# 8.3 二阶方法
# 8.4 随机梯度下降

纪一仑

Harbin Institute of Technology,ShenZhen

Group Meeting, January 2023

# 目录

# 目录

# 8.3.1 牛顿法

## 牛顿法

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \mathbf{H}_t^{-1} \boldsymbol{g}_t$$

where

$$\mathbf{H}_t \triangleq \nabla^2 \mathcal{L}(\boldsymbol{\theta})\Big|_{\boldsymbol{\theta}_t} = \nabla^2 \mathcal{L}\left(\boldsymbol{\theta}_t\right) = \mathbf{H}\left(\boldsymbol{\theta}_t\right)$$

# 8.3.1 牛顿法

## 推导

考虑用二阶泰勒级数近似在 $\boldsymbol{\theta}_t$ 周围的 $\mathcal{L}(\boldsymbol{\theta})$

$$\mathcal{L}_{quad}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}_t) + \mathbf{g}_t^\top(\boldsymbol{\theta} - \boldsymbol{\theta}_t) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_t)^\top \mathbf{H}_t(\boldsymbol{\theta} - \boldsymbol{\theta}_t)$$

最小值点是 $\boldsymbol{\theta} = \boldsymbol{\theta}_t - \mathbf{H}_t^{-1}\mathbf{g}_t$，需要 $\mathbf{H}_t$ 正定

## 线性回归应用

$$\mathrm{RSS}(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(y_n - \boldsymbol{w}^\top \boldsymbol{x}_n\right)^2 = \frac{1}{2}\|\mathbf{X}\boldsymbol{w} - \boldsymbol{y}\|_2^2 = \frac{1}{2}(\mathbf{X}\boldsymbol{w} - \boldsymbol{y})^\top(\mathbf{X}\boldsymbol{w} - \boldsymbol{y})$$
$$= \frac{1}{2}\boldsymbol{y}^\top\boldsymbol{y} - \boldsymbol{y}^\top\mathbf{X}\boldsymbol{w} + \frac{1}{2}\boldsymbol{w}^\top\mathbf{X}^\top\mathbf{X}\boldsymbol{w}$$

损失函数就是二阶泰勒级数，使用牛顿法可以直接得到结果

计算 Hessian 矩阵 $\mathbf{H}_t$ 的代价比较大，为了减小这个代价，类牛顿法使用 Pseudo Hessian 矩阵 $\mathbf{B}_t$ 来近似 $\mathbf{H}_t$ 。

## Pseudo Hessian 的要求

- 1. $\mathbf{B}_t$ 正定对称
- 2. $\mathbf{B}_{t+1}$ 由之前迭代步骤的梯度/更新方向得到
- 3. $\mathbf{B}_{t+1}$ 需要逼近 $\mathbf{B}_t$，来保证收敛
- 4. $\mathbf{B}_{t+1}$ 的更新开销需要尽可能小
- 5. $\mathbf{B}_{t+1}(\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1}) = \boldsymbol{g}_t - \boldsymbol{g}_{t-1}$

## BFGS 公式的推导

求如下的优化问题

$$\min_{\mathbf{B}_+} \|\mathbf{B}_+ - \mathbf{B}\|_F$$

$$\text{s.t.}\ \mathbf{B}_+ \boldsymbol{\delta} = \boldsymbol{\gamma} \text{ and } \mathbf{B}_+ = \mathbf{B}_+^\top \text{ and } \mathbf{B}_+ \succ 0$$

参考链接：https://zhuanlan.zhihu.com/p/573703008

## BFGS 公式

$$\mathbf{B}_{t+1} = \mathbf{B}_t + \frac{\boldsymbol{y}_t \boldsymbol{y}_t^\top}{\boldsymbol{y}_t^\top \boldsymbol{s}_t} - \frac{(\mathbf{B}_t \boldsymbol{s}_t)(\mathbf{B}_t \boldsymbol{s}_t)^\top}{\boldsymbol{s}_t^\top \mathbf{B}_t \boldsymbol{s}_t}$$

$$\boldsymbol{s}_t = \boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1}$$

$$\boldsymbol{y}_t = \boldsymbol{g}_t - \boldsymbol{g}_{t-1}$$

$\mathbf{B}_t$ 保持正定需要满足下面的条件

### Wolfe Conditions

- $\mathcal{L}(\boldsymbol{\theta}_t + \eta \boldsymbol{d}_t) \leq \mathcal{L}(\boldsymbol{\theta}_t) + c\eta \boldsymbol{d}_t^\top \nabla \mathcal{L}(\boldsymbol{\theta}_t)$
- $-\boldsymbol{d}_t^\top \nabla \mathcal{L}(\boldsymbol{\theta}_t + \eta \boldsymbol{d}_t) \leq -c_2 \boldsymbol{d}_t^\top \nabla \mathcal{L}(\boldsymbol{\theta}_t)$

参考链接：https://www.zhihu.com/question/49600881

其中 $0 < c < c_2 < 1$
第一个条件确保函数单调下降。
第二个条件确保函数的梯度也下降。

## 其他需要注意的事情

- 可以使用 SWM 公式直接维护 $\mathbf{B}_{t+1}$ 的逆 $\mathbf{C}_{t+1}$

$$\mathbf{C}_{t+1} = \left(\mathbf{I} - \frac{\boldsymbol{s}_t \boldsymbol{y}_t^\top}{\boldsymbol{y}_t^\top \boldsymbol{s}_t}\right) \mathbf{C}_t \left(\mathbf{I} - \frac{\boldsymbol{y}_t \boldsymbol{s}_t^\top}{\boldsymbol{y}_t^\top \boldsymbol{s}_t}\right) + \frac{\boldsymbol{s}_t \boldsymbol{s}_t^\top}{\boldsymbol{y}_t^\top \boldsymbol{s}_t}$$

- 存储 $\mathbf{B}_t$ 需要 $O(D^2)$ 的空间，对于很大的问题，可以只存储最近的 $M$ 对 $(\boldsymbol{s}_t, \boldsymbol{y}_t)$，然后用这些 $(\boldsymbol{s}_t, \boldsymbol{y}_t)$ 直接计算 $\mathbf{B}_t$，这样只需要 $O(MD)$ 的空间。这就是 limited memory BFMS，即 L-BFMS。

牛顿法得到的是多元二次函数的驻点，所以当 $\mathbf{H}_t$ 不正定时，$\boldsymbol{d}_t = -\mathbf{H}_t^{-1}\boldsymbol{g}_t$ 可能不是函数下降的方向。

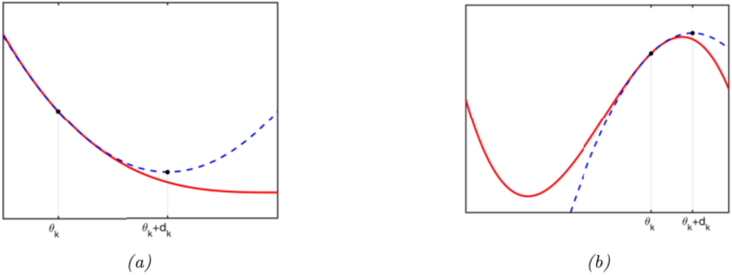



Figure: 牛顿法不一定向着函数下降的方向

# 8.3.3 Trust region method

但是，这个多元二次函数在一定区域内肯定有最小值。

**Trust-region optimization**

$$\boldsymbol{\delta}^* = \arg\min_{\boldsymbol{\delta}\in\mathcal{R}_t} M_t(\boldsymbol{\delta})$$

其中 $\boldsymbol{\delta} = \boldsymbol{\theta} - \boldsymbol{\theta}_t$，$\mathcal{R}_t$ 是所取区域，$M(\boldsymbol{\delta})$ 是目标函数或者它的估计。

## Tikhonov damping

通常情况下，假设

$$M_t(\boldsymbol{\delta}) = \mathcal{L}_{quad}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}_t) + \mathbf{g}_t^\top \boldsymbol{\delta} + \frac{1}{2}\boldsymbol{\delta}^\top \mathbf{H}_t \boldsymbol{\delta}$$

$$\mathcal{R}_t = \{\boldsymbol{\delta} : \|\boldsymbol{\delta}\|_2 \leq r\}$$

使用拉格朗日乘数法将有约束问题转化为无约束问题

$$\boldsymbol{\delta}^* = \arg\min_{\boldsymbol{\delta}} M(\boldsymbol{\delta}) + \lambda\|\boldsymbol{\delta}\|_2^2 = \arg\min_{\boldsymbol{\delta}} \mathbf{g}^\top \boldsymbol{\delta} + \frac{1}{2}\boldsymbol{\delta}^\top (\mathbf{H} + \lambda\mathbf{I})\boldsymbol{\delta}$$

当 $\lambda$ 足够大，这个函数的 Hessian，即 $\mathbf{H} + \lambda\mathbf{I}$ 肯定是正定的，于是有最小值点

$$\boldsymbol{\delta} = -(\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{g}$$

# 8.3.3 Trust region method

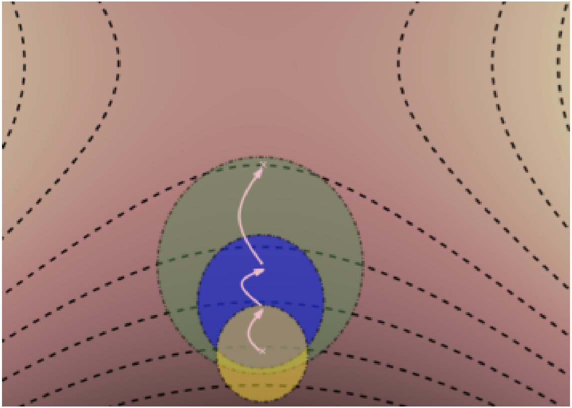


Figure: trust region method

# 目录

目标函数:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{z})}[\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{z})]$$

这里 $\boldsymbol{z}$ 是随机输入，且 $\boldsymbol{z} \sim q$。
在每次迭代中，假设我们观测到 $\mathcal{L}_t(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{z}_t)$，如果分布 $q(\boldsymbol{z})$ 独立于我们优化的参数 $\boldsymbol{\theta}$，我们可以使用 $\boldsymbol{g}_t = \nabla_{\boldsymbol{\theta}}\mathcal{L}_t(\boldsymbol{\theta}_t)$ 来近似 $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}_t)$

于是，迭代公式为:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \nabla \mathcal{L}(\boldsymbol{\theta}_t, \boldsymbol{z}_t) = \boldsymbol{\theta}_t - \eta_t \boldsymbol{g}_t$$

经验风险最小化的损失函数:

$$\mathcal{L}\left(\boldsymbol{\theta}_t\right)=\frac{1}{N}\sum_{n=1}^{N}\ell\left(\boldsymbol{y}_n, f(\boldsymbol{x}_n;\boldsymbol{\theta}_t)\right)=\frac{1}{N}\sum_{n=1}^{N}\mathcal{L}_n\left(\boldsymbol{\theta}_t\right)$$

此损失函数的梯度:

$$\boldsymbol{g}_t=\frac{1}{N}\sum_{n=1}^{N}\nabla_{\boldsymbol{\theta}}\mathcal{L}_n(\boldsymbol{\theta}_t)=\frac{1}{N}\sum_{n=1}^{N}\nabla_{\boldsymbol{\theta}}\ell\left(\boldsymbol{y}_n, f(\boldsymbol{x}_n;\boldsymbol{\theta}_t)\right)$$

可以看到，想要精确计算出梯度，需要在所有 N 个样本点上计算。

# 8.4.1 有限和问题的应用

当 $N$ 很大的时候，计算梯度会很慢。幸运的是，我们可以用少量样本上的梯度估计所有样本上的梯度。
在所有 $N$ 个样本中随机取 $B \ll N$ 个样本，作为一个 minibatch，计算 minibatch 上的梯度：

$$\boldsymbol{g}_t \approx \frac{1}{|\mathcal{B}_t|} \sum_{n \in \mathcal{B}_t} \nabla_{\boldsymbol{\theta}} \mathcal{L}_n(\boldsymbol{\theta}_t) = \frac{1}{|\mathcal{B}_t|} \sum_{n \in \mathcal{B}_t} \nabla_{\boldsymbol{\theta}} \ell\left(\boldsymbol{y}_n, f(\boldsymbol{x}_n; \boldsymbol{\theta}_t)\right)$$

因为 minibatch 是随机取的，所以这个是无偏估计。

线性回归的损失函数：

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2N}\sum_{n=1}^{N}\left(\boldsymbol{x}_n^\top\boldsymbol{\theta} - y_n\right)^2 = \frac{1}{2N}\|\mathbf{X}\boldsymbol{\theta} - \boldsymbol{y}\|_2^2$$

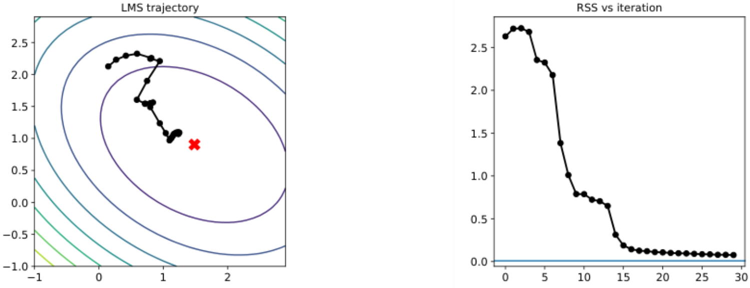
梯度：

$$\boldsymbol{g}_t = \frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{\theta}_t^\top\boldsymbol{x}_n - y_n)\boldsymbol{x}_n$$

使用 SGD，minibatch 的大小取 $B = 1$，迭代公式为：

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t(\boldsymbol{\theta}_t^\top\boldsymbol{x}_n - y_n)\boldsymbol{x}_n$$

# 8.4.2 实例：SGD 用于线性回归




(a) (b)

Figure: SGD for fitting linear regression