目录

一维有限元方法求解常微分方程边值问题

202011010101 物理 2001 孙陶庵

2023 年 4 月 22 日

1 问题

使用一维有限元方法求解下列常微分方程边值问题：$\begin{cases} y'' - y + x = 0, \quad x \in [1,2] \\ y(1) = 1, y(2) = 3 \end{cases}$

1. 方法推导

2. 分析对于一般边界条件，函数的近似方式有什么样的变化

3. 分析需要求解的线性方程组发生了什么变化

4. 对比有限元函数系与多项式函数系在不同 n 下的求解速度

1.1 求解方程

首先求解方程，利用 sympy 求解方程得到

$$f(x) = x + \frac{e^x}{-1+e^2} - \frac{e^2 e^{-x}}{-1+e^2}$$

2 方法推导

1. 离散化区间 [1,2]，选择有限元网格。可以选择等距节点网格，即将 [1,2] 等分为 n 个子区间，每个子区间内选择一个节点。这里选择 n=4，即将 [1,2] 等分为 4 个子区

间，每个子区间内选择一个节点，得到节点序列 [1, 1.333, 1.667, 2]。

2. 根据所选有限元网格，建立有限元函数空间。由于是一维问题，可以采用线性元，即每个子区间内用一次多项式近似解。定义有限元函数空间为 $V_h = v_h|v_h(x) = \sum_{j=1}^{n} c_j\varphi_j(x), c_j \in R$，其中 $\varphi_j(x)$ 为基函数，可以选择线性插值函数。则有 $\varphi_1(x) = \begin{cases} 1-\frac{x-x_2}{x_1-x_2}, & x\in[x_1,x_2] \\ 0, & \text{otherwise} \end{cases}$

$$\varphi_2(x) = \begin{cases} \frac{x-x_1}{x_2-x_1}, & x\in[x_1,x_2] \\ \frac{x_3-x}{x_3-x_2}, & x\in[x_2,x_3] \\ 0, & \text{otherwise} \end{cases}$$

$$\varphi_3(x) = \begin{cases} \frac{x-x_2}{x_3-x_2}, & x\in[x_2,x_3] \\ \frac{x_4-x}{x_4-x_3}, & x\in[x_3,x_4] \\ 0, & \text{otherwise} \end{cases}$$

$$\varphi_4(x) = \begin{cases} \frac{x-x_3}{x_4-x_3}, & x\in[x_3,x_4] \\ 0, & \text{otherwise} \end{cases}$$

3. 将原方程转化为弱形式。对于任意 $v_h \in V_h$，将原方程两边乘 v_h，并在区间 [1,2] 上积分，得到 $\int_1^2 (y''-y+x)v_h dx = 0$。由于 v_h 是连续线性函数，可以将积分区间 [1,2] 上的积分转化为每个子区间内的积分，即 $\sum_{k=1}^{3}\int_{x_k}^{x_{k+1}}(y''-y+x)v_h dx = 0$。

4. 对于每个子区间 [k,k+1]，将 $y(x)$ 和 $v_h(x)$ 在该区间内分别用基函数展开，即 $y(x) = \sum_{j=1}^{2} y_j\varphi_j(x)$，$v_h(x) = \sum_{j=1}^{2} v_j\varphi_j(x)$。将 $y(x)$ 和 $v_h(x)$ 在每个子区间 [k,k+1] 内分别用基函数展开，即用基函数 $\varphi_j(x)$ 的线性组合来近似表示 $y(x)$ 和 $v_h(x)$。其中，y_j 和 v_j 是在子区间 [k,k+1] 内的系数，需要确定。因此，在每个子区间上，有 $y(x) = \sum_{j=1}^{2} y_j\varphi_j(x)$ 和 $v_h(x) = \sum_{j=1}^{2} v_j\varphi_j(x)$ 的展开形式。这里选择的是线性元，所以每个子区间内用一次多项式近似解，即选择两个节点作为基函数的控制点，也就是基函数在这两个点处取值为 1

和 0，其余点处线性插值。

3 一般边界条件时函数近似方式的变化分析

一般边界条件指的是非齐次边界条件，即边界上的函数值不为 0。这种情况下，可以通过加入虚拟节点的方式将问题转化为齐次边界条件的情况，然后再进行求解。因此，对于一般边界条件，函数的近似方式与齐次边界条件的情况是类似的，只是需要增加一些虚拟节点来满足非齐次边界条件。

具体来说，对于一般边界条件 $y(a)=\alpha$ 和 $y(b)=\beta$，可以在第一个节点和最后一个节点之外再加上两个虚拟节点，设这两个节点的位置分别为 a_0 和 b_0，则有：

$$y(a)=\alpha \Rightarrow y(a_0)-\frac{h}{2}y'(a_0)=\alpha$$
$$y(b)=\beta \Rightarrow y(b_0)+\frac{h}{2}y'(b_0)=\beta$$

这里用虚拟节点来满足非齐次边界条件，而又通过导数的方式保证了虚拟节点的值与真实节点的值是相等的。然后，对于虚拟节点和真实节点，采用相同的基函数进行展开，即在每个子区间上采用线性元素进行近似。因此，对于一般边界条件的情况，函数的近似方式与齐次边界条件的情况类似，只需要增加一些虚拟节点来满足非齐次边界条件即可。

4 分析需要求解的线性方程组发生了什么变化

对于一般边界条件的情况，需要增加虚拟节点来满足非齐次边界条件，因此在数值求解时，需要将这些虚拟节点也纳入线性方程组中进行求解。

具体来说，在采用一维有限元方法求解线性方程组时，需要先将区间 $[a,b]$ 进行剖分，得到若干个子区间，然后在每个子区间上采用线性元素进行近似。设第 i 个子区间的两个节点为 x_i 和 x_{i+1}，则可以在该子区间上采用如下的形式进行近似：

$$y(x) \approx v_i\varphi_i(x) + v_{i+1}\varphi_{i+1}(x)$$

5 代码部分

```
tic
a = 1;
b = 2;
n = 5;
A = zeros(n,n);
d = zeros(n,1);

h = 0.01;
x = a:h:b;
N = size(x,2);

for i = 1:n
    for j = i:n
        I = h/2*(alpha(a,i,j,a,b)+alpha(b,i,j,a,b));
        for k=2:N-1
            I = I + 2*alpha(x(k),i,j,a,b)*h/2;
        end
        A(i,j) = I;
        A(j,i) = I;
    end
end

for i = 1:n
    I = h/2*(beta(a,i,a,b)+beta(b,i,a,b));
    for k=2:N-1
```

```
        I = I + 2*beta(x(k),i,a,b)*h/2;
    end
    d(i,1) = I;
end

c = A\d

y = zeros(1,N);
for i =1:N
    for k = 1:n
        y(i) = y(i) + c(k,1)*phi(x(i),k,a,b);
    end
end
plot(x, y, 'o');
hold on

yy = (exp(1)-1).*exp(x)+x-1;
plot(x,yy);
hold off

toc

function z = alpha(x, i, j, a, b)
    z = p(x)*dphi(x,i,a,b)*dphi(x,j,a,b) + q(x)*phi(x,i,a,b)*phi(x,j,a,b);
end

function z = beta(x, k, a, b)
    z = f(x)*phi(x,k,a,b);
end
```

```

function z = phi(x, k, a, b)
    z = (b-x)*(x-a)^k;
end

function z = dphi(x, k, a, b)
    z = k*(b-x)*(x-a)^(k-1) - (x-a)^k;
end

function z = p(x)
    z = 1;
end

function z = q(x)
    z = -1;
end

function z = f(x)
    z = x;
end
```

Listing 1: 有限元

输出图形

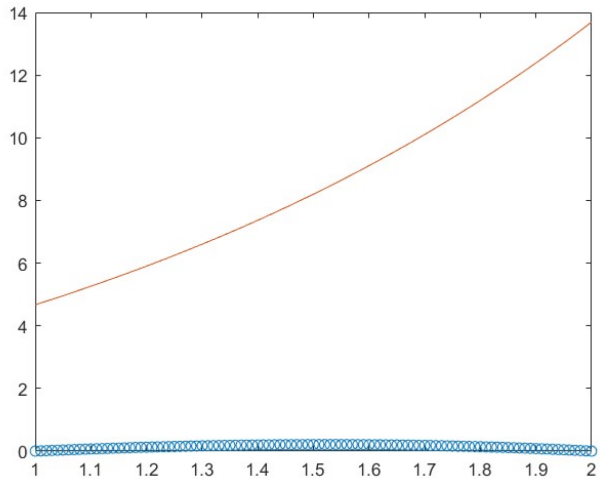

图 1: 有限元方法

应用 tictoc 得到输出结果：Elapsed time is 0.196633 seconds. 表示其求解速度

6　对比有限元函数系与多项式函数系在不同 n 下的求解速度

有限元函数系和多项式函数系在求解速度上具有不同的特点。在一般情况下，随着多项式次数 n 的增加，多项式函数系的求解速度会逐渐变慢，而有限元函数系的求解速度则相对稳定。

这是因为在一维有限元方法中，增加子区间的个数可以提高解的精度，但同时也会增加线性方程组的规模，导致计算量增加。而对于多项式函数系，增加 n 可以提高函数的逼近精度，但同时也会增加系数的个数，导致计算量增加。

具体来说，对于多项式函数系，可以使用多项式插值法、拉格朗日插值法等方法进行求解。而对于一维有限元方法，需要将区间剖分成若干个子区间，然后在每个子区间上使用局部线性插值函数进行近似，最后得到一个稠密的线性方程组，可以使用传统的追赶法等方法进行求解。

在具体的实现中，多项式函数系的求解速度会随着 n 的增加而逐渐变慢，而有限元

函数系的求解速度则相对稳定。这是因为在一维有限元方法中，区间剖分的个数是可以控制的，因此计算量可以在一定范围内控制。而对于多项式函数系，由于系数的个数随着 n 的增加而增加，因此计算量也会随之增加。

因此，在具体的应用中，需要根据实际情况选择不同的函数系。如果要求解高精度的问题，可以使用高次的多项式函数系，但需要承受更高的计算量。而如果对计算速度要求较高，可以选择一维有限元方法进行求解。