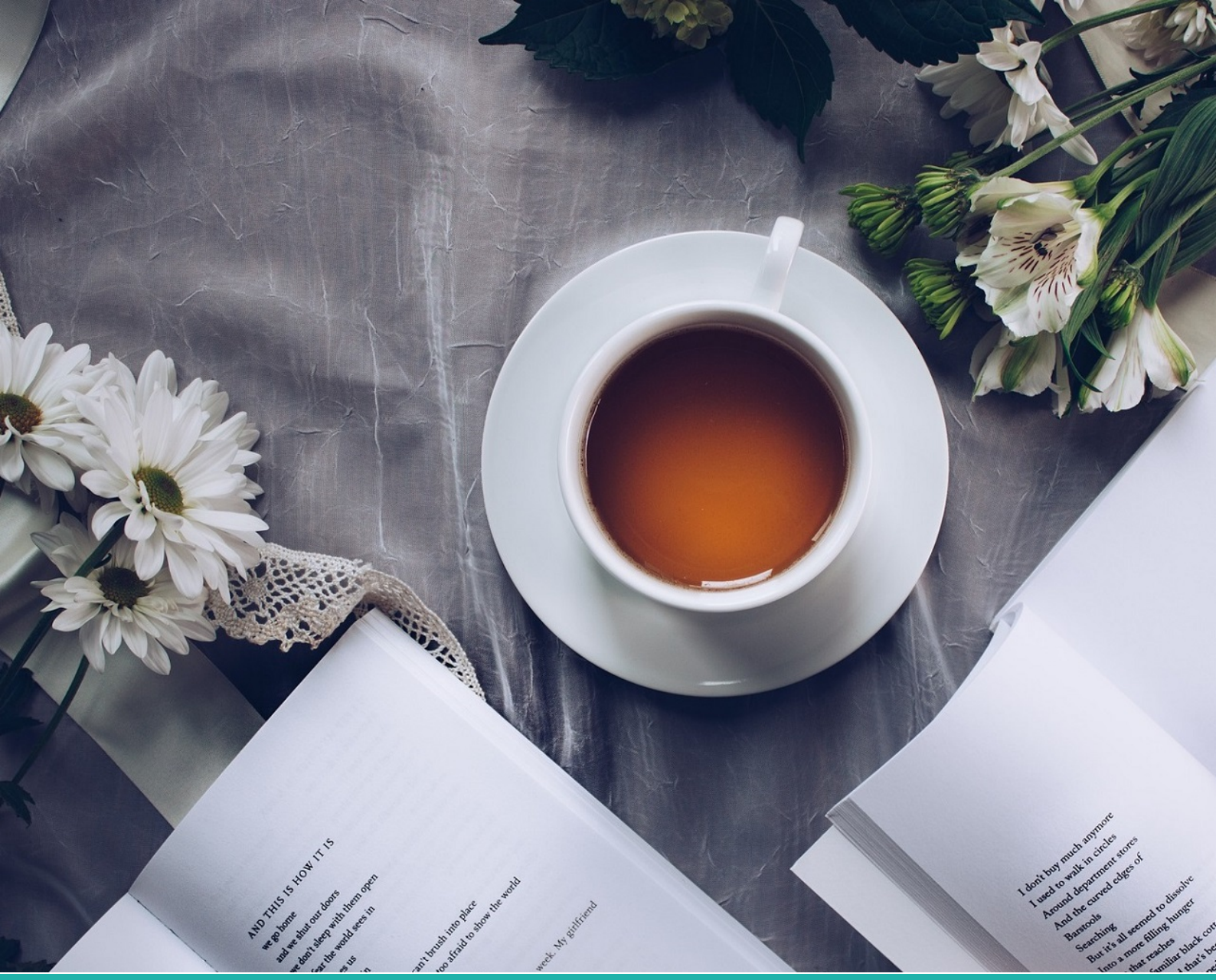

Note For Finite Element Methods

Zhejiang University

作者：Shuang Hu

组织：Zhejiang University

时间：Sept 14, 2022

版本：1.0

简介：2022 秋冬学季“有限元方法”课程笔记

ElegantLaTeX Program

目录

第 1 章 引入

1.1 为什么需要有限元方法？

此前在《微分方程数值解》课程中，我们已经学习了有限差分法和有限体积法。这两种方法有不少优点：首先，比较直观，只要知道如何利用差分近似导数即可得到对应的差分公式；其次，在一些情形下，有限差分和有限体积方法可以实现较高的计算精度。

但是，这两种算法有一些明显的缺陷。

- 算法稳定性的分析比较复杂。
- 处理不规则区域的问题时较为麻烦，需要多次利用插值近似。
- 只是求解离散格点的近似点值/离散网格的近似积分平均值，未能给出函数整体的近似。

为此，基于函数逼近论的**有限元方法**被提出。该算法能弥补有限差分法的一些明显缺陷，目前是最主流的数值算法之一。

1.2 从一维边值问题说起

考虑如下例子：

$$\begin{cases} -u'' + u = f(x), x \in (0,1) \\ u(0) = u(1) = 0. \end{cases} \tag{1.1}$$

类似于“偏微分方程”课程中对弱解的讨论方式，在(1.1)两边同时乘某个函数 v 并在 $[0,1]$ 上积分，得到如下形式：

$$\int_0^1 (-u'' + u)v\mathrm{d}x = \int_0^1 fv\mathrm{d}x. \tag{1.2}$$

定义函数空间 V 如下：

$$V := \left\{ v \mid v(0) = v(1) = 0, \int_0^1 ((v')^2 + v^2)\mathrm{d}x < \infty \right\}. \tag{1.3}$$

如果函数 $v \in V$, 利用分部积分法，(1.1)可以转化为以下问题：

例题 1.1 记 $a(u,v) = \int_0^1 (u'v' + uv)\mathrm{d}x$, $h(v) = \int_0^1 fv\mathrm{d}x$, $v \in V$. 求 $u \in V$，使得 $a(u,v) = h(v)\ \forall v \in V$.

下面的定理说明了该问题可以转化为一个优化问题：

定理 1.1

记泛函 $J(v) := \frac{1}{2}a(v,v) - h(v)$, 问题1.1与最小化 $J(v)$ 的优化问题等价。即：如果 $a(u,v) = h(v) \forall v \in V$, 那么 $J(u) \le J(v) \forall v \in V$。

♡

证明 $\Rightarrow$:

$$\begin{aligned} J(v) - J(u) &= \frac{1}{2}a(v,v) - h(v) - \frac{1}{2}a(u,u) + h(u) \\ &= \frac{1}{2}(a(v,v) - a(u,u) - 2h(v-u)) \\ &= \frac{1}{2}(a(v,v) - a(u,u) - 2a(u,v-u)) \\ &= \frac{1}{2}(a(v,v) + a(u,u) - 2a(v,u)) \\ &= \frac{1}{2}(a(v-u,v-u)) \ge 0. \end{aligned} \tag{1.4}$$

由(1.4)可得，如果 $a(u,v) = h(v)$，那么 $J(u) \le J(v)$.

$\Leftarrow$: $\forall v \in V, t \in \mathbb{R}$, 有 $J(u+tv) \geq J(u)$。我们定义函数 $g(t) := J(u+tv)$，根据上面的讨论可知:$g'(0) = 0$。另一方面，计算 $g(t)$ 的表达式，有:

$$\begin{aligned} g(t) &= J(u+tv) \\ &= \frac{1}{2}\int_0^1 ((u'+tv')^2 + (u+tv)^2)\mathrm{d}x - \int_0^1 f(u+tv)\mathrm{d}x \end{aligned} \tag{1.5}$$

对(1.5)求一阶导数，可得:

$$g'(0) = \int_0^1 (u'v' + uv)\mathrm{d}x - \int_0^1 fv\mathrm{d}x. \tag{1.6}$$

根据 g 的一阶条件，可得 $a(u,v) = h(v)$。又由于 v 的任意性，结论得证。

如此，我们把一个解微分方程的问题，利用1.1和1.2转化为了一个变分问题。由于 V 是一个无穷维空间，我们不能期望利用算法给出这个变分问题的精确解，但我们可以考虑对空间 V 进行有限维近似，并在有限维空间上近似求解这个变分问题。

1.3 有限元思想的导出

接下来，根据上一节的思路，我们继续问题(1.1)的近似求解。根据上面的分析，我们的数值算法需要解决两个问题:

- 如何对函数空间 V 进行有限维近似?
- 在进行有限维近似之后，如何在有限维空间中对变分问题进行求解?

首先，我们考虑第二个问题。根据上面的讨论，近似求解变分问题有两种不同的思路，分别对应的是 **Galerkin 近似方法**和 **Ritz 近似方法**。

1.3.1 Galerkin 近似

假设已经给出有限维子空间 $V_N \leq V$，Galerkin 近似的目的是求解 $u_N \in V_N$ 使得

$$a(u_N, v_N) = h(v_N) \tag{1.7}$$

对所有 $v_N \in V_N$ 都成立。

由于 V_N 是有限维的空间，我们可以找到这个空间中的一组基函数 $\{\phi_1, \cdots, \phi_N\}$，注意到 $a(u,v)$ 是对称双线性函数，设 $u_N = \alpha_1\phi_1(x) + \cdots + \alpha_N\phi_N(x)$，$v_N(x) = \phi_i(x)$，代入(1.7)，可得一个线性方程组:

$$\begin{bmatrix} a(\phi_1,\phi_1) & a(\phi_1,\phi_2) & \cdots & a(\phi_1,\phi_N) \\ a(\phi_2,\phi_1) & a(\phi_2,\phi_2) & \cdots & a(\phi_2,\phi_N) \\ \vdots & \vdots & \ddots & \vdots \\ a(\phi_N,\phi_1) & a(\phi_N,\phi_2) & \cdots & a(\phi_N,\phi_N) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix} = \begin{bmatrix} h(\phi_1) \\ h(\phi_2) \\ \vdots \\ h(\phi_N) \end{bmatrix} \tag{1.8}$$

解这个线性方程组，得到系数向量，即可给出该方程的近似解。

1.3.2 Ritz 方法

同样，假设有限维子空间 $V_N \leq V$ 已经给出，**Ritz 方法**的思路是求解有关 $J(u)$ 的优化问题，即: 求 $u_N \in V_N$，使得

$$J(u_N) \leq J(v) \forall v \in V_N. \tag{1.9}$$

这里 $J(v) := \frac{1}{2}a(v,v) - h(v)$。

给出这个问题之后，我们在有限维空间中，利用最优化算法求解该问题。

这两个思路都是建立在有限维子空间 V_N 已经给出的前提下的。但这个有限维空间如何构造?

一个很容易想到的思路是利用 $v(0) = v(1)$ 这一性质，构造三角函数系作为基底。这种选取思路对于问题(1.1)而言当然是极好的，三角函数系的正交性也使得(1.8)中的系数矩阵变得相当简单易求解。但这个方案的可扩展性并不强，如果扩展到二维平面上，乃至更高维度的椭圆偏微分方程，就很难找到像这样全局定义的基函数。如果问题区域非规则或是存在不同方程的耦合，则更是如此。

有限元方法由此引出。

1.4 有限元方法

在上面两种思路的基础上，我们需要一个方便推广的构建有限维子空间的方法。

多项式函数空间是最容易表示的函数空间，因此这是我们的首选。但全局定义的多项式很难保证其符合边界条件。于是，借助样条插值的思想，我们转为考虑分段多项式空间。

1.4.1 线性有限元空间

我们先针对问题(1.1)，考虑最简单的近似形式–分段线性近似。在这种情形下，有限维子空间 V_h 由 $(0,1)$ 上的分段线性函数表示。

定义 1.1 (线性有限元空间)

线性有限元空间的定义为：

$$V_h := \{v_h \in C(0,1) : v_h(0) = v_h(1) = 0, v_h|_{[x_i,x_{i+1}]} \in \mathbb{P}_1\}. \tag{1.10}$$

其中 $\{x_i\}_{i=0}^n$ 为 $[0,1]$ 上给定的互异节点，$x_0 = 0$，$x_n = 1$。

♣

首先需要讨论的是，空间 V_h 的维数和基底。空间(1.10)的形式很容易联想到数值分析课程中学习过的 **B-样条空间**。特别地，一维 B-样条基函数为所谓的 “hat-function”，定义如下：

$$\phi_i(x) = \begin{cases} \dfrac{x - x_{i-1}}{x_i - x_{i-1}}, x_{i-1} \le x \le x_i \\ \dfrac{x_{i+1} - x}{x_{i+1} - x_i}, x_i \le x \le x_{i+1} \\ 0, otherwise \end{cases} \tag{1.11}$$

图 1.1: hat 函数的示意图

容易验证，如此定义的 $(\phi_1, \cdots, \phi_{n-1})$ 构成了空间 V_h 的一组基。从而可得 $\dim(V_h) = n - 1$。

1.4.2 刚度矩阵与负载

此处我们采用 Galerkin 近似的思路，下面我们需要讨论的是方程组(1.8)的导出与求解。

定义 1.2 (刚度矩阵)

线性有限元空间1.1的刚度矩阵定义为：

$$A = (a(\phi_i, \phi_j))_{i,j=1}^{n-1}. \tag{1.12}$$

其中 ϕ_i 为(1.11)中定义的基函数。

♣

定义 1.3 (负载)

方程(1.1)关于线性有限元空间1.1的负载向量定义为：

$$\mathbf{b} := \left(\int_0^1 f(x)\phi_i(x)\mathrm{d}x \right)_{i=1}^{n-1}. \tag{1.13}$$

♣

下面讨论(1.1)负载的计算。由于我们是用一次样条多项式近似真实解的，在选取数值积分公式的时候只需要使用一阶代数精度的公式即可不损失计算精度。因此此处采用复化梯形公式进行负载的近似计算。

记 $h_i = x_i - x_{i-1}$，负载的近似计算公式如下：

$$b_i = \int_0^1 f(x)\phi_i(x)\mathrm{d}x = \int_{x_{i-1}}^{x_{i+1}} f(x)\phi_i(x) \approx \frac{1}{2}(h_i + h_{i+1})f(x_i). \tag{1.14}$$

对于刚度矩阵的计算，虽然基函数 ϕ_i 在整个区间 $(0,1)$ 上并不可导，但不可导点为零测集，不会对积分的计算产生影响。将基函数 ϕ_i 的表达式代入刚度矩阵，计算得：

$$\begin{cases} a_{ij} = 0, |i-j| > 1 \\ a_{ii} = \dfrac{1}{h_i} + \dfrac{1}{h_{i+1}} + \dfrac{1}{3}(h_i + h_{i+1}) \\ a_{i-1,i} = -\dfrac{1}{h_i} + \dfrac{1}{6}h_i \\ a_{i,i+1} = -\dfrac{1}{h_{i+1}} + \dfrac{1}{6}h_{i+1} \end{cases} \tag{1.15}$$

由(1.15)可知，刚度矩阵是一个三对角矩阵，因此如果采用传统的矩阵存储方式和求解方式可能会造成计算资源的浪费。下面给出一种采用“局部矩阵”的形式存储(1.15)的技巧。

记负载矩阵 $A = \sum_{k=1}^{n} A^{(k)}$，其中 $A^{(k)} := (a_{ij}^{(k)})$，$a_{ij}^{(k)} := \int_{x_{k-1}}^{x_k} [\phi_i'(x)\phi_j'(x) + \phi_i(x)\phi_j(x)]\mathrm{d}x$，由定义可得，$a_{ij}^{(k)}$ 只在 $i,j \in \{k-1, k\}$ 时取非零值。所以每个 $A^{(k)}$ 都可以存储为一个 2×2 的矩阵，而整体的负载矩阵 A 可以视为把所有的 $A^{(k)}$ “装配”起来的结果。在求解最终的线性方程组的时候，由于矩阵 A 是一个三对角对称正定矩阵，我们可以采用一系列数值代数算法加速方程组的求解。

1.5 误差分析