

- [RBF 基本原理](#)
- [RBF 算法伪代码](#)
- [关于怎样解线性方程组 \$Aw = y\$ 的讨论](#)

RBF 基本原理

RBF 是一种对多维散点数据进行数值拟合的插值方法.

给定一组包含 n 条数据的输入

$$\mathbf{x} \in \mathbb{R}^m, \quad \{(\mathbf{x}_i, y_i)\}_{i=1, \dots, n}$$

RBF 对这组输入求得一个插值函数

$$y = f(\mathbf{x}) = \sum_{i=1}^n w_i \phi(\|\mathbf{x} - \mathbf{x}_i\|),$$

其中 $\phi(x)$ 是手动指定的 RBF kernel, 我们选用的是

$$thin_plate : \phi(x) = x^2 \log(x).$$

基于这个插值得到的函数,

对每一个新的 m 维变量 \mathbf{x} 都能求得一个对应的 $y = f(\mathbf{x})$

可以看到插值函数 $y = f(\mathbf{x})$ 由已知输入 $\{\mathbf{x}_i\}_{i=1, \dots, n}$ 和

一组权重 $\mathbf{w} = [w_1 \quad \dots \quad w_n]^T \in \mathbb{R}^n$ 构成

求得了这组权重我们就求得了这个插值函数.

RBF 的构造过程就是通过给定输入 $\mathbf{x} \in \mathbb{R}^m$, $\{(\mathbf{x}_i, y_i)\}_{i=1, \dots, n}$ 求权重的过程

$$\text{记 } \Phi = \begin{bmatrix} \phi_{1,1} & \dots & \phi_{1,n} \\ \vdots & \ddots & \vdots \\ \phi_{n,1} & \dots & \phi_{n,n} \end{bmatrix} \in \mathbb{R}^{n \times n}, \quad \phi_{i,j} = \phi(\|\mathbf{x}_i - \mathbf{x}_j\|)$$

通过解 $\Phi \mathbf{w} = \mathbf{y}$ 这样一个方程我们就可以求得 \mathbf{w}

考虑 Smoothing/Regularization, 这个方程被写为 $(\Phi + \lambda \mathbf{I}) \mathbf{w} = \mathbf{y}$

基于 LU 分解, 我们可以解这个方程求得权重 \mathbf{w}

RBF 算法伪代码

给定一组输入 $X = (x_{ij}) \in \mathbb{R}^{m \times n}$, $\mathbf{y} = (y_j) \in \mathbb{R}^{n \times 1}$

其中 X 表示由 n 组 m 维向量组成的 $m \times n$ 矩阵, \mathbf{y} 表示 n 组向量对应的 n 个 y

即我们有 n 组输入 (\mathbf{x}, y) , 每个 \mathbf{x} 由 m 个分量 x_i 构成

RBF (插值) 算法伪代码如下:

1. 对输入矩阵 X 的 n 组向量两两之间求 $L2$ 范数欧式距离

得到一个矩阵 $D = (d_{ij}) \in \mathbb{R}^{n \times n}$ where $d_{ij} = \|X_{*,i}, X_{*,j}\|$

2. 对矩阵 D 的每个元素 $r = d_{ij}$ 求其对应的 $\phi(d_{ij}) = r^2 \log(r)$ 得到矩阵

$A = (\phi(d_{ij})) \in \mathbb{R}^{n \times n}$

3. 考虑正则化, 矩阵 $A = A + \lambda I$

4. 解线性方程组 $A\mathbf{w} = \mathbf{y}$ 求得权重向量 $\mathbf{w} = [w_1 \cdots w_n]^T \in \mathbb{R}^n$

在python中

步骤1中求矩阵列向量两两之间欧式距离的操作 可以通过 `scipy.cdist(X,X)` 来完成

步骤2中 $r^2 \log(r)$ 操作可以通过 `scipy.xlogy` 来完成

在 `eigen/xtensor` 中, 我们都没有现成的函数可以使用, 必须手动实现这两个功能

手动实现 `scipy.cdist` 时思路如下:

$$\begin{aligned} d_{ij} &= \|A_{*,i}, B_{*,j}\| \\ &= \sqrt{(A_{0i} - B_{0j})^2 + (A_{1i} - B_{1j})^2 + \cdots + (A_{(m-1)i} - B_{(m-1)j})^2} \\ &= \sqrt{(A_{0i}^2 + \cdots + A_{(m-1)i}^2) + (B_{0j}^2 + \cdots + B_{(m-1)j}^2) - 2(A_{0i}B_{0j} + \cdots + A_{(m-1)i}B_{(m-1)j})} \end{aligned}$$

所以

$$\text{cdist}(A, B) = \sqrt{\text{norm}(A, \text{by_col}) + \text{norm}(B, \text{by_col}) - 2\text{dot}(A^T, B)}$$

即我们可以通过对矩阵 A, B 按列求 norm 然后求矩阵乘

就可以批量的求得矩阵 A, B 列向量两两之间的欧式距离

手动实现 `scipy.xlogy` 时思路如下: $xlogy = x \times \log(y)$

公式本身并不复杂, 但需要注意的是, $xlogy$ 的参数是两个向量之间的欧式距离, 值可能为0

而 $\log(0) = \text{inf}$, `scipy`中的`xlogy`能自动处理这种情况

手动实现时需要我们自己手动处理输入为0的情况

在`eigen`中, 我们这样处理

```
logr = (logr.array().isInf() == 1).select(0, logr);
```

在`xtensor`中, 我们这样处理

```
xt::log(xt::where(input > 0, input, 1));
```

相应地, 求得BBF插值函数之后, 给定新输入求值过程如下:

插值函数由已知的 X 和 \mathbf{w} 构成: $y = \sum_{i=1}^n w_i \phi(\|\mathbf{x} - \mathbf{x}_i\|)$

给定一个新的输入 $\mathbf{x} = (x_i) \in \mathbb{R}^{m \times 1}$, 求值过程如下:

1. 求新输入向量和已知 X 的 n 个列向量之间的欧式距离得到一个向量

$$D = (d_{ij}) \in \mathbb{R}^{1 \times n}$$

2. 对矩阵 D 的每个元素 $r = d_{ij}$ 求其对应的 $\phi(d_{ij}) = r^2 \log(r)$ 得到矩阵 $A = (\phi(d_{ij})) \in \mathbb{R}^{1 \times n}$
3. $\text{dot}(D, \mathbf{w})$ 求得 y

关于怎样解线性方程组 $Aw = y$ 的讨论

参考链接

一份对RBF求解过程清晰直观的介绍

<https://yuki-koyama.github.io/mathtoolbox/rbf-interpolation/>

lu 分解的问题

<https://answer-id.com/47315471>

https://scc.ustc.edu.cn/zlsc/sugon/intel/mkl/mkl_manual/index.htm

<https://software.intel.com/content/www/us/en/develop/documentation/mkl-developer-reference-c/top/overview.html> <http://geowu.blogspot.com/2013/05/lapack-dgesv.html>

批量求 矩阵之间的 欧氏距离的原理是什么

<https://stackoverflow.com/questions/1401712/how-can-the-euclidean-distance-be-calculated-with-numpy>

<https://blog.csdn.net/frankzd/article/details/80251042>

<https://stackoverflow.com/questions/27948363/numpy-broadcast-to-perform-euclidean-distance-vectorized>

RBF 理论详解

Ken Anjyo, J. P. Lewis, and Frédéric Pighin. 2014. Scattered data interpolation for computer graphics. In ACM SIGGRAPH 2014 Courses (SIGGRAPH '14). ACM, New York, NY, USA, Article 27, 69 pages.

DOI: <https://doi.org/10.1145/2614028.2615425>

<http://scribblethink.org/Courses/ScatteredInterpolation/scatteredinterpcoursenotes.pdf>

<https://diginole.lib.fsu.edu/islandora/object/fsu:405601/datastream/PDF/view>