**湘潭大学毕业论文**

**题 目： 二阶椭圆界面问题的浸入界面有限元方法**

**学 院： 数学与计算科学学院**

**班 级： 2018信息与计算科学2班**

**学 号： 201805750521**

**姓 名： 刘威**

**指导教师： 王华**

**完成日期： 2022年5月**

**湘 潭 大 学**

**毕业论文任务书**

毕业论文题目： 二阶椭圆界面问题的浸入界面有限元方法

学号： 201805750521 姓名： 刘威 专业： 信息与计算科学

指导教师： 王华 系主任：

一、主要内容及基本要求

主要内容：

1. 了解界面问题的物理背景以及研究现状；

2.比较拟合网格方法和非拟合网格方法的优缺点，选择一种方法，进行深入研提相

应的改进方案；

3.完成一类界面问题的有限元误差的理论分析，并编程实现。

基本要求：

1.选题体现专业特点及教学要求，难度和份量适虫；

2.搜集、精读、归纳与选题有关的文献不少干10篇；

3.撰写研究综述的现状时要有学术观点提炼和文献引用,并进行辩证评析；

4.要求论文结构合理，内容充实，论据可靠，论证有力，主题明确，概念准确，层次清楚，重点突出，文字简练，文理通顺，有自己的见解；

5.按照毕业论文的各项进度要求完成工作，主动并及时与指导老师联系；

6.毕业论文的正文篇幅不少于10000字。

二、重点研究的问题

1. 界面单元形函数的构造与插值误差估计；

2. 如何改进方法得到与跳跃系数无关的有限元误差估计；

3. 界面问题的有限元程序设计。

三、进度安排

|  |  |  |
| --- | --- | --- |
| 序号 | 各阶段完成的内容 | 完成时间 |
| 1 | 毕业论文选题 | 2021年9-10月 |
| 2 | 围绕选题搜集、阅读、整理文献资料 | 2021年9-10月 |
| 3 | 开题：完成文献综述和论文框架 | 2021年10月中旬 |
| 4 | 中期检查：提交论文初稿 | 2022年3月下旬 |
| 5 | 论文查重检测 | 2022年4月下旬 |
| 6 | 论文定稿后排版打印、交稿 | 2022年5月初 |
| 7 | 毕业论文答辩 | 2022年5月中旬 |
| 8 | 答辩后的论文修改 | 2022年5月下旬 |

四、应收集的资料及主要参考文献

[1] 李荣华,刘播. 微分方程数值解法[M]. 第四版. 北京: 高等教育出版社, 2010.

[2]崔文文. 求解二阶椭圆界面问题的二次浸入型有限元方法[D]. 郑州: 郑州大学, 2017.

[3]冯亚芳. 二阶椭圆界面问题浸入界面有限元方法的多重网格算法[D]. 南京: 南京师范大学, 2017.

[4]Ji Haifeng, Chen Jinru, Li Zhilin. A Symmetric and Consistent Immersed Finite Element Method for Interface Problems[J]. Journal of Scientific Computing, 2014, 61(3).

[5]王华. 界面问题的增扩有限元方法[D]. 南京: 南京师范大学, 2018.

[6]王楠. 界面问题的扩展有限元方法[D]. 南京: 南京师范大学, 2020.

[7]和晓晓. 界面问题的匹配及非匹配有限元方法[D]. 南京: 南京大学, 2019.

**湘 潭 大 学**

**毕业论文评阅表**

学号： 201805750521 姓名： 刘威 专业： 信息与计算科学

毕业论文题目： 二阶椭圆界面问题的浸入界面有限元方法

|  |  |
| --- | --- |
| 评价项目 | 评 价 内 容 |
| 选题 | 1.是否符合培养目标，体现学科、专业特点和教学计划的基本要求，达到综合训练的目的；  2.难度、份量是否适当；  3.是否与生产、科研、社会等实际相结合。 |
| 能力 | 1.是否有查阅文献、综合归纳资料的能力；  2.是否有综合运用知识的能力；  3.是否具备研究方案的设计能力、研究方法和手段的运用能力；  4.是否具备一定的外文与计算机应用能力；  5.工科是否有经济分析能力。 |
| 论文质量 | 1.立论是否正确，论述是否充分，结构是否严谨合理；实验是否正确，设计、计算、分析处理是否科学；技术用语是否准确，符号是否统一，图表图纸是否完备、整洁、正确，引文是否规范；  2.文字是否通顺，有无观点提炼，综合概括能力如何；  3.有无理论价值或实际应用价值，有无创新之处。 |
| 综  合  评  价 | 该生论文选题符合专业培养目标，能够达到综合训练目标，文章篇幅符合学院规定，内容较为完整，实证分析较为科学，态度较为认真，参考了丰富的文献资料，有一定的个人见解。    同意其参加答辩。  评阅人：  2022年5月12日 |

**湘 潭 大 学**

**毕业论文鉴定意见**

学号： 201805750521 姓名： 刘威 专业： 信息与计算科学

毕业论文题目： 二阶椭圆界面问题的浸入界面有限元方法

|  |
| --- |
| 内容提要  本论文主要介绍了二阶椭圆界面问题的浸入界面有限元方法, 此方法的关键是修改界面单元上的基函数, 得到线性协调元空间的修正空间, 即浸入界面有限元空间, 通过修正空间, 可以更好地对界面单元进行求解, 而且不需要对整体多项式进行构造也能逼近真解, 因此界面的位置和大小与网格剖分无关。  浸入界面有限元方法的优点在于它的自由度比其他方法的少, 并且对于界面的形状和位置是稳定的。 对此方法的理论, 我们证明了浸入界面基函数的唯一可解性和相容性, 同时了解到能唯一决定浸入有限元空间包括顶点函数值和界面跳跃条件。 在得到近似解时, 利用与误差估计, 为所定义的模型解决了真解与近似解的偏差问题, 最后也对浸入界面有限元方法做了六组数值实验, 得到的图像表明数值解有明显的拐点, 说明如果使用一般的有限元方法的误差会很大, 而且通过数值实验所得到的误差证明了对不同的界面形状在范数, 半范数上具有最优逼近性。 |
| 指导教师评语  论文选题符合专业培养目标，全文结构基本合理科学，逻辑思路清晰，观点表达准确，语言流畅，论证方法较合理，参考的文献符合主题要求，从主题到内容符合专业要求，衔接的比较紧密，但个别引文没有标著出来，真正属于自己创新的内容还不是很多，个别概念比较模糊，总体上达到毕业论文要求。  同意其参加答辩，建议成绩评定为 良好 。  指导教师：  2022年5月10 日 | | |
| 答辩简要情况及评语  该生能较好地运用所学理论和专业知识, 按期圆满地完成论文, 有一定地独立工作能力, 但在完成任务书所要求地设计内容上有欠缺。论文分析合理，层次分明，文字通畅，论文格式基本符合要求。  该生在答辩过程中，表达清晰，能基本正确地回答答辩委员提出的主要问题。  根据答辩情况，结合毕业论文写作水平、指导教师意见、评阅人意见和答辩专家意见等综合评定，**答辩小组同意其成绩评定为 良好 。**  答辩小组组长：  2022年 5 月15日 | | |
| 答辩委员会意见  该生能在规定时间内比较流利、清晰地阐述论文的主要内容, 陈述论文思路清晰、能准确回答所提出的问题。  经答辩委员会讨论，同意该毕业论文成绩评定为 良好 。  答辩委员会主任：  2022年5月20日 | | |

# 摘 要

客观世界下的很多能接触到的材料都不仅仅是单一的材料构成, 往往由不同种材料复合而来. 其不同材料复合而来的物理现象所构成的数学模型即为界面问题. 材料之间的间断面称之为界面, 且在上述模型中, 模型的解通常受间断系数的影响需要满足某种跳跃条件, 即服从某种意义上的守恒律. 界面问题在足够光滑的界面条件下,其解也会在此区域上光滑. 但是由于解在界面上的跳跃, 导致使用一般的数值解法很难在解的整体光滑性较差的情况下还能有比较理想的逼近效果和逼近精度. 我们有许多种求解界面问题的方法, 有限差分法是最先用于求解的. 但是在网格不能与界面良好匹配的情况下, 使用有限差分是不太理想的. 于是便要用到网格与界面非匹配的方法，扩展有限元法以及浸入界面有限元方法是非匹配网格法的两种主要方法. 本文采用的是非匹配网格法中的浸入界面有限元法来对一维情况下二阶椭圆界面问题进行求解.

**关键词：** 二阶椭圆方程；界面问题；浸入界面有限元方法

Abstract

Many of the materials that can be accessed in the objective world are not just composed of a single material, but are often compounded from different materials. The mathematical model composed of the physical phenomena composed of different materials is the interface problem. The section between the materials is called the interface, and in the above model, the solution of the model is usually affected by the discontinuity coefficient and needs to meet certain jump conditions, that is, obey the conservation law in some sense. The solution to an interface problem that is sufficiently smooth will also be smooth in this area. However, due to the jump in the interface of the solution, it is difficult to use the general numerical solution method to have a relatively ideal approximation effect and approximation accuracy under the condition that the overall smoothness of the solution is poor. We have many ways to solve interface problems, and the finite difference method was the first to be used to solve them. But in cases where the mesh doesn't match the interface well, using finite difference is less than ideal. Therefore, it is necessary to use the grid and interface non-matching method, extended finite element method and immersion interface finite element method is the two main methods of non-matching mesh method. In this paper, the immersion interface finite element method in the unmatched mesh method is used to solve the second-order elliptic interface problem in one-dimensional situation.

**Key words:** Second order elliptic equation; Interface problem; Immersed finite element method

**目 录**

[摘要 I](#_Toc103763116)

[Abstract II](#_Toc103763117)

[1 前言 1](#_Toc103763118)

[1.1 研究背景 1](#_Toc103763119)

[1.2 国内外研究现状 1](#_Toc103763120)

[2 有限元方法的数学理论 2](#_Toc103763121)

[2.1 Sobolev空间初步 2](#_Toc103763122)

[2.2 Galerkin方法 4](#_Toc103763123)

[2.3 有限元离散 5](#_Toc103763124)

[2.4 误差估计 7](#_Toc103763125)

[3 浸入界面有限元方法 10](#_Toc103763126)

[3.1 模型问题 10](#_Toc103763127)

[3.2 浸入界面有限元空间 10](#_Toc103763128)

[3.3 有限元离散 12](#_Toc103763129)

[3.4 误差估计 14](#_Toc103763130)

[4 数值例子 16](#_Toc103763131)

[总结 20](#_Toc103763132)

[参考文献 24](#_Toc103763133)

[致谢 25](#_Toc103763134)

[附录A 论文检测报告 26](#_Toc103763135)

[附录B 数值程序 24](#_Toc103763136)

# 前言

## 研究背景

工程应用中普遍存在界面问题, 即由间断系数不同的介质耦合而产生的问题, 比如声音在水中的传递、海市蜃楼现象等, 同时在数值模拟等场景中也有着大量的界面问题. 有限元法、有限体积法以及有限差分法这三种数值方法常被用于求解界面问题: 有限差分法通过合适的数值微分公式, 离散化为差分格式从而对方程组求解; 有限体积法的用法很强大, 虽然对于高精度的构造略显麻烦, 但是能有效快捷地对其他有限差分领域进行求解, 并且对于双曲、抛物、椭圆型都可以进行运算, 但是在对于椭圆方程求解时, 有限元法却更为适配, 并且在求解椭圆型方程的过程中没有间断. 综上所述, 对计算区域复杂度较高的方程采用有限元和有限体积法, 有限差分法就不再那么合适, 本课题采取的求解方法为浸入界面有限元法.

浸入界面有限元方法是对界面附近的逼近函数进行修改, 对网格划分下各个单元求近似解, 而不需要对整体多项式进行构造也能逼近真解. 由此看来,界面的位置和大小与网格剖分无关.

## 国内外研究现状

每一项技术的产生与发展都是顺应时代的需要. 而新技术的出现也会经历层层磨难, 用来检验是否符合生产力的需要. 有限元法最初应用领域较为单一, 且由于在上世纪90年代以前, 计算机成本高、体积大等缺点导致计算机的普及很困难, 因此许多有限元分析模拟很难进行实践, 大多数都停留在理论阶段. 当时社会背景下, 有限元的分析建模大都针对力学、流体以及电磁场等物理场问题进行单一模拟, 局限性很大. 但是单一物理场的问题在客观世界下并不存在, 所以有限元方法的未来还有更广阔的空间. 在上世纪40年代, 新兴崛起的航空航天事业让人们对飞机的内部结构设计越发精密, 而有限元分析法能进行精确的设计和计算, 在此契机下, 有限元方法得以加速发展.

1943年, R. W. Courant首次提出有限元法的核心思想. 1956年, R. W. Clough等四位教授与工程师在科技期刊上发表一篇计算飞机机翼强度的论文, 且把此种解法称之为刚性法(Stiffness), 是有限元法在工程学界上的开端. 1960年, R. W. Clough教授发表的平面弹性论文中, “有限元法”这个名称被首次使用, 同时也将有限元方法扩展到土木工程上. 1963年, Richard MacNeal博士与Robert Schwendler联手创办了MSC公司, 并开发了第一个软件程序, 名为SADSAM, 即数字仿真模拟结构分析,标志着有限元方法(FEA)由理论向程序的转变, 1964-1965年, O. C. Zienkiewicz等人发表关于利用极小位能原理和虚功原理, 以新的思路推导出有限元法.

我国有限元发展之路中较为著名的有: 冯康(有限单元法理论), 钱令希(余能定理), 钱伟长(广义变分原理)等等. 但是受限于当时的国际与国内环境, 我国的学者在有限元方法上的进一步研究困难重重, 很难跟上国际潮流方向, 遗憾的被国外拉开距离. 在之后的十年, 伴随着国内有限元软件和有限元方法理论的诞生和发展, 大型工程也逐渐使用有限元方法来计算, 如水利、机械等多个领域, 并且也都取得了不错的效果.上世纪90年代, 国外的有限元软件大批量地涌入国内市场, 涉及到各个领域. 国外的学者专家也都进入各大学、工厂与企业进行宣传他们所掌握的技术与使用技巧. 国内人员大都选择更加先进的国外技术, 导致国内有限元发展更加困难. 管理部门对有限元软件的认知上产生了偏差, 对此失去了必要的支持, 核心技术掌握在国外, 所以至上世纪的最后十几年里, 国内自主技术创新的进度十分缓慢. 但是进入21世纪后, 国内自主知识产权的软件逐渐市场化, 获得了一定的发展, 同时也获得了国家对有限元技术的关注, 逐渐走出低迷状态, 有限元技术也不再仅仅停留在高校和企业中.

# 有限元方法的数学理论

## Sobolev空间初步

考虑弦的平衡问题: 一根长为的弦, 端点固定在和上, 如果没有外力作用在弦上时, 弦的位置在水平方向与轴重合, 如果存在强度为外荷载垂直向下作用在弦上时, 弦会发生形变. 假定荷载强度很小, 则弦发生的形变也会很小. 由力的平衡条件可知, 外荷载作用于弦上所达到的平衡位置满足如下微分方程[1]:

其中是弦的张力.

边值条件:

则弦的平衡问题就等价于求解上述方程.根据极小位能原理, 弦的平衡位置是在所有满足边值条件的位置中,使位能最小的位置, 计算此时的应变能:

外力所做的功:

总位能: . 由极小位能原理可知 是如下变分问题的解:

为了精确地表述变分问题, 结合位能的计算公式, 需要对u 作出适当的限制, 下面引入这样的限制空间:

一维区间上的Sobolev空间: 设. 其内积和范数分别为:

*.*

可得是Hilbert空间. 同时满足Cauchy收敛. 分析, 假定, 为了使有意义, 则需 所以我们要求的限制空间得包含如此的可测函数, 但是如果仅限于此条件,限制空间是不够的, 真实物理情况复杂, 许多构建出来的函数不可微, 所以需要进一步的扩充. 对所有在上一次连续可微的函数, 利用分部积分法, 可得:

其中. 由此推广得到广义导数的概念: 设若存在, 满足如下等式:

则说明在上有广义导数, 记为.在一般情况下有对应的广义导数, 则 也是在广义意义下的导数, 但是反过来结论不成立. 函数的广义导数并不唯一, 但不唯一的广义导数几乎相等, 利用变分法的基本引理可对此证明.

变分法的基本引理：设满足

则 几乎处处为0.

定义

,

其中 是 在下的广义导数. 可以得到是线性空间. 将 引入内积

范数：

得到是完全内积空间, 所以是Hilbert空间, 称之为Sobolev空间[2]. 类似地对阶的Sobolev空间 进行定义内积和范数:

.

当 时, 就是 空间,

## Galerkin方法

作为数值分析方法的一种, Galerkin方法利用样条函数方法, 提供对局部基函数地选取方法, 大大降低了传统有限元方法对基函数地选取困难, 并且在选取有限个多项式函数(又称基函数或形函数)之后, 原求解变分问题等价为求基函数系数的问题, 利用标准有限元法求解即可. 在方法推导上Galerkin法比Ritz方法更直接, 适用面更广, 如不需对称, 从而大大加强Galerkin法的实用性[1].

设是的一组基, 则对, 有：

考虑如下两点边值问题:

满足Lax-Milgram theorem定理, 建立两点边值问题的变分问题：求使得:

其中 .根据Lax-Milgram theorem定理, 变分问题存在唯一解, 由于Galerkin法对基函数的选取方法, 得到的解并非是原变分方程的精确解, 而是各个单元之和的近似解.

引理2.2.1 (Lax-Milgram定理) **:** 设V是Hilbert空间, 是上的双线性泛函, 且满足连续性和V椭圆的, f是V上有界线性泛函, 则变分问题.有唯一解, 其中f是V上的有界线性泛函[2].

## 有限元离散

1有限元离散化简称线性元求解, 计算过程首先是对所要求解的区间进行网格剖分:

网格剖分节点：

相邻节点 之间的单元被记作, 同时也被称为第 号单元, 为部分步长,定义线性元空间:

,

2 取Sobolev空间 内的子空间 : 元素时多项式, 且多项式个数不超过在(为正整数). 且进行网格剖分的区间属于Sobolev空间,即. 按照上述方法取得的 , 且 可以得出是试探函数空间. 分段线性函数由节点上的一组值

按线性插值公式:

确定. 其空间称之为一次有限元空间, (2.3.1)为单元形状函数, 被试探函数限制在单元上. 为了使得按段插值标准化, 引入标准(或参考)单元[0,1], 给出标准插值基函数(分别是关于标准单元的左右端点的线性拉格朗日因子):

引入仿射变换:

和逆变换

则

得到 是 维线性空间.

将(2.3.1)代入泛函数: ,

由文献可知, . 利用(2.3.3)可解出

令

可由 (2.3.7)得到有限元方程.

通过分析每一单元的局部二次形及单元矩阵, 可以得到单元刚度矩阵, 由多个单元刚度矩阵可以形成总刚度矩阵. 利用刚度矩阵求解的方法直观灵活, 且在程序的实现上也较为简单. 从单刚度矩阵形成总刚度矩阵的步骤: 计算第个单元刚度矩阵的元素, 并标出在总刚度矩阵的行列号; 总刚度矩阵的第行第列元素实际上, 当与不相邻时, 即时当与相邻时, 即时非零元素只出现在第个节点相邻两单元刚度矩阵, 不作用于其他刚度矩阵. 按照上述法则, 可以得到总刚度矩阵为三对角矩阵以及右端方程的联系[1-4].

现用Galerkin法推导, 虽然可以取的基底很多, 但是有些基底不可取, 考虑剖分单元上的线性插值公式和单刚对于总纲上的系数, 对每个剖分单元节点 构造山形函数:

显然线性无关, 可以组成的基底.边值问题相应的线性形式为

从而Galerkin方程为

利用上述构造的山形函数和仿射变换, 可得:

则对 , 基函数可写成:

可由运算可得

综上, 总刚度矩阵即为 (2.3.10)的系数矩阵.

3若左边值条件非齐次, 则增加一基函数:

将表为 , 自由度还是n, 有限元方程:

若右边值条件非齐次, 则右端修改为, 有限元方程:

## 误差估计

定理2.4.1 设 是 的解,是Galerkin方程 的解, 则有与无关的常数使误差满足不等式[2]：

其中(2.4.1)不等式左边表示与的距离, 不等式右边表示和子空间的最小距离, 且在不考虑常数的情况下,(2.4.1)估计式是最佳的. 因此可由(2.4.1)得到收敛性, 若对完全:

证明:从(2.4.1)推(2.4.2)是直接的. 实际上, 对任意 有线性组合 使

由(2.4.1), 当 时,

这证明了(2.4.2).

今证(2.4.1), 因 , 依次满足方程：

故对任意,

两式相减, 得

利用

和(2.4.3), 有

由

因此

消去共同因子 , 并令 ,则得

两端关于 取下确界, 即得(2.4.1).

由定理(2.4.4)得出的不等式对有限元解的误差估计起到关键作用:

应用到线性元就是

由上得问题已经化为纯属逼近论的问题了. 现介绍一种逼近方法, 取分段线性插值函数, 来逼近:

在任一单元 内考虑差

显然 . 由Rolle定理, 有 , 使 , 于是

从而

这样便得插值误差估计

右端 是 模. 可得有限元解的估计：

若注意到对 , 还可得[4-9].

可见当 时 一致趋于 .

# 浸入界面有限元方法

## 模型问题

有以下二阶椭圆界面问题:

在界面上的跳跃条件:

其中是矩形区域, 且被界面分成、, 即, 表示区域Ω的边界. 假设, 系数是分片正常数, 即:

考虑到为一维情况, 所以用、代替、. 则由原问题的双线性形式可得其变分形式为：求, 满足:

其中

由Lax-Milgram定理可知：变分形式有解且唯一.

## 浸入界面有限元空间

浸入界面有限元方法的关键是修改界面单元上的基函数, 得到线性协调元空间的修正空间, 即浸入界面有限元空间, 通过修正空间, 可以更好地对界面单元进行求解. 现假设界面落在单元 内, 在节点 和 处的标准插值基函数为:

界面条件:

设, 满足如下条件

根据待定系数法, 由, (3.2.1), (3.2.2), 跳跃条件可求出修改后的两个点的基函数[3][6]

其中

线性方程组修正 , 通过修正得到

对于二阶椭圆界面问题, 若则, 且满足 由Cea引理, 可得

## 有限元离散

对于二阶椭圆界面问题

对(3.3.1)左右同乘 ,

并积分

,

,

.

求 使得

由于 是无限维的, 用 逼近.

求解：

设,

取得到以下等式

我们取则

我们取则

因为由基函数的定义可知, 当时, . 所以可化为如下形式：

记为 Ku=b.

当时, 方程的左端只有三个非零系数：；

当时, 方程的左端只有两个非零系数：；

当时, 方程的左端只有两个非零系数：.

由于界面落在单元 , 为便于区分, 现记界面单元为.

需要对包含点处的基函数的元素进行修改, 即： .

方程右端所需修改的元素：.

令u的扩充向量, 将u的计算转换为其扩充向量的计算. 则(n+1)\*(n+1)阶三带状矩阵 和(n+1)维向量

其中为总刚度矩阵, 为总荷载向量.

下面给出总刚度矩阵和总荷载向量用单元积分的表示公式.

其中(3.3.6), (3.3.7)分别表示双线性泛函在剖分单元上的单元刚度矩阵和部分单元上的单元荷载向量. 通过文献, 建立总刚与单刚的元素之间的关系:

对角元

非对角元.

可知，所需要修改的元素:

单元上需分段考虑, 记 故(3.3.8)等价于:

建立总载与单载的元素之间的关系:

元素

同理得, 方程右端所需要修改的元素:

由(3.2.1), (3.2.3), (3.2.4)即可解得所需修改的元素并且考虑到(3.3.1)二阶椭圆界面问题的Dirichlet边界条件 ,则, 可得:

需要对界面点所在的剖分单元左边的单元乘, 对右边的单元乘. 结合上文方程左端和右端需要修改的元素, 对(3.3.7)中的和进行处理, 重新组装求解.显然重新组装的的解向量中的u也是方程Ku=b的解[1][11][12].

## 误差估计

求 误差估计, 我们考虑其适定性问题, 即其解是否唯一. 步骤如下：

定义双线性形式

线性泛函

问题(3. 3. 1)抽象地写成

其强制性如下：

.

其有界性(连续性)如下:

.

记：

(庞加莱不等式)

...(cauchy-schwarz不等式)

由Lax-Milgram定理可知问题(3.3.2)存在唯一解, 且满足. 假设 是强制的, 有界点的和 是有界的, 由 Cea引理, 可得

# 数值例子

考虑下面二阶椭圆界面问题

其中是一维空间中的有界区域, 界面点

跳跃条件:

构造精确解：

令

通过对函数求导 , 有：

,

由原二阶椭圆方程得

所以

对系数 ()做一组梯度变化:

1. 当=1, =100

**表 4‑1 系数比为1/100误差**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| h | 1/8 | 1/16 | 1/32 | 1/64 | 1/128 |
|  | 0. 1408 | 0. 1107 | 0. 0689 | 0. 0313 | 0. 0078 |
|  | 0. 0053 | 0. 0033 | 0. 0006 | 3. 52e-05 | 5. 49e-07 |

|  |  |
| --- | --- |
| **图 4‑1 系数比为1/100时精确解图像** | **图 4‑2系数比为1/1000时数值解图像** |

1. 当=1, =1000

**表 4‑2 系数比为1/1000 误差**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| h | 1/8 | 1/16 | 1/32 | 1/64 | 1/128 |
|  | 0. 1411 | 0. 1114 | 0. 0695 | 0. 0316 | 0. 0079 |
|  | 0. 0054 | 0. 0033 | 6. 08e-04 | 3. 55e-05 | 5. 54e-07 |

|  |  |
| --- | --- |
| **图 4‑3系数比为1/1000时精确解图像** | **图 4‑4系数比为1/1000时数值解图像** |

1. 当=1, =10000

**表 4‑3 系数比为1/10000 时的误差**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| h | 1/8 | 1/16 | 1/32 | 1/64 | 1/128 |
|  | 0. 1411 | 0. 1115 | 0. 0695 | 0. 0316 | 0. 0079 |
|  | 0. 0054 | 0. 0033 | 6. 09e-04 | 3. 56e-05 | 5. 55e-07 |

|  |  |
| --- | --- |
| **图 4‑5系数比为1/10000时精确解图像** | **图 4‑6系数比为1/1000时数值解图像** |

1. 当=100, =1

**表 4‑4 系数比为100/1 时的误差**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| h | 1/8 | 1/16 | 1/32 | 1/64 | 1/128 |
|  | 0. 1750 | 0. 1224 | 0. 0702 | 0. 0314 | 0. 0078 |
|  | 0. 0075 | 0. 0036 | 6. 13e-04 | 3. 53e-05 | 5. 50e-07 |

|  |  |
| --- | --- |
| **图 4‑7系数比为100/1时精确解图像** | **图 4‑8系数比为100/1时数值解图像** |

1. 当=1000, =1

**表 4‑5 系数比为1000/1 时的误差**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| h | 1/8 | 1/16 | 1/32 | 1/64 | 1/128 |
|  | 0. 1752 | 0. 1230 | 0. 0707 | 0. 0317 | 0. 0079 |
|  | 0. 0075 | 0. 0036 | 6. 18e-04 | 3. 56e-05 | 5. 55e-07 |

|  |  |
| --- | --- |
| **图 4‑9系数比为1000/1时精确解图像** | **图 4‑10系数比为1000/1时数值解图像** |

1. 当=10000, =1

**表 4‑6 系数比为10000/1 时的误差**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| h | 1/8 | 1/16 | 1/32 | 1/64 | 1/128 |
|  | 0. 1752 | 0. 1231 | 0. 0708 | 0. 0317 | 0. 0079 |
|  | 0. 0075 | 0. 0036 | 6. 18e-04 | 3. 56e-05 | 5. 55e-07 |

|  |  |
| --- | --- |
| **图 4‑11系数比为10000/1时精确解图像** | **图 4‑12系数比为10000/1时数值解图像** |

# 总结

本文就二阶椭圆方程的一维情况提出使用浸入界面有限元方法. 利用此方法的核心思想, 即得到线性协调元的修正空间来逼近问题的真实解. 也对一维情况下的浸入界面方程模型进行定义, 借助参考文献证明了模型解的存在唯一性, 并且了解到能唯一决定浸入有限元空间包括顶点函数值和界面跳跃条件. 在得到近似解时, 利用与误差估计, 为所定义的模型解决了真解与近似解的偏差问题, 丰富了浸入界面有限元方法的理论结果.

在之后的研究中, 可以选择精确度更高的误差估计算法, 以弥补本文中对误差估计的不足; 同时也可对本文中浸入界面有限元方法的一维情况进行总结, 从而可以推广到多维情况或者多界面点情况, 其次, 利用计算机软件的编程实现问题的仿真模拟, 对此方法的优越性和精确性进行验证, 使方法的可行性和实用性大大提高, 从而推动此方法的发展, 让其更加适配现实情况的复杂问题的仿真模拟.

# 参考文献

1. 李荣华, 刘播. 微分方程数值解法[M]. 第四版. 北京: 高等教育出版社, 2010.
2. 崔文文. 求解二阶椭圆界面问题的二次浸入型有限元方法[D]. 郑州: 郑州大学, 2017.
3. 冯亚芳. 二阶椭圆界面问题浸入界面有限元方法的多重网格算法[D]. 南京: 南京师范大学, 2017.
4. 和晓晓. 界面问题的匹配及非匹配有限元方法[D]. 南京: 南京大学, 2019.
5. Zhilin Li and Kazufumi Ito. The immersed interface method: numerical solutions of PDEs involving interfaces and irregular dimains[J]. SIAM, 2016.
6. 张聪聪. 二阶椭圆界面问题的混合元方法及其理论分析[D]. 山东: 山东师范大学, 2011.
7. 彭玉成. 有限元方法若干问题研究[D]. 郑州: 郑州大学, 2006.
8. 张帅. 三维空间中二阶椭圆界面问题的浸入有限元方法[D]. 山东: 山东师范大学, 2012.
9. Frei Stefan, Richter Thomas. A locally modified parametric finite element method for interface problems[J]. SIAM Journal on Numerical Analysis, 2014, 52(5).
10. Haifeng Ji, Jinru Chen,Zhilin Li. A Symmetric and Consistent Immersed Finite Element Method for Interface Problems.[J]. J. Sci. Comput., 2014, 61(3).
11. 高艳妮. 界面问题的有限体积元法研究[D]. 吉林: 吉林大学, 2016.

# 致 谢

时光荏苒, 岁月如梭, 转眼间大学生活来到了最后阶段. 当我写完这篇毕业论文的时候, 有一种如释重负的感觉, 感慨颇多. 回首大学四年, 得到过太多人的帮助了. 首先诚挚的感谢我的论文指导老师王华老师. 本文的研究工作都是在王华老师的悉心指导下完成的. 王老师平易近人, 严谨务实, 由于我知识储备不足, 在论文撰写过程中遇到了许多困难和疑惑, 王华老师都及时给予指点, 耐心解释所犯的错误, 投入了大量的心血和精力, 更是不厌其烦地帮我察看论文中的小漏洞. 王华老师对我的帮忙和关怀实在无法用言语表明. 还要感谢所有的老师们, 正是因为有了他们的督促和教导才能让我在这四年的学习生活里受益匪浅, 快速汲取专业知识, 提升专业能力. 同时也要感谢组内的同学们, 是他们以极大的热情来解答我在理论和程序上的疑问, 帮忙收集资料, 让平淡的日子不再那么枯燥乏味. 最后还要感谢我的家人, 是他们的支持与付出才给了我学习的机会, 感谢一直对我的理解, 这是我不断前进的动力.

# 附录A 论文检测报告

# 





# 附录B 数值程序

附录1. 编写intervalmesh. m文件：

function [node, h, elem, bdFlag] = intervalmesh(a, b, h)

node = a:h:b;

node = node';

N = length(node);

elem = [1:N-1;2:N];

elem = elem';

bdFlag = false(N, 1);

bdFlag([1, N]) = true;

附录2. 编写getH1error1d. m文件：

function H1err = getH1error1d(node, k, h, jd, beta1, beta2, elem, uh, quadOrder)

NT = size(elem, 1);

lens = node(elem(:, 2))-node(elem(:, 1));

Dphi = [-1. /lens, 1. /lens];

[lambda, weight] = quadpts1d(quadOrder);

phi = lambda;

lens = node(elem(:, 2))-node(elem(:, 1));

Dphi = [-1. /lens, 1. /lens];

nQuad = length(weight);

Dexactu1=@(x) (sin(pi. \*x)+(x-jd). \*pi. \*cos(pi. \*x)). /beta1;

Dexactu2=@(x) (sin(pi. \*x)+(x-jd). \*pi. \*cos(pi. \*x)). /beta2;

uh1=uh(1:k+1, 1);

uh2=uh(k:end, 1);

Duh1=uh(elem(1:k, 1)). \*Dphi(1:k, 1) + uh(elem(1:k, 2)). \*Dphi(1:k, 2);

Duh2=uh(elem(k+1:end, 1)). \*Dphi(k+1:end, 1)+uh(elem(k+1:end, 2)). \*Dphi(k+1:end, 2);

H1err1=zeros(k, 1);

for i = 1:nQuad

px = node(elem(1:k, 1))\*phi(i, 1) + node(elem(1:k, 2))\*phi(i, 2);

H1err1 = H1err1 + weight(i). \*(Dexactu1(px)-Duh1). ^2;

end

H1err2=zeros(1/h-k, 1);

for i = 1:nQuad

px=node(elem(k+1:end, 1))\*phi(i, 1)+node(elem(k+1:end, 2))\*phi(i, 2);

H1err2 = H1err2 + weight(i). \*(Dexactu2(px)-Duh2). ^2;

end

H1err=[H1err1;H1err2];

H1err=H1err. \*lens;

H1err = sqrt(sum(H1err));

附录3. 编写getL2error1d. m文件：

function L2err = getL2error1d(node, k, h, jd, beta1, beta2, elem, uh, quadOrder)

NT = size(elem, 1);

lens = node(elem(:, 2))-node(elem(:, 1));

Dphi = [-1. /lens, 1. /lens];

[lambda, weight] = quadpts1d(quadOrder);

phi = lambda;

nQuad = length(weight);

uh1=uh(1:k+1, 1);

uh2=uh(k:end, 1);

exactu1=@(x) (x-0. 35). \*sin(pi\*x). /beta1;

exactu2=@(x) (x-0. 35). \*sin(pi\*x). /beta2;

err1=zeros(k, 1);

err2=zeros(1/h-k, 1);

for i = 1:nQuad

uhp = uh(elem(1:k, 1))\*phi(i, 1) + uh(elem(1:k, 2))\*phi(i, 2);

px = node(elem(1:k, 1))\*phi(i, 1) + node(elem(1:k, 2))\*phi(i, 2);

err1 = err1+ weight(i). \*(exactu1(px) - uhp). ^2;

end

for i = 1:nQuad

uhp = uh(elem(k+1:end, 1))\*phi(i, 1) + uh(elem(k+1:end, 2))\*phi(i, 2);

px = node(elem(k+1:end, 1))\*phi(i, 1) + node(elem(k+1:end, 2))\*phi(i, 2);

err2 = err2 + weight(i). \*(exactu2(px) - uhp). ^2;

end

L2err=[err1;err2];

L2err = L2err. \*lens;

format short

L2err = sqrt(sum(L2err));

附录4. 编写quadpts1d. m文件：

function [lambda, weight] = quadpts1d(order)

numPts = ceil((order+1)/2);

if numPts > 10

numPts = 10;

end

switch numPts

case 1

A = [0 2. 0000000000000000000000000];

case 2

A = [0. 5773502691896257645091488 1. 0000000000000000000000000

-0. 5773502691896257645091488 1. 0000000000000000000000000];

case 3

A = [0 0. 8888888888888888888888889

0. 7745966692414833770358531 0. 5555555555555555555555556

-0. 7745966692414833770358531 0. 5555555555555555555555556];

case 4

A = [0. 3399810435848562648026658 0. 6521451548625461426269361

0. 8611363115940525752239465 0. 3478548451374538573730639

-0. 3399810435848562648026658 0. 6521451548625461426269361

-0. 8611363115940525752239465 0. 3478548451374538573730639];

case 5

A = [0 0. 5688888888888888888888889

0. 5384693101056830910363144 0. 4786286704993664680412915

0. 9061798459386639927976269 0. 2369268850561890875142640

-0. 5384693101056830910363144 0. 4786286704993664680412915

-0. 9061798459386639927976269 0. 2369268850561890875142640];

case 6

A = [0. 2386191860831969086305017 0. 4679139345726910473898703

0. 6612093864662645136613996 0. 3607615730481386075698335

0. 9324695142031520278123016 0. 1713244923791703450402961

-0. 2386191860831969086305017 0. 4679139345726910473898703

-0. 6612093864662645136613996 0. 3607615730481386075698335

-0. 9324695142031520278123016 0. 1713244923791703450402961];

case 7

A = [0 0. 4179591836734693877551020

0. 4058451513773971669066064 0. 3818300505051189449503698

0. 7415311855993944398638648 0. 2797053914892766679014678

0. 9491079123427585245261897 0. 1294849661688696932706114

-0. 4058451513773971669066064 0. 3818300505051189449503698

-0. 7415311855993944398638648 0. 2797053914892766679014678

-0. 9491079123427585245261897 0. 1294849661688696932706114];

case 8

A = [0. 1834346424956498049394761 0. 3626837833783619829651504

0. 5255324099163289858177390 0. 3137066458778872873379622

0. 7966664774136267395915539 0. 2223810344533744705443560

0. 9602898564975362316835609 0. 1012285362903762591525314

-0. 1834346424956498049394761 0. 3626837833783619829651504

-0. 5255324099163289858177390 0. 3137066458778872873379622

-0. 7966664774136267395915539 0. 2223810344533744705443560

-0. 9602898564975362316835609 0. 1012285362903762591525314];

case 9

A = [0 0. 3302393550012597631645251

0. 3242534234038089290385380 0. 3123470770400028400686304

0. 6133714327005903973087020 0. 2606106964029354623187429

0. 8360311073266357942994298 0. 1806481606948574040584720

0. 9681602395076260898355762 0. 0812743883615744119718922

-0. 3242534234038089290385380 0. 3123470770400028400686304

-0. 6133714327005903973087020 0. 2606106964029354623187429

-0. 8360311073266357942994298 0. 1806481606948574040584720

-0. 9681602395076260898355762 0. 0812743883615744119718922];

case 10

A = [0. 1488743389816312108848260 0. 2955242247147528701738930

0. 4333953941292471907992659 0. 2692667193099963550912269

0. 6794095682990244062343274 0. 2190863625159820439955349

0. 8650633666889845107320967 0. 1494513491505805931457763

0. 9739065285171717200779640 0. 0666713443086881375935688

-0. 1488743389816312108848260 0. 2955242247147528701738930

-0. 4333953941292471907992659 0. 2692667193099963550912269

-0. 6794095682990244062343274 0. 2190863625159820439955349

-0. 8650633666889845107320967 0. 1494513491505805931457763

-0. 9739065285171717200779640 0. 0666713443086881375935688];

end

lambda1 = (A(:, 1)+1)/2;

lambda2 = 1 - lambda1;

lambda = [lambda1, lambda2];

weight = A(:, 2)/2;

附录5. 编写Poisson1d. m文件：

function [uh, k] = Poisson1d(node, jd, h, elem, bdFlag, beta1, beta2)

N = size(node, 1); NT = size(elem, 1); Ndof = N;

lens = node(elem(:, 2))-node(elem(:, 1));

Dphi = [-1. /lens, 1. /lens];

%找出界面点所在的单元IK [xk-1, xk]

%需要修改的单元 得出K

for k=0:1/h

if h\*k<jd

k=k+1;

else

break

end

end

D=h-(beta2-beta1)\*(h\*k-jd)/beta2;

bet=beta1/beta2;

%构建A

A = sparse(NT, NT);

for m=1:1/h

if m==1

A(m, m)=2\*beta1/h;

A(m+1, m)=-beta1/h; %D边界条件

elseif m==1/h

A(m-1, m)=-beta2/h;

A(m, m)=2\*beta2/h;

elseif m==k

A(m-1, m)=-beta1/D^2\*(jd-(k-1)\*h)-beta2\*bet^2/D^2\*(k\*h-jd);

A(m, m)=beta1/D^2\*(jd-(k-1)\*h)+beta2\*(k\*h-jd)\*bet^2/D^2+beta2/h;

A(m+1, m)=-beta2/h;

elseif m==k-1

A(m-1, m)=-beta1/h;

A(m, m)=beta1/h+beta1\*(jd-(k-1)\*h)/D^2+beta2\*bet^2\*(k\*h-jd)/D^2;

A(m+1, m)=-beta1/D^2\*(jd-(k-1)\*h)-beta2\*bet^2/D^2\*(k\*h-jd);

elseif m<k-1

A(m-1, m)=-beta1/h;

A(m, m)=2\*beta1/h;

A(m+1, m)=-beta1/h;

elseif m>k

A(m-1, m)=-beta2. /h;

A(m, m)=2\*beta2. /h;

A(m+1, m)=-beta2. /h;

end

end

%标准高斯积分点及其权重

lambda=[0 0.7745966692414833770358531 -0. 7745966692414833770358531 ];

weight=[0. 8888888888888888888888889 0. 5555555555555555555555556 0. 5555555555555555555555556]; %高斯积分点及其权重

b=zeros(NT, 1);

for i=1:1/h

if i==1/h

s = h. \*lambda. /2+(i\*h+(i-1)\*h)/2;

jac = h/2;% dx = jac \* ds

fun=@(x) [pi^2. \*(x-0. 35). \*sin(pi. \*x)-2. \*pi. \*cos(pi. \*x)]. \*((x-(i-1). \*h). /h);

mf = fun(s);

f = weight. \* mf . \* jac;

f=sum(f);

elseif i==k-1

%(k-2, k-1)

s = h. \*lambda. /2+((k-1)\*h+(k-2)\*h)/2;

jac = h/2;% dx = jac \* ds

fun=@(x) [pi^2. \*(x-0. 35). \*sin(pi. \*x)-2\*pi. \*cos(pi. \*x)]. \*((x-(k-2)\*h). /h);

mf = fun(s);

f = weight. \* mf . \* jac;

%k-1, jd

s2=(jd-(k-1)\*h). \*lambda. /2+(jd+(k-1)\*h)/2;

jac2=(jd-(k-1)\*h)/2;

fun2=@(x) [pi^2. \*(x-0. 35). \*sin(pi. \*x)-2\*pi. \*cos(pi. \*x)]. \*(1+((k-1)\*h-x). /D);

mf2=fun2(s2);

f2= weight. \* mf2 . \* jac2;

%(jd, k)

s3=(k\*h-jd). \*lambda. /2+(jd+k\*h)/2;

jac3=(k\*h-jd)/2;

fun3=@(x) [pi^2. \*(x-0. 35). \*sin(pi. \*x)-2\*pi. \*cos(pi. \*x)]. \*(bet. \*(k\*h-x). /D);

mf3=fun3(s3);

f3= weight. \* mf3 . \* jac3;

f=sum(f)+sum(f2)+sum(f3);

elseif i==k

%(k-1, jd)

s = (jd-(k-1)\*h). \*lambda. /2+(jd+(k-1)\*h)/2;

jac = (jd-(k-1)\*h)/2;% dx = jac \* ds

fun=@(x) [pi^2. \*(x-0. 35). \*sin(pi. \*x)-2\*pi. \*cos(pi. \*x)]. \*((x-(k-1)\*h). /D);

mf = fun(s);

f = weight. \* mf . \* jac;

%jd, k

s2=(k\*h-jd). \*lambda. /2+(jd+k\*h)/2;

jac2=(k\*h-jd)/2;

fun2=@(x) [pi^2. \*(x-0. 35). \*sin(pi. \*x)-2\*pi. \*cos(pi. \*x)]. \*(1+bet. \*(x-k\*h). /D);

mf2=fun2(s2);

f2= weight. \* mf2 . \* jac2;

%(k, k+1)

s3=h. \*lambda. /2+(k\*h+(k+1)\*h)/2;

jac3=h/2;

fun3=@(x) [pi^2. \*(x-0. 35). \*sin(pi. \*x)-2\*pi. \*cos(pi. \*x)]. \*(((k+1)\*h-x). /h);

mf3=fun3(s3);

f3= weight. \* mf3 . \* jac3;

f=sum(f)+sum(f2)+sum(f3);

else

%i-1, i

s = h. \*lambda. /2+(i\*h+(i-1)\*h)/2;

jac = h/2;% dx = jac \* ds

fun=@(x) [pi^2. \*(x-0. 35). \*sin(pi. \*x)-2\*pi. \*cos(pi. \*x)]. \*((x-(i-1)\*h). /h);

mf = fun(s);

f = weight. \* mf . \* jac;

%i, i+1

s2=h. /2. \*lambda+((i+1)\*h+i\*h)/2;

jac2=h/2;

fun2=@(x) [pi^2. \*(x-0. 35). \*sin(pi. \*x)-2\*pi. \*cos(pi. \*x)]. \*(1-(x-i\*h). /h);

mf2=fun2(s2);

f2= weight. \* mf2 . \* jac2;

f=sum(f)+sum(f2);

end

b(i, 1)=f;

end

%根据边界条件组装A b 并计算uh

zo=zeros(1/h, 1);

zs=zo';

A=[1 zs;zo A];

b=[0;b];

isFixed = bdFlag;

isFree = ~isFixed;

uh = zeros(1/h+1, 1);

uh(isFixed) = 0;

b = b - A\*uh;

uh(isFree) = A(isFree, isFree)\b(isFree);

附录6. 编写脚本文件main\_test

h = 1/8;

jd=0. 35;

beta1=1;

beta2=10;

a = 0;

b = 1;

quadOrder = 3;

maxIt = 5;

N = zeros(maxIt, 1);

L2err=zeros(maxIt, 1);

H1err=zeros(maxIt, 1);

for i = 1:maxIt

[node, h, elem, bdFlag] = intervalmesh(a, b, h/2^(i-1));

[uh, k] = Poisson1d(node, jd, h, elem, bdFlag, beta1, beta2);

N(i) = size(elem, 1);

name = ['solut on' int2str(N(i))];

save(name, 'node', 'elem', 'uh');

L2err(i, 1)= getL2error1d(node, k, h, jd, beta1, beta2, elem, uh, quadOrder);

H1err(i, 1) = getH1error1d(node, k, h, jd, beta1, beta2, elem, uh, quadOrder);

end

figure(1)

plot(node, uh)

% %精确解图

figure(2)

nn=0:0. 01:jd;

exactu1=(nn-jd). \*sin(pi\*nn). /beta1;

plot(nn, exactu1)

hold on

nn2=jd:0. 01:1;

exactu2=(nn2-jd). \*sin(pi\*nn2). /beta2;

plot(nn2, exactu2)

format short

L2err

H1err