

统计计算

王璐

1 Gradient Descent (GD)

梯度下降法 (Gradient Descent) 也称最速下降法 (Steepest Descent), 是常用的求目标函数极小值的一阶算法。这里的“一阶”是指在优化过程中只用到目标函数和其一阶导数 (梯度) 的信息, 没有用到更高阶导数的信息。当目标函数 $f(\mathbf{x})$ 一阶可导时, 由于函数在任意一点的负梯度方向是函数下降最快的方向, 我们总可以沿函数负梯度 $-\nabla f(\mathbf{x})$ 的方向搜索, 使得每一步的目标函数值都减小。

与二阶优化算法 (e.g. Newton-Raphson) 相比, 一阶算法通常收敛得很慢。但对于变量个数很多的高维优化问题, 一阶算法可能更有优势, 因为每步迭代计算简单, 而求二阶导数 (Hessian matrix) 的成本可能太高或不可行。特别当目标函数不是凸函数 (convex) 时, 有时找到一个局部极小值点就足够了, 而 Newton-Raphson 找到的可能是“鞍点” (saddle point)。

考虑下面的优化问题

$$\min_{\mathbf{x}} f(\mathbf{x})$$

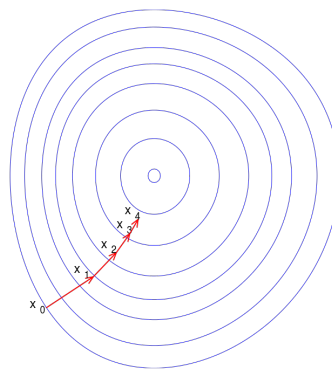
其中 $\mathbf{x} \in \mathbb{R}^d$, f 一阶可导。对于 \mathbb{R}^d 上的任意单位向量 \mathbf{v} , f 沿 \mathbf{v} 方向的改变率 (方向导数)

$$\nabla_{\mathbf{v}} f(\mathbf{x}) = \lim_{h \rightarrow 0} \frac{f(\mathbf{x} + h\mathbf{v}) - f(\mathbf{x})}{h} = \nabla f(\mathbf{x})^\top \cdot \mathbf{v} = \|\nabla f(\mathbf{x})\| \cos(\theta)$$

其中 θ 是 \mathbf{v} 与梯度 $\nabla f(\mathbf{x})$ 的夹角, 当 \mathbf{v} 与负梯度 $-\nabla f(\mathbf{x})$ 方向相同时, f 下降得最快。

• Gradient Descent

1. 选取初始值 $\mathbf{x}^{(0)} \in \mathbb{R}^d$
2. 重复以下迭代直到收敛:
 - 计算 $\mathbf{g}_{t-1} = \nabla f(\mathbf{x}^{(t-1)})$
 - 选取步长 λ_t
 - 令 $\mathbf{x}^{(t)} = \mathbf{x}^{(t-1)} - \lambda_t \mathbf{g}_{t-1}$



在上述算法中，“迭代收敛”可以是梯度模长小于某个临界值 $\|\nabla f(\mathbf{x}^{(t)})\| < \varepsilon$ ，或两次迭代间函数变化小于某个临界值 $\|f(\mathbf{x}^{(t+1)}) - f(\mathbf{x}^{(t)})\| < \varepsilon$ 。

如果步长 λ_t 足够小，目标函数值会随着迭代进行不断减小， $f(\mathbf{x}^{(0)}) \geq f(\mathbf{x}^{(1)}) \geq f(\mathbf{x}^{(2)}) \geq \dots$ 。如果函数 f 存在有界的下限， $f(\mathbf{x}) \geq f^*, \forall \mathbf{x}$ ，那么序列 $\{\mathbf{x}^{(t)}\}$ 会收敛到 f 的一个局部极小值点。

1.1 步长 λ_t 的选取

如果选取固定步长，比如令 $\lambda_t = \lambda, t = 1, 2, \dots$ ，当 λ 很大时，GD 算法可能会发散，如图1的左图所示；如果 λ 很小，GD 会收敛得很慢，如图1中间的图所示。通过后面的收敛性分析，我们可能会找到一个较合适的步长，如图1的右图所示。

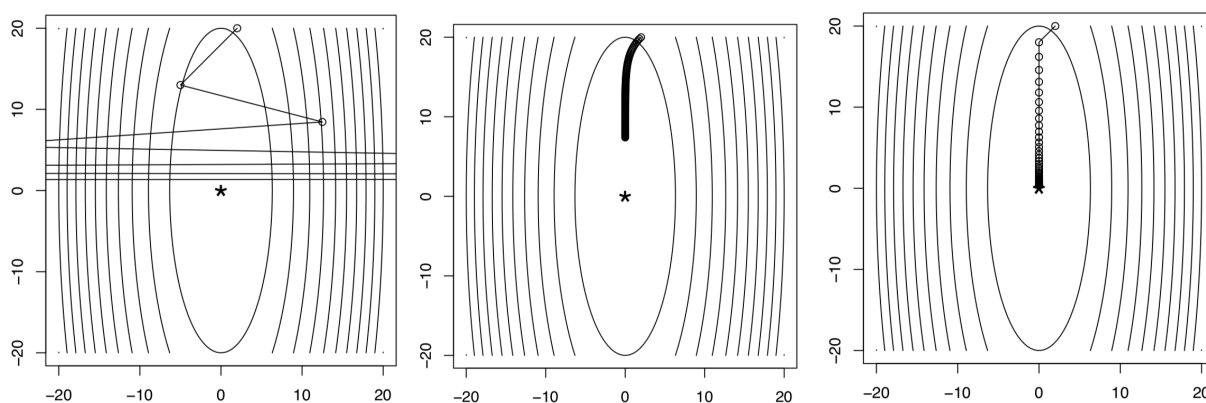


Figure 1: Gradient descent 中步长的选取对算法收敛的影响。Picture source: Ryan Tibshirani

更好的做法是让 λ_t 随迭代变化，每一步根据当前函数值和梯度选取较优的步长。Backtracking line search（一维搜索）是一种常用的自适应 (adaptively) 选取步长的方法。

• Backtracking line search

1. 选取两个固定参数 $0 < \beta < 1$ 和 $0 < \alpha \leq 1/2$ 。
2. 在第 t 步迭代开始时，选取一个较大的初始步长 γ_0 ，然后不断缩小步长，令 $\gamma_j = \beta\gamma_{j-1}, j = 1, 2, \dots$ ，直到

$$f(\mathbf{x}^{(t-1)} - \gamma_j \mathbf{g}_{t-1}) \leq f(\mathbf{x}^{(t-1)}) - \alpha \gamma_j \|\mathbf{g}_{t-1}\|_2^2. \quad (1)$$

(1)被称为 Armijo 条件。

3. 将满足 Armijo 条件(1)的步长 γ_j 选为第 t 步步长 λ_t 。

在图1的例子中使用 backtracking line search 选取步长， $\alpha = \beta = 0.5$ ，迭代的轨迹如图2所示。

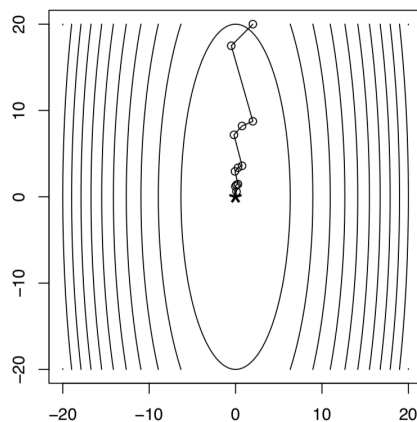


Figure 2: 12 outer steps, 40 steps total. Picture source: Ryan Tibshirani

Remarks

1. 为了简化 backtracking line search 的调参，一般令 $\beta = 1/2$.
2. 有时人们会选取非常小的 α , e.g. $\alpha = 10^{-4}$, 原因是宁可让目标函数下降得少一点，也不想尝试太多 γ_j .
3. 更加复杂的一维搜索方法还有 Wolfe condition (Nocedal and Wright, 2006), 可以确保 α 不会太小。
4. 寻找最优步长的 exact line search:

$$\lambda_t = \underset{\lambda > 0}{\operatorname{argmin}} f(\mathbf{x}^{(t-1)} - \lambda \mathbf{g}_{t-1})$$

在实践中一般不可行，因为计算成本太高。

1.2 收敛性分析

本节我们想回答下面两个问题：

- 步长 λ_t 对 GD 收敛的影响？
- 当 GD 收敛时， $\nabla f(\mathbf{x}^{(t)}) \rightarrow 0, t \rightarrow \infty$. 但可能对任何有限的 t , $\nabla f(\mathbf{x}^{(t)})$ 不会严格为 $\mathbf{0}$. 那么对任意一个给定的 $\varepsilon > 0$, 需要迭代多少步才能达到 $\|\nabla f(\mathbf{x}^{(t)})\| < \varepsilon$?

为了回答以上问题，我们需要假设

1. f 存在有界的下限 f^* .

2. f 的梯度是 Lipschitz 连续。

Definition 1.1. 如果存在一个常数 $L > 0$, 使得对任意的 $\mathbf{x}_1, \mathbf{x}_2$ 都有

$$\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\|_2 \leq L \|\mathbf{x}_1 - \mathbf{x}_2\|_2$$

称 f 的梯度 $\nabla f(\mathbf{x})$ 是 **Lipschitz 连续**。

Remarks

- 梯度 Lipschitz 连续要求函数 f 的梯度不能“变化得太快”。
- 梯度 Lipschitz 连续是一个很弱的假设，很多统计模型的 log-likelihood 都满足该条件，比如线性回归，logistic regression.
- 对于 C^2 函数 (二阶连续可导) f , 梯度的 Lipschitz 连续意味着 $\nabla^2 f(\mathbf{x}) \preceq LI, \forall \mathbf{x}$. 即 $\nabla^2 f(\mathbf{x}) - LI$ 总是一个半负定 (negative semi-definite, n.s.d) 矩阵: $\forall \mathbf{x}, \mathbf{h}$,

$$\mathbf{h}^\top \nabla^2 f(\mathbf{x}) \mathbf{h} \leq L \|\mathbf{h}\|_2^2.$$

- 对于 C^2 函数 f , 可将 L 选为 f 的 Hessian matrix 特征值的一个上界。比如在线性回归中, log-likelihood 的 Hessian 是 $X^\top X$, 因此 L 最小可取为 $X^\top X$ 的最大特征根。

对于 C^2 函数 f , 将 $f(\mathbf{x})$ 在 $\mathbf{x}^{(t)}$ 处 Taylor 展开:

$$f(\mathbf{x}) = f(\mathbf{x}^{(t)}) + \nabla f(\mathbf{x}^{(t)})^\top (\mathbf{x} - \mathbf{x}^{(t)}) + \frac{1}{2} (\mathbf{x} - \mathbf{x}^{(t)})^\top \nabla^2 f(\tilde{\mathbf{x}}) (\mathbf{x} - \mathbf{x}^{(t)}). \quad (2)$$

由 ∇f 的 Lipschitz 连续得

$$f(\mathbf{x}) \leq f(\mathbf{x}^{(t)}) + \nabla f(\mathbf{x}^{(t)})^\top (\mathbf{x} - \mathbf{x}^{(t)}) + \frac{L}{2} \|\mathbf{x} - \mathbf{x}^{(t)}\|_2^2. \quad (3)$$

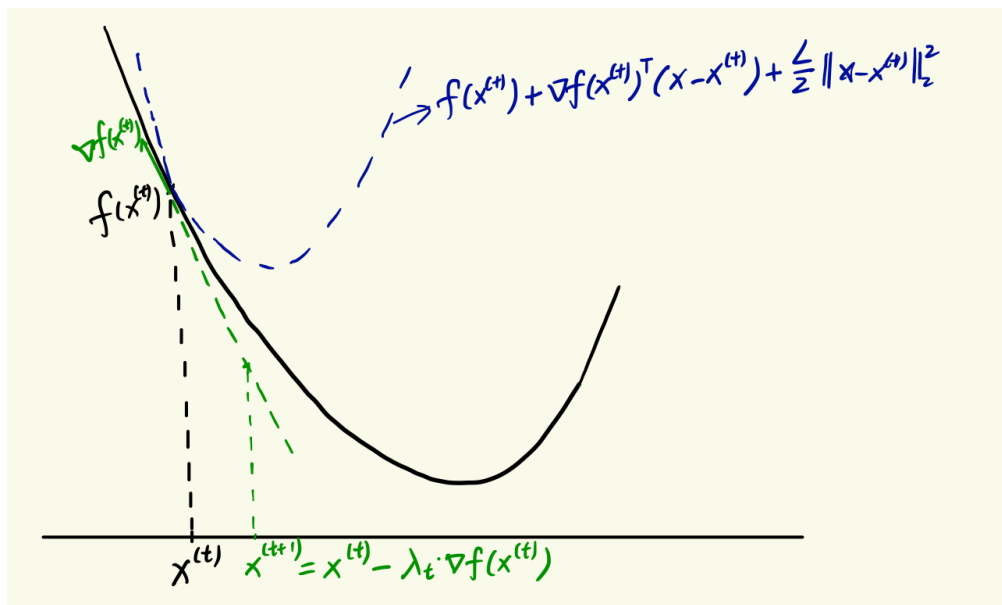
(3)的右端是过点 $(\mathbf{x}^{(t)}, f(\mathbf{x}^{(t)}))$ 的一个 f 的二次上界函数，如图3所示。

将 GD 迭代公式 $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \lambda_t \mathbf{g}_t$ 代入(3)，其中 $\mathbf{g}_t = \nabla f(\mathbf{x}^{(t)})$ ，得到

$$f(\mathbf{x}^{(t+1)}) \leq f(\mathbf{x}^{(t)}) - \|\mathbf{g}_t\|_2^2 \lambda_t + \frac{L}{2} \|\mathbf{g}_t\|_2^2 \lambda_t^2. \quad (4)$$

因此选取步长

$$\lambda_t = 1/L$$



下面计算将误差降到 ε 以下所需 GD 迭代的步数。

为了简化问题, 我们将步长固定在 $\lambda_t = 1/L$. (5)表明, 从某个初始值 $f(\mathbf{x}^{(0)})$ 开始, GD 在每一步迭代中都会将 f 减小 $\frac{1}{2L} \|\mathbf{g}_t\|_2^2$. 对 (5)重新整理可得

$$\|\mathbf{g}_t\|_2^2 \leq 2L \left(f(\mathbf{x}^{(t)}) - f(\mathbf{x}^{(t+1)}) \right)$$

由此得到:

$$\begin{aligned} \|\mathbf{g}_{t-1}\|_2^2 &\leq 2L \left(f(\mathbf{x}^{(t-1)}) - f(\mathbf{x}^{(t)}) \right) \\ &\vdots \\ \|\mathbf{g}_0\|_2^2 &\leq 2L \left(f(\mathbf{x}^{(0)}) - f(\mathbf{x}^{(1)}) \right) \end{aligned}$$

将以上不等式相加得到

$$\sum_{k=0}^{t-1} \|\mathbf{g}_k\|_2^2 \leq 2L \left(f(\mathbf{x}^{(0)}) - f(\mathbf{x}^{(t)}) \right)$$

由于 $f(\mathbf{x}^{(t)}) \geq f^*, \forall t$,

$$t \cdot \min_k \|\mathbf{g}_k\|_2^2 \leq \sum_{k=0}^{t-1} \|\mathbf{g}_k\|_2^2 \leq 2L \left(f(\mathbf{x}^{(0)}) - f(\mathbf{x}^{(t)}) \right) \leq 2L \left(f(\mathbf{x}^{(0)}) - f^* \right)$$

最终得到

$$\min_k \|\mathbf{g}_k\|_2^2 \leq \frac{2L \left(f(\mathbf{x}^{(0)}) - f^* \right)}{t} = O\left(\frac{1}{t}\right). \quad (8)$$

(8)表明, GD 迭代 t 步后, 至少在某一步 $k \in \{1, \dots, t\}$ 有 $\|\mathbf{g}_k\|_2^2 = \|\nabla f(\mathbf{x}^{(k)})\|_2^2 = O(1/t)$. 即第 t 步迭代的误差是 $O(1/t)$, 也称 GD 的收敛速率 (convergence rate) 是 $O(1/t)$.

为了将梯度模长的平方降到 ε 以下, 令

$$\frac{2L \left(f(\mathbf{x}^{(0)}) - f^* \right)}{t} < \varepsilon$$

得到

$$t > \frac{2L \left(f(\mathbf{x}^{(0)}) - f^* \right)}{\varepsilon}$$

所以 GD (最多) 需要 $t = O(1/\varepsilon)$ 步迭代使得 $\|\nabla f(\mathbf{x}^{(k)})\|_2^2 < \varepsilon$.

像(8)这样, 误差 ε 以 $O(1/t)$ 或 $O(1/t^2)$ 的速率减小的情况, 被称为 **sublinear convergence**. 如果误差以 $O((1-\delta)^t)$ ($0 < \delta < 1$) 的速率减小, 称为 **linear convergence**. 这样命名的原因是后者在 $\log(\varepsilon)$ vs t 的图中看起来是线性下降的, 如图4的右图所示. 如果实践中只需要一个低精度的解, sublinear rate 的算法可能就足够了。

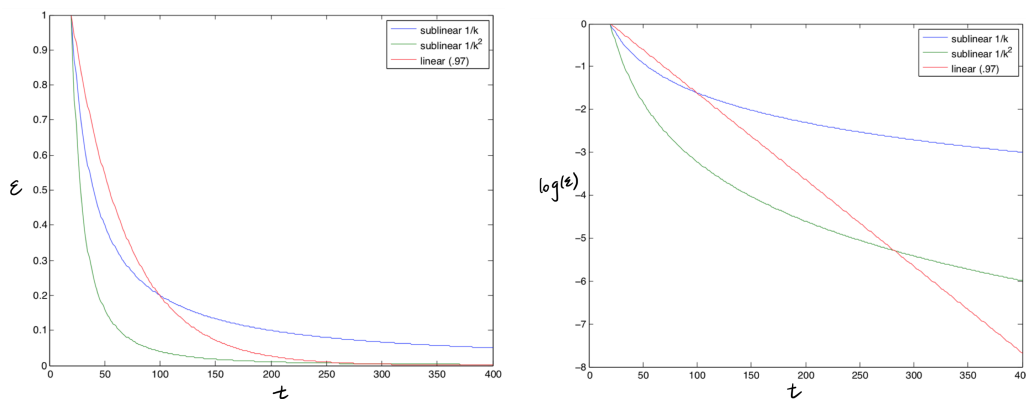


Figure 4: 不同收敛速度的算法迭代误差 ϵ vs 迭代步数 t (左), $\log(\epsilon)$ vs t (右)。Picture source: Stephen Wright

总结

1. 以上我们证明了当目标函数的梯度是 Lipschitz 连续的, 选取足够小的步长可以保证 GD 收敛。
2. t 步迭代的误差是 $O(1/t)$. 这意味着要达到 $\|\nabla f(\mathbf{x}^{(k)})\|_2^2 < \epsilon$, 需要 $t = O(1/\epsilon)$ 步迭代。

Remarks

1. 虽然在以上理论分析中, 我们选取的步长是 $\lambda_t \equiv 1/L$, 但实践中一般不使用该步长, 因为
 - L 通常很难计算
 - 即使可以找到满足条件的 L , $1/L$ 一般特别小, 会导致 GD 收敛得很慢, 只有在最坏的情况下、为保证收敛才使用。
2. Backtracking line search 是更实用的选取步长的方法, 与(5)相比, 很多情况下(1)中的 $\alpha\gamma_j > 1/(2L)$, 这样每一次迭代取得的进步比选取步长 $1/L$ 大。
3. GD 不适用于目标函数不可导的优化问题。

1.3 随机梯度下降 (Stochastic Gradient Descent)

统计模型中的目标函数通常是 joint log-likelihood, 且可以写成单个 log-likelihood 和的形式, 即 $l(\theta | y_1, \dots, y_n) = \sum_{i=1}^n l(\theta | y_i)$. 考虑如下优化问题

$$\min_{\theta} \sum_{i=1}^n f_i(\theta)$$

由于 $\nabla \sum_{i=1}^n f_i(\boldsymbol{\theta}) = \sum_{i=1}^n \nabla f_i(\boldsymbol{\theta})$, GD 迭代格式如下:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \lambda_t \sum_{i=1}^n \nabla f_i(\boldsymbol{\theta}^{(t)}), \quad t = 0, 1, \dots$$

随机梯度下降 (SGD) 的迭代格式为:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \lambda_t \nabla f_{i_t}(\boldsymbol{\theta}^{(t)}), \quad t = 0, 1, \dots$$

其中 $i_t \in \{1, \dots, n\}$.

有两种选择 i_t 的方式:

1. 循环式: 令 $i_t = 1, 2, \dots, n, 1, 2, \dots, n, \dots$
2. 随机式: 在每步迭代 t 随机选取 $i_t \in \{1, \dots, n\}$.

实践中常用的是随机式。从计算量角度, n 步 SGD 迭代 \approx 1 步 GD 迭代。它们在迭代精度上有什么区别? 为了简化分析, 我们采用固定步长和循环式 SGD.

- n 步 SGD 迭代: $\boldsymbol{\theta}^{(t+n)} = \boldsymbol{\theta}^{(t)} - \lambda \sum_{i=1}^n \nabla f_i(\boldsymbol{\theta}^{(t+i-1)})$
- 1 步 GD 迭代: $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \lambda \sum_{i=1}^n \nabla f_i(\boldsymbol{\theta}^{(t)})$

二者在方向上相差

$$\sum_{i=1}^n \left[\nabla f_i(\boldsymbol{\theta}^{(t+i-1)}) - \nabla f_i(\boldsymbol{\theta}^{(t)}) \right].$$

如果每个 f_i 的梯度 $\nabla f_i(\boldsymbol{\theta})$ 不会随着 $\boldsymbol{\theta}$ 发生剧烈改变, 则 SGD 与 GD 在更新方向上应该相差不大, 也会收敛。

References

Nocedal, J. and Wright, S. (2006). *Numerical optimization*. Springer Science & Business Media.