样条曲线设计文档

一、设计思路

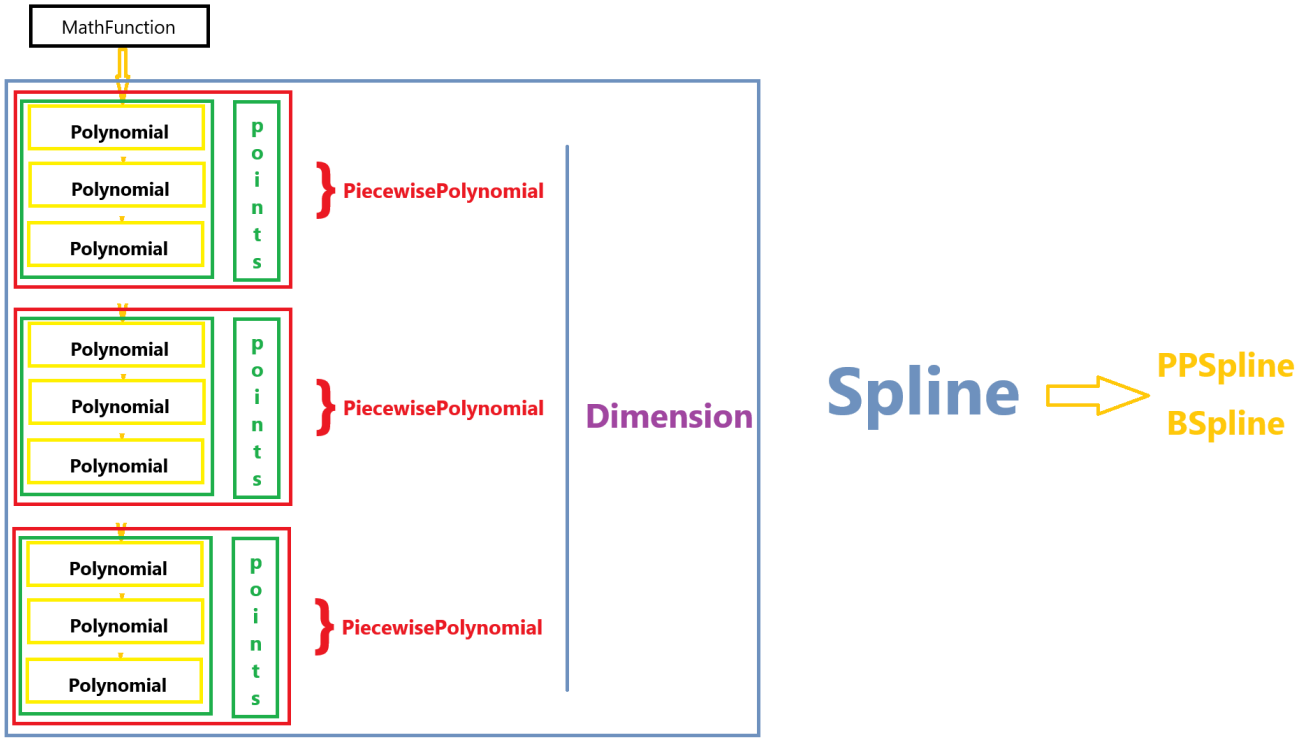
1. **抽象与继承**：
   - 抽象出通用的 `MathFunction` 类和 `Spline` 基类，具体样条曲线通过继承实现扩展。
2. **模块化与复用性**：
   - 每个功能单元独立封装，如多项式运算、分段插值和样条生成，提升模块的复用性。
3. **数值稳定性与效率**：
   - 封装 LAPACK 库加速线性方程组求解，提高数值稳定性和效率。

二、程序结构说明

程序主要分为以下模块：

类	描述	属性/功能
MathFunction	表示数学函数，支持函数值计算。	—
Polynomial	表示多项式，支持多项式的创建、运算及求导。	`coefficients` - 各项系数
PiecewisePolynomial	表示分段多项式，支持分段多项式的创建、运算及求导。	`polynomials` - 各分段的多项式 `points` - 分段起点
Spline	定义通用样条曲线接口，支持参数方程表示。	`dimensions` - 参数数量 `spline_order` - 多项式次数
PPSpline	继承自 `Spline`，实现分段多项式样条。	—
BSpline	继承自 `Spline`，实现 B 样条的基函数计算及生成。	—
SphereSpline	用于处理球面上的样条曲线。	—
辅助函数	提供工具函数：弦长计算、坐标变换、边界条件解析等。	—
方程求解器	使用 LAPACK 库求解线性方程组。	—
JSON 解析器	用于读取 JSON 文件，构造样条曲线。	—

注： Spline类的dimensions属性表示参数方程参数的数量。例如:
参数方程 r(t) = (x(t), y(t)) 的 dimensions = 2，
普通的函数 y = y(x) 的 dimensions = 1。



三、类的功能接口

类	接口
`MathFunction` 类	`MathFunction(double (*func)(double x))`：通过函数指针初始化函数 `double evaluate(double x)`：计算函数在点 xx 的值。
`Polynomial` 类	**构造多项式** `Polynomial(const std::vector<double>& coef)`：直接使用系数构造 `Polynomial(const std::vector<double>& x_values, const std::vector<double>& y_values)`：通过 Newton 插值构造 **输出表达式** `void print()`：从低到高输出多项式各项系数 **运算操作** 重载 `+`，`-`，`*` 运算符 **求导与计算** `Polynomial derivative()`：返回导数 `double evaluate(double x)`：计算多项式值。
`PiecewisePolynomial` 类	**构造分段多项式** `PiecewisePolynomial(const std::vector<Polynomial>& p, const std::vector<double>& x)`：构造分段多项式 **求值与打印** `double evaluate(double x)`：计算分段多项式值 `void print()`：输出分段公式。
`Spline` 类	`std::vector<double> operator()(double t)`：计算样条在参数 tt 处的值 `void print()`：输出样条公式。
`PPSpline` 类	**构造方法** 1.针对给定的函数和分段点进行构造

类	接口
	2.针对给定函数、区间、分段数进行构造，包括均匀选点和累积弦长法选点 3.通过读取json文件构造样条 4.通过给定的散点坐标构造样条
`BSpline` 类	**构造方法** 1.针对给定的函数和分段点进行构造 2.针对给定函数、区间、分段数进行构造，包括均匀选点和累积弦长法选点 3.通过读取json文件构造样条 4.通过给定的散点坐标构造样条