梯度下降方法

Yimeng Ren

1 次梯度与次微分

对于一个函数如果不可导，那么如何使用梯度方法?

1.1 次梯度（subgradient）

对于凸实值函数 $f:\mathbb{R}^n \to \mathbb{R}$ 和向量 $g \in \mathbb{R}^n$，如果对于任意的 y，都有 $f(y) \geq f(x) + g^T(y-x)$，那么称 g 是一个 f 在 x 点处的次梯度。

Note:

- **函数必须是凸函数**，否则是不能定义次梯度的，因为如果不是凸函数，线性函数就不会是函数的一个低估，因此次梯度就无法保证存在;
- **次梯度并不唯一**，它可以是一个集合。对于复杂的问题，很多时候找到所有的次梯度并不现实，所以会退而求其次，找到某一个次梯度，但这就足够用上次梯度方法了;
- 如果 x 处函数可导，那么次梯度唯一，就是 $\nabla f(x)$。

因为次梯度不唯一，所以还有一个对应的次微分的定义。

1.2 次微分（subdifferential）

定义 $\partial f(x)$ 为所有 x 点处 f 的次梯度的集合，称其为次微分。

若有 $f(\beta) = |\beta|$，则:

$$\partial f(\beta) = \begin{cases} \{1\}, \beta > 0 \\ \{-1\}, \beta < 0 \\ [-1, 1], \beta = 0 \end{cases}$$

极值点一阶条件的推广: 判断 $\beta^\star$ 是否为 $f(\beta)$ 的极值点（local optimal），可以通过 $0 \in \partial f(\beta)$ 来判断。

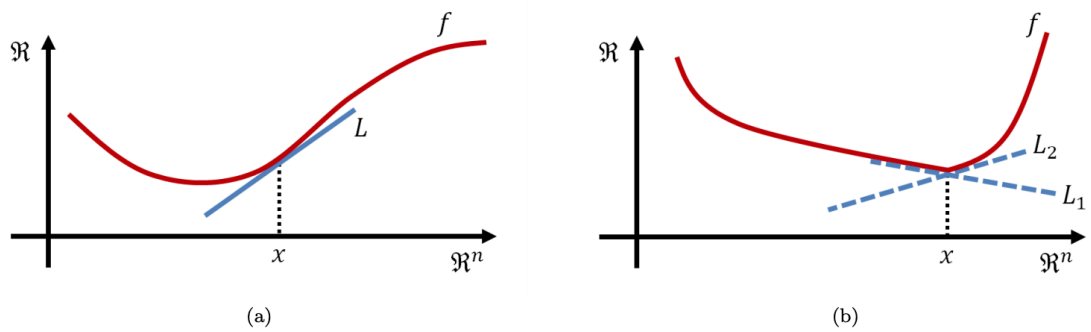


Figure 6.1

To say that a function$f : \Re^n \mapsto \Re$is differentiable at x is to say that there is a single unique linear tangent such as shown in Fig 6.1a that under estimates the function:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \forall x, y$$

While in Fig 6.1b we see the function f at x has many possible linear tangents that may fit appropriately. A **subgradient** is any $g \in \Re^n$ such that:

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

Thus, if a function is differentiable at a point x then it has a unique subgradient at that point $(\nabla f(x))$.

2 无约束优化问题

无约束优化问题：minimize $f(x)$，其中 $f : \mathbf{R}^n \to \mathbf{R}$ 是二次可微凸函数（dom f 是开集）。假定该问题存在全局最优解 $x^\star$，用 $p^\star$ 表示最优值 $\inf_x f(x) = f(x^\star)$。由于 f 是可微凸函数，则最优点 $x^\star$ 应当满足：

$$\nabla f(x^\star) = 0$$

3 下降方法

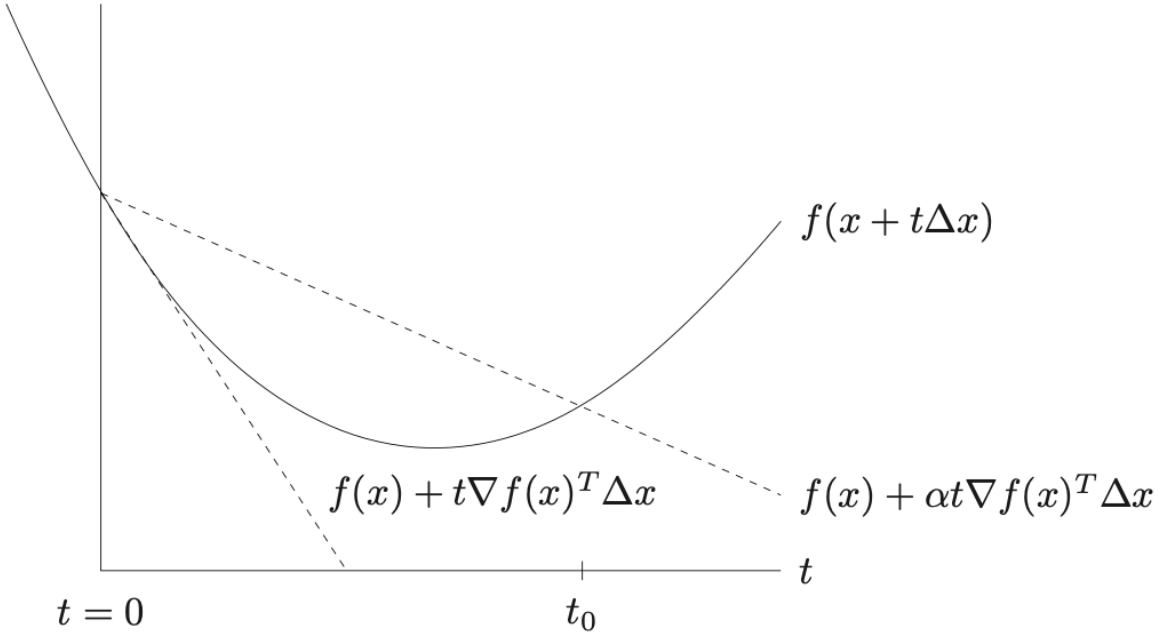
这一系列算法将产生一个优化点列 $x^{(k)}, k = 1, 2, \ldots$，其中，

$$x^{(k+1)} = x^{(k)} - t^{(k)}\Delta x^{(k)}$$

$t^{(k)} \geq 0$ 表示第 k 次迭代的步长，Δx 表示 $\mathbf{R}^n$ 的一个向量，$-\Delta x$ 为搜索方向。

3.1 回溯直线搜索（backtracking rule）

是一种找 $t^{(k)}$ 的方法，将 t 初始化为一个较大的数字，如 1。参数 $\alpha \in (0, 0, 5), \beta \in (0, 1)$，当满足 $f(x + t\Delta x) > f(x) + \alpha t \nabla f(x)^T \Delta x$ 时，不断缩小 $t := \beta t$。



$f(x + t\Delta x)$ 是一个 t 的一元函数，在 $t = 0$ 处的切线可以由 $y = f(x) + t\nabla f(x)^T \Delta x$ 表示。在切线的斜率上乘一个 $\alpha < 1$，则得到图中的割线。

算法启动时，我们尝试性沿着 Δx 方向走 t 这么长的距离。我们想要控制 t，防止走得太远反而令函数值增大。以**割线和函数的交点**为分界 $(f(x + t\Delta x) > f(x) + \alpha t \nabla f(x)^T \Delta x)$，要是 t 过大，就缩小 t，控制 $t < t_0$，保证算法收敛。

回溯搜索从单位步长开始，按照比例逐渐减小，直到满足停止条件 $f(x + t\Delta x) \leqslant f(x) + \alpha t \nabla f(x)^T \Delta x$。由于 $-\Delta x$ 是下降方向，$-\nabla f(x)^T \Delta x < 0$，所以只要 t 足够小，就有：

$$f(x + t\Delta x) \approx f(x) + t\nabla f(x)^T \Delta x < f(x) + \alpha t \nabla f(x)^T \Delta x$$

3.2 投影梯度下降（Projected Gradient Method）

老师的笔记：

$$\beta^{t+1} := \underset{\beta \in C}{\operatorname{argmin}} \left\{ f(\beta^t) + \langle \nabla f(\beta^t)., \beta - \beta^t \rangle + \frac{1}{2S^t} \|\beta - \beta^t\|_2^2 \right\}$$

当 constraints 对应一个 convex set，即增加约束 $x \in C$，并且计算到该集合的投影非常容易的时候，我们可以用一种叫做 Projected Gradient Method 的迭代算法来进行优化求解。考虑问题：

$$\min_{x \in S} f(x)$$

其中 S 是非空闭凸集。因为我们并不关心其约束集的具体描述形式，只要假设有一个 blackbox 的投影算子 $P_S(x)$，能计算 $\arg\min_{y \in S} \|x - y\|$ 即可，相当于在普通的 Gradient Descent 算法中间多加了一步 projection 的步骤，保证解在 feasible region 里面。

假设我们现在有一个 blackbox 可以很容易地计算 $P_S(x)$，则算法由根据下迭代规则进行迭代直到收敛为止：

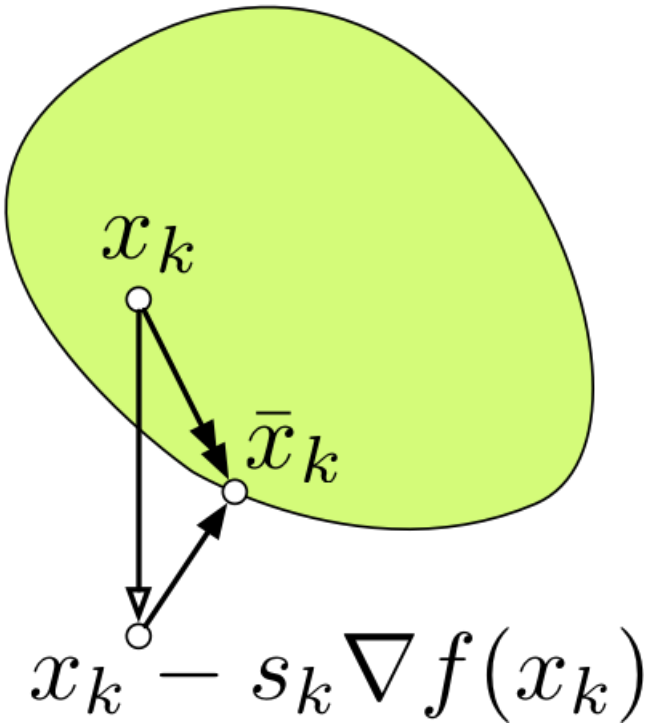
$$\begin{aligned} \bar{x}_k &= P_S\left(x_k - s_k \nabla f\left(x_k\right)\right), s_k > 0 \\ x_{k+1} &= x_k + \alpha_k\left(\bar{x}_k - x_k\right), \alpha_k \in (0, 1] \end{aligned}$$

其中 $s_k > 0$ 是一个步长参数，例如取 $\alpha_k = 1$ 的时候，上面的算法其实就变成了如下的迭代：

$$x_{k+1} = P_S\left(x_k - s_k \nabla f\left(x_k\right)\right)$$

也就是通过一次普通的 gradient descent 然后再利用 Projection Operator $P_S(\cdot)$ 投影到 feasible set 里来。加上 α_k 参数之后实际上就是再多加了一步步长选择或者 line search 的过程，我们要求 $\alpha_k \in (0, 1]$ 这样可以保证不会再移动到 feasible set 外面去。

另外有一点值得一提的是，投影之后的移动方向 $\bar{x}_k - x_k$ 一定是一个下降方向：也就是说它和 $-\nabla f\left(x_k\right)$ 成锐角。



举一个投影算子非常容易计算的例子：LASSO。LASSO 对应如下的一个优化问题：

$$\min_{w} \sum_{i=1}^{N} \left(w^T x_i - y_i\right)^2 + \lambda \sum_{j=1}^{n} \left|w^j\right|$$

注意到任意实数 w^j 都可以表示成其正部和负部之差 $w^j = w_+^j - w_-^j$，其中 $w_+^j = \max\left\{w^j, 0\right\} \geq 0, w_-^j = \max\left\{-w^j, 0\right\} \geq 0$，此时绝对值可以简单地表示成 $\left|w^j\right| = w_+^j + w_-^j$。根据这个性质，我们在 LASSO 问题中引入新的变量 $w_+^j, w_0^j \geq 0, \quad j = 1, \ldots, n$。LASSO 问题等价于：

$$\begin{aligned} \min_{w_+, w_-} & \sum_{i=1}^{N} \left(\left(w_+ - w_-\right)^T x_i - y_i\right)^2 + \lambda \sum_{j=1}^{n} \left(w_+^j + w_-^j\right) \\ \text{s.t. } & w_+ \succeq 0, w_- \succeq 0 \end{aligned}$$

现在问题变成了 constrained 优化问题，并且目标函数是可导的（二次函数），约束集也是投影算子非常容易计算的 non-negative cone。为了码代码方便我们把上面的目标函数写成矩阵形式：

$$\begin{aligned} \min_{w_+, w_-} & \left\|X w_+ - X w_- - y\right\|^2 + \lambda w_+^T \mathbf{1} + \lambda w_-^T \mathbf{1} \\ \text{s.t.} & w_+ \succeq 0, w_- \succeq 0 \end{aligned}$$

其中 $X = \left(x_1, \ldots, x_N\right)^T$ 是数据矩阵。令 $\tilde{w} = \left(w_+^T, w_-^T\right)^T, \quad \tilde{X} = (X, -X)$，则问题转化为：

$$\begin{aligned} \min_{\tilde{w}} \quad & \|\tilde{X}\tilde{w} - y\|^2 + \lambda \tilde{w}^T \mathbf{1} \\ \text{s.t.} \quad & \tilde{w} \succeq 0 \end{aligned}$$

这样一来 gradient 的计算和投影的计算都变得非常明了了。

3.3 近端梯度下降法 (proximal gradient descent)

近端梯度下降法是众多梯度下降 (gradient descent) 方法中的一种，其英文名称为 proximal gradident descent，其中，术语中的 proximal 一词比较耐人寻味，将 proximal 翻译成 “近端” 主要想表达 “（物理上的）接近”。与经典的梯度下降法和随机梯度下降法相比，近端梯度下降法的适用范围相对狭窄。对于凸优化问题，当其目标函数存在不可微部分（例如目标函数中有 [公式] -范数或迹范数）时，近端梯度下降法才会派上用场。

假设目标函数 $f(x) = g(x) + h(x)$ 是由 $g(x)$ 和 $h(x)$ 叠加而成，其中，限定 $g(x)$ 是可微的凸函数、$h(x)$ 是不可微 (或局部不可微) 的凸函数。（可以理解为**目标函数 = 损失函数 + 惩罚**）

如果 f 可微，则梯度下降更新公式为：

$$x^+ = x - t \cdot \nabla f(x)$$

在 x 附近最小化 f 的二次近似，使用 $\frac{1}{t}I$ 代替 $\nabla^2 f(x)$：

$$x^+ = \underset{z}{\operatorname{argmin}} \underbrace{f(x) + \nabla f(x)^T (z - x) + \frac{1}{2t}\|z - x\|_2^2}_{\bar{f}_t(z)}$$

如果 f 是不可微的，则将 $h(x)$ 单独写出来，即更新公式为：

$$\begin{aligned} x^+ &= \underset{z}{\operatorname{argmin}} \bar{g}_t(z) + h(z) \\ &= \underset{z}{\operatorname{argmin}} g(x) + \nabla g(x)^T (z - x) + \frac{1}{2t}\|z - x\|_2^2 + h(z) \\ &= \underset{z}{\operatorname{argmin}} \frac{1}{2t}\|z - (x - t\nabla g(x))\|_2^2 + h(z) \end{aligned}$$

定义 proximal mappping：

$$\text{prox}_{h,t}(x) = \underset{z}{\operatorname{argmin}} \frac{1}{2t}\|x - z\|_2^2 + h(z)$$

则初始化 $x^{(0)}$，近端梯度下降的更新公式可以写成：

$$x^{(k)} = \text{prox}_{h,t_k}\left(x^{(k-1)} - t_k \nabla g\left(x^{(k-1)}\right)\right), \quad k = 1, 2, 3, ...$$

以 LASSO 问题为例：

$$f(\beta) = \underbrace{\frac{1}{2}\|y - X\beta\|_2^2}_{g(\beta)} + \underbrace{\lambda\|\beta\|_1}_{h(\beta)}$$

proximal mapping 为：

$$\begin{aligned}\text{prox}_t(\beta) &= \underset{z}{\text{argmin}} \frac{1}{2t}\|\beta - z\|_2^2 + \lambda\|z\|_1 \\ &= S_{\lambda t}(\beta)\end{aligned}$$

其中 $S_\lambda(\beta)$ 为软阈值算子（soft-thresholding operator）：

$$[S_\lambda(\beta)]_i = \begin{cases} \beta_i - \lambda & \text{if } \beta_i > \lambda \\ 0 & \text{if } -\lambda \le \beta_i \le \lambda, \\ \beta_i + \lambda & \text{if } \beta_i < -\lambda \end{cases} \quad i = 1, ..., n$$

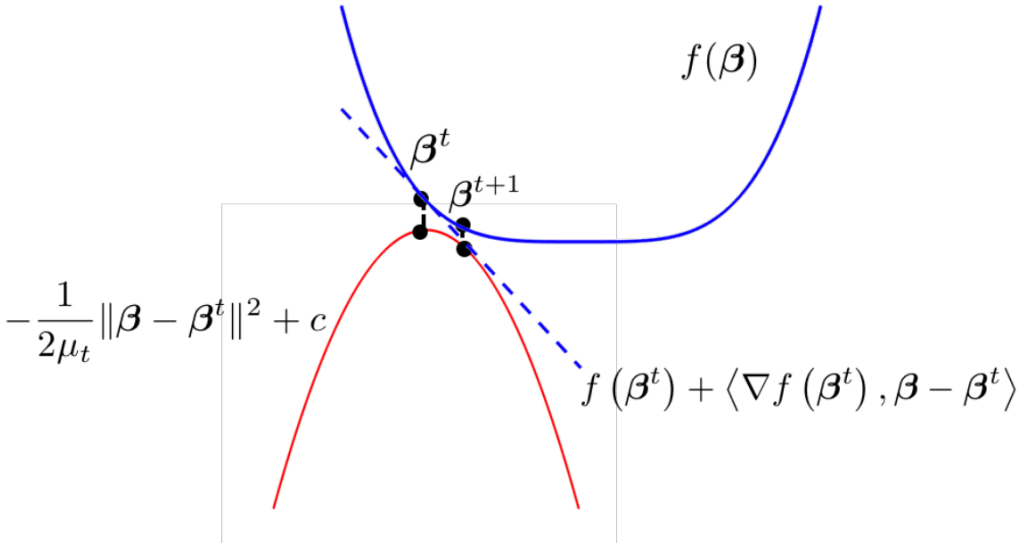
由于 $\nabla g(\beta) = -X^T(y - X\beta)$，则近端梯度更新公式为：

$$\beta^+ = S_{\lambda t}\left(\beta + tX^T(y - X\beta)\right)$$

3.3.1 examples

普通的梯度下降更新公式：$\beta^{t+1} = \beta^t - \mu_t \nabla f\left(\beta^t\right)$

从 proximal point 的角度出发看 GD：



$$\beta^{t+1}=\arg\min_{\beta}\{\underbrace{f\left(\beta^t\right)+\left\langle\nabla f\left(\beta^t\right),\beta-\beta^t\right\rangle}_{\text{linear approximation}}+\underbrace{\frac{1}{2\mu_t}\left\|\beta-\beta^t\right\|^2}_{\text{proximal term}}\}$$

当 μ_t 很小时，$\beta^{t+1}\approx\beta^t$

proximal operator 定义为：

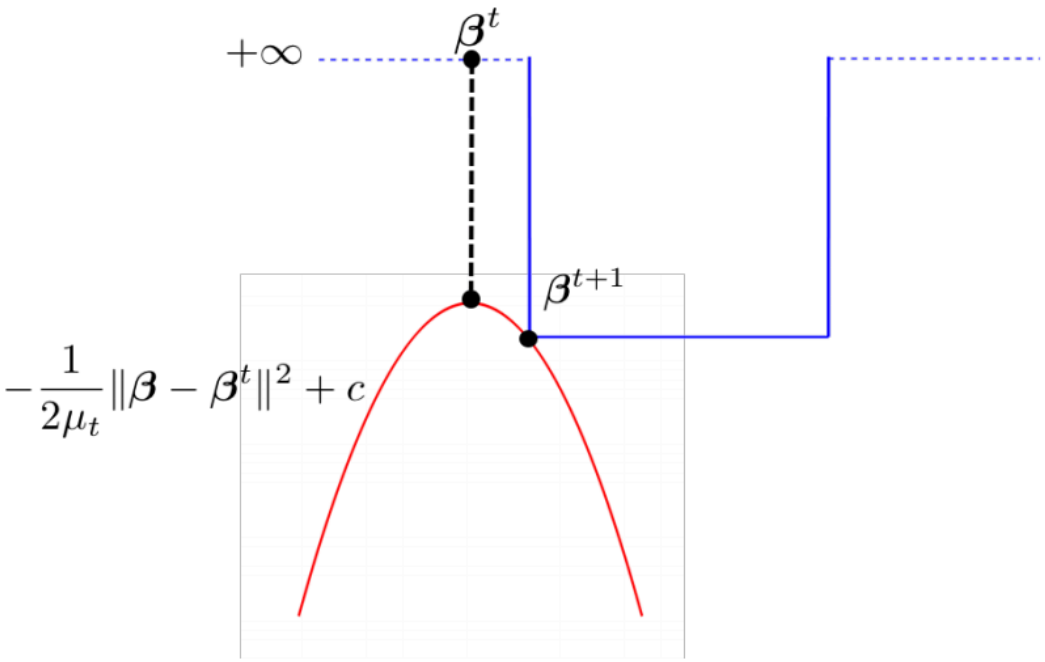
$$\text{prox}_h(b):=\arg\min_{\beta}\left\{\frac{1}{2}\|\beta-b\|^2+h(\beta)\right\}$$

则对于任意的凸函数 h，有：

$$\beta^{t+1}=\text{prox}_{\mu_t f_t}\left(\beta^t\right)$$

其中：$f_t(\beta):=f\left(\beta_t\right)+\left\langle\nabla f\left(\beta_t\right),\beta-\beta_t\right\rangle$

- 如果 h 为示性函数（characteristic function）：

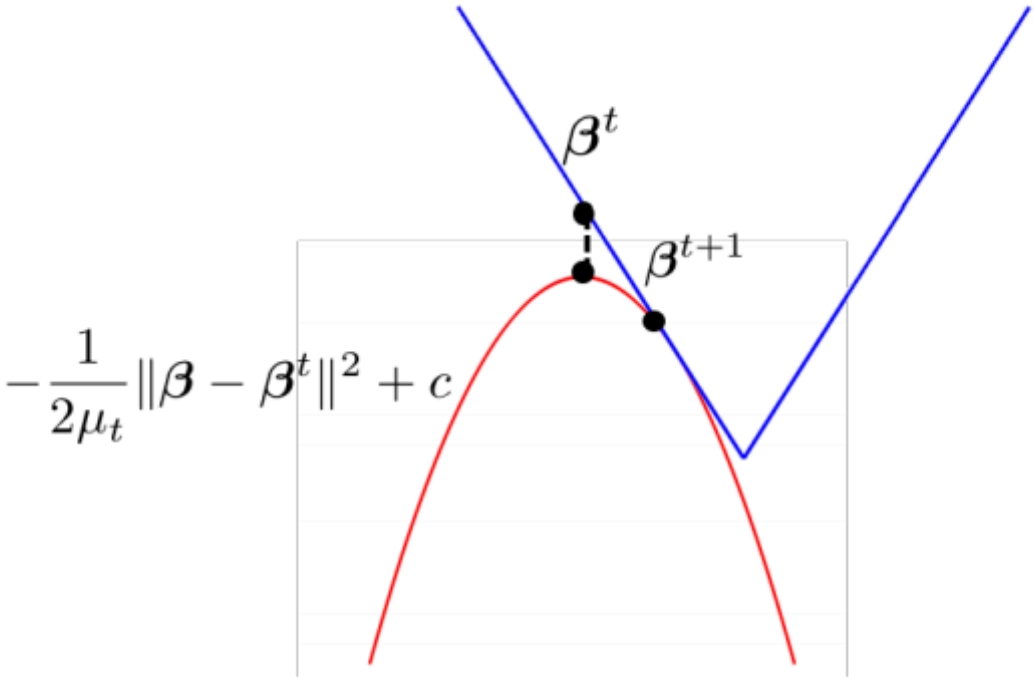


$$h(\beta)=\begin{cases}0, & \text{if } \beta\in\mathcal{C}\\ \infty, & \text{else}\end{cases}$$

那么：

$$\text{prox}_h(b)=\arg\min_{\beta\in\mathcal{C}}\|\beta-b\|_2 \quad (\text{ Euclidean projection })$$

- 如果 h 为 l_1 $norm$：



$$h(\beta)=\|\beta\|_1$$

那么：

$$\text{prox}_{\lambda h}(b)=\psi_{\text{st}}(b;\lambda)$$

其中 $\psi_{\text{st}}(b;\lambda)$ 为软阈值算子。

- 如果 h 为 l_2 $norm$：

$$h(\beta)=\|\beta\|$$

那么：

$$\text{prox}_{\lambda h}(b)=\left(1-\frac{\lambda}{\|b\|}\right)_+ b$$

其中 $a_+ := \max\{a,0\}$。（block soft thresholding）

3.4 加速梯度下降（Nesterov Accelerated Gradient）

它仅仅是在 Momentum 算法的基础上做了一点微小的工作，形式上发生了一点看似无关痛痒的改变，却能够显著地提高优化效果。Momentum 改进自 SGD 算法，让每一次的参数更新方向不仅仅取决于当前位置的梯度，还受到上一次参数更新方向的影响：

$$
\begin{aligned}
d_i &= \beta d_{i-1} + g\left(\theta_{i-1}\right) \\
\theta_i &= \theta_{i-1} - \alpha d_i
\end{aligned}
$$

其中，d_i 和 d_{i-1} 分别是这一次和上一次的更新方向，$g(\theta)$ 表示目标函数在 θ 处的梯度，超参数 β 是对上一次更新方向的衰减权重，所以一般是 0 到 1 之间，α 是学习率。总的来说，在一次迭代中总的参数更新量包含两个部分，第一个是由上次的更新量得到的 $\alpha\beta d_{i-1}$，第二个则是由本次梯度得到的 $\alpha g\left(\theta_{i-1}\right)$。

所以 Momentum 的想法很简单，就是多更新一部分上一次迭代的更新量，来平滑这一次迭代的梯度。从物理的角度上解释，就像是一个小球滚落的时候会受到自身历史动量的影响，所以才叫动量（Momentum）算法。这样做直接的效果就是使得梯度下降的的时候转弯掉头的幅度不那么大了，于是就能够更加平稳、快速地冲向局部最小点：

Image 2: SGD without momentum

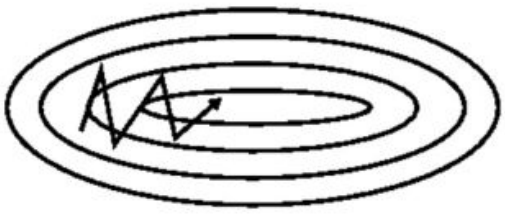


Image 3: SGD with momentum

NAG 跟上面 Momentum 公式的唯一区别在于，梯度不是根据当前参数位置 θ_{i-1}，而是根据先走了本来计划要走的一步后，达到的参数位置 $\theta_{i-1} - \alpha\beta d_{i-1}$ 计算出来的：

$$
\begin{aligned}
d_i &= \beta d_{i-1} + g\left(\theta_{i-1} - \alpha\beta d_{i-1}\right) \\
\theta_i &= \theta_{i-1} - \alpha d_i
\end{aligned}
$$

应用在 LASSO 问题的求解：FISTA 算法

3.5 交替方向乘子法（Alternating Direction Method of Multipliers, ADMM）

3.5.1 问题模型

通常用于解决存在两个优化变量的只含等式约束的优化类问题，其一般形式为：

$$\begin{aligned} \min_{x,z} \quad & f(x)+g(z) \\ \text{s.t.} \quad & Ax+Bz=c \end{aligned}$$

其中，$x \in R^n, z \in R^m$ 是优化变量，等式约束中 $A \in R^{p\times n}, B \in R^{p\times m}, c \in R^p$，$f, g$ 都是凸函数。

3.5.2 增广拉格朗日函数（Augmented Lagrangian）

定义增广拉格朗日函数:

$$L_\rho(x,z,u) = f(x)+g(z)+u^T(Ax+Bz-c)+\frac{\rho}{2}\|Ax+Bz-c\|_2^2$$

增广拉格朗日函数就是在关于原问题的拉格朗日函数之后增加了一个和约束条件有关的惩罚项，其中 $\rho > 0$

3.5.3 算法流程

每一步只更新一个变量而固定另外两个变量，如此交替重复更新。即，对于 $k = 1, 2, 3, ...$，重复如下步骤:

$$\begin{aligned} &\text{step1}: x^{(k)} = \arg\min_x L_\rho\left(x, z^{(k-1)}, u^{(k-1)}\right) \\ &\text{step2}: z^{(k)} = \arg\min_z L_\rho\left(x^{(k)}, z, u^{(k-1)}\right) \\ &\text{step3}: u^{(k)} = u^{(k-1)} + \rho\left(Ax^{(k)} + Bz^{(k)} - c\right) \end{aligned}$$

ADMM 算法提供了一个将多优化变量问题转化为单优化变量问题的转化方式（即，交替方向），并未涉及具体的下降方法，其中关于 x 和 z 的更新过程需要结合具体的下降类算法，如梯度下降算法等。

为简化形式，如果令 $w = \frac{u}{\rho}$, 则增广拉格朗日函数可以写成:

$$L_\rho(x,z,w) = f(x)+g(z)+\frac{\rho}{2}\|Ax+Bz-c+w\|_2^2 - \frac{\rho}{2}\|w\|_2^2$$

上面这个式子被称为是 ADMM 的缩放形式。

相应地，更新步骤变为:

$$\begin{aligned} \text{step1}: \quad & x^{(k)} = \arg\min_x f(x) + \tfrac{\rho}{2}\left\|Ax + Bz^{(k-1)} - c + w^{(k-1)}\right\|_2^2 \\ \text{step 2}: \quad & z^{(k)} = \arg\min_z g(z) + \tfrac{\rho}{2}\left\|Ax^{(k)} + Bz - c + w^{(k-1)}\right\|_2^2 \\ \text{step3}: \quad & w^{(k)} = w^{(k-1)} + Ax^{(k)} + Bz^{(k)} - c \end{aligned}$$

note that: $w^{(k)} = w^{(0)} + \sum_{i=1}^{k} \left(Ax^{(i)} + Bz^{(i)} - c\right)$，即第 k 次迭代的 $w^{(k)}$ 等于其初始值 $w^{(0)}$ 加上约束条件残差的累加和。

3.5.4 ADMM 求解 LASSO 问题

考虑 LASSO 问题：$\min_\beta \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1$

重写为：

$$\begin{aligned} &\min_{\beta,\alpha} \tfrac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\alpha\|_1 \\ &\text{s.t.} \quad \beta - \alpha = 0 \end{aligned}$$

构造增光拉格朗日函数：

$$L_\rho(\beta, \alpha, u) = \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\alpha\|_1 + u^T(\beta - \alpha) + \frac{\rho}{2}\|\beta - \alpha\|_2^2$$

更新步骤：

首先更新 β，由于 L_p 对 β 是可导的，令 $\frac{\partial L}{\partial \beta} = 0$，有

$$\begin{aligned} &\frac{\partial L}{\partial \beta} = -X^T(y - X\beta) + u + \rho(\beta - \alpha) = 0 \\ \Rightarrow\ &\beta = \left(X^TX + \rho I\right)^{-1}\left(X^Ty + \rho\alpha - u\right) \\ \Rightarrow\ &\beta = \left(X^TX + \rho I\right)^{-1}\left(X^Ty + \rho(\alpha - w)\right) \end{aligned}$$

然后求 α 的更新值：

$$\begin{aligned} \arg\min_\alpha L_\rho(\beta, \alpha, u) &= \arg\min_\alpha \left\{\lambda\|\alpha\|_1 + u^T(\beta - \alpha) + \frac{\rho}{2}\|\beta - \alpha\|_2^2\right\} \\ &= \arg\min_\alpha \left\{\lambda\|\alpha\|_1 + u^T(\beta - \alpha) + \frac{\rho}{2}\|\beta - \alpha\|_2^2\right\} \times \frac{2}{\rho} \\ &= \arg\min_\alpha \left\{\frac{2\lambda}{\rho}\|\alpha\|_1 + \left\|\frac{u}{\rho} + (\beta - \alpha)\right\|_2^2 - \left\|\frac{u}{\rho}\right\|_2^2\right\} \\ &= \arg\min_\alpha \left\{\frac{2\lambda}{\rho}\|\alpha\|_1 + \left\|\alpha - \left(\frac{u}{\rho} + \beta\right)\right\|_2^2\right\} \\ &= S_{\frac{\lambda}{\rho}}\left(\frac{u}{\rho} + \beta\right) = S_{\frac{\lambda}{\rho}}(w + \beta) \end{aligned}$$

其中最后一步用到了软阈值算子，原问题转化为了具有解析解形式的问题。

综上，LASSO 问题的更新步骤为：

$$\begin{aligned}
\beta^{(k)} &= \left(X^T X + \rho I\right)^{-1}\left(X^T y + \rho\left(\alpha^{(k-1)} - w^{(k-1)}\right)\right) \\
\alpha^{(k)} &= S_{\frac{\lambda}{\rho}}\left(w^{(k-1)} + \beta^{(k)}\right) \\
w^{(k)} &= w^{(k-1)} + \beta^{(k)} - \alpha^{(k)}
\end{aligned}$$

Moreover, large-scale ADMM using distributed computation is denoted as:

$$\begin{aligned}
&\min_{x,z} \sum_{i=1}^{N} f_i\left(x_i\right) + g(z) \\
&\text{s.t. } A_i x_i + B_i z = C_i
\end{aligned}$$

增广拉格朗日函数为:

$$L(x,z,u) = \sum_{i=1}^{N} f_i\left(x_i\right) + g(z) + \sum_{i=1}^{N}\left[u^{\top}\left(A_i x_i + B_i z - C_i\right) + \frac{\rho}{2}\left\|A_i x_i + B_i z - C_i\right\|_2^2\right]$$

重复上述的三个更新步骤即可。