**基于面向对象的偏微分方程求解项目设计说明**

**1. 项目概述**

本项目旨在构建一个可扩展的偏微分方程（PDE）求解系统，支持多种 PDE 类型（如拉普拉斯方程、椭圆型方程、抛物型方程等）和数值求解方法（有限差分法等）。系统采用面向对象设计，通过封装、继承和多态实现代码复用与扩展，并结合工厂模式解耦对象创建与使用，最终实现高效灵活的 PDE 求解流程。

**2. 核心类设计与成员说明**

**2.1 基础几何与数据类**

**Mesh 类（网格基类）**

**成员变量：**start（起始位置）、h（间距）、N（点数）、x\_vec（坐标数组）。​

**成员函数**：\_\_init\_\_（初始化网格）、\_\_str\_\_（返回网格信息）。

**Field 类（场数据类，继承自Mesh）**

**成员变量：**继承Mesh变量，新增val（场值数组）。

**成员函数：**\_\_init\_\_（初始化场值）、\_\_str\_\_（返回场信息）。

**Source 类（源项类，继承自Mesh）**

**成员变量：**继承Mesh变量，新增val（源项值数组）。​

**成员函数：**\_\_init\_\_（初始化源项）、make\_sin（生成正弦源项）、make\_gaussian（生成高斯源项）、\_\_str\_\_（返回源项信息）。

**2.2 PDE 核心类**

**PDE 类（偏微分方程基类）**

**成员变量：**mesh（网格对象）、source（源项对象）、coefficients（系数字典）、boundary\_conditions（边界条件字典）。​

**成员函数：**\_\_init\_\_（初始化参数）、set\_boundary\_conditions（设置边界条件）、set\_coefficients（设置系数）、get\_solution\_domain（抽象方法，获取解域）。

**EllipticPDE 类（椭圆型 PDE，继承自PDE）**

**成员变量：**继承PDE变量，新增elliptic\_params（椭圆型特有参数）。**​**

**成员函数：**\_\_init\_\_（初始化参数）、set\_elliptic\_parameters（设置特有参数）、get\_solution\_domain（返回静态解域）。

**LaplaceEq 类（拉普拉斯方程，继承自EllipticPDE）**

**成员变量：**继承EllipticPDE变量，新增u（解场对象）、A（差分矩阵）。**​**

**成员函数：**\_\_init\_\_（初始化解场和矩阵）、build\_diff\_matrix（构建差分矩阵）、solve（求解方程组）、save\_fig（保存结果图像）。

**ParabolicPDE 类（抛物型 PDE，继承自PDE）**

**成员变量：**继承PDE变量，新增time\_step（时间步长）、time\_range（时间范围）。

**成员函数：**\_\_init\_\_（初始化时间参数）、set\_time\_parameters（设置时间参数）、get\_solution\_domain（返回时空解域）。

**2.3 求解器类**

**PDESolver 类（PDE 求解器基类）**

**成员变量：**discretization\_method（离散化方法）、tolerance（精度容限）。

**成员函数：**\_\_init\_\_（初始化方法和精度）、set\_discretization\_method（修改方法）、solve（抽象方法，求解 PDE）。

**FiniteDifferenceSolver类（有限差分求解器，继承自PDESolver）**​

**成员变量**：继承PDESolver变量，新增grid\_size（网格尺寸）、space\_step（空间步长）。​

**成员函数**：\_\_init\_\_（初始化网格参数）、solve（实现有限差分求解）。

**2.4 工厂类**

**PDEFactory 类（PDE 与求解器工厂）**

**成员函数：**create\_pde（静态方法，按类型创建 PDE 对象）、create\_solver（静态方法，按类型创建求解器）。

**3.类间继承关系**

**PDE 体系**：PDE为基类，EllipticPDE、ParabolicPDE继承自PDE，LaplaceEq继承自EllipticPDE。​

**基础数据类**：Mesh为基类，Field、Source继承自Mesh。​

**求解器体系**：PDESolver为基类，FiniteDifferenceSolver继承自PDESolver。

**4. 程序设计范式应用**

**4.1 面向对象范式**

**封装：**类封装特定职责（如Mesh封装网格信息，LaplaceEq封装求解逻辑），隐藏内部状态，暴露必要接口。**​**

**继承：**子类复用父类功能（如Field复用Mesh的坐标生成），减少冗余。​

**多态：**抽象方法在子类中实现不同逻辑（如get\_solution\_domain在椭圆型、抛物型 PDE 中返回不同解域）。

**4.2 工厂模式**

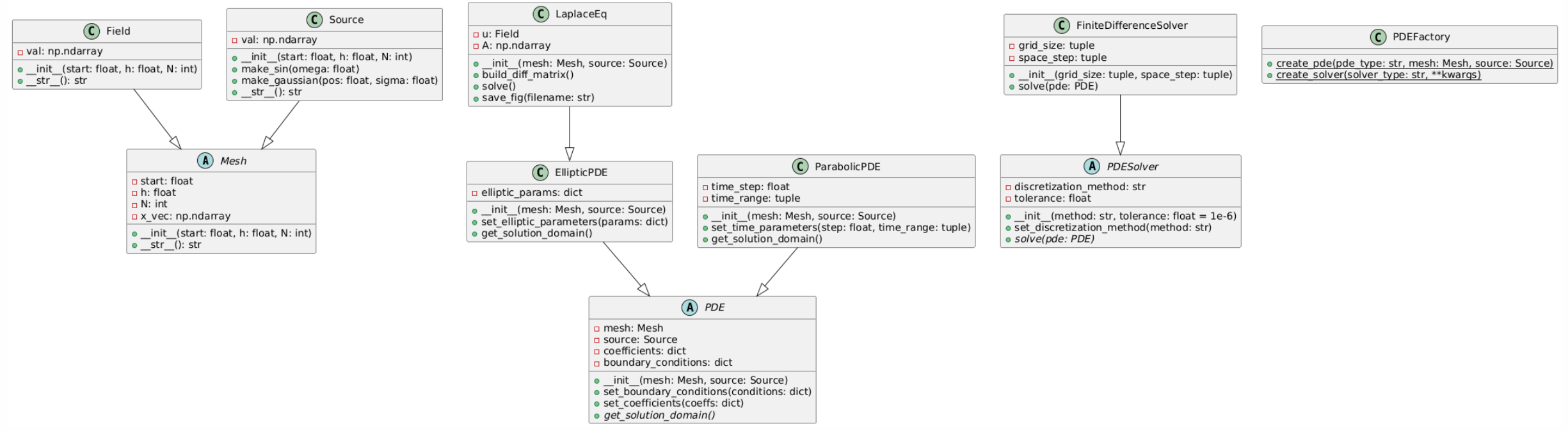
解耦对象创建与使用，通过PDEFactory创建对象，客户端无需依赖具体类。​

新增 PDE 类型时，仅需扩展工厂方法，符合 “开闭原则”。

**4.3 依赖倒置原则**

高层模块（如主程序）依赖PDE、PDESolver等抽象基类，而非具体子类（如LaplaceEq）；

低层模块（如LaplaceEq、FiniteDifferenceSolver）实现抽象接口，确保高层模块稳定性（如主程序可通过PDEFactory动态切换求解的方程类型）。



**GitHub 仓库说明**

**仓库信息**

**仓库名称**：pde-solver-oop

**仓库 URL**：<https://github.com/>hyh101923/pde-solver-oop