Research group
Discrete Geometries of Mathematics and Physics
www.mathischeap.com/dgmp

Discrete Geometries of Mathematics and Physics
创新实验

实验指导书

一类基函数的构造与应用

Yi Zhang (张仪)
: www.mathischeap.com
@: zhangyi_aero@hotmail.com
: https://github.com/mathischeap

1 实验基本信息

1.1 实验名称

一类基函数的构造与应用

1.2 实验指导老师联系方式

张仪(www.mathischeap.com), 数学与计算科学学院，花江慧谷4#-408A

- 手机： 199 7417 1867
- email: zhangyi_aero@hotmail.com
- QQ: 360 788 903

1.3 所属课程名称

创新实验

2 实验的目的意义要求等

2.1 实验目的

学习拟谱元法的基函数的构造方法并完成它们的编程实现与测试验证。

2.2 实验意义

- 学习理解有限元方法的一些基本思想
- 训练针对数学问题的编程能力
- 培养对于有限元方法这一领域的兴趣

2.3 实验要求

- 具备基础Python编程（包含阅读文档、自行查阅开源函数使用方法等）能力。Python作为最简单实用的编程语言，十分易学。鼓励无基础的同学借助本实验入门Python编程。
- 完成实验报告并发回你的程序。你可以通过查询网络（或询问chatgpt等）搜集信息，但不允许从网络直接复制或要求chatgpt生成代码或实验报告等。

3 实验步骤

3.1 理论基础

有限元方法包含了多种精细的数学逻辑与设计。其中最重要的一个是**用有限维度的空间近似无限维度的空间**。其具体表现为**使用有限个已知函数的线性组合对某个函数进行近似**。比如，一般函数$f(x)$可以被近似表达成

$$f(x) \approx f^h(x) = \alpha_0 b^0(x) + \alpha_1 b^1(x) + \alpha_2 b^2(x) + \cdots + \alpha_N b^N(x) = \sum_{i=0}^{N} \alpha_i b^i(x), \tag{1}$$

其中$b^0(x)$, $b^1(x)$, $b^2(x)$, $\cdots$, $b^N(x)$是$N+1$个已知函数（一般称为基函数, basis functions），α_0, α_1, α_2, $\cdots$,α_N是$N+1$个实系数(coefficients)，也被称为展开系数。显然，一旦我们能够确定这$N+1$个实系数的值，我们就能确定$f(x)$的近似表达式，即$f^h(x)$。

不同的有限元方法往往使用不同的基函数。作为有限元方法中较新的一种，拟谱元法[1, 2]使用了一种特殊的基函数，边多项式（edge polynomials）[3]。通过本实验，你将学习拟谱元法的基函数的构造方法并完成它们的编程实现与测试验证，从而学习理解有限元方法的一些基本思想，训练针对数学问题的基础编程能力，培养对于有限元方法这一领域的兴趣。

3.1.1 拉格朗日多项式

拉格朗日多项式（Larange polynomials）是一种最常用的基函数。在$x \in [-1, 1]$这一区间上，我们取$N+1$个递增的点，x_0，x_1， x_2，$\cdots$，x_N, 也就是

$$-1 = x_0 < x_1 < x_2 < \cdots < x_N = 1. \tag{2}$$

基于这$N+1$个点，我们可以构造$N+1$个拉格朗日多项式，即

$$l^i(x) := \prod_{j=0, j\neq i}^{N} \frac{x - x_j}{x_i - x_j}, \quad i \in \{0, 1, 2, \cdots, N\}. \tag{3}$$

注释 1： 上式中$\prod$为连乘符号，即

$$\prod_{j=0, j\neq i}^{N} y_j(x) = y_0(x)y_1(x)y_2(x)\cdots y_{i-1}(x)y_{i+1}(x)\cdots y_N(x).$$

很显然，拉格朗日多项式有一个显著性质：

$$l^i(x_j) = \delta^i_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{else} \end{cases}, \quad i, j \in \{0, 1, 2, \cdots, N\}, \tag{4}$$

其中δ^i_j一般被称作克罗内克德尔塔（Kronecker delta）符号。

3.1.2 边多项式

边多项式（edge polynomials）是通过对拉格朗日多项式的一阶导数进行线性组合得到的。它们的构造方法为

$$e^i(x) := \sum_{j=i}^{N} \frac{\mathrm{d}l^j(x)}{\mathrm{d}x} = -\sum_{j=0}^{i-1} \frac{\mathrm{d}l^j(x)}{\mathrm{d}x}, \quad i \in \{1, 2, 3, \cdots, N\}, \tag{5}$$

其中，$e^i(x)$即为N个与拉格朗日多项式(3)对应的边多项式。边多项式具备的一个典型特征是：

$$\int_{x_{j-1}}^{x_j} e^i(x)\mathrm{d}x = \delta^i_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{else} \end{cases}, \quad i, j \in \{1, 2, 3, \cdots, N\}. \tag{6}$$

性质(4)和(6)是本实验着重探索的对象。

3.2 投影

本小节我们将介绍如何使用拉格朗日多项式和边多项式完成有限元近似，即使用$l^i(x)$或$e^i(x)$替代(1)中的$b^i(x)$将$f(x)$近似成$f^h(x)$的过程。这个过程一般被称为***投影（projection）***，记作π，也就是

$$f^h(x) = \pi\left(f(x)\right) \approx f(x).$$

从(1)可以看出，如果能够计算得到展开系数α_i，就能完成投影，得到$f^h(x)$。下面我们分别介绍使用拉格朗日多项式和边多项式作为基函数时计算展开系数的方法。

3.2.1 拉格朗日多项式展开系数

给定一个函数$f(x)$，$x \in [-1, 1]$，拉格朗日多项式的展开系数(expansion coefficients)为

$$\alpha_i = f(x_i), \quad i \in \{0, 1, 2, \cdots, N\}. \tag{7}$$

那么，拉格朗日多项式投影(或拉格朗日展开)就能够表达成

$$f^h(x) = \pi_l\left(f(x)\right) = \sum_{i=0}^{N} \alpha_i l^i(x), \tag{8}$$

其中$l^i(x)$即为拉格朗日多项式(3).

3.2.2 边多项式展开系数

给定一个函数$g(x)$， $x \in [-1,1]$，边多项式的展开系数为

$$\beta_i = \int_{x_{i-1}}^{x_i} g(x)\mathrm{d}x, \quad i \in \{1,2,3,\cdots,N\}. \tag{9}$$

进而边多项式投影(或边多项式展开)就能够表示成

$$g^h(x) = \pi_e\left(g(x)\right) = \sum_{i=1}^{N} \beta_i e^i(x), \tag{10}$$

其中$e^i(x)$即为边多项式(5).

3.3 详细实验步骤

本节中，我们将介绍具体的实验步骤。本实验将使用Python作为编程语言。你可以在下面链接中下载本实验所需的所有样板程序。

（步骤一）实验准备: 请访问
https://mathischeap.com/contents/teaching/innoexp/msem_basis_function_study/main#course-innoexp-bfs
下载本实验所需的样板程序。解压后的文件夹中包含了所有的（完整的及待完成的）*Python*代码文件:

- *_0_lobatto_nodes.py*
- *_1_lagrange.py*
- *_2_edge.py*
- *_3_lagrange_expansion.py*
- *_4_edge_expansion.py*
- *_5_error_visualization.py*

请根据本实验指导书补全其中待完成代码并完成测试。本实验需要用的*Python*库为*numpy, scipy, matplotlib*。如果你没有安装某个库，请运行比如*$ pip install numpy*命令安装相应库。

3.3.1 拉格朗日多项式的实现与测试

我们使用Lobatto点集作为我们构造拉格朗日多项式和边多项式的基础点集，即(2)。产生该点集的（完整的）代码为_0_lobatto_nodes.Lobatto函数：

$$\text{Lobatto}(N) \to \text{Lobatto\_nodes}.$$

该函数输入使用正实数N作为输入变量，生成含有$N+1$个升序点的（即阶数为N的）Lobatto点集。

（步骤二） 拉格朗日多项式的实现与测试：请补全*_1_lagrange.Lagrange_polynomials*函数中的待完成部分（即代码中“****待补全****”，“****补全完成****”之间的部分），实现拉格朗日多项式的计算：

$$Lagrange\_polynomials(N, x) \to Lagrange\_polynomial\_values.$$

该函数旨基于阶数为N的*Lobatto*点集构建$N+1$个拉格朗日多项式并计算它们在变量x的值。补全后，请运行*_1_lagrange.py*文件。这将自动调用绘图函数绘制拉格朗日多项式。请观察结果是否满足性质(4)。你可以改变“*if __name__ == "__main__":*”判断语句后的变量N的值，观察不同阶数下拉格朗日多项式的形态和性质。

3.3.2 边多项式的实现与测试

样板程序中已经完成了边多项式的实现。请查看_2_edge.edge_polynomials函数。该函数与上述_1_lagrange.Lagrange_polynomials输入变量一样，但它输出N个边多项式在变量x的值。

（步骤三） 边多项式的测试：请运行*_2_edge.py*文件。它将自动调用绘图函数绘制边多项式。请观察结果是否满足性质(6)。请改变“*if __name__ == "__main__":*”判断语句后的变量N的值，观察不同阶数下边多项式的形态和性质。

3.3.3 拉格朗日多项式投影

给定函数

$$f(x) = \ln\left(\sin(2\pi x) + \frac{3}{2}\right), \quad x \in [-1, 1].$$

该函数的Python版本位于_3_lagrange_expansion.f。我们对其进行拉格朗日多项式展开，也就是(8).

（步骤四） 拉格朗日多项式的展开系数：请根据(7)，补全函数
_3_lagrange_expansion.Lagrange_coefficients，

$$Lagrange\_coefficients(N) \to Lagrange\_expression\_coefficients$$

它计算基于阶数为N的*Lobatto*点集的拉格朗日多项式的展开系数。待补全的代码位于“****待补全****”，“****补全完成****”之间。

在计算得到拉格朗日多项式展开系数之后，我们可以用它们作为系数将拉格朗日多项式进行线性组合，最终得到$f(x)$的拉格朗日投影$f^h(x)$，也就是(8)。

（步骤五） 拉格朗日多项式投影：请根据(8)，补全函数
_3_lagrange_expansion.Lagrange_expansion，

$$Lagrange\_expansion(N, x) \to Lagrange\_expansion\_value$$

它将求解使用基于阶数为N的*Lobatto*点集的拉格朗日展开式$f^h(x)$在x的值。待补全的代码位于“****待补全****”，“****补全完成****”之间。请运行*_3_lagrange_expansion.py*文件。它

将自动调用绘图函数绘制边$f(x)$与$f^h(x)$，并调用误差函数计算$f(x)$与$f^h(x)$间的误差。请改变“*if __name__ == "__main__":*”判断语句后的变量N的值观察图像与误差的变化。

3.3.4 边多项式投影

给定函数

$$g(x) = \frac{\mathrm{d}\left(f(x)\right)}{\mathrm{d}x} = \frac{2\pi\cos(2\pi x)}{\sin(2\pi x) + \frac{3}{2}}, \quad x \in [-1, 1].$$

该函数的Python版本位于*_4_edge_expansion.g*。我们对其进行边多项式展开，也就是(10).

（步骤六） 边多项式的展开系数：请根据(9)，补全函数 *_4_edge_expansion.edge_coefficients*，

$$edge\_coefficients(N) \to edge\_expression\_coefficients$$

它将计算基于阶数为N的*Lobatto*点集的边多项式的展开系数。同样，待补全的代码位于“*****待补全*****”，“*****补全完成*****”之间。注意，我们可以使用*scipy*库的积分函数*quad*计算定积分，如：

```
from scipy.integrate import quad

def f(x):
  return 3.0*x*x + 1.0

I, err = quad(f, 0, 1)
print(I)
print(err)
```

这个例子中的代码计算了$\int_0^1 3x^2 + 1\mathrm{d}x$。积分值为$I$，误差为$err$。

在计算得到边多项式展开系数之后，我们可以用它们作为系数将边多项式进行线性组合，最终得到$g(x)$的拉格朗日投影$g^h(x)$。

（步骤七） 边多项式投影：请根据(10)，补全函数 *_4_edge_expansion.edge_expansion*，

$$edge\_expansion(N, x) \to edge\_expansion\_values$$

它求解使用基于阶数为N的*Lobatto*点集的边多项式展开式$g^h(x)$在x的值。待补全的代码位于“*****待补全*****”，“*****补全完成*****”之间。请运行*_4_edge_expansion.py*文件。它将自动调用绘图函数绘制边$g(x)$与$g^h(x)$，并调用误差函数计算$g(x)$与$g^h(x)$间的误差。请改变“*if __name__ == "__main__":*”判断语句后的变量N的值观察图像与误差的变化。

3.3.5 误差可视化

在完成拉格朗日多项式和变多项式的投影及误差计算后，我们可以计算不同N对应的误差，并进行作图分析。比如，拉格朗日多项式投影和边多项式投影的误差分别记作err_N^l和err_N^e，我们可以搜集数据，分别制作以下表格：

N	1	2	3	4	5	6	8	9	10	$\cdots$
err_N^l										

N	1	2	3	4	5	6	8	9	10	$\cdots$
err_N^e										

（步骤八）误差可视化：利用搜集的误差数据，在*_5_error_visualization.py*中使用*matplotlib*库作图进行可视化。我建议使用*semilogy*函数作图，横轴为N，纵轴为误差，err_N^l或err_N^e，制作两张图。*matplotlib*库中*semilogy*函数的使用方法非常简单。请自行上网调查。

3.4 实验报告

在完成上述实验整理实验报告时，请尽可能多地加入你的观察、理解、对实验结果的讨论、遇到的问题及解决方法等可以反映你主观思考的内容。同时，请也将你的程序打包发送至实验课程负责老师的邮箱。

References

[1] Y. Zhang, A. Palha, M. Gerritsma, L. G. Rebholz, A mass-, kinetic energy- and helicity-conserving mimetic dual-field discretization for three-dimensional incompressible Navier-Stokes equations, part I: Periodic domains, Journal of Computational Physics 451 (2022) 110868.

[2] Y. Zhang, Mimetic Spectral Element Method and Extensions toward Higher Computational Efficiency, 2022.

[3] M. Gerritsma, Edge functions for spectral element methods, Spectral and High Order Methods for Partial Differential Equations, Springer (2011) 199–207.