a function defined on an open interval I and $supp(u)$ is a compact subset (that is, a closed and bounded subset), then u is saied to be compactly supported in I.

Lemma

A function compactly supported in an open interval I is zero on and near the boudary of I.

Definition

$C_0^\infty(I)$ is the set of all functions that are infinitely differentiable on I and compactly supported in I.

- Recall integration by parts:

$$\int_a^b u'v\,dx = uv\Big|_a^b - \int_a^b uv'\,dx$$

$$= u(b)v(b) - u(a)v(a) - \int_a^b uv'\,dx.$$

- For $v \in C_0^\infty(I)$, we have $v(a) = v(b) = 0$. Then

$$\int_a^b u'v\,dx = -\int_a^b uv'\,dx.$$

1D Sobolev spaces

Definition (weak derivative)

Suppose u is a real-valued function defined on an open interval $I = (a, b)$ and that u is integrable over every compact subset of I. If there exists another locally integrable function w defined on I such that

$$\int_a^b wv\,dx = -\int_a^b uv'\,dx$$

for all $v \in C_0^\infty(I)$, then u is said to be weakly differentiable and w is called the weak derivative of u.

1D Sobolev spaces

Lemma

If u is differentiable, then u is weakly differentiable and its weak derivative is u'.

Remark

In the Sobolev spaces, which will be defined below, u' is used to represent the weak derivative.

Definition (L^2 space)

$$L^2(I) = \left\{ v : I \to \mathbb{R} : \int_a^b v^2 \, dx < \infty \right\}, \quad I = (a, b).$$

Definition (H^1 space)

$$H^1(I) = \left\{ v \in L^2(I) : v' \in L^2(I) \right\}, \quad I = (a, b).$$

Definition (H_0^1 space)

$$H_0^1(I) = \left\{ v \in H^1(I) : v(a) = v(b) = 0 \right\}, \quad I = (a, b).$$

weak formulation

- Weak formulation: find $u \in H^2(I)$ such that

$$\int_a^b cu'v'\,dx = \int_a^b fv\,dx$$

for any $v \in H_0^1(I)$.

- Let

$$a(u,v) = \int_a^b cu'v'\,dx, \quad (f,v) = \int_a^b fv\,dx.$$

- Weak formulation: find $u \in H^2(I)$ such that

$$a(u,v) = (f,v)$$

for any $v \in H_0^1(I)$.

Galerkin formulation

- Assume there is a finite dimensional subspace $U_h \subset H^1[a, b]$.
- Galerkin formulation: find $u_h \in U_h$ such that

$$a(u_h, v_h) = (f, v_h) \quad \text{i.e.,} \quad \int_a^b c u_h' v_h' \, dx = \int_a^b f v_h \, dx$$

  for any $v_h \in U_h$.
- Basic idea of Galerkin formulation: use finite dimensional space to approximate infinite dimensional space.
- Question: How to obtain U_h?

Mesh

- Assume that we have a uniform partition of $[a, b]$ into N elements with mesh size $h = \frac{b-a}{N}$.
- Let $x_i = a + (i-1)h (i = 1, \cdots, N+1)$ denote the mesh nodes.
- Let $E_n = [x_n, x_{n+1}] (n = 1, \cdots, N)$ denote the mesh element.

1D linear finite element space

- Define 1D linear finite element space:

$$U_h = \{\phi \in C[a, b] : \phi(x) \text{ is linear on each } E_n (n = 1, 2, \cdots, N)\}.$$

- U_h is actually a piecewise linear function space based on the mesh generated before.

1D linear finite element space

Theorem

U_h is an $(N+1)$-dimensional subspace of $C[a,b]$.

Proof:

- First, it is easy to verify that U_h is a subspace of $C[a,b]$.
- If we can find a continuous piecewise linear basis of $N+1$ functions for U_h, then the proof is completed.
- Consider $\phi_j(x) \in U_h$ such that

$$\phi_j(x_i) = \delta_{ij} = \begin{cases} 0, & j \neq i, \\ 1, & j = i \end{cases}$$

for $i, j = 1, \cdots, N+1$.

1D linear finite element space

Continued proof:

- In fact,

$$\phi_1(x) = \begin{cases} \frac{x_2-x}{h}, & x_1 \le x \le x_2, \\ 0, & \text{otherwise}, \end{cases} \tag{1}$$

$$\phi_j(x) = \begin{cases} \frac{x-x_{j-1}}{h}, & x_{j-1} \le x \le x_j, \\ \frac{x_{j+1}-x}{h}, & x_j \le x \le x_{j+1}, \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

$$\phi_{N+1}(x) = \begin{cases} \frac{x-x_N}{h}, & x_N \le x \le x_{N+1}, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

- In order to show that $\phi_j(x)(j = 1, \cdots, N+1)$ form a basis of U_h, we need to show the linear independecne of $\{\phi_j\}_{j=1}^{N+1}$ and $U_h = span\{\phi_j\}_{j=1}^{N+1}$.

1D linear finite element space

Continued proof:
Linear independence: consider

$$\sum_{j=1}^{N+1} c_j \phi_j(x) = 0$$

for any $x \in [a, b]$.

- Let $x = x_i (i = 1, \cdots, N+1)$, then

$$\phi_j(x_i) = \delta_{ij} \quad \Rightarrow \quad c_i = 0 \ (i = 1, \cdots, N+1).$$

- So $\phi_j(x) (j = 1, \cdots, N+1)$ are linearly independent.

1D linear finite element space

Continued proof:

Span: Given any $f \in U_h$, let $c_j = f(x_j)$ and consider

$$g(x) = \sum_{j=1}^{N+1} c_j \phi_j(x) = \sum_{j=1}^{N+1} f(x_j)\phi_j(x).$$

- First, $g(x_i) = c_i = f(x_i)(i = 1, \cdots, N+1)$.
- Second, both $f(x)$ and $g(x)$ are linear in each piece $[x_i, x_{i+1}](i = 1, \cdots, N)$.
- Hence $f(x) = g(x)$ in each piecewise $[x_i, x_{i+1}](i = 1, \cdots, N)$.
- Then $f(x) = g(x) = \sum_{j=1}^{N+1} c_j \phi_j(x)$.
- This implies $U_h = span\{\phi_j\}_{j=1}^{N+1}$.
- Therefore $\phi_j(x)(j = 1, \cdots, N+1)$ form a basis of U_h.

Discretization formulation

- Recall the Galerkin formulation: find $u_h \in U_h$ such that

$$a(u_h, v_h) = (f, v_h) \quad \Leftrightarrow \quad \int_a^b c u_h' v_h' \, dx = \int_a^b f v_h \, dx$$

  for any $v_h \in U_h$.

- Since $u_h \in U_h = span\{\phi_j\}_{j=1}^{N+1}$, then

$$u_h = \sum_{j=1}^{N+1} u_j \phi_j$$

  for some coefficients $u_j (j = 1, \cdots, N+1)$.

Discretization formulation

- If we can set up a linear algebraic system for $u_j (j = 1, \cdots, N+1)$ and solve it, then we can obtain the finite element solution u_h.
- Therefore, we choos the test function $v_h = \phi_i (i = 1, \cdots, N+1)$. Then the finite element formulation gives

$$\int_a^b c \left(\sum_{j=1}^{N+1} u_j \phi_j \right)' \phi_i' \, dx = \int_a^b f \phi_i \, dx, \quad i = 1, \cdots, N+1$$

$$\Rightarrow \quad \sum_{j=1}^{N+1} u_j \left(\int_a^b c \phi_j' \phi_i' \, dx \right) = \int_a^b f \phi_i \, dx, \quad i = 1, \cdots, N+1$$

Discretization formulation

- Define the stiffness matrix

$$\boldsymbol{A} = [a_{ij}]_{i,j=1}^{N+1} = \left[\int_a^b c\phi_j'\phi_i' \, dx \right]_{i,j=1}^{N+1}$$

- Define the load vector

$$\boldsymbol{b} = [b_i]_{i=1}^{N+1} = \left[\int_a^b f\phi_i \, dx \right]_{i=1}^{N+1}.$$

- Define the unknown vector

$$\boldsymbol{x} = [u_j]_{j=1}^{N+1}.$$

- Then we obtain the linear algebraic system

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}.$$

- Here $\boldsymbol{A}$ is symmetric positive-definite if the original elliptic equation is symmetric positive-definite.

Discretization formulation

Remark

- *In fact, since*

$$\phi_j(x_k) = \delta_{jk},$$

then

$$u_h(x_k) = \sum_{j=1}^{N+1} u_j \phi_j(x_k) = u_k.$$

- *Hence the coeffient u_j is actually the numerical solution at the node $x_j (j = 1, \cdots, N+1)$.*
- *Once $\boldsymbol{x} = [u_j]_{j=1}^{N+1}$ is obtained, the finite element solution $u_h = \sum_{j=1}^{N+1} u_j \phi_j$ and the numerical solution at all the mesh nodes are obtained.*

Assembly of the stiffness matrix

- In this section we will first introduce the matrix and vector assembly by using a special method. In the later section "FE method", we will discuss a different universal framework.
- From the definition of $\phi_j (j = 1, \cdots, N+1)$, we can see that ϕ_j are tje non-zero only on the elements adjacent to the node x_j, but 0 on all the other elements.
- This observation motivates us to think about

$$a_{ij} = \int_a^b c\phi_j'\phi_i' \, dx = \sum_{n=1}^{N} \int_{n=1}^{N} \int_{x_n}^{x_{n+1}} c\phi_j'\phi_i' \, dx, \quad i, j = 1, \cdots, N+1.$$

- It is easy to see that most of $\int_{x_n}^{x_{n+1}} c\phi_j'\phi_i' \, dx$ will be 0.
- So we only need to use numerical integration to compute those nonzero integrals.

Assembly of the stiffness matrix

Case 1: when $|i-j|>1$, x_i and x_j are not neighouring mesh nodes.

- On any element $[x_n, x_{n+1}](n=1,\cdots,N)$, at least one of ϕ_j and ϕ_i is 0.
- Hence,

$$\int_{x_n}^{x_{n+1}} c\phi_j'\phi_i'\,dx = 0(n=1,\cdots,N)$$

$$\Rightarrow \quad a_{ij} = \sum_{n=1}^{N} \int_{x_n}^{x_{n+1}} c\phi_j'\phi_i'\,dx = 0.$$

Assembly of the stiffness matrix

Case 2: when $i = j + 1 (j = 1, \cdots, N)$

- the only element, on which both ϕ_j and ϕ_i are not zero, is $[x_j, x_{j+1}]$.
- Hence,

$$\int_{x_n}^{x_{n+1}} c\phi_j'\phi_i' \, dx = 0 (n = 1, \cdots, j-1, j+1, \cdots, N)$$

$$\Rightarrow \quad a_{ij} = \sum_{n=1}^{N} \int_{x_n}^{x_{n+1}} c\phi_j'\phi_i' \, dx = \int_{x_j}^{x_{j+1}} c\phi_j'\phi_i' \, dx$$

$$\Rightarrow \quad a_{j+1,j} = \int_{x_j}^{x_{j+1}} c(x) \left(\frac{x_{j+1} - x}{h}\right)' \left(\frac{x - x_j}{h}\right)' \, dx = -\frac{1}{h^2} \int_{x_j}^{x_{j+1}} c(x) \, dx.$$

Assembly of the stiffness matrix

Case 3: when $i = j-1 (j = 2, \cdots, N+1)$

- the only element, on which both ϕ_j and ϕ_i are not zero, is $[x_{j-1}, x_j]$.
- Hence,

$$\int_{x_n}^{x_{n+1}} c\phi_j'\phi_i' \, dx = 0 (n = 1, \cdots, i-1, i+1, \cdots, N)$$

$$\Rightarrow \quad a_{ij} = \sum_{n=1}^{N} \int_{x_n}^{x_{n+1}} c\phi_j'\phi_i' \, dx = \int_{x_{j-1}}^{x_j} c\phi_j'\phi_i' \, dx$$

$$\Rightarrow \quad a_{j-1,j} = \int_{x_{j-1}}^{x_j} c(x) \left(\frac{x - x_{j-1}}{h}\right)' \left(\frac{x_j - x}{h}\right)' \, dx = -\frac{1}{h^2} \int_{x_{j-1}}^{x_j} c(x) \, dx.$$

Assembly of the stiffness matrix

Case 4: when $i = j (j = 2, \cdots, N)$

- the only elements, on which both ϕ_j and ϕ_i are not zero, are $[x_{j-1}, x_j]$ and $[x_j, x_{j+1}]$.
- Hence,

$$\int_{x_n}^{x_{n+1}} c\phi_j'\phi_i'\,dx = 0 (n = 1, \cdots, j-2, j+1, \cdots, N)$$

$$\Rightarrow \quad a_{ij} = \sum_{n=1}^{N} \int_{x_n}^{x_{n+1}} c\phi_j'\phi_i'\,dx = \int_{x_{j-1}}^{x_j} c\phi_j'\phi_i'\,dx + \int_{x_j}^{x_{j+1}} c\phi_j'\phi_i'\,dx$$

$$\Rightarrow \quad a_{jj} = \int_{x_{j-1}}^{x_j} c(x) \left(\frac{x - x_{j-1}}{h}\right)' \left(\frac{x - x_{j-1}}{h}\right)' dx$$

$$+ \int_{x_j}^{x_{j+1}} c(x) \left(\frac{x_{j+1} - x}{h}\right)' \left(\frac{x_{j+1} - x}{h}\right)' dx$$

$$= \frac{1}{h^2} \int_{x_{j-1}}^{x_j} c(x)\,dx + \frac{1}{h^2} \int_{x_j}^{x_{j+1}} c(x)\,dx.$$

Assembly of the stiffness matrix

Case 5: when $i = j = 1$

- the only element, on which both ϕ_j and ϕ_i are not zero, are $[x_1, x_2]$.
- Hence,

$$\int_{x_n}^{x_{n+1}} c\phi_1'\phi_1'\,dx = 0 (n = 2, \cdots, N)$$

$$\Rightarrow \quad a_{11} = \sum_{n=1}^{N} \int_{x_n}^{x_{n+1}} c\phi_1'\phi_1'\,dx = \int_{x_1}^{x_2} c\phi_1'\phi_1'\,dx$$

$$\Rightarrow \quad a_{jj} = \int_{x_1}^{x_2} c(x) \left(\frac{x_2 - x}{h}\right)' \left(\frac{x_2 - x}{h}\right)'\,dx$$

$$= \frac{1}{h^2} \int_{x_1}^{x_2} c(x)\,dx.$$

Assembly of the stiffness matrix

Case 6: when $i = j = N+1$

- the only elements, on which both ϕ_j and ϕ_i are not zero,is $[x_N, x_{N+1}]$.
- Hence,

$$\int_{x_n}^{x_{n+1}} c\phi'_{N+1}\phi'_{N+1}\,dx = 0 (n = 1, \cdots, j-2, j+1, \cdots, N-1)$$

$$\Rightarrow \quad a_{N+1,N+1} = \sum_{n=1}^{N} \int_{x_n}^{x_{n+1}} c\phi'_{N+1}\phi'_{N+1}\,dx = \int_{x_N}^{x_{N+1}} c\phi'_{N+1}\phi'_{N+1}\,dx$$

$$\Rightarrow \quad a_{N+1,N+1} = \int_{x_N}^{x_{N+1}} c(x) \left(\frac{x-x_N}{h}\right)' \left(\frac{x-x_N}{h}\right)' dx$$

$$= \frac{1}{h^2} \int_{x_N}^{x_{N+1}} c(x)\,dx.$$

Assembly of the stiffness matrix

- From the above discussion, we can see that most of the elements $a_{ij}(i, j = 1, \cdots, N+1)$ are 0.
- Hence the stiffness matrix $\boldsymbol{A}$ is a sparse matrix.
- We can also see that we only need to compute the integrals on local elements instead of the whole domain, which later will lead to the "local assembly" idea of finite elements.

Assembly of the stiffness matrix

Algorithm I

- Initialize the matrix: $\boldsymbol{A} = sparse(N+1, N+1)$;
- Compute the integrals and assemble them into $\boldsymbol{A}$:

FOR $j = 1, \cdots, N+1$:

IF $j \leq N$, THEN

Compute $A(j+1, j) = -\frac{1}{h^2} \int_{x_j}^{x_{j+1}} c(x)\, dx$

END

IF $j \geq 2$, THEN

Compute $A(j-1, j) = -\frac{1}{h^2} \int_{x_{j-1}}^{x_j} c(x)\, dx$

END

IF $2 \leq j \leq N$, THEN

Compute $A(j, j) = \frac{1}{h^2} \int_{x_{j-1}}^{x_{j+1}} c(x)\, dx$

END

END

Compute $A(1,1) = \frac{1}{h^2} \int_{x_1}^{x_2} c(x)\, dx$

Compute $A(N+1, N+1) = \frac{1}{h^2} \int_{x_N}^{x_{N+1}} c(x)\, dx$

Assembly of the load vector

The idea for the assembly of the load vector is similar. We have

$$b_i = \int_a^b f\phi_i \, dx = \sum_{n=1}^{N} \int_{x_n}^{x_{n+1}} f\phi_i \, dx, \, i = 1, \cdots, N+1.$$

- Case 1: when $2 \leq i \leq N$, the only two elements, on which ϕ_i is not zero, are $[x_{i-1}, x_i]$ and $[x_i, x_{i+1}]$. Then

$$\int_{x_n}^{x_{n+1}} f\phi_i \, dx = 0, \quad n = 1, \cdots, i-2, i+1, \cdots, N,$$

Assembly of the load vector

The idea for the assembly of the load vector is similar. We have

$$b_i = \int_a^b f\phi_i \, dx = \sum_{n=1}^{N} \int_{x_n}^{x_{n+1}} f\phi_i \, dx, \, i = 1, \cdots, N+1.$$

- Case 1: when $2 \leq i \leq N$, the only two elements, on which ϕ_i is not zero, are $[x_{i-1}, x_i]$ and $[x_i, x_{i+1}]$. Then

$$\int_{x_n}^{x_{n+1}} f\phi_i \, dx = 0, \quad n = 1, \cdots, i-2, i+1, \cdots, N,$$

$$\Rightarrow b_i = \int_{x_{i-1}}^{x_i} f\phi_i \, dx + \int_{x_i}^{x_{i+1}} f\phi_i \, dx$$

$$= \int_{x_{i-1}}^{x_i} f(x) \frac{x - x_{i-1}}{h} \, dx + \int_{x_i}^{x_{i+1}} f(x) \frac{x_{i+1} - x}{h} \, dx$$

Assembly of the load vector

- Case 2: when $i = 1$, the only element, on which ϕ_1 is not zero, is $[x_1, x_2]$. Then

$$\int_{x_n}^{x_{n+1}} f\phi_i \, dx = 0, \quad n = 2, \cdots, N,$$

Assembly of the load vector

- Case 2: when $i = 1$, the only element, on which ϕ_1 is not zero, is $[x_1, x_2]$. Then

$$\int_{x_n}^{x_{n+1}} f\phi_i \, dx = 0, \quad n = 2, \cdots, N,$$

$$\Rightarrow b_1 = \int_{x_1}^{x_2} f\phi_1 \, dx = \int_{x_1}^{x_2} f(x) \frac{x_2 - x}{h} \, dx$$

Assembly of the load vector

- Case 2: when $i = 1$, the only element, on which ϕ_1 is not zero, is $[x_1, x_2]$. Then

$$\int_{x_n}^{x_{n+1}} f\phi_i \, dx = 0, \quad n = 2, \cdots, N,$$

$$\Rightarrow b_1 = \int_{x_1}^{x_2} f\phi_1 \, dx = \int_{x_1}^{x_2} f(x) \frac{x_2 - x}{h} \, dx$$

- Case 3: when $i = N + 1$, the only element, on which ϕ_{N+1} is not zero, is $[x_N, x_{N+1}]$. Then

$$\int_{x_n}^{x_{n+1}} f\phi_{N+1} \, dx = 0, \quad n = 1, \cdots, N-1,$$

Assembly of the load vector

- Case 2: when $i = 1$, the only element, on which ϕ_1 is not zero, is $[x_1, x_2]$. Then

$$\int_{x_n}^{x_{n+1}} f\phi_i \, dx = 0, \quad n = 2, \cdots, N,$$

$$\Rightarrow b_1 = \int_{x_1}^{x_2} f\phi_1 \, dx = \int_{x_1}^{x_2} f(x) \frac{x_2 - x}{h} \, dx$$

- Case 3: when $i = N + 1$, the only element, on which ϕ_{N+1} is not zero, is $[x_N, x_{N+1}]$. Then

$$\int_{x_n}^{x_{n+1}} f\phi_{N+1} \, dx = 0, \quad n = 1, \cdots, N-1,$$

$$\Rightarrow b_{N+1} = \int_{x_N}^{x_{N+1}} f\phi_{N+1} \, dx = \int_{x_N}^{x_{N+1}} f(x) \frac{x - x_N}{h} \, dx$$

Algorithm II

- Initialize the matrix: $\boldsymbol{b} = zeros(N+1, 1)$;
- Compute the integrals and assemble them into $\boldsymbol{b}$:

  FOR $i = 2, \cdots, N$:

  Compute $b(i) = \int_{x_{i-1}}^{x_i} f(x) \frac{x - x_{i-1}}{h} \, dx + \int_{x_i}^{x_{i+1}} f(x) \frac{x_{i+1} - x}{h} \, dx$

  END

  Compute $b(1) = \int_{x_1}^{x_2} f(x) \frac{x_2 - x}{h} \, dx$

  Compute $b(N+1) = \int_{x_N}^{x_{N+1}} f(x) \frac{x - x_N}{h} \, dx$

Dirichlet boundary condition

Basically, the Dirichlet boundary condition $u(a) = g_a, u(b) = g_b$ give the solutions at $x_1 = a$ and $x_{N+1} = b$.

Since the coefficient u_j in the finite element solution $u_h = \sum_{j=1}^{N+1} u_j \phi_j$ is actually the numerical solution at the node $x_j (j = 1, \cdots, N+1)$, we actually know that $u_1 = u(a) = g_a$ and $u_{N+1} = u(b) = g_b$.

Therefore, we don't really need the first and last equations in the linear system since they are set up for u_1 and u_{N+1} by using ϕ_1 and ϕ_{N+1}.

Dirichlet boundary condition

One way to impose the Dirichlet boundary condition is to replace the first and last equations in the linear system by the following two equations

$$u_1 = g_a \quad \Rightarrow \quad 1 \cdot u_1 + 0 \cdot u_2 + \cdots + 0 \cdot u_{N+1} = g_a,$$

$$u_{N+1} = g_b \quad \Rightarrow \quad 0 \cdot u_1 + \cdots + 0 \cdot u_N + 1 \cdot u_{N+1} = g_b.$$

That is, the first and last rows of the matrix $\boldsymbol{A}$ should become

$$(1, 0, \cdots, 0) \quad \text{and} \quad (0, \cdots, 0, 1)$$

respectively, and the first and last elements of the vector $\boldsymbol{b}$ should become g_a and g_b respectively.

Dirichlet boundary condition

- Deal with the Dirichlet boundary conditions:

$$A(1,:) = 0;$$

$$A(1,1) = 1;$$

$$A(N+1,:) = 0;$$

$$A(N+1,N+1) = 0;$$

$$b(1) = g_a;$$

$$b(N+1) = g_b;$$

Basic algorithm

- Input a, b and N, compute $h = (b-a)/N$ and $x_j = a + (j-1)h$ $(j = 1, \cdots, N+1)$.
- Assemble the stiffness matrix $\boldsymbol{A}$ by using Algorithm I.
- Assemble the load vector $\boldsymbol{b}$ by using Algorithm II.
- Deal with the Dirichlet boundary condition by using Algorithm III.
- Solve $\boldsymbol{Ax} = \boldsymbol{b}$ for $\boldsymbol{x}$ by using a direct or iterative method.

Basic algorithm

Remark

The above algorithm uses the Algorithms I, II and III, which are designed for some particular cases with a special method. It is not general enough to deal with different types of PDEs. Therefore, we will discuss a more universal framework in the following.

Universal framework of FEM

- Generate the information matrix: $\boldsymbol{P}$, $\boldsymbol{T}$, $\boldsymbol{E}$;
- Assemble the matrices and vectors: local assemly based on $\boldsymbol{P}$, $\boldsymbol{T}$, $\boldsymbol{E}$ only;
- Deal with the boundary conditions: boundary information matrix and local assembly;
- Solve linear systems.

Mesh information matrices

- Define your global indices for all the mesh elements and mesh nodes. Let N denote the number of mesh elements and N_m denote the number of mesh nodes. Here $N_m = N + 1$.
- Define matrix $\boldsymbol{P}$ to be an information matrix consisting the coordinates of all mesh nodes.
- Define matrix $\boldsymbol{T}$ to be an information matrix consisting of the global node indices of the mesh nodes of all the mesh elements.

Mesh information matrices

For example, for the mesh used in this chapter, we can use the j^{th} column of the matrix $\boldsymbol{P}$ to store the coordinates of the j^{th} mesh node and the n^{th} column of the matrix $\boldsymbol{T}$ to store the global node indices of the mesh nodes of the n^{th} mesh elements:

$$\begin{aligned}\boldsymbol{P} &= \begin{pmatrix} x_1 & x_2 & \cdots & x_{N_m-1} & x_{N_m} \end{pmatrix} \\ &= \begin{pmatrix} x_1 & x_2 & \cdots & x_N & x_{N+1} \end{pmatrix}\end{aligned}$$

$$\begin{aligned}\boldsymbol{T} &= \begin{pmatrix} 1 & 2 & \cdots & N_m-2 & N_m-1 \\ 2 & 3 & \cdots & N_m-1 & N_m \end{pmatrix} \\ &= \begin{pmatrix} 1 & 2 & \cdots & N-1 & N \\ 2 & 3 & \cdots & N & N+1 \end{pmatrix}\end{aligned}$$

Note: The above mesh information matrices $\boldsymbol{P}$ and $\boldsymbol{T}$ are for the mesh nodes.

Finite element information matrices

We also need similar finite element information matrices $\boldsymbol{P}_b$ and $\boldsymbol{T}_b$ for the finite elements nodes, which are the nodes correspoinding to the finite basis function.

For example, the finite element nodes of the linear finite element are the same as those mesh nodes since all the linear basis functions are corresponding to mesh nodes.

Finite element information matrices

We also need similar finite element information matrices $\boldsymbol{P}_b$ and $\boldsymbol{T}_b$ for the finite elements nodes, which are the nodes correspoinding to the finite basis function.

For example, the finite element nodes of the linear finite element are the same as those mesh nodes since all the linear basis functions are corresponding to mesh nodes.

Note: For the nodal finite element basis functions, the correspondence between the finite elements nodes and the finite element basis nodes is one-to-one in a straightforward way. But it could be more complicated for other types of finite element basis functions in the future.

Finite element information matrices

- Define your global indices for all the mesh elements and finite element nodes (or the FE basis functions). Let N_b denote the total number of the FE basis functions (= the number of unknowns = the total number of the FE nodes). Here, $N_b = N + 1$.
- Then

$$u_h = \sum_{j=1}^{N_b} u_j \phi_j.$$

Finite element information matrices

- Define matrix $\boldsymbol{P}_b$ to be an information matrix consisting of the cooordinates of all FE nodes;
- Define matrix $\boldsymbol{T}_b$ to be an information matrix consisting of the global node indices of the FE nodes of all the mesh elements.

Finite element information matrices

For example, for the linear FE we use here, $\boldsymbol{P}_b = \boldsymbol{P}$ and $\boldsymbol{T}_b = \boldsymbol{T}$ since the nodes of the linear FE basis functions are the same as those of the mesh. We can use the j^{th} column of the matrix $\boldsymbol{P}_b$ to store the coordinates of the j^{th} FE node and the n^{th} column of the matrix $\boldsymbol{T}_b$ to store the global node indices of the FE nodes of the n^{th} mesh elements:

$$\begin{aligned}\boldsymbol{P}_b &= \begin{pmatrix} x_1 & x_2 & \cdots & x_{N_b-1} & x_{N_b} \end{pmatrix} \\ &= \begin{pmatrix} x_1 & x_2 & \cdots & x_N & x_{N+1} \end{pmatrix}\end{aligned}$$

$$\begin{aligned}\boldsymbol{T}_b &= \begin{pmatrix} 1 & 2 & \cdots & N_b-2 & N_b-1 \\ 2 & 3 & \cdots & N_b-1 & N_b \end{pmatrix} \\ &= \begin{pmatrix} 1 & 2 & \cdots & N-1 & N \\ 2 & 3 & \cdots & N & N+1 \end{pmatrix}\end{aligned}$$

Finite element information matrices

Remark

For many types of FE, such as quadratic elements which will be dicussed later and some elements which will be introduced in the next chapter, $\boldsymbol{P}_b$ and $\boldsymbol{T}_b$ are different from $\boldsymbol{P}$ and $\boldsymbol{T}$ since the nodes for the FE basis functions are different from those of the mesh.

Local assembly

Observation based on Algorithm I:

- All the non-zero entries in the stiffness matrix $\boldsymbol{A}$ comes from the non-zero local integrals defined on the mesh elements;
- In each non-zero local integral, the trial and test basis functions are only corresponding to the nodes of the element which is the integral interval.
- On each element, all the local integrals, whose trial and test basis functions are corresponding to the nodes of this element, have non-trivial contribution to some non-zero entries of the stiffness matrix $\boldsymbol{A}$.

Local assembly

New assembly idea for the stiffness matrix $\boldsymbol{A}$:

- Loop over all the mesh elements;
- Compute all non-zero local integrals on each element for $\boldsymbol{A}$;
- Assemble these non-zero local integrals into the corresponding entries of $\boldsymbol{A}$.

Local assembly

Compute all non-zero local integrals on each element for $\boldsymbol{A}$:

- On the n^{th} element $E_n = [x_n, x_{n+1}]$, we get non-zero local integrals only when the trial and test basis functions are corresponding to the FE nodes of the element.
- That is, we only consider the trial and test basis functions to be ϕ_n or ϕ_{n+1}.
- These are only four non-zero local integrals on E_n with the global basis function ϕ_n and ϕ_{n+1}:

$$\int_{x_n}^{x_{n+1}} c\phi_n'\phi_n'\,dx, \qquad \int_{x_n}^{x_{n+1}} c\phi_{n+1}'\phi_n'\,dx,$$

$$\int_{x_n}^{x_{n+1}} c\phi_n'\phi_{n+1}'\,dx, \qquad \int_{x_n}^{x_{n+1}} c\phi_{n+1}'\phi_{n+1}'\,dx.$$

- They can be rewritten as

$$\int_{x_n}^{x_{n+1}} c\phi_j'\phi_i'\,dx (i,j = n, n+1).$$

Local assembly

Recall

$$\phi_1(x) = \begin{cases} \frac{x_2 - x}{h}, & x_1 \le x \le x_2, \\ 0, & otherwise \end{cases}$$

$$\phi_j(x) = \begin{cases} \frac{x - x_{j-1}}{h}, & x_{j-1} \le x \le x_j, \\ \frac{x_{j+1} - x}{h}, & x_j \le x \le x_{j+1}, \\ 0, & otherwise \end{cases}$$

$$\phi_{N+1}(x) = \begin{cases} \frac{x - x_N}{h}, & x_N \le x \le x_{N+1}, \\ 0, & otherwise \end{cases}$$

Define two local linear basis functions:

$$\psi_{n1} = \phi_n\big|_{E_n} = \frac{x_{n+1} - x}{h}, \psi_{n2} = \phi_{n+1}\big|_{E_n} = \frac{x - x_n}{h},$$

so in one element, the number of local basis functions N_{lb}=2.

Local assembly

Then the only four non-zero local integrals become

$$\int_{x_n}^{x_{n+1}} c\psi'_{n1}\psi'_{n1}\,dx, \quad \int_{x_n}^{x_{n+1}} c\psi'_{n2}\psi'_{n1}\,dx,$$

$$\int_{x_n}^{x_{n+1}} c\psi'_{n1}\psi'_{n2}\,dx, \quad \int_{x_n}^{x_{n+1}} c\psi'_{n2}\psi'_{n2}\,dx.$$

Local assembly

Then the only four non-zero local integrals become

$$\int_{x_n}^{x_{n+1}} c\psi'_{n1}\psi'_{n1}\,dx, \quad \int_{x_n}^{x_{n+1}} c\psi'_{n2}\psi'_{n1}\,dx,$$

$$\int_{x_n}^{x_{n+1}} c\psi'_{n1}\psi'_{n2}\,dx, \quad \int_{x_n}^{x_{n+1}} c\psi'_{n2}\psi'_{n2}\,dx.$$

That is, instead of the original four non-zero local integrals with the global basis functions ϕ_n and ϕ_{n+1}, we will compute the following four non-zero local integrals with the local basis functions ψ_{n1} and ψ_{n2}:

$$\int_{x_n}^{x_{n+1}} c\psi'_{n\alpha}\psi'_{n\beta}\,dx(\alpha,\beta=1,2)$$

Local assembly

Question: How to compute these integrals?

Local assembly

Question: How to compute these integrals?

Gauss quadrature. The needed information is stored in the matrices $\boldsymbol{P}$ and $\boldsymbol{T}$.

Local assembly

Assemble the non-zero local integrals into $\boldsymbol{A}$:

Based on Algorithm I, when the trial function is ϕ_j and the test function is ϕ_j, the corresponding non-zero local integrals should be assembled to a_{ij}.

For example,

- $\int_{x_n}^{x_{n+1}} c\phi_n'\phi_n'\,dx$ should be assmbled to a_{nn};
- $\int_{x_n}^{x_{n+1}} c\phi_{n+1}'\phi_n'\,dx$ should be assmbled to $a_{n,n+1}$;
- $\int_{x_n}^{x_{n+1}} c\phi_n'\phi_{n+1}'\,dx$ should be assmbled to $a_{n+1,n}$;
- $\int_{x_n}^{x_{n+1}} c\phi_{n+1}'\phi_{n+1}'\,dx$ should be assmbled to $a_{n+1,n+1}$.

Local assembly

Therefore, if we find the global node indices of the trial and test basis functions, we can easily locate where to assemble a non-zero local integral.

Local assembly

Therefore, if we find the global node indices of the trial and test basis functions, we can easily locate where to assemble a non-zero local integral.

Question: Since we compute

$$\int_{x_n}^{x_{n+1}} c\psi'_{n\alpha}\psi'_{n\beta}\,dx(\alpha,\beta=1,2)$$

instead of

$$\int_{x_n}^{x_{n+1}} c\phi'_j\phi'_i\,dx(i,j=n,n+1),$$

how do we obtain the corresponding global node indices of the local trial and test basis functions $\psi_{n\alpha}$ and $\psi_{n\beta}(\alpha,\beta=1,2)$?

Local assembly

Therefore, if we find the global node indices of the trial and test basis functions, we can easily locate where to assemble a non-zero local integral.

Question: Since we compute

$$\int_{x_n}^{x_{n+1}} c\psi'_{n\alpha}\psi'_{n\beta}\,dx(\alpha,\beta=1,2)$$

instead of

$$\int_{x_n}^{x_{n+1}} c\phi'_j\phi'_i\,dx(i,j=n,n+1),$$

how do we obtain the corresponding global node indices of the local trial and test basis functions $\psi_{n\alpha}$ and $\psi_{n\beta}(\alpha,\beta=1,2)$?

Information matrix $\boldsymbol{T}_b$!

Local assembly

Recall that the n^{th} column of the matrix $\boldsymbol{T}_b$ stores the global node indices of the FE nodes of the n^{th} mesh element:

$$\boldsymbol{T}_b = \begin{pmatrix} 1 & 2 & \cdots & N-1 & N \\ 2 & 3 & \cdots & N & N+1 \end{pmatrix}.$$

Hence $\boldsymbol{T}_b(\alpha, n)$ and $\boldsymbol{T}_b(\beta, n)$ give the global node indices of the local trial and test basis function $\psi_{n\alpha}$ and $\psi_{n\beta}(\alpha, \beta = 1, 2)$.

That is, for $n = 1, \cdots, N$,

$$\int_{x_n}^{x_{n+1}} c\psi'_{n\alpha}\psi'_{n\beta}\, dx(\alpha, \beta = 1, 2)$$

should be assembled to a_{ij}, where

$$i = \boldsymbol{T}_b(\beta, n), \quad j = \boldsymbol{T}_b(\alpha, n).$$

Local assembly

Algorithm IV:

- Initilize the matrix: $\boldsymbol{A} = sparse(N_b, N_b)$;
- Compute the integrals and assemble them into $\boldsymbol{A}$:

FOR $n = 1, \cdots, N$:

FOR $\alpha = 1, \cdots, N_{lb}$:

FOR $\beta = 1, \cdots, N_{lb}$:

Compute $r = \int_{x_n}^{x_{n+1}} c\psi'_{n\alpha}\psi'_{n\beta}\, dx$

Add r to $\boldsymbol{A}(\boldsymbol{T}_b(\beta, n), \boldsymbol{T}_b(\alpha, n))$

END

END

END

Local assembly

Algorithm IV (alternative version):

- Initilize the matrix: $\boldsymbol{A} = sparse(N_b, N_b), \boldsymbol{S} = zeros(N_{lb}, N_{lb})$;
- Compute the integrals and assemble them into $\boldsymbol{A}$:

FOR $n = 1, \cdots, N$:

FOR $\alpha = 1, \cdots, N_{lb}$:

FOR $\beta = 1, \cdots, N_{lb}$:

Compute $S(\beta, \alpha) = \int_{x_n}^{x_{n+1}} c\psi'_{n\alpha}\psi'_{n\beta}\, dx$

END

END

$\boldsymbol{A}(\boldsymbol{T}_b(:, n), \boldsymbol{T}_b(:, n)) = \boldsymbol{A}(\boldsymbol{T}_b(:, n), \boldsymbol{T}_b(:, n)) + \boldsymbol{S}$;

END

Local assembly

To make a general subroutine for different cases, more information are needed for computing and assembling the integral should be treated as input parameters or input function of this subroutine:

- the coefficient function c;
- the Gauss quadrature points and weights for numerical integrals;
- the mesh information matrices $\boldsymbol{P}$ and $\boldsymbol{T}$, which can also provide the number of the mesh elements $N = size(\boldsymbol{T}, 2)$ and the number of mesh nodes $N_m = size(\boldsymbol{P}, 2)$;
- the FE information matrices $\boldsymbol{P}_b$ and $\boldsymbol{T}_b$ for the trial and test functions repectively, which can also provide the number of local basis functions $N_{lb} = size(\boldsymbol{T}_b, 1)$ and the number of the global basis functions $N_b = size(\boldsymbol{P}_b, 2) =$ the number of unknowns
- the type of the basis function for the trial and test functions, respectively.