数理方程经典问题专题整理

非齐次问题的处理方法

方程和定解条件的齐次和非齐次是这门课程中的重要基本概念之一. 我们之前有探讨过,其实对于任意一个定解问题,我们可以给它很多个标签,从不同角度来描述这个定解问题. 比如,方程的类型、方程的阶、方程是否齐次、定解条件是否齐次、方程中变量是否有界、方程中坐标变量整体所描述的区域是否有界,等等.

而一般情况下在学习一种求解方法或者处理一个特定问题时,我们往往只关注某些标签. 比如,我们在使用分离变量法时,往往关注的是方程中是否存在定义域是有界的变量、方程是否齐次、定解条件是否齐次、是否是二阶方程 (这个主要考虑到固有值问题要满足施刘定理,但是对于不满足情形我们之前作业中也有遇到,在作业讲解和复习指导中有详细解释如何处理这类问题. 另外这门课程主要研究二阶线性方程问题,所以往往都是二阶的固有值问题).

在这些概念中方程和定解条件的齐次和非齐次是非常重要的一组概念,这一专题专门讨论如何理解这一概念以及对应地在处理定解问题中的应用.

首先,在这门课程中,有这些地方对于方程和定解条件的齐次提出了要求,分别是

- 行波法处理的一维无界区域波动方程的泛定方程要求是齐次的
- 齐次化原理在使用的时候 (用于将非齐次发展方程转化为齐次方程) 要求初始条件是齐次的
- 分离变量法在处理有界区域问题时要求方程和边界条件是齐次的

在处理非齐次方程的时候, 我们主要学习三种方法, 分别是

- 固有函数展开法 (只能用于配合分离变量法处理有界区域的问题)
- 齐次化原理(只能用于处理非齐次发展方程,并且要求初始条件齐次)
- 特解法 (从原则上讲可以用于处理任何问题,但考虑到特解求解的难度,我们一般认为主要是用于求解非齐次项为独立变量线性组合形式的非齐次问题)

在处理非齐次定解条件的时候, 我们主要学习的方法就是

• 特解法 (和处理非齐次方程的时候方法是一致的)

在这一专题,我们详细地说明这几种方法的本质和使用注意事项.

固有函数展开法

- 本质:将非齐次项在固有函数系上展开,从而通过比较系数得到解在固有函数系上展开的结果,进而归入齐次方程处理流程
- 适用范围: 有界区域问题,需配合分离变量法使用,只能处理非齐次方程齐次边界问题
- 使用方法:将非齐次项在求解得到的固有函数系上展开,以非齐次项形式并入其他常微分方程的求解,求解其他常微分方程后得到解在固有函数系上展开的系数,例如 $T_n(t)$,和固有函数系对应项相乘并线性叠加,进而得到形式解.之后的系数求解和一般分离变量法中的求解过程是一致的

齐次化原理

- 本质:基于时间变量分片使用微元法,建立原始定解问题和时间微元内的定解问题的对应关系,即通过转化为时间微元内的定解问题并求解,最后通过叠加方法(积分)转化为定解问题的解
- 适用范围: 非齐次发展方程, 齐次初始条件 (如果是非齐次初始条件, 只要满足发展方程, 也可以考虑使用叠加原理处理)
- 使用方法:根据定解问题写出时间分片内的定解问题,求解得到的解代入积分公式得到定解问题的解.在这里要注意,时间变量偏移问题

特解法

- 本质:基于叠加原理,将定解问题分解为多个(一般是两个)定解问题分别求解. 当我们发现用已知方法无法直接求解非齐次问题时,我们的一个直观想法就是将 其转化为齐次问题,进而可以直接求解.所以,我们转化的目标就是齐次定解问 题.所以现在,问题变成,如何将非齐次问题转化为齐次问题.一个直观的想法是, 应用叠加原理,将非齐次问题写成一个齐次问题和一个非齐次问题的叠加.这里, 齐次问题可以直接求解,非齐次问题对应的就是特解满足的定解问题
- 适用范围:任何线性定解问题.但这只是从理论上说,因为任何线性问题都满足叠加原理,从而都可以使用特解法.但是考虑到特解的求解难易程度,也就是转化后的两个问题中,非齐次问题的求解难易程度,所以我们得到的结论是,一般情况下,我们使用特解法主要是求解非齐次项是独立变量的线性组合形式的问题.这个结论的出发点在于,如果我们处理的非齐次项含有多个变量且不是独立变量线性组合情形,那么,解一定是含有多个变量的,所以非齐次问题仍然不能退化为常微分方程问题.而我们最初使用特解法就是为了用一个特解,将我们不会求

解的非齐次问题转化为齐次问题从而求解,那么如果转化后特解满足的定解问题和原来定解问题类似,仍然是非齐次的偏微分方程问题,那么其实特解的求解难度仍然是非常大的(一般很难求解). 所以结论: 特解法一般只能用于处理非齐次项是若干独立变量线性组合情形,而且如果是这种情形,往往第一选择是特解法

• 使用方法:根据非齐次项的特点,往往是独立变量线性组合情形,则特解可以设成对应的独立变量函数的线性组合情形,代入其满足的定解问题(将原定解问题转化为齐次问题和非齐次问题叠加中的非齐次问题),即可得到特解.求解出齐次问题,将解叠加即可得到原定解问题的解

