次梯度算法

芦星宇

April 2021

目录

1 次梯度方法

对于可导的凸函数，我们通常采用常规的梯度下降法处理，但当目标函数不可导（在某些点上导数不存在）时，我们没法采用常规的梯度下降法处理。于是引入次梯度 (Subgradient) 用于解决此类目标函数并不总是处处可导的问题。

1.1 次梯度定义

对于凸函数 f, 如果它可导，那么对 $\forall x, y \in dom f$，都有

$$f(y) \geq f(x) + \bigtriangledown f(x)^T (y - x)$$

此即凸函数的一阶条件，就是对于凸函数，其切线总是在函数的下方。

类比上式，给定函数 f, 对 $\forall y$，如果满足

$$f(y) \geq f(x) + g^T (y - x)$$

则称 g 是函数 f 在点 x 处的 ** 次梯度 (Subgradient)**。

将 f 在 x 处所有次梯度构成的集合称为 f 在 x 处的 ** 次微分 (Subdifferential)**，记作 $\partial f(x)$。

设函数 f 在 x_0 处不一定可导，以一维情况为例，

f 在 x_0 处的左导数：

$$a = \lim_{x \to x_0^-} \frac{f(x) - f(x_0)}{x - x_0}$$

f 在 x_0 处的右导数：

$$b = \lim_{x \to x_0^+} \frac{f(x) - f(x_0)}{x - x_0}$$

凸函数 f 的次微分区间 $[a, b]$ 中任何一个取值都是次梯度。

如果凸函数 f 在点 x 处是可导的，即 $a = b$，次微分中只有一个元素，此时次梯度就是梯度，即 g 就等于 $\bigtriangledown f(x)$；

如果凸函数 f 在点 x 处是不可导的，即 $a \neq b$，此时次梯度是次微分中的任意一个取值，它是不唯一的。

1.2 次梯度迭代算法

类似于梯度下降算法，只是将梯度更换称了次梯度，初始化 $x^{(0)}$，重复：

$$x^{(k)} = x^{(k-1)} - t_k g^{(k-1)}, k = 1, 2, 3...$$

其中 t_k 为步长，$g^{(k-1)} \in \partial f(x^{(k-1)})$, 即从次微分中随机选择一个次梯度作为梯度。

为了使更新的参数呈递减的趋势，对第 k 次的参数更新同步使用如下策略：

$$f(x_{best}^{(k)}) = \min_{i=0,\ldots,k} f(x^{(i)})$$

次梯度算法使用如下步长选择准则:

1) 固定的步长 $t_k = t, k = 1, 2, 3, ..., k$

2) 衰减的步长 t_k 满足如下条件:

$$\sum_{k=1}^{\infty} < \infty, \sum_{k=1}^{\infty} = \infty$$

例如, 取 $t_k = \frac{1}{k}, k = 1, 2, ..., \infty$, 则 $\sum_{k=1}^{\infty} t_k^2 = \sum_{k=1}^{\infty} t_k = \frac{1}{k^2}$ 收敛且 $\sum_{k=1}^{\infty} t_k = \sum_{k=1}^{\infty} t_k = \frac{1}{k}$（为调和级数，即 p=1）发散。

衰减的步长能够保证步长在逐步减小趋近于 0 的同时变化幅度不会太大。

只要选择的步长合适，算法总会收敛，只是算法收敛速度慢。

1.3 次梯度法的一般方法

1) $t = 1$ 选择有限的正的迭代步长 $(a_t)_{t=1}^{\infty}$

2) 计算一个次梯度 $g = \partial f(x^t)$

3) 更新 $x^{t+1} = x^t - \alpha_t g^t$

4) 若是算法没有收敛，则 $t = t + 1$ 返回第二步继续计算。

2 总结

采用 Python 重构次梯度算法问题，与 Matlab 生成图片有所偏差，在 1000 轮的时候梯度已不再变化，对于消失步长的线条，在 10 轮的时候会有突变。

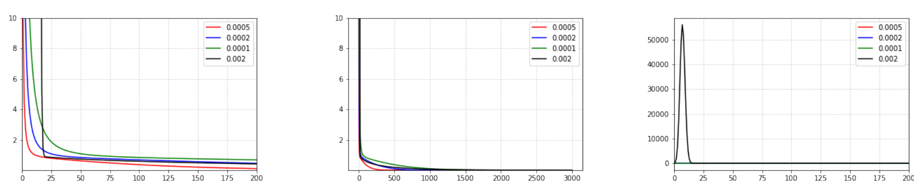

图 1: 次梯度算法解决 LASSO 问题

参考文献

[1]. [次梯度方法（Subgradient method）- 知乎