插值、拟合与应用

杨勇

南京航空航天大学理学院

07/13/2020

Section 1

Julia 简介

Julia 语言

年轻但是功能强大

*Julia is a relatively **young** programming language. The initial design work on the Julia project began at **MIT in August 2009**, and by **February 2012**, it became open source...*

- Data science, machine learning, parallel computing, visualization
- Scientific domain:
  - differential equations ecosystem (DifferentialEquations.jl)
  - optimization tools (JuMP.jl and Optim.jl)
  - iterative linear solvers (IterativeSolvers.jl),
  - a robust framework for Fourier transforms (AbstractFFTs.jl),
  - biology (BioJulia), operations research (JuliaOpt),
  - image processing (JuliaImages),
  - quantum physics (QuantumBFS, QuantumOptics),
  - nonlinear dynamics (JuliaDynamics)
  - https://julialang.org/

特性

- 开源
  - Matlab：商业软件；多个工具箱；许多是业界标准
- 速度快
  - Python << Julia < C/C++≈Fortran
- 易调试
  - Python ≈ Julia >>C/C++≈Fortran
- third-party packages 越来越多、越来越好
  - Python >Julia >>C/C++≈Fortran
- 可以用于数学建模：运筹优化、微分方程、概率决策、统计分析、图论等等

安装

- 下载安装
  1. 官网下载 https://julialang.org/downloads/
     - 按操作系统进行选择 Mac/Windows/Linux
  2. Windows 下双击安装，参考 https://datatofish.com/install-julia/
  3. 安装完成，即可使用 REPL (read-eval-print loop)

安装

- 下载安装
  1. 官网下载 https://julialang.org/downloads/
     - 按操作系统进行选择 Mac/Windows/Linux
  2. Windows 下双击安装，参考 https://datatofish.com/install-julia/
  3. 安装完成，即可使用 REPL (read-eval-print loop)
- 编辑器 jupyter notebook
  1. 双击安装 Miniconda
     - 地址 https://docs.conda.io/en/latest/miniconda.html
  2. `conda install jupyter`
  3. 利用 Pkg，安装 `IJulia`
- 其它编辑器如：Juno，VS code，Vim，Emacs ...

基本规则

像 Matlab

- 下标从 1 开始
- for i in 1:N 包含最后一位 N
- 使用 end，故无需缩进对齐
- 多维数组
  - 按列优先存储
  - a=[1,2,3]
  - b=[1;2;3]
  - c=[1 2 3; 4 5 6]

像 Python

- 多种数据结构：list, tuple, Dictionary
- `using DataStructure`

演示实例

1. 求解线性方程组 $AX = B$:

$$\begin{bmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \cdots & 0 \\ 0 & -1 & 2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 2 \end{bmatrix}_{N\times N} X = \begin{bmatrix} 1/(N+1)^2 \\ . \\ . \\ \vdots \\ . \end{bmatrix}_{N\times 1}$$

2. 速度比较: $N = 100, N = 1000$

不同点

- Macro: `@time`, `@show`
- Multiple dispatch
  - 使用起来像 Python 的 Duck-typing
  - 内部实现不同，故速度有提升
  - 比如

    ```
    function power_2(x)
        return x^2
    end
    ```

  - Types

Section 2

问题

案例 1

- 黄河小浪底调水调沙问题

试验获得一组数据，是关于“时间–水流量–排沙量”（此处省略）。根据试验数据建立数学模型研究下面问题：

1. 任意时刻排沙量及总排沙量的方法
2. 确定排沙量与水流量的关系

- 求解思路
  1. 建立关系 $s(t)$
  2. 计算总排沙量 $\int_a^b s(t)dt$
  3. 建立关系 $s(w)$
- 参考：数学建模算法与程序，司守奎

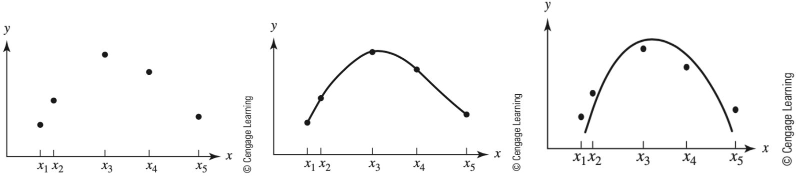
案例 2

- 估计水塔水流量

试验获得一组数据，是关于“时间–水塔水位”（此处省略该数据表、及水塔形状参数信息）。问题：试估计在任何时刻，水从水塔流出的流量 $f(t)$，并估计一天的总用水量。

- 思路
  1. 处理数据、注意数据单位、分析开始充水和停止充水时间
  2. 由水位 h_i得到体积 V_i，建立关系 $V(t)$，得到流量 $f(t) = |V'(t)|$
  3. 或者，使用差分法得到 f_i，再建立关系 $f(t)$
  4. 总用水量为 $\int_{one.day} f(t)dt$
- 参考：数学建模竞赛教程，李尚志

插值与拟合



- 插值 = 过已知数据点，求近似函数；拟合 = 不要求过数据点，求近似函数
- 相同点：由离散数据点，构造函数作为近似
- 不同点：相信模型、还是相信数据；数学方法不同
- 如何选择：对实际问题，如何选择，有时容易确定，有时并不明显
- 其它建立函数的方法：机器学习、建立 neural network、训练参数获得关系 $cat.or.dog = f(image)$

Section 3

插值

例题

已知相应年份的人口数据（单位：千人）如下表

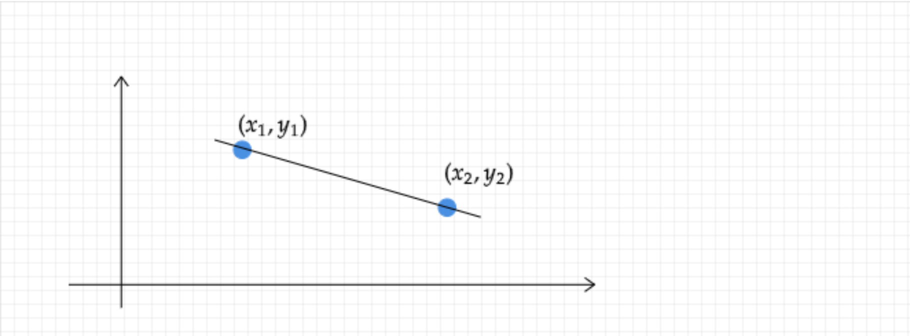
1960	1970	1980	1990	2000	2010
179323	203302	226524	249633	281422	308746

如何估计 1975 年的人口数？

- 可以使用插值求解。

线性插值

已知两个点 $(x_1, y_1), (x_2, y_2)$，如何得到一个线性函数？



解

1. 假设该直线方程是 $y = a_1 + a_2 x$
2. 因为两个点在该直线上，所以得到关于未知量 a_1, a_2的线性方程组

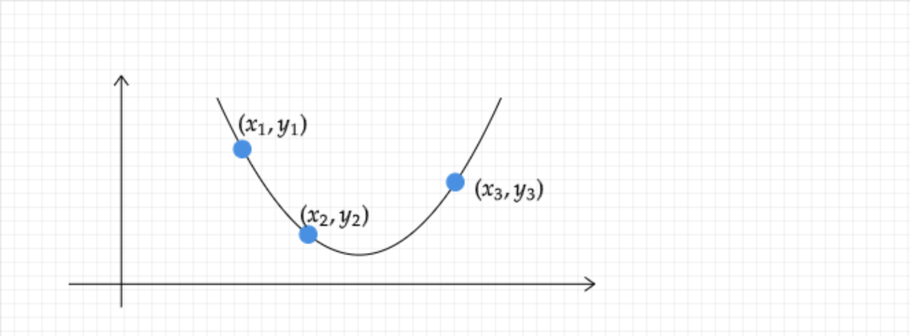
$$\begin{cases} a_1 + a_2 \ x_1 & = & y_1 \\ a_1 + a_2 \ x_2 & = & y_2 \end{cases}$$

3. 求解该线性方程组，得到 a_1, a_2。也就得到了该直线方程 $y = a_1 + a_2 x$
4. 存在唯一性 iff $x_1 \neq x_2$
5. 也可以写成

$$y = y_1 \frac{x - x_2}{x_1 - x_2} + y_2 \frac{x - x_1}{x_2 - x_1}$$

二次插值

已知三个点 $(x_1, y_1), (x_2, y_2), (x_3, y_3)$，如何得到一个二次多项式函数？



- 同样的推理，得到

$$y = y_1 \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} + y_2 \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)} + y_3 \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)}$$

n 次插值多项式

- 已知 $(n+1)$点 (x_i, y_i)，如果 x_i不同，则过这些点，存在唯一的 n多项式。可以表示为

$$P(x) = \sum_{i=0}^{n} y_i \Pi_{k=0,k\neq i}^{n} \frac{x - x_k}{x_i - x_k}$$

- 误差：如果 $y_i = f(x_i), f \in C^{n+1}[a,b], x_i \in [a,b]$，则对于任意 $x \in [a,b]$

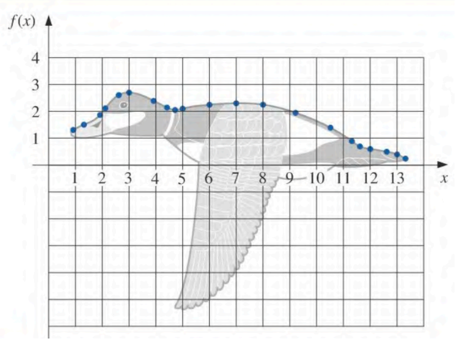
$$f(x) = P(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x - x_0)\cdots(x - x_n)$$

- 多项式次数不是越多越好
  - $f^{(n+1)} \approx \mathcal{O}(n!)$，除非整函数
  - $(x - x_0)\cdots(x - x_n)$

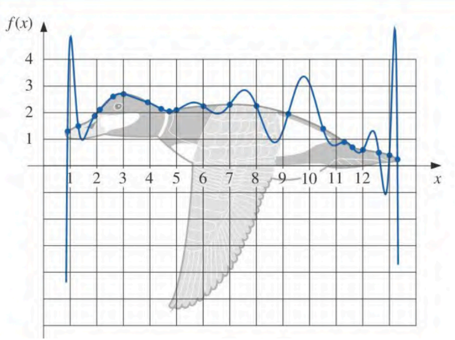
例题

如何利用数据，估计鸟的背部流线外形

x	0.9	1.3	1.9	2.1	2.6	3.0	3.9	4.4	4.7	5.0	6.0	7.0	8.0	9.2	10.5	11.3	11.6	12.0	12.6	13.0	13.3
$f(x)$	1.3	1.5	1.85	2.1	2.6	2.7	2.4	2.15	2.05	2.1	2.25	2.3	2.25	1.95	1.4	0.9	0.7	0.6	0.5	0.4	0.25

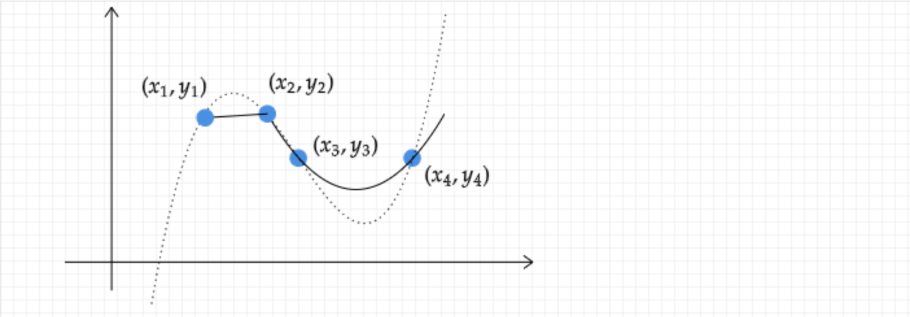


使用 20 次多项式



分片多项式

在每个区间，使用不同的 Lagrange 插值多项式



- 使用 $(x_1, y_1), (x_2, y_2)$线性插值
- 使用 (x_2, y_2),$(x_3, y_3), (x_4, y_4)$二次多项式插值

样条插值

在每个点处，增加导数连续条件

- 三次样条
  - 每个区间 (x_i, x_{i+1})是一个 3 次多项式 $P_i, i = 0, \cdots, n-1$。整体插值函数 $S(x) \in C^2[a, b]$
  - 在每个内点 x_i上，$i = 1, \cdots, n-1$，满足光滑性条件：$P_i(x_{i+1}) = P_{i+1}(x_{i+1}), P_i'(x_{i+1}) = P_{i+1}'(x_{i+1}), P_i''(x_{i+1}) = P_{i+1}''(x_{i+1})$
  - 未知数个数：$4n - 3(n-1) = n + 3$
  - 已知条件 (x_i, y_i)个数:$n + 1$
  - 补充两个条件：
    1. free boundary: $S''(x_0) = 0, S''(x_{n+1}) = 0$
    2. Clamped boundary: 已知 $S'(x_0), S'(x_n)$
  - 称为 Natural spline, clamped spline

存在性、唯一性、误差

- 可以证明：只要有 $(n+1)$个不同点 x_i，则存在唯一的 clamped spline
  - 误差为

$$|f(x) - S(s)| \leq \frac{5}{384} \max_{x\in[a,b]} |f^{(4)}(x)| \max_{0\leq j\leq n-1} (x_{j+1} - x_j)^4$$

- 存在唯一的 natural spline
  - 误差同样为 4 阶，但是表达式复杂。

Natural spline 求解算法

记每个区间 (x_i, x_{i+1})上的 3 次多项式

$P_i = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, i = 0, \cdots, n-1$

1. $a_j = S(x_j) = y_j, j = 0, \cdots, n$
2. 解关于 c_i的线性方程组

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ h_0 & 2(h_0+h_1) & h_1 & \cdots & 0 & 0 & 0 \\ 0 & h_1 & 2(h_1+h_2) & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} 0 \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a_0) \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 0 \end{bmatrix}$$

3. $b_j h_j = a_{j+1} - a_j - \frac{h_j^2}{3}(c_{j+1} + 2c_j)$
4. $d_j h_j = \frac{1}{3}(c_{j+1} - c_j)$

Clamped spline 求解算法

记每个区间 (x_i, x_{i+1})上的 3 次多项式

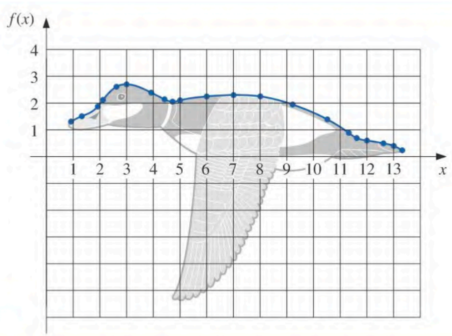
$$P_i = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, i = 0, \cdots, n-1$$

1. $a_j = S(x_j) = y_j, j = 0, \cdots, n$
2. 解关于 c_i的线性方程组

$$\begin{bmatrix} 2h_0 & h_0 & 0 & \cdots & 0 & 0 & 0 \\ h_0 & 2(h_0+h_1) & h_1 & \cdots & 0 & 0 & 0 \\ 0 & h_1 & 2(h_1+h_2) & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & h_{n-1} & 2h_{n-1} \end{bmatrix} \begin{bmatrix} c_0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ c_n \end{bmatrix} = \begin{bmatrix} \frac{3}{h_0}(a_1 - a_0) - 3f'(a) \\ \frac{3}{h_1}(a_2 - a_1) - \frac{3}{h_0}(a_1 - a_0) \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ 3f'(b) - \frac{3}{h_{n-1}}(a_n - a_{n-1}) \end{bmatrix}$$

3. $b_j h_j = a_{j+1} - a_j - \frac{h_j^2}{3}(c_{j+1} + 2c_j)$
4. $d_j h_j = \frac{1}{3}(c_{j+1} - c_j)$

鸟背的外形的三次样条近似



其它包

```
using Pkg
Pkg.add("Dierckx")
```

- wrapper of a Fortran library
- 2D
- High order spline
- Spline fitting using a smooth factor

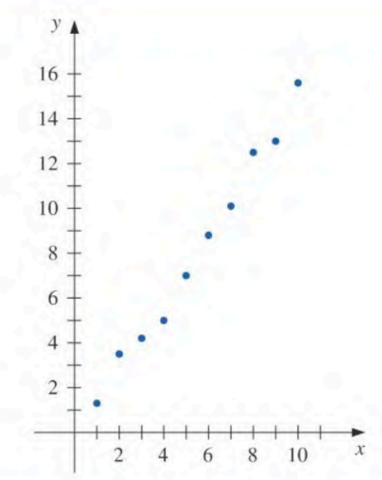
Section 4

拟合

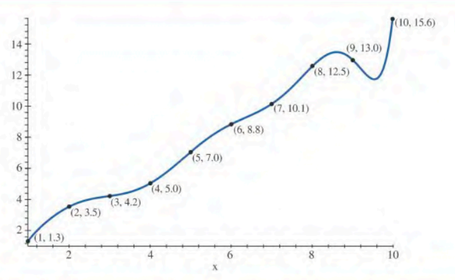
例题

如何近似如下数据

x_i	y_i	x_i	y_i
1	1.3	6	8.8
2	3.5	7	10.1
3	4.2	8	12.5
4	5.0	9	13.0
5	7.0	10	15.6



9 次 Lagrange 多项式插值



线性最小二乘拟合

一种方案是使用**线性最小二乘拟合**：寻找直线 $y = a_0 + a_1 x$最小化误差

$$E(a_0, a_1) = \sum_{i=1}^{10} [y_i - (a_0 + a_1 x_i)]^2$$

- 两个未知数 a_0, a_1
- 两个必要条件 $\begin{cases} 0 = \frac{\partial E}{\partial a_0} = (-2) \sum_{i=1}^{10} (y_i - (a_0 + a_1 x_i)) \\ 0 = \frac{\partial E}{\partial a_1} = (-2) \sum_{i=1}^{10} (y_i - (a_0 + a_1 x_i)) x_i \end{cases}$
- Normal equation $\begin{cases} \left(\sum_{i=1}^{10} 1\right) a_0 + \left(\sum_{i=1}^{10} x_i\right) a_1 = \sum_{i=1}^{10} y_i \\ \left(\sum_{i=1}^{10} x_i\right) a_0 + \left(\sum_{i=1}^{10} x_i^2\right) a_1 = \sum_{i=1}^{10} x_i y_i \end{cases}$

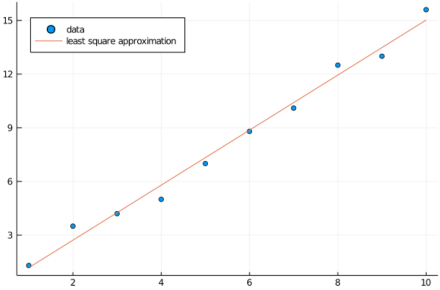
求解公式

定义（内积）

对于 $\mathbb{R}^{10}$ 中向量 $X := (x_1, \cdots, x_{10}), Y := (y_1, \cdots, y_{10})$，内积定义为 $(X, Y) := \sum_{i=1}^{10} x_i y_i$

- Normal equation 可以写成 $\begin{cases} (1,1)a_0 + (1,X)a_1 = (1,Y) \\ (X,1)a_0 + (X,X)a_1 = (X,Y) \end{cases}$
- 求解此线性方程组得到 $\begin{cases} a_0 = \frac{(1,Y)(X,X)-(X,Y)(1,X)}{(1,1)(X,X)-(1,X)^2} \\ a_1 = \frac{(1,1)(X,Y)-(1,Y)(1,X)}{(1,1)(X,X)-(1,X)^2} \end{cases}$
- 可解的充分必要条件是 $1 \nparallel X$；否则直线方程为 $x = x_1$

结果



高阶多项式最小二乘拟合

$$\min E(a_0, a_1, a_2) := \sum_i [y_i - (a_0 + a_1 x_i + a_2 x_i^2)]^2$$

由于 $\frac{\partial E}{\partial a_0} = \frac{\partial E}{\partial a_1} = \frac{\partial E}{\partial a_2} = 0$，得到

Normal equation 可以写成 $\begin{cases} (1,1)a_0 + (1,X)a_1 + (1,X^2)a_2 = (1,Y) \\ (X,1)a_0 + (X,X)a_1 + (X,X^2)a_2 = (X,Y) \\ (X^2,1)a_0 + (X^2,X)a_1 + (X^2,X^2)a_2 = (X^2,Y) \end{cases}$

通过求解此线性方程组，得到 a_0, a_1, a_2

其它类型函数最小二乘拟合

如果使用 $y = be^{ax}$ 逼近数据，

- 方法 1：求解 $\min \sum_{i=1}^{m}(y_i - be^{ax_i})^2$
  - Normal equation $\begin{cases} 0 = \frac{\partial E}{\partial b} = (-2)\sum_{i=1}^{10}(y_i - be^{ax_i})e^{ax_i} \\ 0 = \frac{\partial E}{\partial a} = (-2)\sum_{i=1}^{10}(y_i - be^{ax_i})be^{ax_i}x_i \end{cases}$
  - 困难：关于 a, b 的非线性方程组
- 方法 2：先处理数据
  - $\ln y = \ln b + ax$
  - 求解 $\min \sum_{i=1}^{m}(\ln y_i - B - ax_i)^2$，$b = e^B$
  - 问题不同，但可以使用 normal equation

数据拟合模型

Q：如何选取函数模型呢？

A: 观察数据分布情况；熟悉基本函数 $y = x, y = x^2, y = e^x, y = \ln(x), y = \frac{1}{x}$形状；尝试不同插值。

总结

- Julia 编程的简单介绍
- 两种近似函数方法：插值、拟合
- 二维插值与拟合：$f(x,y) = \sum_i w_i \phi_i(x,y)$
- 同一问题可以尝试不同插值方法（interpolation）、不同拟合（model-fitting）方法
- 参考文献
  - Think Julia https://benlauwens.github.io/ThinkJulia.jl/latest/book.html
  - https://julialang.org/learning/
  - A First Course in Mathematical Modeling by Frank R. Giordano, William P. Fox, Steven B. Horton
  - Numerical Analysis 10th Edition by Richard L. Burden, J. Douglas Faires, Annette M. Burden
- 联系方式：yongyang@nuaa.edu.cn