最优化算法

杨周旺

中国科学技术大学
数学科学学院

2023年2月

Outline I

Outline II

Description

- The course is devoted to the mathematical fundamentals of optimization and the practical algorithms of optimization.
- The course covers the topics of nonlinear continuous optimization, sparse optimization, and optimization methods for machine learning.

Objectives

Objectives of the course are

- to develop an understanding of the fundamentals of optimization;
- to learn how to analyze the widely used algorithms for optimization;
- to become familiar with the implementation of optimization algorithms.

Prerequisites

- Knowledge of Linear Algebra, Real Analysis, and Mathematics of Operations Research are very important for this course.
- Simultaneously, the ability to write computer programs of algorithms is also required.

Topics Covered

- Unconstrained Optimization
- Constrained Optimization
- Convex Optimization
- Sparse Optimization
- Optimization Methods for Large-scale Machine Learning

Textbook and References

1. R. Fletcher. Practical Methods of Optimization (2nd Edition), John Wiley & Sons, 1987.
2. J. Nocedal and S. J. Wright. Numerical Optimization (2nd Edition), Springer, 2006.
3. S. Boyd and L. Vandenberghe. Convex Optimization, Cambridge University Press, 2004.
4. M. Elad. Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing. Springer, 2010.
5. L. Bottou, F.E. Curtis, J. Nocedal. Optimization methods for large-scale machine learning. SIAM Review, 60(2): 223-311, 2018.

Grading

(1) Homework (10%)

(2) Project (30%)

(3) Final Exam (60%)

Outline I

Outline II

无约束最优化

无约束最优化问题

$$\min_{x \in \mathbb{R}^n} \quad f(x) \tag{1}$$

其目标函数f是定义在$\mathbb{R}^n$上的实值函数，决策变量x的可取值之集合是全空间 $\mathbb{R}^n$.

梯度类求解算法

梯度向量$\nabla f(\mathrm{x})$是函数f在点x处增加最快的方向，故它成为最优化时的重要工具。实际上针对无约束最优化问题，大家所知的求解算法中大多属于下面的梯度方法类。

GRADIENT （梯度法类）

(0) 初始化：选取适当的初始点$\mathrm{x}^{(0)} \in \mathbb{R}^n$, 令$k := 0$.

(1) 计算搜索方向：利用适当的正定对称阵H_k计算搜索方向向量$\mathrm{d}^{(k)} := -H_k \nabla f(\mathrm{x}^{(k)})$. （如果$\nabla f(\mathrm{x}^{(k)}) = 0$, 则结束计算）

(2) 确定步长因子：解一维最优化问题$\min\limits_{\alpha \geq 0} f(\mathrm{x}^{(k)} + \alpha \mathrm{d}^{(k)})$, 求出步长$\alpha = \alpha_k$, 令$\mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} + \alpha_k \mathrm{d}^{(k)}$, $k := k + 1$, 回到第(1)步。

构造搜索方向：负梯度方向

无约束最优化问题：$\min\limits_{x\in\mathbb{R}^n} f(x)$

$$f(x) = f(x^{(k)}) + \nabla f(x^{(k)})^T(x - x^{(k)}) + O(\|x - x^{(k)}\|^2) \tag{2}$$

取负梯度方向

$$d^{(k)} = -\nabla f(x^{(k)}),$$

则当α_k足够小时，总能使

$$f(x^{(k)} + \alpha_k d^{(k)}) < f(x^{(k)}).$$

构造搜索方向：牛顿方向

$$
\begin{aligned}
f(\mathrm{x}) = \quad & f(\mathrm{x}^{(k)}) + \nabla f(\mathrm{x}^{(k)})^T(\mathrm{x} - \mathrm{x}^{(k)}) \\
& + \frac{1}{2}(\mathrm{x} - \mathrm{x}^{(k)})^T \nabla^2 f(\mathrm{x}^{(k)})(\mathrm{x} - \mathrm{x}^{(k)}) + O(\|\mathrm{x} - \mathrm{x}^{(k)}\|^3)
\end{aligned} \tag{3}
$$

取搜索方向

$$
\mathrm{d}^{(k)} = -G_k^{-1} \nabla f(\mathrm{x}^{(k)}),
$$

其中$G_k = \nabla^2 f(\mathrm{x}^{(k)})$为函数$f$在$\mathrm{x}^{(k)}$点处的Hesse矩阵。

确定步长因子：一维搜索

在迭代格式中，通过解一维最优化问题

$$\min_{\alpha \geq 0} \varphi(\alpha) = f(\mathrm{x}^{(k)} + \alpha \mathrm{d}^{(k)}) \tag{4}$$

确定步长因子的方法称为**一维搜索**(Line Search).

确定步长因子：一维搜索

若以问题(4)的最优解为步长，此时称为**精确一维搜索**(Exact Line Search).

经常用到的精确一维搜索有黄金分割法和插值迭代法。即使说是精确一维搜索，通过有限次计算求出问题(4)的严密解一般也是不可能的，实际上在得到有足够精度的近似解时，就采用它作为步长。

确定步长因子：一维搜索

在实际计算中，往往不是求解一维最优化问题(4)，而是找出满足某些适当条件的粗略近似解作为步长，此时称为**非精确一维搜索**(Inexact Line Search).

与精确一维搜索相比，在很多情况下采用非精确一维搜索可以提高整体计算效率。

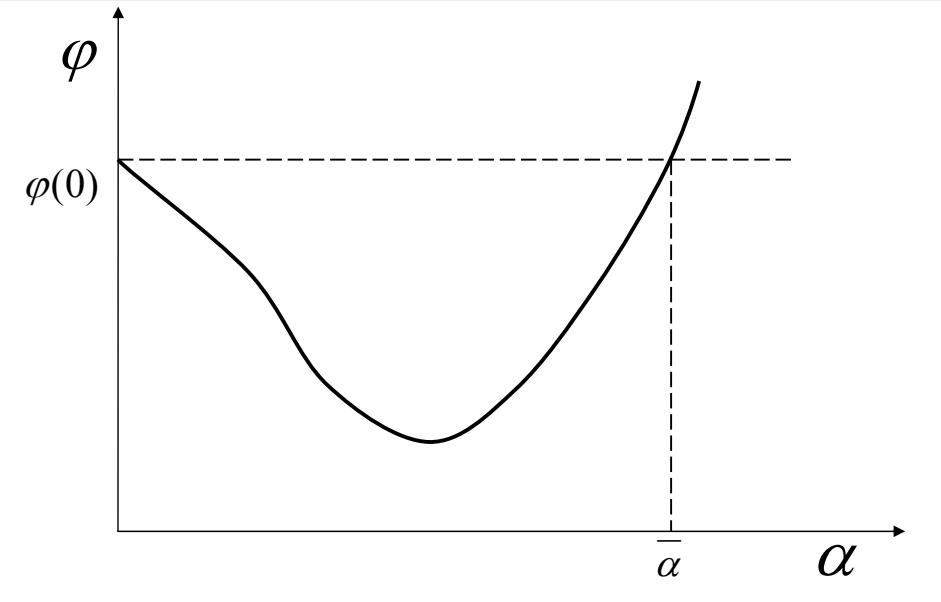
确定步长因子：一维搜索

设$\bar{\alpha}_k$是使得

$$f(\mathrm{x}^{(k)} + \alpha \mathrm{d}^{(k)}) = f(\mathrm{x}^{(k)})$$

的最小正数α.

于是，我们将在区间$[0, \bar{\alpha}_k]$内求得满足适当条件的可接受的步长因子，即$\alpha \in [0, \bar{\alpha}_k]$.



确定步长因子：一维搜索

Goldstein(1965) conditions:

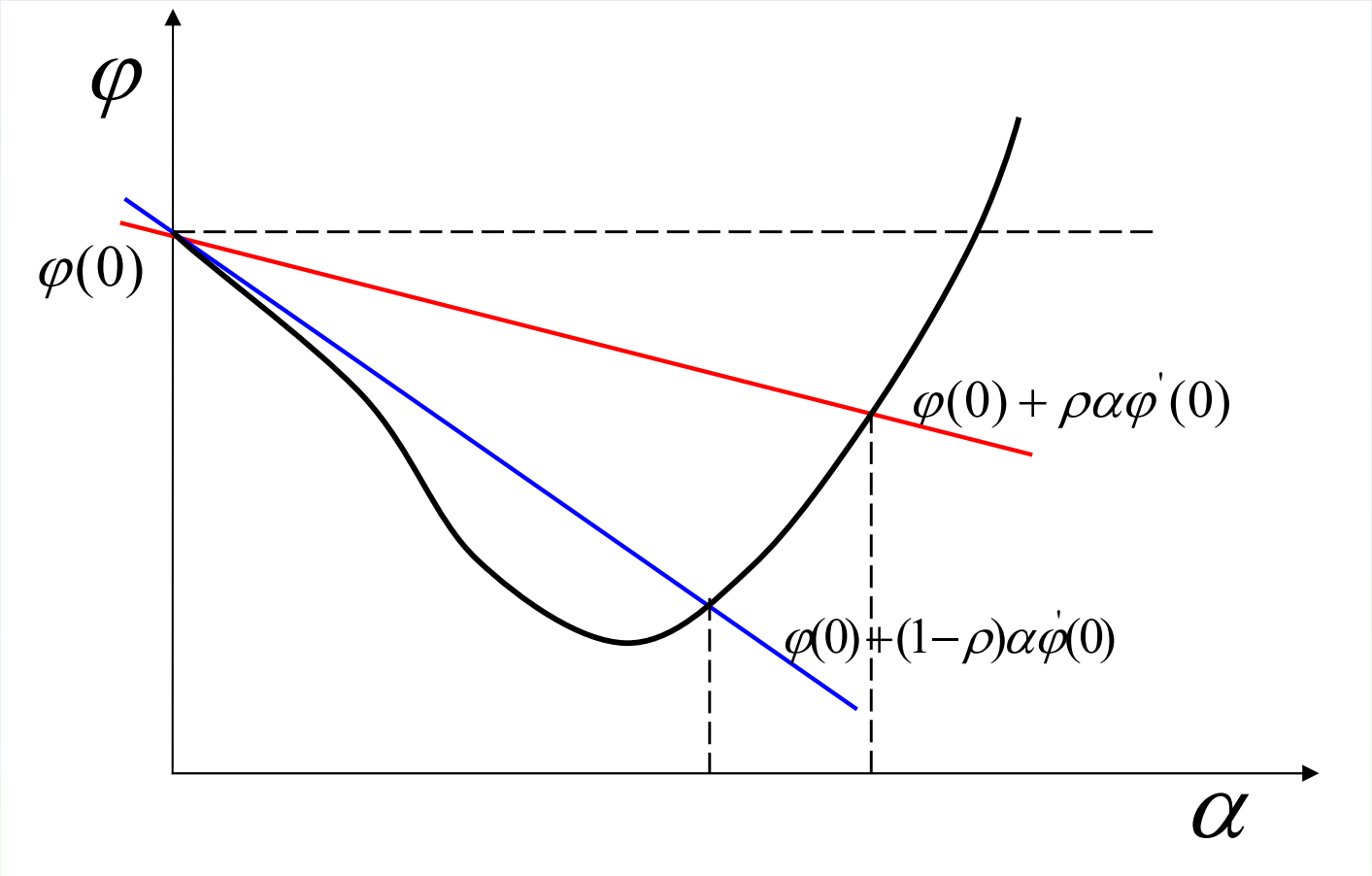
$$\varphi(\alpha) \leq \varphi(0) + \rho\alpha\varphi'(0) \tag{5}$$

$$\varphi(\alpha) \geq \varphi(0) + (1-\rho)\alpha\varphi'(0) \tag{6}$$

其中$\rho \in (0, 1/2)$是一个固定参数。

确定步长因子：一维搜索

Goldstein(1965) conditions:



确定步长因子：一维搜索

Wolfe(1968)-Powell(1976) conditions:

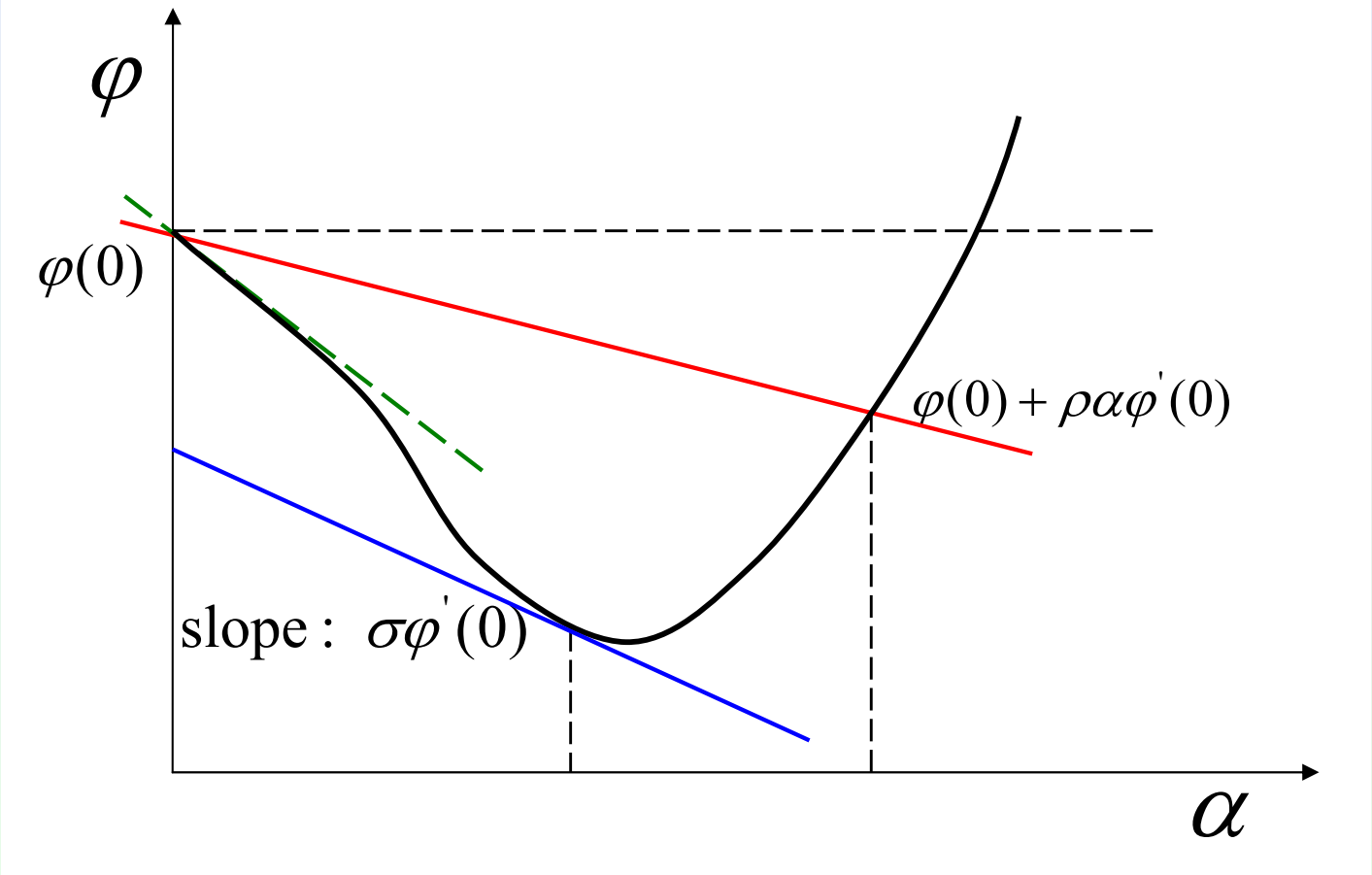
$$\varphi(\alpha) \leq \varphi(0) + \rho\alpha\varphi'(0) \tag{7}$$

$$\varphi'(\alpha) \geq \sigma\varphi'(0) \tag{8}$$

其中$\sigma \in (\rho, 1)$是另一个固定参数。

确定步长因子：一维搜索

Wolfe(1968)-Powell(1976) conditions:



确定步长因子：一维搜索

在很多实际算法中，式(8)常被强化的双边条件所取代

$$|\varphi^{'}(\alpha)| \leq -\sigma\varphi^{'}(0) \tag{9}$$

确定步长因子：一维搜索

基于Wolfe-Powell准则的非精确一维搜索算法：

(0) 给定初始一维搜索区间$[0, \bar{\alpha}]$, 以及$\rho \in (0, 1/2)$, $\sigma \in (\rho, 1)$.
计算$\varphi_0 = \varphi(0) = f(\mathrm{x}^{(k)})$, $\varphi_0^{'} = \varphi^{'}(0) = \nabla f(\mathrm{x}^{(k)})^T \mathrm{d}^{(k)}$.
并令$a_1 = 0, a_2 = \bar{\alpha}, \varphi_1 = \varphi_0, \varphi_1^{'} = \varphi_0^{'}$.
选取适当的$\alpha \in (a_1, a_2)$.

确定步长因子：一维搜索

基于Wolfe-Powell准则的非精确一维搜索算法：

(1) 计算$\varphi = \varphi(\alpha) = f(\mathrm{x}^{(k)} + \alpha \mathrm{d}^{(k)})$. 若$\varphi(\alpha) \leq \varphi(0) + \rho\alpha\varphi'(0)$, 则转到第(2)步。否则，由$\varphi_1, \varphi_1', \varphi$构造两点二次插值多项式$p^{(1)}(t)$, 并得其极小点

$$\hat{\alpha} = a_1 + \frac{1}{2}\frac{(a_1 - \alpha)^2\varphi_1'}{(\varphi_1 - \varphi) - (a_1 - \alpha)\varphi_1'}.$$

于是置$a_2 = \alpha, \alpha = \hat{\alpha}$, 重复第(1)步。

确定步长因子：一维搜索

基于Wolfe-Powell准则的非精确一维搜索算法：

(2) 计算$\varphi^{'} = \varphi^{'}(\alpha) = \nabla f(\mathrm{x}^{(k)} + \alpha \mathrm{d}^{(k)})^T \mathrm{d}^{(k)}$. 若$\varphi^{'}(\alpha) \geq \sigma\varphi^{'}(0)$, 则输出$\alpha_k = \alpha$, 并停止搜索。否则，由$\varphi, \varphi^{'}, \varphi_1^{'}$构造两点二次插值多项式$p^{(2)}(t)$, 并得其极小点

$$\hat{\alpha} = \alpha - \frac{(a_1 - \alpha)\varphi^{'}}{\varphi_1^{'} - \varphi^{'}}.$$

于是置$a_1 = \alpha, \alpha = \hat{\alpha}, \varphi_1 = \varphi, \varphi_1^{'} = \varphi^{'}$, 返回第(1)步。

确定步长因子：一维搜索

[思考题：请写出上述基于Wolfe-Powell准则的非精确一维搜索算法中插值多项式$p^{(1)}(t), p^{(2)}(t)$的具体表达式。]

全局收敛

从任意初始点出发，如果某迭代算法产生的点列的极限（聚点），在适当假定下可保证恒为问题的最优解（或者稳定点），则称该迭代法具有全局收敛性(Global Convergence).

与此相对，如果仅在解的附近选取初始点时，才可以保证所生成的点列收敛于该解，则称这样的迭代法有局部收敛性(Local Convergence).

全局收敛

为了证明迭代法的下降性，我们应尽量避免搜索方向与负梯度方向几乎正交的情形，即要求$\mathrm{d}^{(k)}$偏离$\mathrm{g}^{(k)} = \nabla f(\mathrm{x}^{(k)})$的正交方向远一些。否则，${\mathrm{g}^{(k)}}^T \mathrm{d}^{(k)}$接近于零，$\mathrm{d}^{(k)}$几乎不是下降方向。

为此，我们假设$\mathrm{d}^{(k)}$与$-\mathrm{g}^{(k)}$的夹角θ_k满足

$$\theta_k \leq \frac{\pi}{2} - \mu, \ \forall k \tag{10}$$

其中$\mu > 0$（与k无关）。

全局收敛

显然$\theta_k \in [0, \pi/2)$, 其定义为

$$\cos\theta_k = \frac{-\mathrm{g}^{(k)^T}\mathrm{d}^{(k)}}{\|\mathrm{g}^{(k)}\|\|\mathrm{d}^{(k)}\|} = \frac{-\mathrm{g}^{(k)^T}\mathrm{s}^{(k)}}{\|\mathrm{g}^{(k)}\|\|\mathrm{s}^{(k)}\|} \tag{11}$$

这里$\mathrm{s}^{(k)} = \alpha_k \mathrm{d}^{(k)} = \mathrm{x}^{(k+1)} - \mathrm{x}^{(k)}$.

全局收敛

下面给出各种步长准则下的下降算法的全局收敛性结论。

全局收敛性定理：

设$\nabla f(\mathrm{x})$在水平集 $L(\mathrm{x}^{(0)}) = \{\mathrm{x} \mid f(\mathrm{x}) \leq f(\mathrm{x}^{(0)})\}$ 上存在且连续。下降算法的搜索方向$\mathrm{d}^{(k)}$与$-\nabla f(\mathrm{x}^{(k)})$ 之间的夹角θ_k满足式(10), 其中步长α_k由三种方法之一确定：

(1) 精确一维搜索

(2) Goldstein准则 (5),(6)

(3) Wolfe-Powell准则 (7),(8)

那么，或者对某个k有$\nabla f(\mathrm{x}^{(k)}) = 0$, 或者$f(\mathrm{x}^{(k)}) \to -\infty$, 或者$\nabla f(\mathrm{x}^{(k)}) \to 0$.

全局收敛

全局收敛性证明：（只证明Wolfe-Powell准则的情形）

假设对所有的k, $\mathrm{g}^{(k)} = \nabla f(\mathrm{x}^{(k)}) \neq 0$ 和$f(\mathrm{x}^{(k)})$有下界，
故$f(\mathrm{x}^{(k)}) - f(\mathrm{x}^{(k+1)}) \to 0$. 由式(7)得，$-\mathrm{g}^{(k)\,T}\mathrm{s}^{(k)} \to 0$.

（反证）若$\mathrm{g}^{(k)} \to 0$不成立，那么存在$\varepsilon > 0$和子列$\{\mathrm{x}^{(k)}\}_{k\in K}$使得$\|\mathrm{g}^{(k)}\| \geq \varepsilon$. 从而由

$$-\mathrm{g}^{(k)\,T}\mathrm{s}^{(k)} = \|\mathrm{g}^{(k)}\|\|\mathrm{s}^{(k)}\|\cos\theta_k \geq \varepsilon\|\mathrm{s}^{(k)}\|\sin\mu$$

以及式(10)有$\|\mathrm{s}^{(k)}\| \to 0$.

全局收敛

全局收敛性证明（续）：

又因为$\mathrm{g}(\mathrm{x}) = \nabla f(\mathrm{x})$在$L(\mathrm{x}^{(0)})$上连续，所以

$$\begin{array}{c} \mathrm{g}^{(k+1)\,T}\mathrm{s}^{(k)} = \mathrm{g}^{(k)\,T}\mathrm{s}^{(k)} + o(\|\mathrm{s}^{(k)}\|) \\ \Downarrow \\ \dfrac{\mathrm{g}^{(k+1)\,T}\mathrm{s}^{(k)}}{\mathrm{g}^{(k)\,T}\mathrm{s}^{(k)}} \to 1. \end{array} \tag{12}$$

全局收敛

全局收敛性证明（续）：

而这与Wolfe-Powell准则的式(8)

$$\frac{\mathrm{g}^{(k+1)^T}\mathrm{s}^{(k)}}{\mathrm{g}^{(k)^T}\mathrm{s}^{(k)}} \leq \sigma < 1 \tag{13}$$

相矛盾。因此有$\mathrm{g}^{(k)} \to 0$

[思考题：请补充证明基于Goldstein准则的非精确一维搜索算法的全局收敛性。]

最速下降法

最速下降法取负梯度作为迭代算法的搜索方向，其迭代格式为

$$x^{(k+1)} = x^{(k)} - \alpha_k \nabla f(x^{(k)}).$$

最速下降法

算法：

(0) 选取初始点$\mathrm{x}^{(0)}$, 设置终止误差$\varepsilon > 0$, 令$k := 0$.

(1) 计算$\mathrm{g}^{(k)} = \nabla f(\mathrm{x}^{(k)})$. 若$\|\mathrm{g}^{(k)}\| < \varepsilon$, 则停止迭代并输出$\mathrm{x}^{(k)}$. 否则进行第(2)步。

(2) 令$\mathrm{d}^{(k)} = -\mathrm{g}^{(k)}$, 并由一维搜索确定步长因子$\alpha_k$使得

$$f(\mathrm{x}^{(k)} + \alpha_k \mathrm{d}^{(k)}) = \min_{\alpha > 0} f(\mathrm{x}^{(k)} + \alpha \mathrm{d}^{(k)}).$$

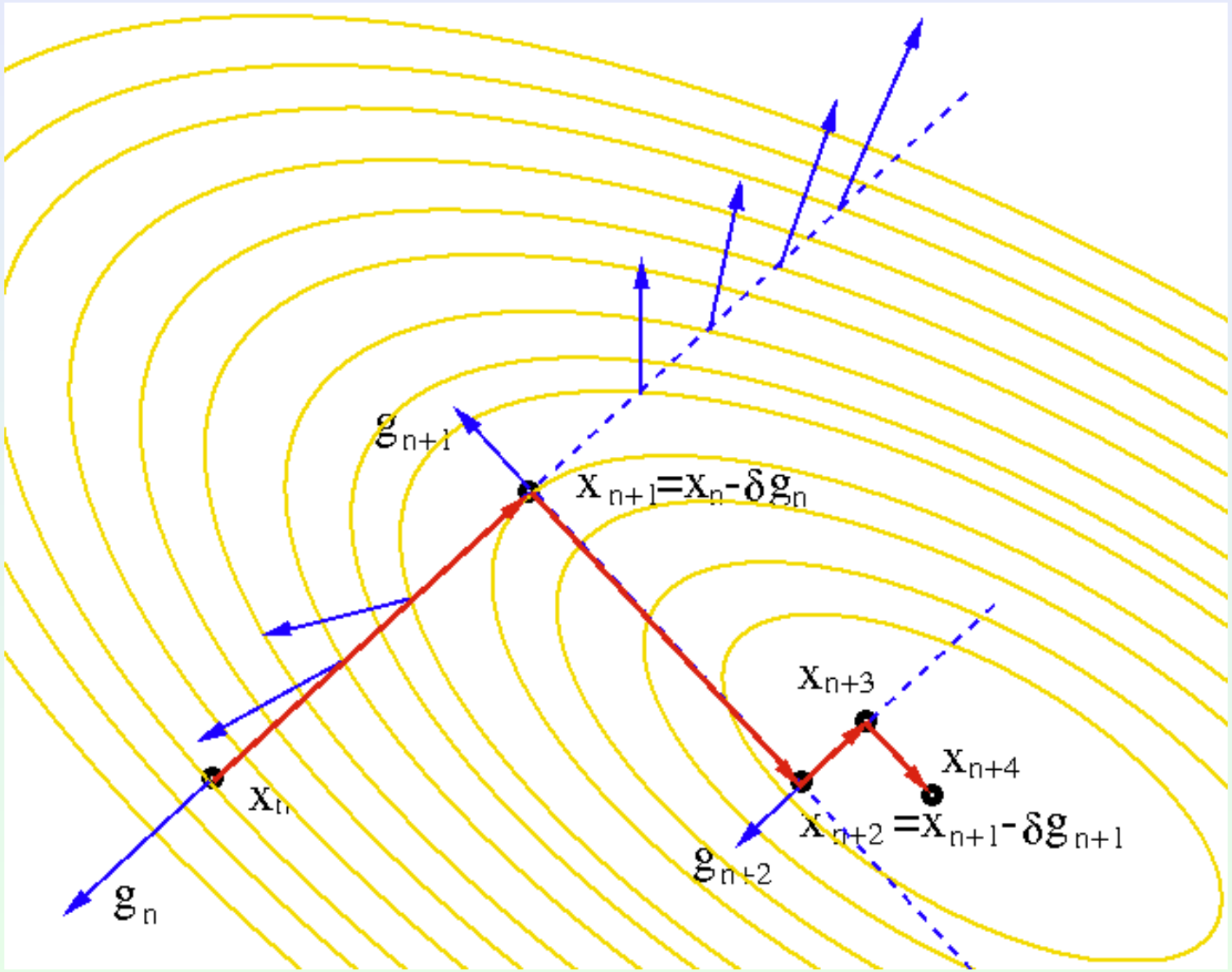
(3) 迭代更新$\mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} + \alpha_k \mathrm{d}^{(k)}$, 置$k := k + 1$, 回到第(1)步。

最速下降法

最速下降法全局收敛性定理：

设$f(x) \in C^1$，在最速下降法中采用（精确或非精确）一维搜索，则产生的迭代点列$\{x^{(k)}\}$的每一个聚点都是驻点。

最速下降法



最速下降法

一般地，最速下降法只有线性收敛速度。

如下例子是一个非常著名的测试函数 (Rosenbrock function)

$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

牛顿法

设$f(\mathrm{x})$是二次可微实函数，在$\mathrm{x}^{(k)}$附近作二阶Taylor展开近似

$$f(\mathrm{x}^{(k)}+\mathrm{s}) \approx q^{(k)}(\mathrm{s}) = f(\mathrm{x}^{(k)}) + \mathrm{g}^{(k)\,T}\mathrm{s} + \frac{1}{2}\mathrm{s}^T G_k \mathrm{s} \tag{14}$$

其中$\mathrm{g}^{(k)} = \nabla f(\mathrm{x}^{(k)}), G_k = \nabla^2 f(\mathrm{x}^{(k)})$.

将$q^{(k)}(\mathrm{s})$极小化便得

$$\mathrm{s} = -G_k^{-1}\mathrm{g}^{(k)}. \tag{15}$$

上式给出的搜索方向$-G_k^{-1}\mathrm{g}^{(k)}$称为牛顿方向(Newton Direction).

牛顿法

在目标函数是正定二次函数

$$f(\mathrm{x}) = \frac{1}{2}\mathrm{x}^T G\mathrm{x} - \mathrm{c}^T\mathrm{x}$$

的情况下(G为正定阵)，对任意的x有$\nabla^2 f(\mathrm{x}) = G$.

在第一次迭代里令$H_0 = G^{-1}$, 则有

$$\mathrm{d}^{(0)} = -H_0\nabla f(\mathrm{x}^{(0)}) = -G^{-1}(G\mathrm{x}^{(0)} - \mathrm{c}) = -(\mathrm{x}^{(0)} - \mathrm{x}^*).$$

这里，$\mathrm{x}^* = G^{-1}\mathrm{c}$是问题的最优解。若$\mathrm{x}^{(0)} \neq \mathrm{x}^*$, 取步长$\alpha_0 = 1$, 于是得$\mathrm{x}^{(1)} = \mathrm{x}^{(0)} + \alpha_0\mathrm{d}^{(0)} = \mathrm{x}^*$. 由此知道，不管初始点$\mathrm{x}^{(0)}$如何取，在一次迭代后即可到达最优解$\mathrm{x}^*$.

牛顿法

根据以上事实，可以认为即使对于一般的非线性函数$f(\mathrm{x})$, 在迭代中令搜索方向

$$\mathrm{d}^{(k)} = -\nabla^2 f(\mathrm{x}^{(k)})^{-1} \nabla f(\mathrm{x}^{(k)})$$

也是较合适的。

特别地，步长$\alpha_k \equiv 1$的迭代公式为

$$\mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} + \mathrm{d}^{(k)} = \mathrm{x}^{(k)} - G_k^{-1} \mathrm{g}^{(k)}. \tag{16}$$

这就是经典的牛顿迭代法

牛顿法

对于正定二次函数而言，牛顿法一步即可达到最优解。对于非二次函数，牛顿法并不能保证经有限次迭代求得最优解。但由于目标函数在极小点附近可用二次函数较好地近似，故当初始点靠近极小点时，牛顿法的收敛速度一般会很快。

可以证明牛顿法的局部收敛性和二阶收敛速率。

牛顿法

牛顿法收敛定理：

设$f \in C^2$, $\mathrm{x}^{(k)}$充分靠近x^*, 其中$\nabla f(\mathrm{x}^*) = 0$. 如果$\nabla^2 f(\mathrm{x}^*)$正定，目标函数的Hesse矩阵$G(\mathrm{x})$满足Lipschitz条件，即存在$\beta > 0$使得对所有$(i, j)$有

$$|G_{ij}(\mathrm{x}) - G_{ij}(\mathrm{y})| \leq \beta \|\mathrm{x} - \mathrm{y}\|. \tag{17}$$

则对一切的k, 牛顿迭代(16)有定义，所得序列$\{\mathrm{x}^{(k)}\}$收敛到x^*, 且具有二阶收敛速率。

牛顿法

证明一：

记 $\mathrm{g}(\mathrm{x}) = \nabla f(\mathrm{x})$, 因为$f \in C^2$, 我们有

$$\mathrm{g}(\mathrm{x} - \mathrm{h}) = \mathrm{g}(\mathrm{x}) - G(\mathrm{x})\mathrm{h} + O(\|\mathrm{h}\|^2).$$

令$\mathrm{x} = \mathrm{x}^{(k)}, \mathrm{h} = \mathrm{h}^{(k)} = \mathrm{x}^{(k)} - \mathrm{x}^*$ 代入上式得

$$0 = \mathrm{g}(\mathrm{x}^*) = \mathrm{g}(\mathrm{x}^{(k)} - \mathrm{h}^{(k)}) = \mathrm{g}(\mathrm{x}^{(k)}) - G(\mathrm{x}^{(k)})\mathrm{h}^{(k)} + O(\|\mathrm{h}^{(k)}\|^2). \quad (18)$$

牛顿法

证明一（续）：

由于$G(x)$满足Lipschitz条件，易证$\left[G(x^{(k)})\right]^{-1}$有界。方程(18)两边同时乘以$\left[G(x^{(k)})\right]^{-1}$得

$$\begin{aligned} 0 &= \left[G(x^{(k)})\right]^{-1} g(x^{(k)}) - h^{(k)} + O(\|h^{(k)}\|^2) \\ &= x^* - (x^{(k)} - \left[G(x^{(k)})\right]^{-1} g(x^{(k)})) + O(\|h^{(k)}\|^2) \\ &= x^* - x^{(k+1)} + O(\|h^{(k)}\|^2) \\ &= -h^{(k+1)} + O(\|h^{(k)}\|^2) \end{aligned}$$

所以$\|h^{(k+1)}\| = O(\|h^{(k)}\|^2)$, 即牛顿迭代法具有二阶收敛速率。

牛顿法

证明二：

对于牛顿迭代法，我们记

$$x^{(k+1)} = x^{(k)} - G_k^{-1} g^{(k)} \triangleq \mathcal{A}(x^{(k)}). \tag{19}$$

注意到$g(x^*) = 0$, $G(x^*)$正定（非奇异），有$\mathcal{A}(x^*) = x^*$.

于是由$x^{(k+1)} - x^* = \mathcal{A}(x^{(k)}) - \mathcal{A}(x^*)$ 得

$$\begin{aligned} \|x^{(k+1)} - x^*\| &= \|\mathcal{A}(x^{(k)}) - \mathcal{A}(x^*)\| \\ &\leq \|\mathcal{A}'(x^*)(x^{(k)} - x^*)\| + \frac{1}{2}\|\mathcal{A}''(\bar{x})\|\|x^{(k)} - x^*\|^2, \end{aligned}$$

其中$\bar{x}$位于$x^{(k)}$和x^*之间的线段上。

牛顿法

证明二（续）：

显然

$$\mathcal{A}'(\mathrm{x}) = [\mathrm{x} - G(\mathrm{x})^{-1}\mathrm{g}(\mathrm{x})]' = -[G(\mathrm{x})^{-1}]'\mathrm{g}(\mathrm{x})$$

所以$\mathcal{A}'(\mathrm{x}^*) = 0$. 从而有

$$\|\mathrm{h}^{(k+1)}\| = \|\mathrm{x}^{(k+1)} - \mathrm{x}^*\| \leq \gamma\|\mathrm{x}^{(k)} - \mathrm{x}^*\|^2 = \gamma\|\mathrm{h}^{(k)}\|^2$$

其中常数γ仅依赖于$f(\mathrm{x})$在x^*附近的三阶导数。

牛顿法

在式(16)的牛顿迭代法里，如果选取的初始点$x^{(0)}$不在解x^*的附近，那么生成的点列$\{x^{(k)}\}$未必收敛于最优解。

为保证算法的全局收敛性，有必要对牛顿法作某些改进。

牛顿法

比如，在牛顿法中也可采用一维搜索来确定步长。

阻尼牛顿法：

(0) 选取初始点$x^{(0)}$, 设置终止误差$\varepsilon > 0$, 令$k := 0$.

(1) 计算$g^{(k)} = \nabla f(x^{(k)})$. 若$\|g^{(k)}\| < \varepsilon$, 停止迭代并输出$x^{(k)}$. 否则进行第(2)步。

(2) 解线性方程组$G_k d = -g^{(k)}$, 求出牛顿方向$d^{(k)}$.

(3) 采用一维搜索确定步长因子α_k, 令$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$, 置$k := k + 1$, 回到第(1)步。

牛顿法

牛顿法面临的主要困难是Hesse矩阵$G_k = \nabla^2 f(\mathrm{x}^{(k)})$不正定。这时二阶近似模型不一定有极小点，即二次函数$q^{(k)}(\mathrm{s})$是无界的。

为了克服这些困难，人们提出了很多修正措施。

牛顿法

Goldstein & Price (1967)

$$d^{(k)} = \begin{cases} -G_k^{-1} g^{(k)}, & \text{if } \cos\theta_k > \eta \\ -g^{(k)}, & \text{otherwise} \end{cases} \tag{20}$$

牛顿法

Levenberg(1944), Marquardt(1963), Goldfeld et. al(1966)

$$(G_k + \mu_k I)\mathrm{d}^{(k)} = -\mathrm{g}^{(k)} \tag{21}$$

牛顿法

设x是函数f的一个不定点，若方向d满足

$$\mathrm{d}^T \nabla^2 f(\mathrm{x})\mathrm{d} < 0,$$

则称d为f在x处的负曲率方向。

当Hesse矩阵$\nabla^2 f(\mathrm{x}^{(k)})$不正定时，负曲率方向法是修正牛顿法的另一种途径。

拟牛顿法

牛顿法的突出优点是局部收敛很快（具有二阶收敛速率），
但运用牛顿法需要计算二阶导，而且目标函数的Hesse矩阵$\nabla^2 f(x^{(k)})$ 可能非正定，甚至奇异。为了克服这些缺点，
人们提出了拟牛顿法。其基本思想是：用不含二阶导数的矩阵H_k近似牛顿法中的Hesse矩阵的逆$G(x^{(k)})^{-1}$.

由构造近似矩阵的方法不同，将出现不同的拟牛顿法。

拟牛顿法

回顾牛顿法的迭代

$$\begin{cases} G_k \mathrm{d} = -\mathrm{g}^{(k)} \\ \mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} + \alpha_k \mathrm{d}^{(k)} \end{cases}$$

为了构造Hesse矩阵逆G_k^{-1}的近似H_k, 我们先分析二阶导$\nabla^2 f(\mathrm{x}^{(k)})$ 与一阶导$\nabla f(\mathrm{x}^{(k)})$的关系。

拟牛顿法

设第k次迭代后得到$x^{(k+1)}$, 将目标函数$f(x)$在$x^{(k+1)}$处二阶Taylor展开：

$$\begin{aligned} f(x) \approx \quad & f(x^{(k+1)}) + \nabla f(x^{(k+1)})^T (x - x^{(k+1)}) \\ & + \frac{1}{2}(x - x^{(k+1)})^T \nabla^2 f(x^{(k+1)})(x - x^{(k+1)}), \end{aligned}$$

进一步有

$$\nabla f(x) \approx \nabla f(x^{(k+1)}) + \nabla^2 f(x^{(k+1)})(x - x^{(k+1)}),$$

于是令$x = x^{(k)}$得

$$\nabla f(x^{(k)}) \approx \nabla f(x^{(k+1)}) + \nabla^2 f(x^{(k+1)})(x^{(k)} - x^{(k+1)}).$$

拟牛顿法

记$\mathsf{s}^{(k)} = \mathsf{x}^{(k+1)} - \mathsf{x}^{(k)}$, $\mathsf{y}^{(k)} = \nabla f(\mathsf{x}^{(k+1)}) - \nabla f(\mathsf{x}^{(k)})$, 则有

$$\nabla^2 f(\mathsf{x}^{(k+1)})\mathsf{s}^{(k)} \approx \mathsf{y}^{(k)} \quad \text{or} \quad \nabla^2 f(\mathsf{x}^{(k+1)})^{-1}\mathsf{y}^{(k)} \approx \mathsf{s}^{(k)}.$$

这样，计算出$\mathsf{s}^{(k)}$和$\mathsf{y}^{(k)}$后，可依上式估计在$\mathsf{x}^{(k+1)}$处的Hesse矩阵的逆。我们有理由要求在迭代中构造出Hesse矩阵逆的近似H_{k+1}, 使其满足

$$H_{k+1}\mathsf{y}^{(k)} = \mathsf{s}^{(k)}. \tag{22}$$

通常把式(22)称作正割条件，也称为拟牛顿条件。

拟牛顿法

拟牛顿迭代算法的一般格式：

(0) 选取初始点$x^{(0)}$, 令$H_0 = I$, $k := 0$.

(1) 计算搜索方向$d^{(k)} = -H_k \nabla f(x^{(k)})$.

(2) 采用一维搜索确定步长因子α_k, 令$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$.

(3) 基于$x^{(k)}$到$x^{(k+1)}$的梯度变化，更新Hesse矩阵逆的近似，即确定满足正割条件的H_{k+1}. 置$k := k + 1$, 返回第(1)步。

拟牛顿法

下面我们就来讨论怎样构造及确定满足拟牛顿条件的Hesse矩阵逆的近似H_{k+1}.

拟牛顿法

设H_k是第k次迭代的Hesse矩阵逆的近似，我们希望以H_k来产生H_{k+1}，即

$$H_{k+1} = H_k + E_k,$$

其中E_k是一个低秩的矩阵。

为此，可采用对称秩一(SR1)校正

$$H_{k+1} = H_k + a\mathrm{u}\mathrm{u}^T, \ (a \in \mathbb{R}, \mathrm{u} \in \mathbb{R}^n).$$

拟牛顿法

由拟牛顿条件(22)知

$$H_{k+1}\mathrm{y}^{(k)} = H_k\mathrm{y}^{(k)} + (a\mathrm{u}^T\mathrm{y}^{(k)})\mathrm{u} = \mathrm{s}^{(k)}$$

故u必与方向$\mathrm{s}^{(k)} - H_k\mathrm{y}^{(k)}$一致，且假定$\mathrm{s}^{(k)} - H_k\mathrm{y}^{(k)} \neq 0$.

不妨取$\mathrm{u} = \mathrm{s}^{(k)} - H_k\mathrm{y}^{(k)}$, 此时$a = \dfrac{1}{\mathrm{u}^T\mathrm{y}^{(k)}}$, 从而得到

$$H_{k+1} = H_k + \frac{(\mathrm{s}^{(k)} - H_k\mathrm{y}^{(k)})(\mathrm{s}^{(k)} - H_k\mathrm{y}^{(k)})^T}{(\mathrm{s}^{(k)} - H_k\mathrm{y}^{(k)})^T\mathrm{y}^{(k)}}. \tag{23}$$

上式称为对称秩一校正。

拟牛顿法

二次终止性

定义： 如果一种迭代法能在确知的有限步内找到二次函数的极小点，则称这种方法具有二次终止性。

拟牛顿法

对称秩一校正的突出性质：

1. 针对二次函数具有遗传性，即 $H_k y^{(\ell)} = s^{(\ell)}, \ell = k-1, \cdots, 1, 0.$
2. 具有二次终止性，即对于二次函数不需要进行一维搜索而具有n步终止性质，且$H_n = [\nabla^2 f(x^*)]^{-1}$.

[思考题：请证明对称秩一校正拟牛顿法的上述性质。]

拟牛顿法

对称秩一校正的缺点是，不能保持迭代矩阵H_{k+1}的正定性。

仅当$(\mathrm{s}^{(k)} - H_k \mathrm{y}^{(k)})^T \mathrm{y}^{(k)} > 0$时，对称秩一校正才能保持正定性。而这个条件往往很难保证，即使$(\mathrm{s}^{(k)} - H_k \mathrm{y}^{(k)})^T \mathrm{y}^{(k)} > 0$ 满足，它也可能很小从而导致数值上的困难。

这些都使得对称秩一校正的拟牛顿法应用有较大局限性。

拟牛顿法

采用对称秩二(SR2)校正

$$H_{k+1} = H_k + a\mathrm{u}\mathrm{u}^T + b\mathrm{v}\mathrm{v}^T,$$

并使得拟牛顿条件(22)成立，则有

$$H_{k+1}\mathrm{y}^{(k)} = H_k\mathrm{y}^{(k)} + (a\mathrm{u}^T\mathrm{y}^{(k)})\mathrm{u} + (b\mathrm{v}^T\mathrm{y}^{(k)})\mathrm{v} = \mathrm{s}^{(k)}.$$

这里u, v显然不是唯一确定的，但有一种明显的选择是：

$$\begin{cases} \mathrm{u} = \mathrm{s}^{(k)}, & a\mathrm{u}^T\mathrm{y}^{(k)} = 1; \\ \mathrm{v} = H_k\mathrm{y}^{(k)}, & b\mathrm{v}^T\mathrm{y}^{(k)} = -1. \end{cases}$$

拟牛顿法

因此有

$$H_{k+1} = H_k + \frac{s^{(k)} s^{(k)T}}{s^{(k)T} y^{(k)}} - \frac{H_k y^{(k)} y^{(k)T} H_k}{y^{(k)T} H_k y^{(k)}}. \tag{24}$$

上式称为 DFP(Davidon-Fletcher-Powell)校正公式，由Davidon(1959)提出，后经Fletcher & Powell(1963)修改而来。

拟牛顿法

DFP校正(24)是典型的拟牛顿校正公式，它有很多重要性质。

（一）对于二次函数（采用精确一维搜索）

1. 遗传性，即$H_k \mathrm{y}^{(\ell)} = \mathrm{s}^{(\ell)}, \ell = k-1, \cdots, 1, 0.$
2. 二次终止性，即$H_n = [\nabla^2 f(\mathrm{x}^*)]^{-1}.$
3. 共轭性，即当取$H_0 = I$时，迭代产生共轭方向。

（二）对于一般非线性函数

1. 校正保持正定性，因而$\mathrm{d}^{(k)}$总是下降方向。
2. 每次迭代需要$3n^2 + O(n)$次乘法运算。
3. 方法具有超线性收敛速度。

拟牛顿法

拟牛顿（正割）条件

$$H_{k+1}\mathrm{y}^{(k)} = \mathrm{s}^{(k)}$$

其中H_{k+1}是Hesse矩阵逆的近似；

$$B_{k+1}\mathrm{s}^{(k)} = \mathrm{y}^{(k)}$$

其中B_{k+1}是Hesse矩阵的近似。

拟牛顿法

由对称秩二校正和拟牛顿条件$H_{k+1}\mathrm{y}^{(k)} = \mathrm{s}^{(k)}$可得到$H_k$的DFP校正公式

$$H_{k+1}^{(DFP)} = H_k + \frac{\mathrm{s}^{(k)}\mathrm{s}^{(k)\,T}}{\mathrm{s}^{(k)\,T}\mathrm{y}^{(k)}} - \frac{H_k\mathrm{y}^{(k)}\mathrm{y}^{(k)\,T}H_k}{\mathrm{y}^{(k)\,T}H_k\mathrm{y}^{(k)}}.$$

拟牛顿法

BFGS (Broyden-Fletcher-Goldfarb-Shanno) 校正

类似地，我们可从拟牛顿条件$B_{k+1}\mathrm{s}^{(k)} = \mathrm{y}^{(k)}$得到关于$B_k$的对称秩二校正公式

$$B_{k+1}^{(BFGS)} = B_k + \frac{\mathrm{y}^{(k)}\mathrm{y}^{(k)\,T}}{\mathrm{y}^{(k)\,T}\mathrm{s}^{(k)}} - \frac{B_k\mathrm{s}^{(k)}\mathrm{s}^{(k)\,T}B_k}{\mathrm{s}^{(k)\,T}B_k\mathrm{s}^{(k)}}. \tag{25}$$

把(25)式称为关于B_k的BFGS校正。

拟牛顿法

如果我们对B_k的BFGS校正“求逆”，就可以得到关于H_k的BFGS校正公式

$$\begin{aligned} H_{k+1}^{(BFGS)} = & H_k + (1 + \frac{\mathrm{y}^{(k)^T} H_k \mathrm{y}^{(k)}}{\mathrm{s}^{(k)^T} \mathrm{y}^{(k)}}) \frac{\mathrm{s}^{(k)} \mathrm{s}^{(k)^T}}{\mathrm{s}^{(k)^T} \mathrm{y}^{(k)}} \\ & - \frac{H_k \mathrm{y}^{(k)} \mathrm{s}^{(k)^T} + \mathrm{s}^{(k)} \mathrm{y}^{(k)^T} H_k}{\mathrm{s}^{(k)^T} \mathrm{y}^{(k)}}. \end{aligned} \tag{26}$$

[思考题：请给出$H_{k+1}^{(BFGS)}$的对称秩二校正的特解，即$a, \mathrm{u}, b, \mathrm{v}$.]

拟牛顿法

进一步，若将(26)式中 $\{H \leftrightarrow B, \mathrm{s} \leftrightarrow \mathrm{y}\}$互换，便得到关于$B_k$的DFP校正公式

$$\begin{aligned} B_{k+1}^{(DFP)} = \quad & B_k + (1 + \frac{\mathrm{s}^{(k)^T} B_k \mathrm{s}^{(k)}}{\mathrm{y}^{(k)^T} \mathrm{s}^{(k)}}) \frac{\mathrm{y}^{(k)} \mathrm{y}^{(k)^T}}{\mathrm{y}^{(k)^T} \mathrm{s}^{(k)}} \\ & - \frac{B_k \mathrm{s}^{(k)} \mathrm{y}^{(k)^T} + \mathrm{y}^{(k)} \mathrm{s}^{(k)^T} B_k}{\mathrm{y}^{(k)^T} \mathrm{s}^{(k)}}. \end{aligned} \tag{27}$$

拟牛顿法

秩一校正的求逆公式

Sherman-Morrison定理： 设$A \in \mathbb{R}^{n\times n}$是非奇异阵，$\mathrm{u}, \mathrm{v} \in \mathbb{R}^n$是任意向量。若$1 + \mathrm{v}^T A^{-1}\mathrm{u} \neq 0$, 则$A$的秩一校正$A + \mathrm{u}\mathrm{v}^T$非奇异，且其逆可以表示为

$$(A + \mathrm{u}\mathrm{v}^T)^{-1} = A^{-1} - \frac{A^{-1}\mathrm{u}\mathrm{v}^T A^{-1}}{1 + \mathrm{v}^T A^{-1}\mathrm{u}}. \tag{28}$$

[思考题：利用秩一校正的求逆公式，由$H_{k+1}^{(DFP)}$推导$B_{k+1}^{(DFP)}$]

拟牛顿法

进一步的参考资料

- R. Fletcher, Practical Methods of Optimization (2nd Edition). John Wiley & Sons, 1987.
- D. C. Liu and J. Nocedal, On the Limited Memory Method for Large Scale Optimization. Mathematical Programming B, 45(3), pp. 503-528, 1999.
- ...

共轭梯度法

共轭方向

定义： 设G是$n \times n$正定阵，$\mathbb{R}^n$中的任一组非零向量$\{\mathrm{d}^{(0)}, \mathrm{d}^{(1)}, \cdots, \mathrm{d}^{(k)}\}$, 如果$\mathrm{d}^{(i)^T} G \mathrm{d}^{(j)} = 0 (i \neq j)$,
则称$\mathrm{d}^{(0)}, \mathrm{d}^{(1)}, \cdots, \mathrm{d}^{(k)}$是$G-$共轭的。

显然共轭是正交概念的推广，当取$G = I$时，共轭即为正交。

共轭梯度法

共轭方向法（类）：

(0) 给定正定阵G, 选取初始点$\mathrm{x}^{(0)}$, 计算$\mathrm{g}^{(0)} = \nabla f(\mathrm{x}^{(0)})$ 并构造$\mathrm{d}^{(0)}$使得${\mathrm{g}^{(0)}}^T \mathrm{d}^{(0)} < 0$. 令$k := 0$.

(1) 求精确的一维搜索步长α_k, 即$\alpha_k = \arg\min_{\alpha>0} f(\mathrm{x}^{(k)} + \alpha \mathrm{d}^{(k)})$.

(2) 更新迭代点$\mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} + \alpha_k \mathrm{d}^{(k)}$, 并构造$\mathrm{d}^{(k+1)}$使得${\mathrm{d}^{(k+1)}}^T G \mathrm{d}^{(j)} = 0,\ j = 0, 1, \cdots, k$.

(3) 置$k := k + 1$, 返回第(1)步。

共轭梯度法

共轭方向法是从研究二次函数的极小化问题中产生的，但它可以推广到处理非二次函数的极小化问题。

共轭方向法的一个重要性质是，只要执行精确一维搜索，迭代算法就具有二次终止性。

共轭梯度法

共轭方向法基本定理： 严格凸二次函数$f(\mathrm{x}) = \frac{1}{2}\mathrm{x}^T G\mathrm{x} + \mathrm{c}^T\mathrm{x}$,
共轭方向法执行精确一维搜索，则每步迭代点$\mathrm{x}^{(k+1)}$是$f(\mathrm{x})$在线性流形

$$\mathcal{V} = \{\mathrm{x} \mid \mathrm{x} = \mathrm{x}^{(0)} + \sum_{j=0}^{k} \beta_j \mathrm{d}^{(j)}, \forall \beta_j \in \mathbb{R}\}$$

中的唯一极小点。

共轭梯度法

证明： 设共轭方向法产生的G-共轭方向为$\mathrm{d}^{(0)}, \mathrm{d}^{(1)}, \cdots, \mathrm{d}^{(k)}$.
由共轭方向的定义知，$\{\mathrm{d}^{(0)}, \mathrm{d}^{(1)}, \cdots, \mathrm{d}^{(k)}\}$线性无关。

下面只要证：对所有$k < n$成立

$$\mathrm{g}^{(k+1)^T}\mathrm{d}^{(j)} = 0,\ j = 0, 1, \cdots, k.$$

即在点$\mathrm{x}^{(k+1)}$处的函数梯度$\mathrm{g}^{(k+1)} = \nabla f(\mathrm{x}^{(k+1)})$ 与子空间$\mathrm{span}\{\mathrm{d}^{(0)}, \mathrm{d}^{(1)}, \cdots, \mathrm{d}^{(k)}\}$正交。

由此易得出定理的结论。

共轭梯度法

证明（续）： 直接由精确一维搜索知，对 $\forall j$ 成立

$$\mathrm{g}^{(j+1)^T}\mathrm{d}^{(j)} = 0.$$

特别地，当 $j = k$ 时，$\mathrm{g}^{(k+1)^T}\mathrm{d}^{(k)} = 0.$

共轭梯度法

证明（续）： 事实上，由于

$$\mathrm{y}^{(k)} = \mathrm{g}^{(k+1)} - \mathrm{g}^{(k)} = G(\mathrm{x}^{(k+1)} - \mathrm{x}^{(k)}) = G\mathrm{s}^{(k)} = \alpha_k G\mathrm{d}^{(k)}.$$

故当$j < k$时有

$$\begin{aligned}
\mathrm{g}^{(k+1)\,T}\mathrm{d}^{(j)} &= \mathrm{g}^{(j+1)\,T}\mathrm{d}^{(j)} + \sum_{i=j+1}^{k} \mathrm{y}^{(i)\,T}\mathrm{d}^{(j)} \\
&= \mathrm{g}^{(j+1)\,T}\mathrm{d}^{(j)} + \sum_{i=j+1}^{k} \alpha_i \mathrm{d}^{(i)\,T} G\mathrm{d}^{(j)} \\
&= 0 + 0 \\
&= 0
\end{aligned}$$

共轭梯度法

证明（续）： 综合上述，从而证明了

$$\mathrm{g}^{(k+1)^T}\mathrm{d}^{(j)} = 0,\ j = 0, 1, \cdots, k.$$

共轭梯度法

推论： 对于严格凸的二次函数，若沿着一组共轭方向搜索，经有限步迭代必达到极小点。

共轭梯度法

由于共轭方向法具有二次终止性，人们希望能给出一个具体的算法（属于共轭方向法类）。通过修改最速下降法，使其搜索方向具有共轭性质，这便是共轭梯度法。

下面我们先针对二次函数，给出共轭梯度法的具体描述。

共轭梯度法

设二次函数$f(\mathrm{x}) = \frac{1}{2}\mathrm{x}^T G\mathrm{x} + \mathrm{c}^T\mathrm{x}$, 其中$G$是$n \times n$正定阵，c是$n$维向量。函数$f$的梯度向量为

$$\mathrm{g}(\mathrm{x}) = \nabla f(\mathrm{x}) = G\mathrm{x} + \mathrm{c}.$$

取$\mathrm{d}^{(0)} = -\mathrm{g}^{(0)}$, 因为$\mathrm{x}^{(1)} = \mathrm{x}^{(0)} + \alpha_0\mathrm{d}^{(0)}$中步长$\alpha_0$由精确一维搜索决定，所以${\mathrm{g}^{(1)}}^T\mathrm{d}^{(0)} = 0$.

现设$\mathrm{d}^{(1)} = -\mathrm{g}^{(1)} + \beta_0^{(1)}\mathrm{d}^{(0)}$, 选择$\beta_0^{(1)}$使${\mathrm{d}^{(1)}}^T G\mathrm{d}^{(0)} = 0$, 即得

$$\beta_0^{(1)} = \frac{{\mathrm{g}^{(1)}}^T\mathrm{g}^{(1)}}{{\mathrm{g}^{(0)}}^T\mathrm{g}^{(0)}}.$$

共轭梯度法

同理，令$\mathrm{d}^{(2)} = -\mathrm{g}^{(2)} + \beta_0^{(2)}\mathrm{d}^{(0)} + \beta_1^{(2)}\mathrm{d}^{(1)}$, 选择$\beta_0^{(2)}, \beta_1^{(2)}$ 使得${\mathrm{d}^{(2)}}^T G\mathrm{d}^{(j)} = 0,\ j = 0, 1.$ 从而有

$$\beta_0^{(2)} = 0,$$
$$\beta_1^{(2)} = \frac{{\mathrm{g}^{(2)}}^T \mathrm{g}^{(2)}}{{\mathrm{g}^{(1)}}^T \mathrm{g}^{(1)}}.$$

共轭梯度法

一般地，在第k次迭代中，令

$$\mathrm{d}^{(k)} = -\mathrm{g}^{(k)} + \sum_{j=0}^{k-1} \beta_j^{(k)} \mathrm{d}^{(j)},$$

选择$\beta_j^{(k)}$ 使得${\mathrm{d}^{(k)}}^T G \mathrm{d}^{(j)} = 0,\ j = 0, 1, \cdots, k-1,$ 则有

$$\beta_j^{(k)} = \frac{{\mathrm{g}^{(k)}}^T G \mathrm{d}^{(j)}}{{\mathrm{d}^{(j)}}^T G \mathrm{d}^{(j)}} = \frac{{\mathrm{g}^{(k)}}^T (\mathrm{g}^{(j+1)} - \mathrm{g}^{(j)})}{{\mathrm{d}^{(j)}}^T (\mathrm{g}^{(j+1)} - \mathrm{g}^{(j)})}.$$

又由于${\mathrm{g}^{(k)}}^T \mathrm{g}^{(j)} = 0,\ j = 0, 1, \cdots, k-1,$ 故得

$$\beta_j^{(k)} = 0,\ j = 0, 1, \cdots, k-2$$

$$\beta_{k-1}^{(k)} = \frac{{\mathrm{g}^{(k)}}^T \mathrm{g}^{(k)}}{{\mathrm{g}^{(k-1)}}^T \mathrm{g}^{(k-1)}}.$$

共轭梯度法

针对二次函数的共轭梯度算法 (Fletcher & Reeves, 1964)

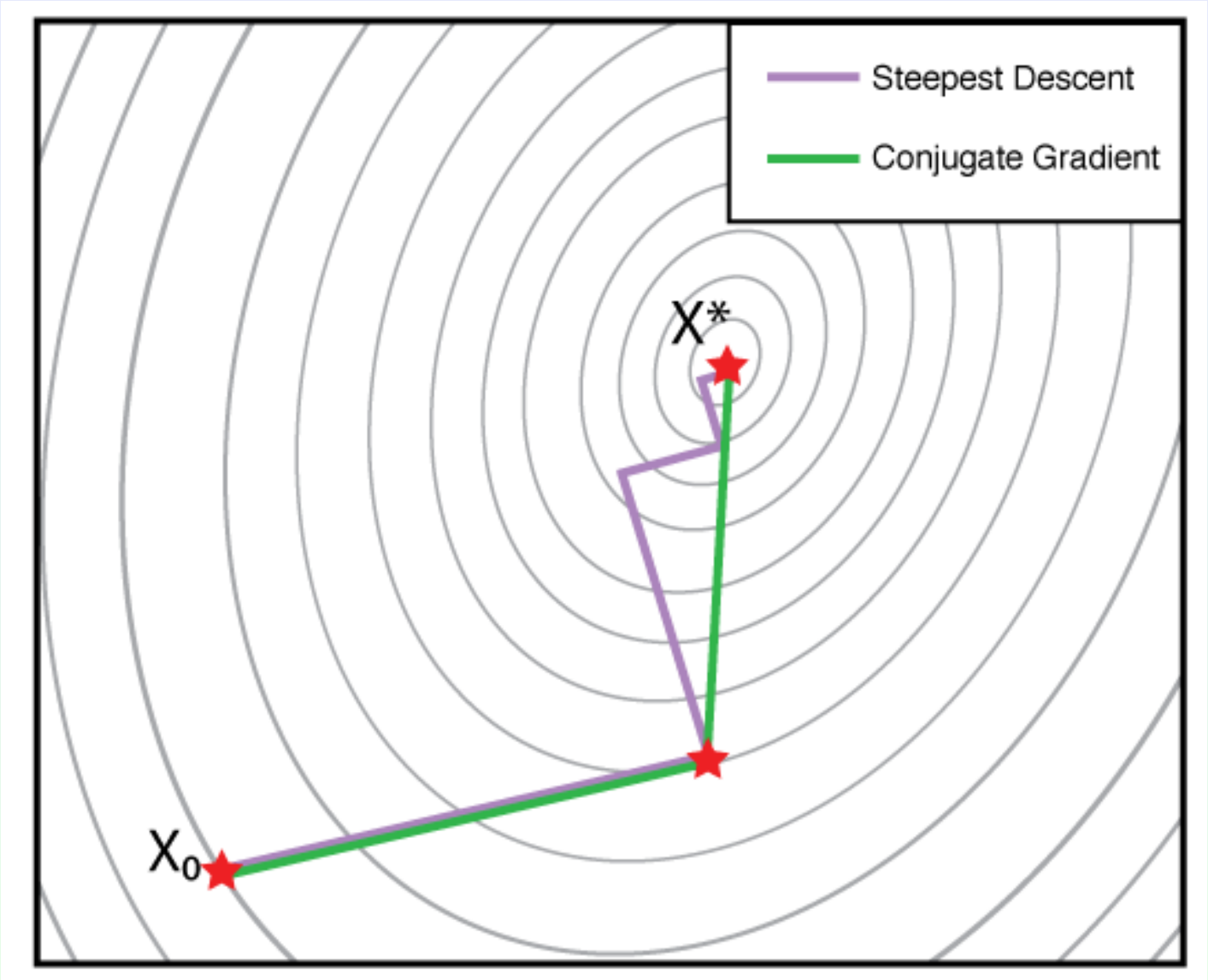
(0) 给定初始点$x^{(0)}$, 计算$g^{(0)} = g(x^{(0)})$, 令$d^{(0)} = -g^{(0)}$, $k := 0$.

(1) 迭代更新$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$, 其中$\alpha_k = \dfrac{{g^{(k)}}^T g^{(k)}}{{d^{(k)}}^T G d^{(k)}}$.

(2) 计算$g^{(k+1)} = g(x^{(k+1)})$, 构造共轭梯度方向$d^{(k+1)} = -g^{(k+1)} + \beta_k d^{(k)}$, 其中$\beta_k = \dfrac{{g^{(k+1)}}^T g^{(k+1)}}{{g^{(k)}}^T g^{(k)}}$.

(3) 置$k := k + 1$, 返回第(1)步。

共轭梯度法



共轭梯度法

共轭梯度法性质定理： 设目标函数$f(\mathrm{x}) = \frac{1}{2}\mathrm{x}^T G\mathrm{x} + \mathrm{c}^T\mathrm{x}$, 则采用精确一维搜索的共轭梯度法经$m \leq n$步迭代后终止，且对所有的$1 \leq k \leq m$成立下列关系式：

$$
\begin{aligned}
&\mathrm{d}^{(k)^T} G\mathrm{d}^{(j)} = 0,\ \mathrm{g}^{(k)^T}\mathrm{g}^{(j)} = 0,\ j = 0, 1, \cdots, k-1 \\
&\mathrm{d}^{(k)^T}\mathrm{g}^{(k)} = -\mathrm{g}^{(k)^T}\mathrm{g}^{(k)} \\
&\mathrm{span}\{\mathrm{g}^{(0)}, \mathrm{g}^{(1)}, \cdots, \mathrm{g}^{(k)}\} = \mathrm{span}\{\mathrm{g}^{(0)}, G\mathrm{g}^{(0)}, \cdots, G^k\mathrm{g}^{(0)}\} \\
&\mathrm{span}\{\mathrm{d}^{(0)}, \mathrm{d}^{(1)}, \cdots, \mathrm{d}^{(k)}\} = \mathrm{span}\{\mathrm{d}^{(0)}, G\mathrm{g}^{(0)}, \cdots, G^k\mathrm{g}^{(0)}\}
\end{aligned}
$$

[思考题：证明上述定理...]

共轭梯度法

将共轭梯度法推广到非二次函数的极小化问题，其迭代为

$$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}.$$

步长α_k由精确或者非精确一维搜索决定，而$d^{(k+1)}$的构造如下：

$$d^{(k+1)} = -g^{(k+1)} + \beta_k d^{(k)}.$$

共轭梯度法

其中

$$\beta_k := \frac{\mathrm{g}^{(k+1)^T}\mathrm{g}^{(k+1)}}{\mathrm{g}^{(k)^T}\mathrm{g}^{(k)}} \quad (\mathrm{Fletcher}-\mathrm{Reeves})$$

$$\beta_k := \frac{\mathrm{g}^{(k+1)^T}(\mathrm{g}^{(k+1)}-\mathrm{g}^{(k)})}{\mathrm{d}^{(k)^T}(\mathrm{g}^{(k+1)}-\mathrm{g}^{(k)})} \quad (\mathrm{Hestenes}-\mathrm{Stiefel})$$

$$\beta_k := \frac{\mathrm{g}^{(k+1)^T}(\mathrm{g}^{(k+1)}-\mathrm{g}^{(k)})}{\mathrm{g}^{(k)^T}\mathrm{g}^{(k)}} \quad (\mathrm{Polak}-\mathrm{Ribiere}-\mathrm{Polyak})$$

$$\beta_k := \frac{\mathrm{g}^{(k+1)^T}\mathrm{g}^{(k+1)}}{-\mathrm{d}^{(k)^T}\mathrm{g}^{(k)}} \quad (\mathrm{Dixon})$$

$$\beta_k := \frac{\mathrm{g}^{(k+1)^T}\mathrm{g}^{(k+1)}}{\mathrm{d}^{(k)^T}(\mathrm{g}^{(k+1)}-\mathrm{g}^{(k)})} \quad (\mathrm{Dai}-\mathrm{Yuan})$$

共轭梯度法

对于非二次函数，共轭梯度法迭代n步以后所产生的搜索方向$\mathrm{d}^{(k+1)} = -\mathrm{g}^{(k+1)} + \beta_k \mathrm{d}^{(k)}$ 可能不再是下降方向（由非精确一维搜索造成的）。

因此，n步以后我们应该周期性采用最速下降方向作为搜索方向，即令$\mathrm{d}^{(\ell n)} = -\mathrm{g}^{(\ell n)},\ \ell = 1, 2, \ldots$

这种策略称为重启动策略，这样的共轭梯度法也称作重启动共轭梯度法。

共轭梯度法

如上所述的共轭梯度法迭代对于一般的非线性函数的最小化也是照样适用的，但迭代更新的步长因子无法显式表达，需要执行数值近似的非精确一维搜索。

共轭梯度法

由于每n步迭代执行重启动策略，若记重新启动时得到的点列为$\{z^{(j)}\}$，则可证明这些相隔n次的迭代点列超线性收敛。受实际计算误差的影响，在很多情形下仅能取得类似线性的收敛速率。

共轭梯度法

从实际计算效率及稳定性来看，共轭梯度法未必比拟牛顿法好。但是，共轭梯度法中搜索方向的计算仅仅用到目标函数的梯度，而不必像拟牛顿法那样在每次迭代中更新Hesse矩阵（或其逆）的近似阵并记忆之。所以，当问题的规模大而且有稀疏结构时，共轭梯度法有高效执行计算的好处。

共轭梯度法

The preconditioned conjugate gradient method

在大多数情况下，为确保共轭梯度法的快速收敛，预条件处理是必要的。

共轭梯度法

进一步的参考资料

- M. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems. Journal of Research of the National Bureau of Standards, 49 (6), 1952.
- K. Atkinson, An Introduction to Numerical Analysis (2nd Edition). John Wiley & Sons, 1988.
- M. Avriel, Nonlinear Programming: Analysis and Methods. Dover Publishing， 2003.
- G. Golub and C. Van Loan, Matrix Computations (3rd Edition). Johns Hopkins University Press.
- ...

信赖域方法

为了保证迭代法的全局收敛性，之前我们采用了一维搜索策略。

一维搜索策略先确定一个搜索方向$\mathrm{d}^{(k)}$，然后沿着这个方向选择适当的步长因子α_k，新的迭代点$\mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} + \alpha_k \mathrm{d}^{(k)}$.

现在，我们讨论另一种全局收敛策略
— 信赖域方法 (Trust-Region Method).

信赖域方法

为了保证迭代法的全局收敛性，之前我们采用了一维搜索策略。

一维搜索策略先确定一个搜索方向$\mathsf{d}^{(k)}$, 然后沿着这个方向选择适当的步长因子α_k, 新的迭代点$\mathsf{x}^{(k+1)} = \mathsf{x}^{(k)} + \alpha_k \mathsf{d}^{(k)}$.

现在，我们讨论另一种全局收敛策略
— 信赖域方法 (Trust-Region Method).

信赖域方法

为了保证迭代法的全局收敛性，之前我们采用了一维搜索策略。

一维搜索策略先确定一个搜索方向$\mathrm{d}^{(k)}$, 然后沿着这个方向选择适当的步长因子α_k, 新的迭代点$\mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} + \alpha_k \mathrm{d}^{(k)}$.

现在，我们讨论另一种全局收敛策略
— 信赖域方法 (Trust-Region Method).

信赖域方法

信赖域方法首先定义当前迭代点$x^{(k)}$的邻域

$$\Omega_k = \{x \in \mathbb{R}^n \mid \|x - x^{(k)}\| \leq e_k\},$$

这里Ω_k称为信赖域，e_k是信赖域半径。

假定在这个邻域里，二次模型$q^{(k)}(s)$是目标函数$f(x)$的一个合适的近似，则在信赖域中极小化二次模型，得到近似极小点$s^{(k)}$，并取$x^{(k+1)} = x^{(k)} + s^{(k)}$.

信赖域方法

信赖域方法利用二次模型在信赖域内求得方向步$s^{(k)}$，使得目标函数的下降比一维搜索更有效。

信赖域方法不仅具有全局收敛性，而且不要求目标函数的Hesse矩阵（或其近似）是正定的。

信赖域方法

信赖域子问题

$$\begin{aligned} \min \quad & q^{(k)}(\mathrm{s}) = f(\mathrm{x}^{(k)}) + \mathrm{g}^{(k)\,T}\mathrm{s} + \frac{1}{2}\mathrm{s}^T B_k \mathrm{s} \\ \text{s.t.} \quad & \|\mathrm{s}\| \le e_k. \end{aligned} \tag{29}$$

其中$\mathrm{s} = \mathrm{x} - \mathrm{x}^{(k)}$, $\mathrm{g}^{(k)} = \nabla f(\mathrm{x}^{(k)})$, 对称阵$B_k$是Hesse矩阵$\nabla^2 f(\mathrm{x}^{(k)})$或其近似，$e_k > 0$为信赖域半径，$\|\cdot\|$为某一范数。

信赖域方法

如何选择信赖域半径e_k?

我们将根据二次模型$q^{(k)}(s)$对目标函数$f(x)$的拟合程度自适应地调整信赖域半径。

信赖域方法

设子问题(29)的解$s^{(k)}$, 令目标函数的下降量

$$\mathrm{Act}_k = f(x^{(k)}) - f(x^{(k)} + s^{(k)})$$

为实际下降量，令二次模型函数的下降量

$$\mathrm{Pre}_k = q^{(k)}(0) - q^{(k)}(s^{(k)})$$

为预测下降量。定义比值

$$r_k = \frac{\mathrm{Act}_k}{\mathrm{Pre}_k} = \frac{f(x^{(k)}) - f(x^{(k)} + s^{(k)})}{q^{(k)}(0) - q^{(k)}(s^{(k)})}.$$

它衡量了二次模型与目标函数之间的一致程度。

信赖域方法

当r_k越接近1，表明二次模型函数$q^{(k)}(s)$与目标函数f的一致性程度越好，此时可以增大半径e_k以扩大信赖域。

如果$r_k > 0$但不接近1，我们保持信赖域半径e_k不变。

如果r_k接近零或取负值，表明$q^{(k)}(s)$与目标函数f的一致性程度不理想，就减小半径e_k以缩小信赖域。

信赖域方法

信赖域算法

(0) 给定初始点$x^{(0)}$, 信赖域半径的上界$\bar{e}$,
$\varepsilon > 0, 0 < \gamma_1 < \gamma_2 < 1, 0 < \eta_1 < 1 < \eta_2$. 取$e_0 \in (0, \bar{e})$, 令$k := 0$.

(1) 如果$\|g^{(k)}\| \leq \varepsilon$, 停止迭代。否则，求解信赖域子问题(29)得到$s^{(k)}$.

(2) 计算比值r_k, 更新迭代点

$$x^{(k+1)} = \begin{cases} x^{(k)} + s^{(k)} & \text{if } r_k > 0, \\ x^{(k)} & \text{otherwise.} \end{cases}$$

信赖域方法

信赖域算法

(3) 调整信赖域半径，令

$$e_{k+1} = \begin{cases} \eta_1 e_k & \text{if } r_k < \gamma_1, \\ e_k & \text{if } \gamma_1 \le r_k < \gamma_2, \\ \min(\eta_2 e_k, \bar{e}) & \text{if } r_k \ge \gamma_2. \end{cases}$$

(4) 置$k := k + 1$, 返回第(1)步。

信赖域方法

信赖域方法的全局收敛性定理：

设水平集$L(\mathrm{x}^{(0)}) = \{\mathrm{x} \mid f(\mathrm{x}) \leq f(\mathrm{x}^{(0)})\}$有界，且$f(\mathrm{x})$在其上$C^2$连续，则由信赖域算法产生的迭代序列存在聚点$\mathrm{x}^{\infty}$，满足一阶和二阶必要条件，即

$$\mathrm{g}^{\infty} = \nabla f(\mathrm{x}^{\infty}) = 0, \quad G_{\infty} = \nabla^2 f(\mathrm{x}^{\infty}) \geq 0.$$

信赖域方法

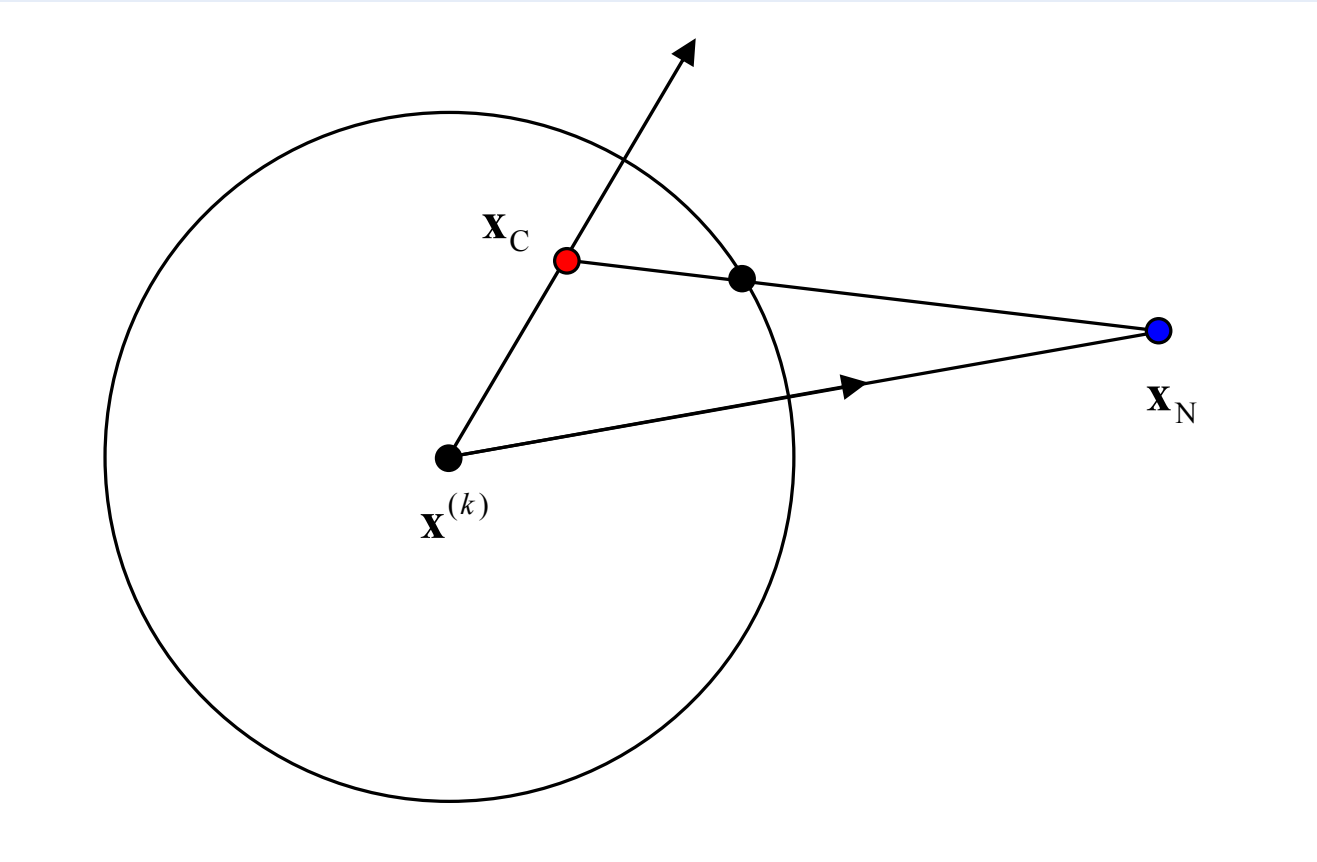
在信赖域算法中关键的一步是，解信赖域子问题(29).

这里我们介绍一种求解信赖域子问题的方法，即由 Powell (1970) 提出的折线法。

所谓折线法，是连接 Cauchy 点（由最速下降法产生的极小点）和牛顿点（由牛顿法产生的极小点），其连线与信赖域边界的交点取为$x^{(k+1)}$.

信赖域方法

折线法图示



信赖域方法

对于二次模型

$$q^{(k)}(-\alpha \mathrm{g}^{(k)}) = f(\mathrm{x}^{(k)}) - \alpha \|\mathrm{g}^{(k)}\|^2 + \frac{1}{2}\alpha^2 {\mathrm{g}^{(k)}}^T B_k \mathrm{g}^{(k)},$$

精确一维搜索的步长因子可表达为

$$\alpha_k = \frac{\|\mathrm{g}^{(k)}\|^2}{{\mathrm{g}^{(k)}}^T B_k \mathrm{g}^{(k)}}.$$

于是 Cauchy 步为

$$\mathrm{s}_C^{(k)} = -\alpha_k \mathrm{g}^{(k)} = -\frac{\|\mathrm{g}^{(k)}\|^2}{{\mathrm{g}^{(k)}}^T B_k \mathrm{g}^{(k)}} \mathrm{g}^{(k)}.$$

信赖域方法

如果$\|s_C^{(k)}\| = \|\alpha_k g^{(k)}\| \geq e_k$, 取

$$s^{(k)} = -\frac{e_k}{\|g^{(k)}\|} g^{(k)},$$

便得

$$x^{(k+1)} = x^{(k)} - \frac{e_k}{\|g^{(k)}\|} g^{(k)}.$$

如果$\|s_C^{(k)}\| = \|\alpha_k g^{(k)}\| < e_k$, 再计算牛顿步

$$s_N^{(k)} = -B_k^{-1} g^{(k)}.$$

信赖域方法

如果$\|s_N^{(k)}\| \leq e_k$, 取

$$s^{(k)} = s_N^{(k)} = -B_k^{-1} g^{(k)},$$

否则，取

$$s^{(k)} = s_C^{(k)} + \lambda(s_N^{(k)} - s_C^{(k)}),$$

其中λ使得

$$\|s_C^{(k)} + \lambda(s_N^{(k)} - s_C^{(k)})\| = e_k.$$

信赖域方法

综上所述，我们得到

$$x^{(k+1)} = \begin{cases} x^{(k)} - \dfrac{e_k}{\|g^{(k)}\|} g^{(k)} & 当 \|s_C^{(k)}\| \geq e_k, \\ x^{(k)} - B_k^{-1} g^{(k)} & 当 \|s_C^{(k)}\| < e_k 且 \|s_N^{(k)}\| \leq e_k, \\ x^{(k)} + s_C^{(k)} + \lambda(s_N^{(k)} - s_C^{(k)}) & 当 \|s_C^{(k)}\| < e_k 且 \|s_N^{(k)}\| > e_k. \end{cases} \tag{30}$$

信赖域方法

折线法满足下列性质：

1) 沿着 Cauchy 点$x_C^{(k+1)}$和牛顿点$x_N^{(k+1)}$的连线，到$x^{(k)}$的距离单调增加；

2) 沿着 Cauchy 点$x_C^{(k+1)}$和牛顿点$x_N^{(k+1)}$的连线，子问题模型函数值单调减少。

[思考题：证明上述性质...]

本章作业（无约束最优化）

- **Exercise 1:** 请写出上述基于Wolfe-Powell准则的非精确一维搜索算法中插值多项式$p^{(1)}(t), p^{(2)}(t)$的具体表达式。
- **Exercise 2:** 请证明基于Goldstein准则的非精确一维搜索算法的全局收敛性。
- **Exercise 3:** 试将非线性方程组求根$F(\mathrm{x}) = 0$的牛顿迭代，用于求解无约束最优化问题 $\min_{\mathrm{x}\in\mathbb{R}^n} f(\mathrm{x})$. 请给出相应的迭代格式，并说明理由。
- **Exercise 4:** 请证明对称秩一校正拟牛顿法具有二次终止性和遗传性。
- **Exercise 5:** 利用秩一校正的求逆公式（Sherman-Morrison定理），由$H_{k+1}^{(DFP)}$推导$B_{k+1}^{(DFP)}$.

本章作业（无约束最优化）

- **Exercise 6:** 请证明共轭梯度法的性质定理。
- **Exercise 7:** 请证明折线法（信赖域方法）子问题模型的函数单调性。
- **Exercise 8:** 在信赖域方法中，请给出一种与调整信赖域半径等效的自适应模式算法。

Outline I

Outline II

二次规划

二次规划(Quadratic Programming)是指，在变量的线性等式和/或不等式限制下求二次函数的极小点问题

$$\begin{aligned} \min \quad & Q(\mathrm{x}) = \frac{1}{2}\mathrm{x}^T G\mathrm{x} + \mathrm{c}^T\mathrm{x} \\ \text{s.t.} \quad & \mathrm{a}_i^T\mathrm{x} = b_i, i \in \mathcal{E} = \{1, \cdots, m_e\} \\ & \mathrm{a}_i^T\mathrm{x} \geq b_i, i \in \mathcal{I} = \{m_e + 1, \cdots, m\} \end{aligned} \tag{31}$$

我们假定G为对称阵，$\mathrm{a}_i (i \in \mathcal{E})$是线性无关的。

二次规划

二次规划的约束可能不相容，也可能没有有限的最小值，这时称二次规划问题无解。

如果矩阵G半正定，问题(31)是凸二次规划问题，它的任意局部解也是整体解。

如果矩阵G正定，问题(31)是正定二次规划问题，只要存在解即是唯一的。

如果矩阵G不定，问题(31)是一般的二次规划问题，有可能出现非整体解的局部解。

二次规划

等式约束二次规划问题

$$\begin{aligned} \min \quad & Q(\mathsf{x}) = \frac{1}{2}\mathsf{x}^T G\mathsf{x} + \mathsf{c}^T\mathsf{x} \\ \text{s.t.} \quad & A\mathsf{x} = \mathsf{b} \end{aligned} \tag{32}$$

这里A是$m \times n$矩阵，且不失一般性可设 $\mathrm{rank}(A) = m$.

二次规划

设有一种基分解$\mathrm{x} = \begin{pmatrix} \mathrm{x}_B \\ \mathrm{x}_N \end{pmatrix}$, 其中$\mathrm{x}_B \in \mathbb{R}^m, \mathrm{x}_N \in \mathbb{R}^{n-m}$, 使得其约束矩阵的对应分块$A = (A_B, A_N)$中$A_B$可逆。于是，等式约束条件可写成

$$\mathrm{x}_B = A_B^{-1}(\mathrm{b} - A_N \mathrm{x}_N),$$

并将上式代入目标函数中得到无约束问题

$$\min_{\mathrm{x}_N \in \mathbb{R}^{n-m}} \frac{1}{2} \mathrm{x}_N^T \hat{G}_N \mathrm{x}_N + \hat{\mathrm{c}}_N^T \mathrm{x}_N. \tag{33}$$

二次规划

在上式中

$$\hat{G}_N = G_{NN} - G_{NB}A_B^{-1}A_N - A_N^T A_B^{-T} G_{BN} + A_N^T A_B^{-T} G_{BB} A_B^{-1} A_N,$$

$$\hat{\mathrm{c}}_N = \mathrm{c}_N - A_N^T A_B^{-T} \mathrm{c}_B + G_{NB} A_B^{-1} \mathrm{b} - A_N^T A_B^{-T} G_{BB} A_B^{-1} \mathrm{b},$$

以及对应分块形式

$$G = \begin{pmatrix} G_{BB} & G_{BN} \\ G_{NB} & G_{NN} \end{pmatrix}, \quad \mathrm{c} = \begin{pmatrix} \mathrm{c}_B \\ \mathrm{c}_N \end{pmatrix}.$$

(1) 如果$\hat{G}_N$正定，则无约束问题的解可唯一地给出

$$x_N^* = -\hat{G}_N^{-1}\hat{c}_N,$$

进一步得原问题(32)的解为

$$x^* = \begin{pmatrix} x_B^* \\ x_N^* \end{pmatrix} = \begin{pmatrix} A_B^{-1}b \\ 0 \end{pmatrix} + \begin{pmatrix} A_B^{-1}A_N \\ -I \end{pmatrix} \hat{G}_N^{-1}\hat{c}_N.$$

设x^*对应的Lagrange乘子向量为λ^*, 则有

$$Gx^* + c = A^T\lambda^* \Longrightarrow \lambda^* = A_B^{-T}(G_{BB}x_B^* + G_{BN}x_N^* + c_B).$$

(2) 如果$\hat{G}_N$是半正定的，则在$(I - \hat{G}_N\hat{G}_N^+)\hat{c}_N = 0$时，无约束问题有界，且它的解可表示为

$$x_N^* = -\hat{G}_N^+\hat{c}_N + (I - \hat{G}_N^+\hat{G}_N)\tilde{y},$$

其中$\tilde{y} \in \mathbb{R}^{n-m}$为任意向量，$\hat{G}_N^+$表示$\hat{G}_N$的广义逆矩阵。此时，原问题的解$x^*$和相应最优乘子$\lambda^*$可类似确定。

当$(I - \hat{G}_N\hat{G}_N^+)\hat{c}_N = 0$不成立时，则可推出无约束问题无下界，从而原问题也无下界。

(3) 如果$\hat{G}_N$不定（即存在负的特征根），显然无约束问题无下界，故原问题不存在有限最优解。

二次规划

上述消去法的不足之处是，当A_B接近奇异时，容易导致数值计算的不稳定。

广义消去法

设$Z = \{z_{m+1}, \cdots, z_n\}$解空间$\mathrm{Ker}(A)$的一组基，
$Y = \{y_1, \cdots, y_m\}$是商空间$\mathbb{R}^n/\mathrm{Ker}(A)$的一组基，
则$\forall x \in \mathbb{R}^n$可作如下分解表达

$$x = Yx_Y + Zx_Z.$$

从而有

$$Ax = b \Longrightarrow AYx_Y + AZx_Z = b \Longrightarrow x_Y = (AY)^{-1}b,$$

所以得

$$x = Y(AY)^{-1}b + Zx_Z,$$

其中$x_Z \in \mathbb{R}^{n-m}$是自由变量。

广义消去法

将上式代入目标函数中得无约束问题

$$\min_{x_Z \in \mathbb{R}^{n-m}} \frac{1}{2} x_Z^T (Z^T G Z) x_Z + [Z^T G Y (AY)^{-1} b + Z^T c]^T x_Z. \tag{34}$$

假定$Z^T GZ$正定，则有

$$x_Z^* = -(Z^T G Z)^{-1} Z^T [GY(AY)^{-1} b + c].$$

广义消去法

从而得到原问题的最优解

$$\mathrm{x}^* = Y(AY)^{-1} - Z(Z^T GZ)^{-1} Z^T [GY(AY)^{-1}\mathrm{b} + \mathrm{c}],$$

相应的Lagrange乘子为

$$\lambda^* = (AY)^{-T} Y^T (G\mathrm{x}^* + \mathrm{c}).$$

二次规划

Lagrange方法是基于求解可行域内的(K-T)点，即Lagrange函数的稳定点。

对于等式约束问题(32), 其Lagrange函数的稳定点就是如下线性方程组的解

$$\begin{cases} Gx + c = A^T\lambda, \\ Ax = b. \end{cases}$$

写成矩阵形式得

$$\begin{pmatrix} G & -A^T \\ -A & 0 \end{pmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} = - \begin{pmatrix} c \\ b \end{pmatrix}.$$

二次规划

设矩阵$\begin{pmatrix} G & -A^T \\ -A & 0 \end{pmatrix}$可逆，则存在矩阵$U \in \mathbb{R}^{n\times n}, V \in \mathbb{R}^{m\times m}, W \in \mathbb{R}^{m\times n}$使得

$$\begin{pmatrix} G & -A^T \\ -A & 0 \end{pmatrix}^{-1} = \begin{pmatrix} U & W^T \\ W & V \end{pmatrix}$$

从而可求得问题的唯一解

$$\begin{cases} \mathrm{x}^* = -U\mathrm{c} - W^T\mathrm{b}, \\ \lambda^* = -W\mathrm{c} - V\mathrm{b}. \end{cases}$$

二次规划

上述Lagrange方法中的矩阵非奇异性并不一定要求G^{-1}存在，可用不同的方法给出分块矩阵U, V, W的表达形式，从而导致不同的计算公式。

二次规划

当G可逆，A行满秩，则$(AG^{-1}A^T)^{-1}$存在，不难验证

$$\begin{cases} U = G^{-1} - G^{-1}A^T(AG^{-1}A^T)^{-1}AG^{-1}, \\ V = -(AG^{-1}A^T)^{-1}, \\ W = -(AG^{-1}A^T)^{-1}AG^{-1}. \end{cases}$$

于是我们得到求解公式

$$\begin{cases} \mathrm{x}^* = -G^{-1}\mathrm{c} + G^{-1}A^T(AG^{-1}A^T)^{-1}(AG^{-1}\mathrm{c} + \mathrm{b}), \\ \lambda^* = (AG^{-1}A^T)^{-1}(AG^{-1}\mathrm{c} + \mathrm{b}). \end{cases}$$

二次规划

如果取Y, Z满足$(Y, Z) = \begin{pmatrix} A \\ B \end{pmatrix}^{-1}$, 即$AY = I_{m\times m}, AZ = 0.$

若另有$Z^T GZ$可逆，则知$\begin{pmatrix} G & -A^T \\ -A & 0 \end{pmatrix}$可逆。此时

$$\begin{cases} U = Z(Z^T GZ)^{-1} Z^T, \\ V = -Y^T GP^T Y, \\ W = -Y^T P. \end{cases}$$

其中$P = I - GZ(Z^T GZ)^{-1} Z^T.$

二次规划

基于A^T的QR分解，可给出(Y, Z)的一种特殊取法：

设

$$A^T = Q \begin{pmatrix} R \\ 0 \end{pmatrix} = (Q_1, Q_2) \begin{pmatrix} R \\ 0 \end{pmatrix},$$

即

$$A = (R^T, 0) \begin{pmatrix} Q_1^T \\ Q_2^T \end{pmatrix}.$$

其中Q为$n \times n$正交阵，R为$m \times m$上三角阵。于是
令$Y = Q_1 R^{-T}, Z = Q_2$, 则有

$$AY = R^T Q_1^T Q_1 R^{-T} = I_{m\times m}, \quad AZ = R^T Q_1^T Q_2 = 0_{m\times(n-m)}.$$

二次规划

一般的二次规划

$$\begin{aligned} \min \quad & Q(\mathrm{x}) = \frac{1}{2}\mathrm{x}^T G\mathrm{x} + \mathrm{c}^T\mathrm{x} \\ \text{s.t.} \quad & \mathrm{a}_i^T\mathrm{x} = b_i, i \in \mathcal{E} = \{1, \cdots, m_e\} \\ & \mathrm{a}_i^T\mathrm{x} \geq b_i, i \in \mathcal{I} = \{m_e + 1, \cdots, m\} \end{aligned} \tag{35}$$

直观上，不积极的不等式约束在解的附近不起作用，可去掉不予考虑；而积极的不等式约束，由于它在解处等号成立，故我们可以用等式约束来代替这些积极的不等式约束。

二次规划

积极集基本定理： 设x^*是一般的二次规划问题(35)的局部极小点，则x^*也必是等式约束问题

$$(\mathrm{EQ})\begin{cases}\min & Q(x)=\frac{1}{2}x^T Gx+c^T x\\ \text{s.t.} & a_i^T x=b_i, i\in\mathcal{E}\cup\mathcal{I}(x^*)\end{cases}$$

的局部极小点。反之，如果x^*是一般问题(35)的可行点，同时是(EQ)的K-T点，且相应的Lagrange乘子λ^*满足 $\lambda_i^*\geq 0, i\in\mathcal{I}(x^*)$，则$x^*$必是原问题(35)的K-T点。

[习题6.1：证明上述定理...]

二次规划

设$x^{(k)}$为当前迭代点，且是问题(35)的可行点。

记$\mathcal{E}_k = \mathcal{E} \cup \mathcal{I}(x^{(k)})$, 考虑等式约束问题

$$(\mathrm{EQ1}) \begin{cases} \min & \frac{1}{2} s^T G s + (G x^{(k)} + c)^T s \\ \text{s.t.} & a_i^T s = 0, i \in \mathcal{E}_k \end{cases}$$

求得(EQ1)的解$s^{(k)}$, 及其相应的Lagrange乘子$\lambda_i^{(k)}, i \in \mathcal{E}_k$.

二次规划

(a) $s^{(k)} \neq 0$时，$x^{(k)}$不可能是原问题的K-T点。

(b) $s^{(k)} = 0$时，$x^{(k)}$是问题

$$(\mathrm{EQ2}) \begin{cases} \min & \frac{1}{2} x^T G x + c^T x \\ \text{s.t.} & a_i^T x = b_i, i \in \mathcal{E}_k \end{cases}$$

的K-T点；如果$\lambda_i^{(k)} \geq 0, i \in \mathcal{I}(x^{(k)})$, 则$x^{(k)}$也是原问题的K-T点。

(c) 否则，由$\lambda_{i_q}^{(k)} = \min\limits_{i \in \mathcal{I}(x^{(k)})} \lambda_i^{(k)} < 0$ 确定i_q, 那么如下问题

$$(\mathrm{EQ3}) \begin{cases} \min & \frac{1}{2} s^T G s + (G x^{(k)} + c)^T s \\ \text{s.t.} & a_i^T s = 0, i \in \hat{\mathcal{E}} = \mathcal{E}_k \setminus \{i_q\}. \end{cases}$$

的解$\hat{s}$是原问题在当前点$x^{(k)}$处的可行方向，即$a_{i_q}^T \hat{s} \geq 0$.

[思考题：证明上述(c)的结论...]

积极集方法(Active Set Method)

(0) 给出可行点$x^{(0)}$, 令$\mathcal{E}_0 = \mathcal{E} \cup \mathcal{I}(x^{(0)})$, $k := 0$.

(1) 求解等式约束问题(EQ1)得$s^{(k)}$, 若$s^{(k)} \neq 0$, 转第(3)步。

(2) 如果$\lambda_i^{(k)} \geq 0, i \in \mathcal{I}(x^{(k)})$, 则停止; 否则由$\lambda_{i_q}^{(k)} = \min\limits_{i \in \mathcal{I}(x^{(k)})} \lambda_i^{(k)} < 0$ 确定i_q并令$\mathcal{E}_k := \mathcal{E}_k \setminus \{i_q\}$, $x^{(k+1)} = x^{(k)}$, 转第(4)步。

(3) 由$\alpha_k = \min\{1, \min\limits_{i \notin \mathcal{E}_k, a_i^T s^{(k)} < 0} \frac{b_i - a_i^T x^{(k)}}{a_i^T s^{(k)}}\}$, 计算$x^{(k+1)} = x^{(k)} + \alpha_k s^{(k)}$. 如果$\alpha_k = 1$, 转第(4)步; 不然定可找到$p \notin \mathcal{E}_k$使得$a_p^T(x^{(k)} + \alpha_k s^{(k)}) = b_p$, 并令$\mathcal{E}_k := \mathcal{E}_k \cup \{p\}$.

(4) $\mathcal{E}_{k+1} := \mathcal{E}_k, k := k + 1$, 返回第(1)步。

本章作业（二次规划）

- **Exercise 1:** 请证明积极集基本定理。
- **Exercise 2:** 请证明前述(c)的结论。
- **Exercise 3 （选做题）：** 试证明存在半正定的能量阵$\mathrm{E}_{T,m}$满足如下模型解

$$\frac{1}{n}\mathrm{y}^T\mathrm{E}_{T,m}\mathrm{y} = \min_{\substack{\phi \in W_2^m(\Omega) \\ \phi(X_i)=y_i,\, i=1,\cdots,n}} |\phi|^2_{T,m}$$

其中$\mathrm{y} = (y_1, \cdots, y_n)^T$, $T = \{X_i\}_{i=1}^n$,

$$|f|^2_{T,m} = \frac{1}{n}\sum_{i=1}^{n}\sum_{|\alpha|=m}\frac{m!}{\alpha!}|D^\alpha f(X_i)|^2.$$

非线性约束最优化

先考虑等式约束问题

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & c(x) = 0, \end{aligned} \tag{36}$$

其中$c(x) = (c_1(x), \cdots, c_m(x))^T$.

非线性约束最优化

记$A(x) = [\nabla c(x)]^T = (\nabla c_1(x), \cdots, \nabla c_m(x))^T$.

由最优性条件知：x是等式约束问题(36)的K-T点当且仅当存在乘子$\lambda \in \mathbb{R}^m$使得

$$\nabla f(x) - A(x)^T \lambda = 0,$$

且x是一可行点，即$c(x) = 0$.

非线性约束最优化

于是得到联立方程组

$$\begin{cases} \nabla f(x) - A(x)^T \lambda = 0, \\ -c(x) = 0. \end{cases}$$

我们可用Newton-Raphson迭代法求解上述联立方程组。

非线性约束最优化

记x和λ的计算增量分别为δ_x, δ_λ, Newton-Raphson迭代满足：

$$\begin{pmatrix} W(x,\lambda) & -A(x)^T \\ -A(x) & 0 \end{pmatrix} \begin{pmatrix} \delta_x \\ \delta_\lambda \end{pmatrix} = - \begin{pmatrix} \nabla f(x) - A(x)^T \lambda \\ -c(x) \end{pmatrix}, \tag{37}$$

其中 $W(x,\lambda) = \nabla^2 f(x) - \sum_{i=1}^{m} \lambda_i \nabla^2 c_i(x)$.

非线性约束最优化

上述方法称为Lagrange-Newton法，最早由Wilson(1963)提出的。

其实质上是用Newton-Raphson迭代求问题(36)的Lagrange函数 $L(x, \lambda)$的稳定点。

在此，我们定义价值函数

$$\psi(\mathrm{x},\lambda)=\|\nabla f(\mathrm{x})-A(\mathrm{x})^T\lambda\|^2+\|\mathrm{c}(\mathrm{x})\|^2. \tag{38}$$

显然，$\psi(\mathrm{x},\lambda)$是关于Lagrange-Newton法的下降函数，即满足

$$\nabla\psi(\mathrm{x},\lambda)^T\begin{pmatrix}\delta_{\mathrm{x}}\\ \delta_\lambda\end{pmatrix}=-2\psi(\mathrm{x},\lambda)\le 0.$$

Lagrange-Newton法：

(0) 给定$x^{(0)}$, $\lambda \in \mathbb{R}^m, \beta \in (0,1), \varepsilon \geq 0$, 令$k := 0$.

(1) 计算价值函数$\psi(x^{(k)}, \lambda^{(k)})$, 如果$\psi(x^{(k)}, \lambda^{(k)}) \leq \varepsilon$, 则停止；否则在$(x^{(k)}, \lambda^{(k)})$处求解(37)得到 $(\delta_{x^{(k)}}, \delta_{\lambda^{(k)}})$, 并令$\alpha_k = 1$.

(2) 若$\psi(x^{(k)} + \alpha_k \delta_{x^{(k)}}, \lambda^{(k)} + \alpha_k \delta_{\lambda^{(k)}}) \leq (1 - \beta\alpha_k)\psi(x^{(k)}, \lambda^{(k)})$, 转第(3)步；否则令$\alpha_k = \frac{1}{4}\alpha_k$, 返回第(2)步。

(3) 置$x^{(k+1)} = x^{(k)} + \alpha_k \delta_{x^{(k)}}, \lambda^{(k+1)} = \lambda^{(k)} + \alpha_k \delta_{\lambda^{(k)}}, k := k + 1$, 返回第(1)步。

非线性约束最优化

Lagrange-Newton法的收敛性结果

定理： 设Lagrange-Newton法产生的迭代点列$\{(x^{(k)}, \lambda^{(k)})\}$有界，如果$f(x)$和$c_i(x)$都是二次连续可微，且逆矩阵

$$\begin{pmatrix} W(x,\lambda) & -A(x)^T \\ -A(x) & 0 \end{pmatrix}^{-1}$$

一致有界，则$\{(x^{(k)}, \lambda^{(k)})\}$的任何聚点都是方程$\psi(x,\lambda)=0$的根，从而$\{x^{(k)}\}$的聚点是问题(36)的K-T点。

注：在一定条件下，还可进一步证明Lagrange-Newton法具有二阶收敛速度。

非线性约束最优化
逐步二次规划法

Lagrange-Newton法的一大重要贡献是，在其基础上发展出了逐步二次规划方法(Sequential Quadratic Programming Methods)。而后者已成为求解一般非线性约束最优化问题的一类十分重要的方法。

我们可将式(37)写成如下形式：

$$\begin{cases} W(x,\lambda)\delta_x + \nabla f(x) = A(x)^T(\lambda + \delta_\lambda), \\ c(x) + A(x)\delta_x = 0. \end{cases}$$

由最优性条件知，$\delta_{x^{(k)}}$即为下列二次规划问题

$$\begin{aligned} \min \quad & \frac{1}{2}d^T W(x^{(k)}, \lambda^{(k)})d + \nabla f(x^{(k)})^T d \\ \text{s.t.} \quad & c(x^{(k)}) + A(x^{(k)})d = 0 \end{aligned} \tag{39}$$

的K-T点。

非线性约束最优化
逐步二次规划法

Lagrange-Newton法可以理解为逐步求解上述等式约束二次规划的方法。

设$\mathrm{d}^{(k)}$是二次规划问题(39)的最优解，那么可迭代更新

$$\mathrm{x}^{(k+1)} = \mathrm{x}^{(k)} + \alpha_k \mathrm{d}^{(k)},$$

其中α_k为第k次迭代的步长。

设$\bar{\lambda}^{(k)}$是(39)对应的Lagrange乘子向量，那么对$k \geq 1$有

$$\lambda^{(k+1)} = \lambda^{(k)} + \alpha_k(\bar{\lambda}^{(k)} - \lambda^{(k)}).$$

非线性约束最优化
逐步二次规划法

现考虑一般的非线性约束最优化问题

$$\begin{aligned} \min \quad & f(\mathrm{x}) \\ \text{s.t.} \quad & c_i(\mathrm{x}) = 0, i \in \mathcal{E} = \{1, \cdots, m_e\}, \\ & c_i(\mathrm{x}) \geq 0, i \in \mathcal{I} = \{m_e + 1, \cdots, m\}. \end{aligned} \tag{40}$$

类似地，在第k次迭代里求解子问题

$$\begin{aligned} \min \quad & \frac{1}{2}\mathrm{d}^T W_k \mathrm{d} + \mathrm{g}^{(k)^T}\mathrm{d} \\ \text{s.t.} \quad & c_i(\mathrm{x}^{(k)}) + \mathrm{a}_i(\mathrm{x}^{(k)})^T\mathrm{d} = 0, i \in \mathcal{E}, \\ & c_i(\mathrm{x}^{(k)}) + \mathrm{a}_i(\mathrm{x}^{(k)})^T\mathrm{d} \geq 0, i \in \mathcal{I}. \end{aligned} \tag{41}$$

非线性约束最优化
逐步二次规划法

在这里，W_k是原问题Lagrange函数的Hesse阵或其近似，
$\mathrm{g}^{(k)} = \nabla f(\mathrm{x}^{(k)})$, $A(\mathrm{x}^{(k)}) = (\mathrm{a}_1(\mathrm{x}^{(k)}), \cdots, \mathrm{a}_m(\mathrm{x}^{(k)}))^T = [\nabla \mathrm{c}(\mathrm{x}^{(k)})]^T$.

记子问题(41)的解为$\mathrm{d}^{(k)}$, 相应Lagrange乘子向量为$\bar{\lambda}^{(k)}$, 故有

$$\begin{cases} W_k \mathrm{d}^{(k)} + \mathrm{g}^{(k)} = A(\mathrm{x}^{(k)})^T \bar{\lambda}^{(k)}, \\ \bar{\lambda}_i^{(k)} \geq 0, i \in \mathcal{I}, \\ \mathrm{c}(\mathrm{x}^{(k)}) + A(\mathrm{x}^{(k)}) \mathrm{d}^{(k)} = 0. \end{cases}$$

非线性约束最优化
逐步二次规划法

逐步二次规划法的迭代以$\mathrm{d}^{(k)}$作为搜索方向。该搜索方向有很好的性质。它是许多罚函数的下降方向，例如L_1罚函数

$$P(x,\sigma)=f(\mathrm{x})+\sigma\left(\sum_{i=1}^{m_e}|c_i(\mathrm{x})|+\sum_{i=m_e+1}^{m}|c_i(\mathrm{x})_-|\right).$$

其中$\mathrm{c}(\mathrm{x})_-$定义如下：

$$\begin{cases} c_i(\mathrm{x})_- = c_i(\mathrm{x}), & i\in\mathcal{E},\\ c_i(\mathrm{x})_- = \min\{0, c_i(\mathrm{x})\}, & i\in\mathcal{I}. \end{cases}$$

非线性约束最优化
逐步二次规划法

下面的算法是Han(1977)提出的逐步二次规划方法：

(0) 给定$x^{(0)}$, $W_0 \in \mathbb{R}^{n\times n}$, $\sigma > 0, \rho \in (0,1), \varepsilon \geq 0$, 令$k := 0$.

(1) 求解子问题(41)给出$d^{(k)}$, 如果$\|d^{(k)}\| \leq \varepsilon$, 则停止；否则求$\alpha_k \in [0, \rho]$使得

$$P(x^{(k)} + \alpha_k d^{(k)}, \sigma) \leq \min_{0\leq\alpha\leq\rho} P(x^{(k)} + \alpha d^{(k)}, \sigma) + \epsilon_k.$$

(2) 置$x^{(k+1)} = x^{(k)} + \alpha_k \delta_{x^{(k)}}$, 计算$W_{k+1}$, 令$k := k + 1$, 返回第(1)步。

非线性约束最优化
逐步二次规划法

可证明前述逐步二次规划法的收敛性结果如下：

定理：假定$f(\mathrm{x})$和$c_i(\mathrm{x})$连续可微，且存在常数$M_1, M_2 > 0$使得

$$M_1\|\mathrm{d}\|^2 \leq \mathrm{d}^T W_k \mathrm{d} \leq M_2\|\mathrm{d}\|^2, \forall k \in \mathbb{N}, \forall \mathrm{d} \in \mathbb{R}^n,$$

如果$\|\lambda^{(k)}\|_\infty \leq \sigma$均成立，则Han(1977)算法产生的点列$\{\mathrm{x}^{(k)}\}$ 的任何聚点都是问题(40)的K-T点。

非线性约束最优化
罚函数法

对于非线性约束最优化问题

$$\begin{aligned} \min \quad & f(\mathrm{x}) \\ \text{s.t.} \quad & c_i(\mathrm{x}) = 0, i \in \mathcal{E} = \{1, \cdots, m_e\} \\ & c_i(\mathrm{x}) \geq 0, i \in \mathcal{I} = \{m_e + 1, \cdots, m\} \end{aligned} \tag{42}$$

的罚函数，是指利用目标函数$f(\mathrm{x})$和约束方程$\mathrm{c}(\mathrm{x})$所构造的具有“罚性质”的函数

$$P(\mathrm{x}) = P(f(\mathrm{x}), \mathrm{c}(\mathrm{x})).$$

所谓“罚性质”，即要求对问题的可行点$x \in S$均有$P(x) = f(x)$, 而当约束条件破坏时有$P(x) > f(x)$.

为了描述约束条件被破坏的程度，我们定义$c(x)_-$如下：

$$\begin{cases} c_i(x)_- = c_i(x), & i \in \mathcal{E}, \\ c_i(x)_- = \min\{0, c_i(x)\}, & i \in \mathcal{I}. \end{cases}$$

罚函数一般可取为目标函数与“罚项”之和，即

$$P(x) = f(x) + \phi(c(x)_-).$$

罚项$\phi(c(x)_-)$是定义在$\mathbb{R}^m$上的函数，它满足

$$\phi(0) = 0, \qquad \lim_{\|c\|\to\infty} \phi(c) = +\infty.$$

非线性约束最优化
罚函数法

如Courant罚函数：

$$P_\sigma(x) = f(x) + \sigma \|c(x)_-\|_2^2,$$

其中$\sigma > 0$是罚因子。

非线性约束最优化
罚函数法

考虑简单罚函数

$$P_\sigma(x) = f(x) + \sigma\|c(x)_-\|^2.$$

记$x(\sigma)$是无约束问题 $\min\limits_{x\in\mathbb{R}^n} P_\sigma(x)$的最优解，我们有如下引理。

引理： 若$x(\sigma)$同时是非线性约束最优化问题(42)的可行点，则$x(\sigma)$也是原问题的最优解。

上述引理表明，只要选取充分大的罚因子$\sigma > 0$, 则通过求解无约束最优化问题应可找到相应约束最优化问题的最优解。

然而在实际计算中，确定大小合适的σ往往比较困难，故通常是选取一个单调增的罚因子序列$\{\sigma_k\}$.

通过求解一系列无约束问题来获得约束最优化问题的解，这称为序贯无约束极小化技术(SUMT)。

至此，我们可以给出罚函数法的迭代步骤：

(0) 任选初始点$x^{(0)}$, 给定初始罚因子$\sigma_0 > 0$及$\beta > 1, \varepsilon > 0$. 令$k := 0$.

(1) 以$x^{(k)}$作为初始迭代点求解无约束问题的极小点，即

$$x(\sigma_k) = \arg\min_{x\in\mathbb{R}^n} P_{\sigma_k}(x).$$

(2) 若$\|c(x(\sigma_k))_-\| < \varepsilon$, 则停止迭代并取$x(\sigma_k)$为原约束问题的近似最优解；否则，置$x^{(k+1)} = x(\sigma_k)$, $\sigma_{k+1} = \beta\sigma_k$, 令$k := k + 1$返回第(1)步。

易得如下三个引理

引理1： 设$\sigma_{k+1} > \sigma_k > 0$, 则有 $P_{\sigma_k}(\mathrm{x}(\sigma_k)) \leq P_{\sigma_{k+1}}(\mathrm{x}(\sigma_{k+1}))$,

$$\|\mathrm{c}(\mathrm{x}(\sigma_k))_-\| \geq \|\mathrm{c}(\mathrm{x}(\sigma_{k+1}))_-\|, \quad f(\mathrm{x}(\sigma_k)) \leq f(\mathrm{x}(\sigma_{k+1})).$$

引理2： 设令$\bar{\mathrm{x}}$是原问题(42)的最优解，则对任意的 $\sigma_k > 0$ 成立

$$f(\bar{\mathrm{x}}) \geq P_{\sigma_k}(\mathrm{x}(\sigma_k)) \geq f(\mathrm{x}(\sigma_k)).$$

引理3： 令$\delta = \|\mathrm{c}(\mathrm{x}(\sigma))_-\|$, 则$\mathrm{x}(\sigma)$也是约束问题

$$\begin{array}{ll} \min & f(\mathrm{x}) \\ \text{s.t.} & \|\mathrm{c}(\mathrm{x})_-\| \leq \delta \end{array}$$

的最优解。

[思考题：证明上述引理1...]

[思考题：证明上述引理3...]

这里只给出引理2的证明：

由题设易得

$$P_{\sigma_k}(x(\sigma_k)) \geq f(x(\sigma_k)).$$

因为$\bar{x}$是原问题的最优解，自然为可行点，于是 $\sigma_k \|c(\bar{x})_-\|^2 = 0.$

又因为$x(\sigma_k) = \arg\min_{x\in\mathbb{R}^n} P_{\sigma_k}(x)$, 则有

$$f(\bar{x}) = P_{\sigma_k}(\bar{x}) \geq P_{\sigma_k}(x(\sigma_k)).$$

关于罚函数法的收敛性，我们有如下结果

定理1： 设罚函数法中的ε满足

$$\varepsilon > \min_{x \in \mathbb{R}^n} \|c(x)_-\|,$$

则算法必有限终止。

[思考题：证明上述定理...]

该定理表明，如果原约束问题存在可行点，则对任意给定的$\varepsilon > 0$, 算法都将有限终止于问题

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & \|c(x)_-\| \leq \delta \end{aligned}$$

的解，且$\delta \leq \varepsilon$.

定理2： 如果算法不有限终止，则必有 $\min\limits_{x\in\mathbb{R}^n} \|c(x)_-\| \geq \varepsilon$,
且 $\lim\limits_{k\to\infty} \|c(x(\sigma_k))_-\| = \min\limits_{x\in\mathbb{R}^n} \|c(x)_-\|$. 此时，$\{x(\sigma_k)\}$的任何聚点$x^*$都是问题

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & \|c(x)_-\| = \min_{y\in\mathbb{R}^n} \|c(y)_-\| \end{aligned}$$

的解。

非线性约束最优化
乘子罚函数

为了叙述简单，仅考虑等式约束问题

$$\begin{aligned} \min \quad & f(\mathrm{x}) \\ \text{s.t.} \quad & \mathrm{c}(\mathrm{x}) = 0 \end{aligned} \tag{43}$$

其中$\mathrm{c}(\mathrm{x}) = (c_1(\mathrm{x}), \cdots, c_{m_e}(\mathrm{x}))^T$.

设x^*是上述问题的最优解且λ^*是相应的Lagrange乘子，由Kuhn-Tucher定理知，x^*必是Lagrange函数

$$L(\mathrm{x}, \lambda^*) = f(\mathrm{x}) - (\lambda^*)^T \mathrm{c}(\mathrm{x})$$

的稳定点。但一般而言，x^*并不是Lagrange函数的极小点。

非线性约束最优化
乘子罚函数

我们考虑乘子罚函数（也称增广Lagrange函数）

$$P(\mathrm{x}, \lambda, \sigma) = L(\mathrm{x}, \lambda) + \frac{\sigma}{2}\|\mathrm{c}(\mathrm{x})\|_2^2.$$

由于增广Lagrange函数的性态，只要取足够大的罚因子σ而不必趋向无穷大，就可通过极小化$P(\mathrm{x}, \lambda, \sigma)$求得原问题的最优解。

非线性约束最优化
乘子罚函数

我们事先并不知道最优乘子向量λ^*, 因此用乘子λ代替，得到增广Lagrange罚函数：

$$P(\mathrm{x}, \lambda, \sigma) = f(\mathrm{x}) - \lambda^T \mathrm{c}(\mathrm{x}) + \frac{\sigma}{2}\|\mathrm{c}(\mathrm{x})\|_2^2.$$

一般的策略是，先给定充分大的σ和乘子向量的初始估计λ, 然后在迭代过程中修正乘子λ力图使之趋向最优乘子λ^*.

如何修正？

非线性约束最优化
乘子罚函数

基于增广Lagrange函数的迭代算法：

(0) 给定初始点$x^{(0)}$和乘子向量初始估计$\lambda^{(0)}$, 给定罚因子$\sigma_0 > 0$, 常数$0 < \alpha < 1, \beta > 1$及容许误差$\varepsilon > 0$. 令$k := 0$.

(1) 以$x^{(k)}$为初点求解无约束问题的极小点，即

$$x^{(k+1)} = \arg\min_{x \in \mathbb{R}^n} P(x, \lambda^{(k)}, \sigma).$$

(2) 若$\|c(x^{(k+1)})\| < \varepsilon$, 则停止迭代并取$x^{(k+1)}$作为原问题的近似最优解；否则，更新乘子向量

$$\lambda^{(k+1)} = \lambda^{(k)} - \sigma c(x^{(k+1)}).$$

(3) 如果$\frac{\|c(x^{(k+1)})\|}{\|c(x^{(k)})\|} \geq \alpha$, 则置$\sigma := \beta\sigma$. 令$k := k + 1$返回第(1)步。

记$A(x) = [\nabla c(x)]^T$, 由于$c(x^*) = 0$, 我们易得

$$\nabla_x P(x^*, \lambda^*, \sigma) = \nabla_x L(x^*, \lambda^*) = 0,$$

$$\nabla^2_{xx} P(x^*, \lambda^*, \sigma) = \nabla^2_{xx} L(x^*, \lambda^*) + \sigma A(x^*) A(x^*)^T.$$

非线性约束最优化
乘子罚函数

设在x^*处满足二阶充分条件，即对$\forall d$使得$A(x^*)^T d = 0$的非零向量，均有

$$d^T \nabla^2_{xx} L(x^*, \lambda^*) d > 0.$$

因此，在二阶充分条件的假定下，对于充分大的σ, 可证$\nabla^2_{xx} P(x^*, \lambda^*, \sigma)$是正定阵。

非线性约束最优化
乘子罚函数

定理： 设x^*和λ^*满足等式约束问题(43)局部最优解的二阶充分条件，则存在σ_0使得当$\sigma > \sigma_0$时，x^*是函数$P(x, \lambda^*, \sigma)$的严格局部极小点。

[思考题：证明上述定理...]

证明： 由题设知（满足局部最优解的二阶充分条件），x^*必为问题(43)的(K-T) 点，所以

$$\nabla_x P(x^*, \lambda^*, \sigma) = \nabla_x L(x^*, \lambda^*) = 0.$$

下面证明，在x^*处的Hessian矩阵$\nabla^2_{xx} P(x^*, \lambda^*, \sigma)$是正定的。

证明：

$$\begin{aligned}\nabla^2_{xx}P(x^*,\lambda^*,\sigma) &= \nabla^2_{xx}L(x^*,\lambda^*)+\sigma A(x^*)A(x^*)^T \\ &= \bar{Q}+\sigma\bar{A}\bar{A}^T\end{aligned}$$

其中$\bar{Q}=\nabla^2_{xx}L(x^*,\lambda^*)$, $\bar{A}=A(x^*)$.

非线性约束最优化
乘子罚函数

定理： 若$\bar{x}$是等式约束问题(43)的可行解，且对于某个$\bar{\lambda}$, $\bar{x}$满足$P(x, \bar{\lambda}, \sigma)$的极小点二阶充分条件，则$\bar{x}$是问题(43)的严格局部最优解。

[思考题：证明上述定理...]

非线性约束最优化
障碍函数

基本思想 在迭代中总是从内点出发，并通过引入障碍函数使之保持在可行域内部进行搜索。因此，这种方法适用于不等式约束的非线性最优化问题。

$$\begin{array}{ll} \min & f(\mathrm{x}) \\ \text{s.t.} & g_i(\mathrm{x}) \geq 0, i = 1, \cdots, m. \end{array} \tag{44}$$

现将可行域内部记作 $\mathrm{int}S$, 其中$S = \{\mathrm{x} \mid g_i(\mathrm{x}) \geq 0, i = 1, \cdots, m\}$. 保持迭代点含于可行域内部的方法是定义如下障碍函数：

$$B(\mathrm{x}, \theta) = f(\mathrm{x}) + \theta\psi(\mathrm{x})$$

其中障碍因子θ是很小的正数，$\psi(\mathrm{x})$是连续函数，当x趋于可行域边界时，$\psi(\mathrm{x}) \to +\infty$.

非线性约束最优化
障碍函数

两种重要的障碍形式是：

$$\psi(\mathrm{x}) = \sum_{i=1}^{m} \frac{1}{g_i(\mathrm{x})} \quad \text{and} \quad \psi(\mathrm{x}) = -\sum_{i=1}^{m} \log g_i(\mathrm{x})$$

这样，当x趋向可行域边界时，函数$B(\mathrm{x}, \theta) \to +\infty$. 否则，由于$\theta$很小，函数$B(\mathrm{x}, \theta)$的取值近似于$f(\mathrm{x})$.

非线性约束最优化
障碍函数

因此，我们可通过求解下列问题得到原问题(44)的近似解：

$$\begin{array}{ll} \min & B(x,\theta) \\ \text{s.t.} & x \in \text{int}S \end{array} \tag{45}$$

由于$\psi(x)$的存在，在可行域边界形成一道“围墙”，因此上述障碍问题(45)的解$\bar{x}(\theta)$必含于可行域的内部。

需要解释的是，障碍问题(45)表面上看起来仍是带约束的最优化问题，且它的约束条件比原来的约束还要复杂。但是，由于函数$\psi(x)$的障碍阻挡作用是自动实现的，因此从计算观点看，求解(45) 完全可当作无约束问题来处理。

因此，我们可通过求解下列问题得到原问题(44)的近似解：

$$\begin{aligned} \min \quad & B(\mathrm{x}, \theta) \\ \text{s.t.} \quad & \mathrm{x} \in \mathrm{int} S \end{aligned} \tag{45}$$

由于$\psi(\mathrm{x})$的存在，在可行域边界形成一道“围墙”，因此上述障碍问题(45)的解$\bar{\mathrm{x}}(\theta)$必含于可行域的内部。

需要解释的是，障碍问题(45)表面上看起来仍是带约束的最优化问题，且它的约束条件比原来的约束还要复杂。但是，由于函数$\psi(\mathrm{x})$的障碍阻挡作用是自动实现的，因此从计算观点看，求解(45) 完全可当作无约束问题来处理。

非线性约束最优化
障碍函数

于是，我们可以给出障碍函数法的计算步骤如下：

(0) 给定初始点$x^{(0)} \in \mathrm{int}S$, 初始障碍因子$\theta_0 > 0$, $\beta \in (0,1)$, $\varepsilon > 0$. 令$k := 0$.

(1) 以$x^{(k)}$作为初始迭代点求解下列问题：

$$\begin{aligned} \min \quad & f(x) + \theta_k \psi(x) \\ \text{s.t.} \quad & x \in \mathrm{int}S \end{aligned}$$

记求得的极小点为$x(\theta_k)$.

(2) 若$\theta_k \psi(x(\theta_k)) < \varepsilon$, 则停止计算并取$x(\theta_k)$为原问题的近似最优解；否则，置$x^{(k+1)} = x(\theta_k)$, $\theta_{k+1} = \beta\theta_k$, 令$k := k+1$返回第(1)步。

非线性约束最优化
障碍函数

定理： 设$\theta_k > \theta_{k+1} > 0$, 记$\mathrm{x}(\theta) = \arg\min_{\mathrm{x}} B(\mathrm{x}, \theta)$, 则有

$$B(\mathrm{x}(\theta_k), \theta_k) \geq B(\mathrm{x}(\theta_{k+1}), \theta_{k+1}),$$

$$\psi(\mathrm{x}(\theta_k)) \leq \psi(\mathrm{x}(\theta_{k+1})),$$

$$f(\mathrm{x}(\theta_k)) \geq f(\mathrm{x}(\theta_{k+1})).$$

[思考题：证明上述定理...]

内点法

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & c_E(x) = 0 \\ & c_I(x) \geq 0 \end{aligned} \tag{46}$$

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & c_E(x) = 0 \\ & c_I(x) - s = 0 \\ & s \geq 0 \end{aligned} \tag{47}$$

内点法

The Karush-Kuhn-Tucker (KKT) conditions for the nonlinear program (47) can be written as

$$\begin{aligned} \nabla f(\mathrm{x}) - A_E(\mathrm{x})^T \mathrm{y} - A_I(\mathrm{x})^T \mathrm{z} &= 0 \\ S\mathrm{z} - \mu 1 &= 0 \\ \mathrm{c}_E(\mathrm{x}) &= 0 \\ \mathrm{c}_I(\mathrm{x}) - \mathrm{s} &= 0 \end{aligned} \tag{48}$$

with $\mu = 0$, together with $\mathrm{s} \geq 0, \mathrm{z} \geq 0$.

Here $A_E(\mathrm{x})$ and $A_I(\mathrm{x})$ are the Jacobian matrices of the functions $\mathrm{c}_E(\mathrm{x})$ and $\mathrm{c}_I(\mathrm{x})$, respectively, and y and z are their Lagrange multipliers. We define S and Z to be the diagonal matrices whose diagonal entries are given by the vectors s and z, respectively, and let $1 = (1, \cdots, 1)^T$.

内点法

Applying Newton's method to the KKT system (48), in the variables $\mathrm{x}, \mathrm{s}, \mathrm{y}, \mathrm{z}$, we obtain

$$\begin{pmatrix} \nabla^2_{xx} L & 0 & -A_E(\mathrm{x})^T & -A_I(\mathrm{x})^T \\ 0 & Z & 0 & S \\ -A_E(\mathrm{x}) & 0 & 0 & 0 \\ -A_I(\mathrm{x}) & I & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathrm{p}_x \\ \mathrm{p}_s \\ \mathrm{p}_y \\ \mathrm{p}_z \end{pmatrix} = - \begin{pmatrix} \nabla f(\mathrm{x}) - A_E(\mathrm{x})^T \mathrm{y} - A_I(\mathrm{x})^T \mathrm{z} \\ S\mathrm{z} - \mu 1 \\ -\mathrm{c}_E(\mathrm{x}) \\ -\mathrm{c}_I(\mathrm{x}) + \mathrm{s} \end{pmatrix} \tag{49}$$

where $L(\mathrm{x}, \mathrm{s}, \mathrm{y}, \mathrm{z})$ denotes the Lagrange function

$$L(\mathrm{x}, \mathrm{s}, \mathrm{y}, \mathrm{z}) = f(\mathrm{x}) - \mathrm{y}^T \mathrm{c}_E(\mathrm{x}) - \mathrm{z}^T (\mathrm{c}_I(\mathrm{x}) - \mathrm{s}).$$

内点法

The system (49) is called the primal-dual system. After the step $\mathrm{p} = (\mathrm{p}_x, \mathrm{p}_s, \mathrm{p}_y, \mathrm{p}_z)$ has been determined, we compute the new iterate $(\mathrm{x}^+, \mathrm{s}^+, \mathrm{y}^+, \mathrm{z}^+)$ as

$$\begin{aligned} \mathrm{x}^+ &= \mathrm{x} + \alpha_s^{\max} \mathrm{p}_x, \quad \mathrm{s}^+ = \mathrm{s} + \alpha_s^{\max} \mathrm{p}_s, \\ \mathrm{y}^+ &= \mathrm{y} + \alpha_z^{\max} \mathrm{p}_y, \quad \mathrm{z}^+ = \mathrm{z} + \alpha_z^{\max} \mathrm{p}_z, \end{aligned}$$

where

$$\begin{aligned} \alpha_s^{\max} &= \max\{\alpha \in (0,1] : \mathrm{s} + \alpha \mathrm{p}_s \geq (1-\tau)\mathrm{s}\}, \\ \alpha_z^{\max} &= \max\{\alpha \in (0,1] : \mathrm{z} + \alpha \mathrm{p}_z \geq (1-\tau)\mathrm{z}\}, \end{aligned} \tag{50}$$

with $\tau \in (0,1)$ (A typical value of τ is 0.995). The condition (50), called the fraction to the boundary rule, prevents the variables s and z from approaching their lower bounds of 0 too quickly.

本章作业（非线性约束最优化）

- **Exercise 4:** 证明(38)中定义的$\psi(x, \lambda)$是关于Lagrange-Newton法的下降函数。
- **Exercise 5:** 证明罚函数法求解带误差界近似问题的算法有限终止性。
- **Exercise 6:** 给出约束最优化问题的二阶充分最优性条件，并用于说明增广Lagrange函数的极小点与原问题最优解的等价性。

Outline I

Outline II

About convex optimization

Convex optimization is a subfield of mathematical optimization that studies the problem of minimizing convex functions over convex sets. Whereas many classes of convex optimization problems admit polynomial-time algorithms, mathematical optimization is in general NP-hard.

We introduce the main definitions and results of convex optimization needed for the analysis of algorithms presented in the section.

Affine set

定义 (affine set)

A set $C \subseteq \mathbb{R}^n$ is *affine* if $\forall x_1, x_2 \in C$ and $\theta \in \mathbb{R}$, we have

$$\theta x_1 + (1-\theta)x_2 \in C$$

i.e., if it contains the line through any two distinct points in it.

It can be generalized to more than two points: If C is an affine set, $x_1, \ldots, x_k \in C$ and $\theta_1 + \ldots + \theta_k = 1$, then $\theta_1 x_1 + \ldots + \theta_k x_k \in C$.

We refer to a point of the form $\theta_1 x_1 + \ldots + \theta_k x_k$ where $\theta_1 + \ldots + \theta_k = 1$, as an *affine combination* of the points $x_1, \ldots, x_k$.

Affine set

If C is an affine set and $x_0 \in C$, then the set

$$V = C - x_0 = \{x - x_0 | \, x \in C\}$$

is a (linear) subspace. We can express C as

$$C = V + x_0 = \{v + x_0 | \, v \in V\}.$$

The *dimesion* of an affine set C is the dimesion of the subspace $V = C - x_0$.

例 (Solution set of linear equations)

For $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, the set $C = \{x | \, Ax = b\}$ is affine. Let $V = \{v | \, Av = 0\}$ be a subspace and $Ax_0 = b$, then $C = V + x_0$.

Affine set

定义 (affine hull)

The set of all affine combinations of points in some set $C \subseteq \mathbb{R}^n$ is called the *affine hull* of C, denoted $\mathbf{aff}C$:

$$\mathbf{aff}C = \{\theta_1 x_1 + \ldots + \theta_k x_k | \, x_1, \ldots, x_k \in C, \theta_1 + \ldots + \theta_k = 1\}.$$

The affine hull is the smallest affine set that contains C.

Convex set

定义 (convex set)

A set C is *convex* if $\forall x_1, x_2 \in C$ and $0 \leqslant \theta \leqslant 1$, we have

$$\theta x_1 + (1-\theta)x_2 \in C$$

i.e., if it contains the line segment between any two points in it.

Generalization to more than two points: for any $k \geqslant 1$, $x_1, \ldots, x_k \in C$ and $\theta_1 + \ldots + \theta_k = 1$ where $\theta_i \geqslant 0$, $i = 1, \ldots, k$, we have

$$\theta_1 x_1 + \ldots + \theta_k x_k \in C.$$

The form $\theta_1 x_1 + \ldots + \theta_k x_k$ is called the *convex combination* of the points $x_1, \ldots, x_k$, where $\theta_1, \ldots, \theta_k \geqslant 0$ and $\sum_{i=1}^k \theta_i = 1$.

Convex hull

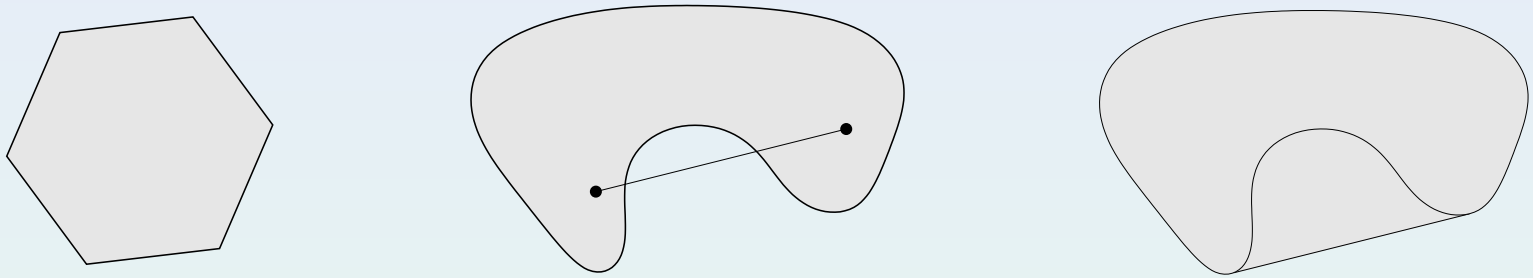
定义 (convex hull)

The *convex hull* of a set C, denoted $\mathbf{conv}\,C$, is the set of all convex combinations of points in C:

$$\mathbf{conv}\,C = \{\theta_1 x_1 + \ldots + \theta_k x_k \,|\, x_i \in C, \theta_i \geqslant 0, i = 1, \ldots, k, \theta_1 + \ldots + \theta_k = 1\}.$$

The convex hull is the smallest convex set that contains C.

Convex set and convex hull



(a) (b) (c)

Figure: **(a)** A convex set (polyhydron). **(b)** A non-convex set. **(c)** The convex hull of (b).

Cone

定义 (cone)

A set C is called a *cone*, if $\forall x \in C$ and $\theta \geqslant 0$ we have θx in C.
A set C is a convex cone if it's convex and a cone, i.e., $\forall x_1, x_2 \in C$ and $\theta_1, \theta_2 \geqslant 0$, we have

$$\theta_1 x_1 + \theta_2 x_2 \in C.$$

A point of the form $\theta_1 x_1 + \ldots + \theta_k x_k$ with $\theta_1, \ldots, \theta_k \geqslant 0$ is called a *conic combination* of $x_1, \ldots, x_k$.

定义 (conic hull)

The conic hull of a set C is the set of all conic combinations of points in C, i.e.,

$$\{\theta_1 x_1 + \ldots + \theta_k x_k \,|\, x_i \in C, \theta_i \geqslant 0, i = 1, \ldots, k\}.$$

Conic hull

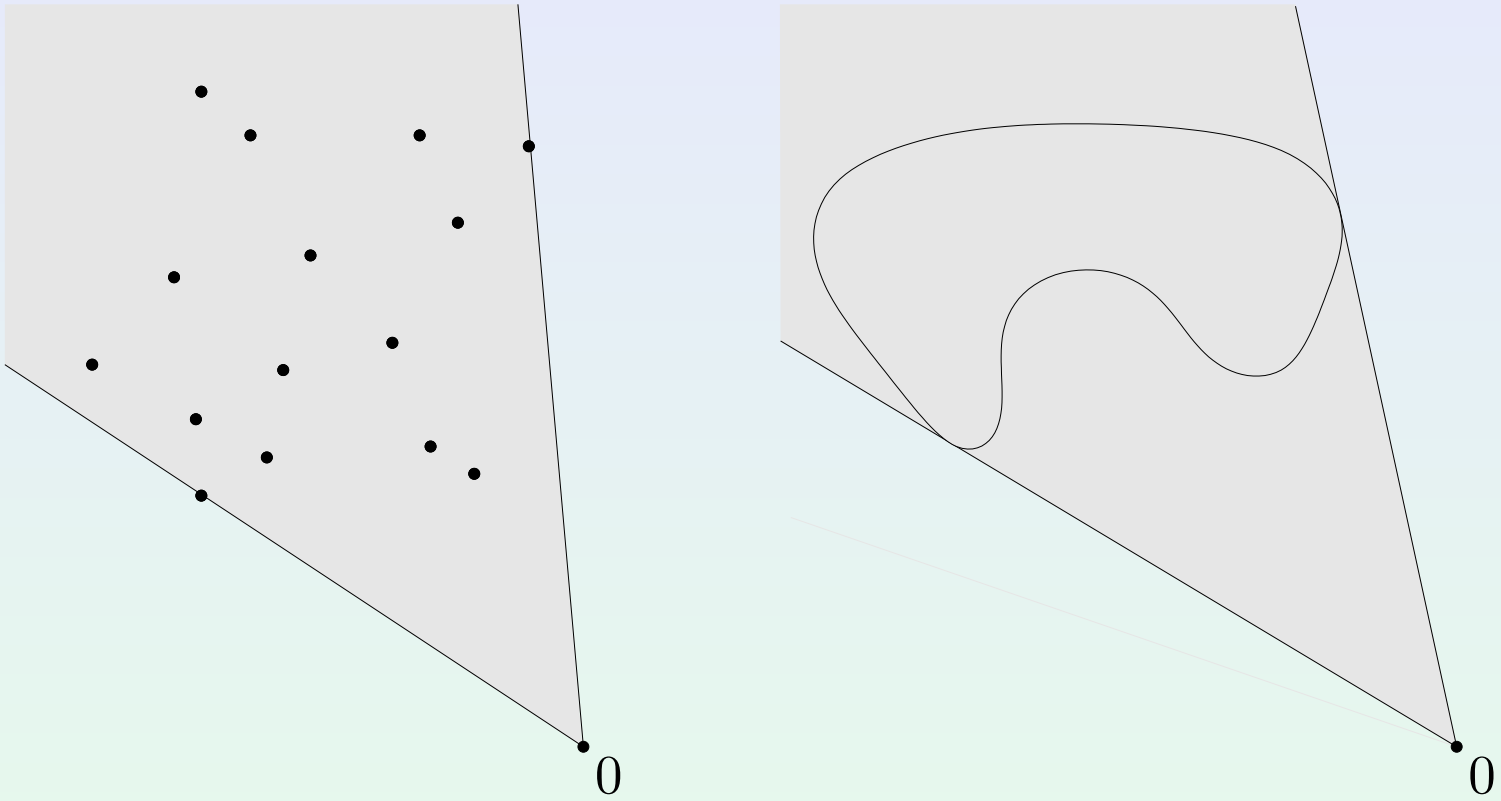


Figure: **Left.** The shaded set is the conic hull of a set of fifteen points (not including the origin). **Right.** The shaded set is the conic hull of the non-convex kidney-shaped set that is surrounded by a curve.

Some important convex examples

- *Hyperplane:* A hyperplane is a set of the form

$$\{x | \, a^\top x = b\}.$$

  It's also affine.

- *Halfspace:* A (closed) halfspace is a set of the form

$$\{x | \, a^\top x \leqslant b\}.$$

  A hyperplane divides $\mathbb{R}^n$ into two halfspaces.

Some important convex examples

- *Polyhedra:* A polyhedron is defined as the solution set of a finite number of linear equalities and inequalities:

$$\mathcal{P} = \{x | \, a_j^\top x \leq b_j, j = 1, \ldots, m, c_k^T x = d_k, k = 1, \ldots, p\}.$$

- *Ball:* A (Euclidean) ball in $\mathbb{R}^n$ has the form

$$B(x_c, r) = \{x | \, \|x - x_c\|_2 \leqslant r\}$$

where $r > 0$ and $\|u\|_2 = (u^\top u)^{1/2}$ denotes the Euclidean norm.

Some important convex examples

- *Norm balls* and *norm cones:*
  Suppose $\|\cdot\|$ is any norm on $\mathbb{R}^n$, a norm ball of radius r and center x_c is given by

$$\{x|\,\|x - x_c\| \leqslant r\}.$$

  The norm cone associated with the norm $\|\cdot\|$ is the set

$$C = \{(x,t)|\,\|x\| \leqslant t\} \subseteq \mathbb{R}^{n+1}.$$

  It's a convex cone.

Some important convex examples

- *The positive semidefinite cone:*
  The set of symmetric $n \times n$ matrices $\mathbf{S}^n$:

$$\mathbf{S}^n = \{X \in \mathbb{R}^{n \times n} \,|\, X = X^\top\},$$

  the set of symmetric positive semidefinite matrices $\mathbf{S}^n_+$:

$$\mathbf{S}^n_+ = \{X \in \mathbf{S}^n \,|\, X \succeq 0\},$$

  and the set of symmetric positive definite matrices $\mathbf{S}^n_{++}$:

$$\mathbf{S}^n_{++} = \{X \in \mathbf{S}^n \,|\, X \succ 0\}$$

  are all convex.

Proper cones and generalized inequalities

A cone $K \subseteq \mathbb{R}^n$ is called a *proper cone* if it satisfies the following:

- K is convex.
- K is closed.
- K is solid, which means it has nonempty interior.
- K is pointed, which means that it contains no line, i.e.,

$$x \in K \text{ and } -x \in K \;\Rightarrow\; x = 0.$$

A proper cone K can be used to define a *generalized inequality*:

$$x \preceq_K y \iff y - x \in K,$$

which is a partial ordering on $\mathbb{R}^n$. Similarly, we define an associated strict partial ordering by

$$x \prec_K y \iff y - x \in \mathbf{int} K$$

Properties of generalized inequalities

- If $x \preceq_K y$ and $u \preceq_K v$, then $x + u \preceq_K y + v$.
- If $x \preceq_K y$ and $y \preceq_K z$ then $x \preceq_K z$.
- If $x \preceq_K y$ and $\alpha \geqslant 0$ then $\alpha x \preceq_K \alpha y$.
- $x \preceq_K x$.
- If $x \preceq_K y$ and $y \preceq_K x$ then $x = y$.
- If $x_i \preceq_K y_i$ for $i = 1, 2, \ldots$, $x_i \to x$ and $y_i \to y$ as $i \to \infty$, then $x \preceq_K y$.

Minimum and minimal elements

- $x \in S$ is the *minimum* element of S (with respect to the generalized inequality $\preceq_K$) if for every $y \in S$ we have $x \preceq_K y$, i.e.,

$$S \subseteq x + K,$$

where $x + K = \{x + z | \, z \in K\}$.

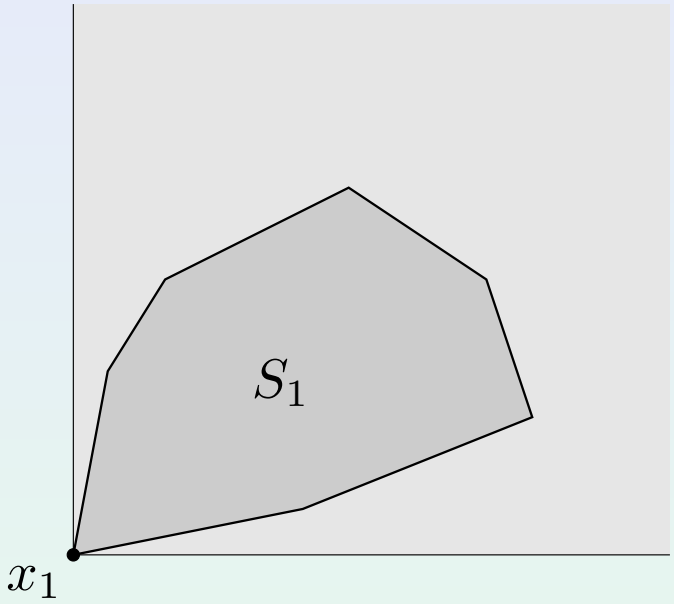
- $x \in S$ is a *minimal* element of S (with respect to the generalized inequality $\preceq_K$) if $y \in S, y \preceq_K x$ only if $y = x$, i.e.,

$$(x - K) \cap S = \{x\},$$

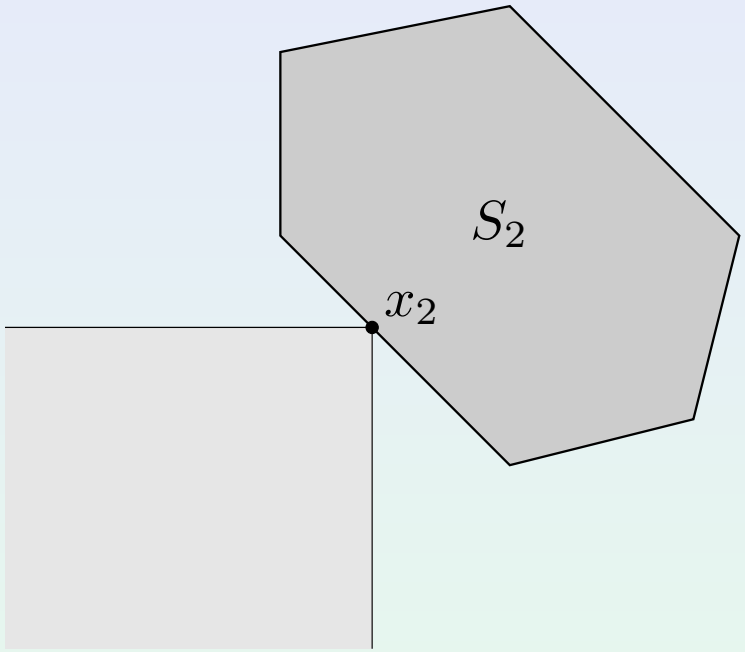
where $x - K = \{x - z | \, z \in K\}$.

- Maximum element and maximal element are defined in a similar way.

Minimum and minimal elements



(a)



(b)

Figure: Let $K = \{(u, v)|u, v \geqslant 0\}$. **(a)** x_1 is the minimum element of S_1. **(b)** x_2 is a minimal element of S_2.

Minimum and minimal elements

If x is the minimum element of S, then x must be a minimal element of S (with respect to the generalized inequality $\preceq_K$).

Brief proof: Suppose $S \subseteq x + K$, and $y \in (x - K) \cap S$, *i.e.*, $\exists z \in K$ such that $y = x - z$. By $y \in S \subseteq x + K$, there exists $w \in K$ such that $y = x + w$. Then we have $w = -z$, which leads to $-w = z \in K$ and $w \in K$. Since K is a proper cone, $w = 0$ and $y = x$.

But the reverse proposition doesn't hold.

Simple example: Let $K = \{(u, v) | \, u, v \geqslant 0\}$ and $L = \{(x, y) | \, x = -y\}$. Then every point of L is a minimal element, but none of them is the minimum element of L.

Definition

定义 (convex function)

A function $f : \mathbb{R}^n \to \mathbb{R}$ is *convex* if $\mathbf{dom} f$ is a convex set and if $\forall x, y \in \mathbf{dom} f$ and θ with $0 \leqslant \theta \leqslant 1$, we have

$$f(\theta x + (1-\theta)y) \leqslant \theta f(x) + (1-\theta) f(y). \tag{51}$$

A function is *strictly convex* if strict inequality holds in (51) whenever $x \neq y$ and $0 < \theta < 1$.

We say f is *concave* if $-f$ is convex, and *strictly concave* if $-f$ is strictly convex.

Definition

Geometrically, Eq.(51) means that the line segment between $(x, f(x))$ and $(y, f(y))$ lies above the graph of f (as shown in Fig.4).

Figure: Graph of a convex function.

First-order conditions

Suppose f is differentiable, i.e., its gradient ∇f exists at each point in $\mathbf{dom} f$.

Function f is convex if and only if $\mathbf{dom} f$ is convex and for $\forall x, y \in \mathbf{dom} f$, the following holds:

$$f(y) \geqslant f(x) + \nabla f(x)^{\top}(y - x).$$

Remark. As a simple result, if $\nabla f(x^*) = 0$, then for all $y \in \mathbf{dom} f$, $f(y) \geqslant f(x^*)$, *i.e.*, x^* is a global minimizer of the function f.

First-order conditions

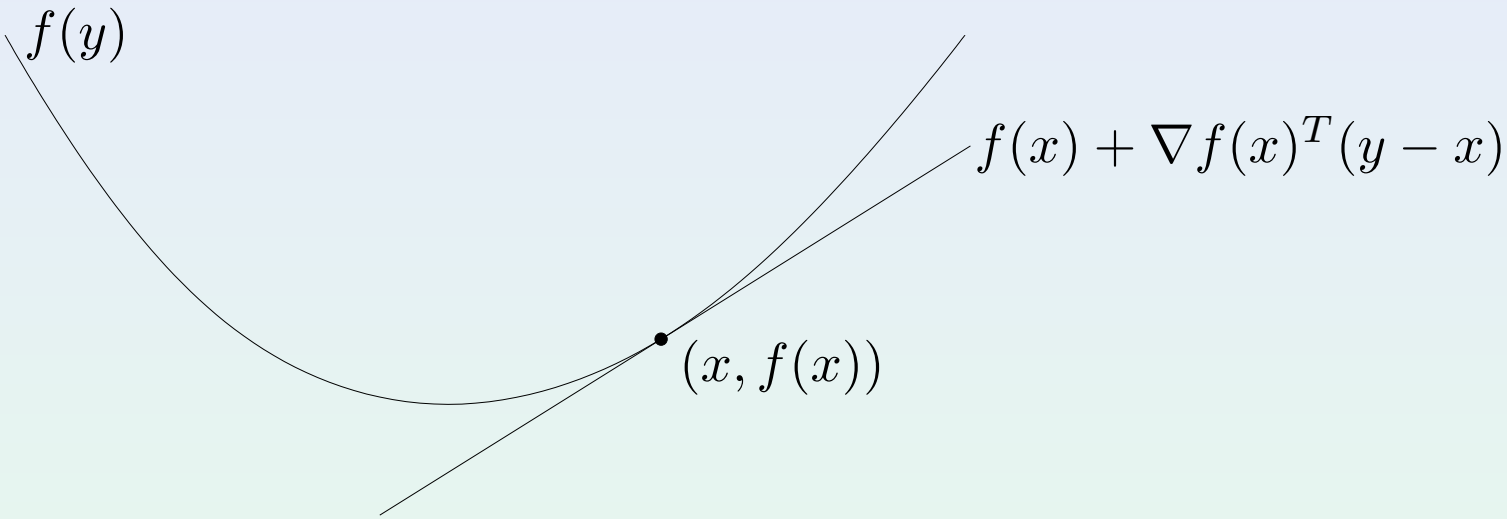


Figure: The tangent to a convex function.

First-order conditions

Function f is strictly convex if and only if $\mathbf{dom} f$ is convex and for $\forall x, y \in \mathbf{dom} f, x \neq y$, we have

$$f(y) > f(x) + \nabla f(x)^\top (y - x).$$

Correspondingly, f is concave if and only if $\mathbf{dom} f$ is convex and for $\forall x, y \in \mathbf{dom} f$, we have

$$f(y) \leqslant f(x) + \nabla f(x)^\top (y - x).$$

Second-order conditions

Assume that f is twice differentiable.

Function f is convex if and only if $\mathbf{dom} f$ is convex and for $\forall x \in \mathbf{dom} f$,

$$\nabla^2 f(x) \succeq 0.$$

Similarly, f is concave if and only if $\mathbf{dom} f$ is convex and $\nabla^2 f(x) \preceq 0$ for $\forall x \in \mathbf{dom} f$.

Second-order conditions

Strict convexity can be partially characterized by second-order conditions.

If $\nabla^2 f(x) \succ 0$ for $\forall x \in \mathbf{dom} f$, then f is strictly convex.

However, the converse is not true. For example, $f : \mathbb{R} \to \mathbb{R}$ given by $f(x) = x^4$ is strictly convex but has zero second derivative at $x = 0$.

Examples

- *Exponential:*
  e^{ax} is convex on $\mathbb{R}$, for any $a \in \mathbb{R}$.
- *Powers:*
  x^a is convex on $\mathbb{R}_{++}$ when $a \geqslant 1$ or $a \leqslant 0$, and concave for $0 \leqslant a \leqslant 1$.
- *Powers of absolute value:*
  $|x|^p$, for $p \geqslant 1$, is convex on $\mathbb{R}$.
- *Logarithm:*
  $\log x$ is concave on $\mathbb{R}_{++}$.
- *Negative entropy:*
  $x \log x$ is convex on $\mathbb{R}_+$, where $0 \log 0$ defined to be 0.

Examples

- *Norms:*
  Every norm on $\mathbb{R}^n$ is convex.

- *Max function:*
  $f(x) = \max\{x_1, \ldots, x_n\}$ is convex on $\mathbb{R}^n$.

- *Log-sum-exp:*
  Then function $f(x) = \log(e^{x_1} + \ldots + e^{x_n})$ is convex on $\mathbb{R}^n$. This function can be interpreted as a differentiable approximation of the max function, since for all x,

$$\max\{x_1, \ldots, x_n\} \leqslant f(x) \leqslant \max\{x_1, \ldots, x_n\} + \log n.$$

- *Geometric mean:*
  $f(x) = (\prod_{i=1}^n x_i)^{1/n}$ is concave on $\mathbf{dom} f = \mathbb{R}^n_{++}$.

- *Log-determinant:*
  $f(X) = \log\det X$ is concave on $\mathbf{dom} f = S^n_{++}$.

Jensen's inequality

The inequality (51), *i.e.*, $f(\theta x + (1-\theta)y) \leqslant \theta f(x) + (1-\theta)f(y)$, is sometimes called *Jensen's inequality.*

It is easily extended to convex combinations of more than two points:

If f is convex, $x_1, \ldots, x_k \in \mathbf{dom} f$, and $\theta_1, \ldots, \theta_k \geqslant 0$ with $\theta_1 + \ldots + \theta_k = 1$, then

$$f(\theta_1 x_1 + \ldots + \theta_k x_k) \leqslant \theta_1 f(x_1) + \ldots + \theta_k f(x_k).$$

Operations that preserve convexity

- *Nonnegative weighted sums:*

  If $f_1, \ldots, f_m$ are convex and $w_1, \ldots, w_m \geqslant 0$, then

  $$f = w_1 f_1 + \ldots + w_m f_m$$

  is convex.

- These properties extend to infinite sums and integrals:

  If $f(x, y)$ is convex in x for each $y \in \mathcal{A}$, and $w(y) \geqslant 0$ for each $y \in \mathcal{A}$, then the function

  $$g(x) = \int_{\mathcal{A}} w(y) f(x, y) dy$$

  is convex in x (provided the integral exists).

Operations that preserve convexity

- *Composition with an affine mapping:*

  Suppose $f : \mathbb{R}^n \to \mathbb{R}$, $A \in \mathbb{R}^{n\times m}$, and $b \in \mathbb{R}^n$. Define $g : \mathbb{R}^m \to \mathbb{R}$ by

  $$g(x) = f(Ax + b),$$

  with $\mathbf{dom} g = \{x | Ax + b \in \mathbf{dom} f\}$. Then if f is convex, so is g; if f is concave, so is g.

Operations that preserve convexity

- *Pointwise maximum:*

  If f_1 and f_2 are convex functions, then

  $$f(x) = \max\{f_1(x), f_2(x)\},$$

  with $\mathbf{dom} f = \mathbf{dom} f_1 \cap \mathbf{dom} f_2$, is also convex.

- Extension to the pointwise supremum:

  If for each $y \in \mathcal{A}$, $f(x, y)$ is convex in x, then

  $$g(x) = \sup_{y \in \mathcal{A}} f(x, y)$$

  is convex in x, where

  $$\mathbf{dom} g = \{x | (x, y) \in \mathbf{dom} f \text{ for all } y \in \mathcal{A}, \sup_{y \in \mathcal{A}} f(x, y) < \infty\}.$$

Functions closed to convex functions

- *Quasi-convex function*: A function $f : \mathbb{R}^n \to \mathbb{R}$ such that its domain and all its sublevel sets

$$S_\alpha = \{x \in \mathbf{dom} f \,|\, f(x) \leqslant \alpha\}, \quad \alpha \in \mathbb{R}$$

  are convex.

- *Log-concave function*: A function $f : \mathbb{R}^n \to \mathbb{R}$ such that $f(x) > 0, \forall x \in \mathbf{dom} f$ and $\log f$ is concave.

Basic terminology

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \leqslant 0, \quad i = 1, \ldots, m \\ & h_j(x) = 0, \quad j = 1, \ldots, p \end{array} \tag{52}$$

$x \in \mathbb{R}^n$	the *optimization variable*
$f_0 : \mathbb{R}^n \to \mathbb{R}$	the *objective function* or *cost function*
$f_i(x) \leqslant 0$	the *inequality constraints*
$f_i : \mathbb{R}^n \to \mathbb{R}$	the *inequality constraint functions*
$h_j(x) = 0$	the *equality constraints*
$h_j : \mathbb{R}^n \to \mathbb{R}$	the *equality constraint functions*

If there are no constraints (*i.e.*, $m = p = 0$) we say the problem is unconstrained.

Basic terminology

- The *domain* of the optimization problem (52) is given as

$$\mathcal{D} = \bigcap_{i=0}^{m} \mathbf{dom} f_i \cap \bigcap_{j=1}^{p} \mathbf{dom} h_j.$$

- A point $x \in \mathcal{D}$ is *feasible* if $f_i(x) \leqslant 0, i = 1, \ldots, m$, and $h_j(x) = 0, j = 1, \ldots, p$.

- The problem (52) is said to be feasible if there exists at least one feasible point, and *infeasible* otherwise.

Basic terminology

The *optimal value* v^* of the problem (52) is defined as

$$v^* = \inf\{f_0(x) | \, f_i(x) \leqslant 0, i = 1, \ldots, m, h_j(x) = 0, j = 1, \ldots, p\}$$

If the problem is infeasible, we have $v^* = \infty$.

- We say x^* is an *optimal point*, or solves the problem (52), if x^* is feasible and $f_0(x^*) = v^*$.
- We say a feasible points $\bar{x}$ is *locally optimal* if there is a constant $\delta > 0$ such that

$$f_0(\bar{x}) = \inf\{f_0(z) | \, f_i(z) \leqslant 0, i = 1, \ldots, m, \\ h_j(z) = 0, j = 1, \ldots, p, \|z - \bar{x}\|_2 \leqslant \delta\}.$$

Convex optimization

A *convex optimization problem* is one of the form

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \leqslant 0, \quad i = 1, \ldots, m \\ & a_j^\top x = b_j, \quad j = 1, \ldots, p \end{array} \tag{53}$$

where $f_0, f_1, \ldots, f_m$ are convex functions.

Any locally optimal point of a convex optimization problem is also globally optimal.

An optimality criterion for differentiable f_0

Suppose that the objective f_0 in a convex optimization problem is differentiable. Let X denote the feasible set, *i.e.*,

$$X = \{x |\, f_i(x) \leqslant 0, i = 1, \ldots, m, h_j(x) = 0, j = 1, \ldots, p\}.$$

Then x is optimal if and only if $x \in X$ and

$$\nabla f_0(x)^\top (y - x) \geqslant 0, \ \forall y \in X. \tag{54}$$

An optimality criterion for differentiable f_0

For an unconstrained problem, the condition (54) reduces to

$$\nabla f_0(x) = 0 \tag{55}$$

for x to be optimal.

An optimality criterion for differentiable f_0

For a convex problem with equality constraints only, *i.e.*,

$$\begin{aligned} \min \ & f_0(x) \\ \text{s.t.} \ & Ax = b \end{aligned}$$

We assume that the feasible set is nonempty. The optimality condition can be expressed as:

$$\nabla f_0(x)^\top u \geqslant 0 \text{ for all } u \in \mathcal{N}(A).$$

In other words,

$$\nabla f_0(x) \perp \mathcal{N}(A).$$

Linear optimization problems

A general *linear program* (LP) has the form

$$\begin{array}{ll} \min & q^\top x + r \\ \text{s.t.} & Gx \leq h \\ & Ax = b \end{array} \tag{56}$$

where $G \in \mathbb{R}^{m \times n}$ and $A \in \mathbb{R}^{p \times n}$. It is common to omit the constant r in the objective function.

Quadratic optimization problems

A convex optimization problem is called *quadratic program* (QP) if it has the form

$$\begin{array}{ll} \min & \frac{1}{2}x^\top Px + q^\top x + r \\ \text{s.t.} & Gx \leq h \\ & Ax = b \end{array} \tag{57}$$

where $P \in \mathbf{S}_+^n$, $G \in \mathbb{R}^{m\times n}$, and $A \in \mathbb{R}^{p\times n}$.

QPs include LPs as a special case by taking $P = 0$.

Quadratic optimization problems

If the objective in (53) as well as the inequality constraint functions are (convex) quadratic, as in

$$\begin{array}{ll} \min & \frac{1}{2}x^\top P_0 x + q_0^\top x + r_0 \\ \text{s.t.} & \frac{1}{2}x^\top P_i x + q_i^\top x + r_i \leqslant 0, \quad i = 1, \ldots, m \\ & Ax = b \end{array} \tag{58}$$

where $P_i \in \mathbf{S}_+^n, i = 0, 1, \ldots, m$, and the problem is called a *quadratically constrained quadratic program* (QCQP).

QCQPs include QPs as a special case by taking $P_i = 0$ for $i = 1, \ldots, m$.

Second-order cone programming

A problem that is closely related to quadratic programming is the *second-order cone program* (SOCP):

$$\begin{array}{ll} \min & f^\top x \\ \text{s.t.} & \|L_i x + g_i\|_2 \leqslant c_i^\top x + d_i, \quad i = 1, \ldots, m \\ & Ax = b \end{array} \tag{59}$$

where $x \in \mathbb{R}^n$ is the optimization variable, $L_i \in \mathbb{R}^{n_i \times n}$, and $A \in \mathbb{R}^{p \times n}$.

When $c_i = 0, i = 1, \ldots, m$, the SOCP is equivalent to a QCQP. However, second-order cone programs are more general than QCQPs (and of course, LPs).

Transform a QCQP into an SOCP

For a QCQP problem (58), let y be an auxiliary variable with constraint:

$$\frac{1}{2}x^\top P_0 x + q_0^\top x + r_0 \leqslant y,$$

then (58) becomes

$$\begin{aligned} \min_{x,y} \quad & y \\ \text{s.t.} \quad & \frac{1}{2}x^\top P_i x + q_i^\top x + r_i \leqslant 0, \quad i = 1, \ldots, m \\ & \frac{1}{2}x^\top P_0 x + q_0^\top x - y + r_0 \leqslant 0 \\ & Ax = b \end{aligned}$$

whose objective is linear. To transform it into an SOCP, we need only translate quadratic constraints into second-order conic ones.

Transform a QCQP into an SOCP

For a quadratic constraint

$$\frac{1}{2}x^\top Px + q^\top x + r \leqslant 0$$

with $P \in \mathbf{S}_+^n$, let $L \in \mathbf{S}_+^n$ be the square root of P, *i.e.*, $LL = P$. Let

$$\tilde{L} = \begin{bmatrix} L \\ q^\top \end{bmatrix}, \quad \tilde{g} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ r + \frac{1}{2} \end{bmatrix} \in \mathbb{R}^{n+1},$$

then the constraint is equivalent to

$$\|\tilde{L}x + \tilde{g}\|_2 \leqslant -(q^\top x + r - \frac{1}{2}).$$

The Lagrangian

Consider an optimization problem in the standard form (52):

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \leqslant 0, \quad i = 1, \ldots, m \\ & h_j(x) = 0, \quad j = 1, \ldots, p. \end{array} \tag{60}$$

We assume its domain $\mathcal{D} = \bigcap_{i=0}^{m} \mathbf{dom} f_i \cap \bigcap_{j=1}^{p} \mathbf{dom} h_j$ is nonempty, and denote the optimal value of (60) by v^*, but do not assume the problem (60) is convex.

The Lagrangian

The basic idea of Lagrangian duality is to take the constraints in (60) into account by augmenting the objective function with a weighted sum of the constraint functions.

The Lagrangian

We define the *Lagrangian* $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ associated with the problem (60) as

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{j=1}^{p} \nu_j h_j(x)$$

with $\mathbf{dom} L = \mathcal{D} \times \mathbb{R}^m \times \mathbb{R}^p$.

- Refer to λ_i as the *Lagrange multiplier* associated with the ith inequality constraint $f_i(x) \leqslant 0$.
- Refer to ν_j as the Lagrange multiplier associated with the jth equality constraint $h_j(x) = 0$.
- The vectors λ and ν are called *Lagrange multiplier vectors* or the *dual variables* associated with the problem (60).

The Lagrange dual function

We define the *Lagrange dual function* (or just *dual function*) $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ as

$$g(\lambda, \nu) = \inf_{x \in \mathcal{D}} L(x, \lambda, \nu) = \inf_{x \in \mathcal{D}} \left(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{j=1}^{p} \nu_j h_j(x) \right).$$

Since the dual function is the pointwise infimum of a family of affine functions of (λ, ν), it is concave, even when the problem (60) is not convex.

Lower bounds on optimal value

Let v^* be the optimal value of the primal problem (60). For any $\lambda \geq 0$ and any ν we have

$$g(\lambda, \nu) \leqslant v^*. \tag{61}$$

Proof.

Suppose $\tilde{x}$ is a feasible point for (60), then we have

$$\sum_{i=1}^{m} \lambda_i f_i(\tilde{x}) + \sum_{j=1}^{p} \nu_j h_j(\tilde{x}) \leqslant 0.$$

Hence

$$g(\lambda, \nu) = \inf_{x \in \mathcal{D}} L(x, \lambda, \nu) \leqslant L(\tilde{x}, \lambda, \nu) \leqslant f_0(\tilde{x}).$$

Since $g(\lambda, \nu) \leqslant f_0(\tilde{x})$ holds for every feasible point $\tilde{x}$, the inequality (61) follows. □

Lower bounds on optimal value

The dual function gives a nontrivial lower bound on v^* only when $\lambda \geq 0$ and $(\lambda, \nu) \in \mathbf{dom} g$, *i.e.*, $g(\lambda, \nu) > -\infty$.

We refer to a pair (λ, ν) with $\lambda \geq 0$ and $(\lambda, \nu) \in \mathbf{dom} g$ as *dual feasible*.

Linear approximation interpretation

Let $I_- : \mathbb{R} \to \mathbb{R} \cup \{\infty\}$ and $I_0 : \mathbb{R} \to \mathbb{R} \cup \{\infty\}$ to be the indicator function for the nonpositive reals and $\{0\}$ respectively:

$$I_-(u) = \begin{cases} 0 & u \leqslant 0 \\ \infty & u > 0 \end{cases}, \quad I_0(u) = \begin{cases} 0 & u = 0 \\ \infty & u \neq 0 \end{cases}.$$

Then the primal problem (60) can be reformulated as an unconstrained problem:

$$\min \quad f_0(x) + \sum_{i=1}^{m} I_-(f_i(x)) + \sum_{j=1}^{p} I_0(h_j(x)). \tag{62}$$

Linear approximation interpretation

We replace the function $I_-(u)$ with the linear function $\lambda_i u$, where $\lambda_i \geqslant 0$, and the function $I_0(u)$ with $\nu_j u$. The objective becomes the Lagrangian function, i.e.,

$$\min \quad L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{j=1}^{p} \nu_j h_j(x).$$

In this formulation, we use a linear or "soft" displeasure function in place of I_- and I_0.

Linear function is an *underestimator* of the indicator function. Since $\lambda_i u \leqslant I_-(u)$ and $\nu_j u \leqslant I_0(u)$ for all u, we see immediately that the dual function yields a lower bound on the optimal value of the primal problem.

The Lagrange dual problem

To attain the *best* lower bound that can be obtained from the Lagrange dual function leads to the optimization problem

$$\begin{array}{ll} \max & g(\lambda, \nu) \\ \text{s.t.} & \lambda \geq 0 \end{array} \tag{63}$$

This problem is called the *Lagrange dual problem* associated with the problem (60). Correspondingly, the problem (60) is called the *primal problem*.

The Lagrange dual problem

The term *dual feasible*, to describe a pair (λ, ν) with $\lambda \geq 0$ and $g(\lambda, \nu) > -\infty$, now makes sense.

We refer to (λ^*, ν^*) as *dual optimal* or *optimal Lagrange multipliers* if they are optimal for the Lagrange dual problem (63).

The Lagrange dual problem (63) is a convex optimization problem no matter the primal problem is convex or not, since the objective to be maximized is concave and the constraint is convex.

Weak duality

For the optimal value of the Lagrange dual problem g^*, we have

$$g^* \leqslant v^*. \tag{64}$$

This property is called *weak duality*.

$v^* - g^*$ is the *optimal duality gap* of the primal problem.

Strong duality and Slater's constraint qualification

If the equality

$$g^* = v^* \tag{65}$$

holds, then we say that *stong duality* holds.

- Strong duality does not, in general, hold.
- For a convex primal problem, there are many additional conditions on the primal problem, under which strong duality holds.

Strong duality and Slater's constraint qualification

One simple condition is *Slater's condition*:

> There exists an $x \in \mathbf{relint}\mathcal{D}$ such that
>
> $$f_i(x) < 0, \quad i = 1, \ldots, m, \qquad Ax = b, \tag{66}$$
>
> where $\mathbf{relint}\mathcal{D} = \{x \in \mathcal{D} | B(x, r) \cap \mathbf{aff}\mathcal{D} \subseteq \mathcal{D} \text{ for some } r > 0\}$. Such a point is called *relative feasible interior point*.

Slater's theorem states that strong duality holds if Slater's condition holds (and the problem is convex).

Optimality conditions

Dual feasible points allow us to bound how suboptimal a given feasible point is, without knowing the exact value of v^*.

If x is primal feasible and (λ, ν) is dual feasible, then

$$f_0(x) - v^* \leqslant f_0(x) - g(\lambda, \nu)$$

and

$$v^* \in [g(\lambda, \nu), f_0(x)], \quad g^* \in [g(\lambda, \nu), f_0(x)].$$

It leads to

$$g(\lambda, \nu) = f_0(x) \Longrightarrow v^* = f_0(x) = g(\lambda, \nu) = g^*.$$

We refer to $f_0(x) - g(\lambda, \nu)$ as the *duality gap* associated with the primal feasible point x and dual feasible point (λ, ν).

Complementary slackness

Suppose that the primal and dual optimal values are attained and equal, let x^* be a primal optimal and (λ^*, ν^*) be a dual optimal points, then

$$\begin{aligned} f_0(x^*) &= g(\lambda^*, \nu^*) \\ &= \inf_x \left(f_0(x) + \sum_{i=1}^m \lambda_i^* f_i(x) + \sum_{j=1}^p \nu_j^* h_j(x) \right) \\ &\leqslant f_0(x^*) + \sum_{i=1}^m \lambda_i^* f_i(x^*) + \sum_{j=1}^p \nu_j^* h_j(x^*) \\ &\leqslant f_0(x^*) \end{aligned}$$

Complementary slackness

By $\lambda_i^* \geqslant 0, f_i(x^*) \leqslant 0, i = 1, \ldots, m$, we have

$$\lambda_i^* f_i(x^*) = 0, \quad i = 1, \ldots, m. \tag{67}$$

This condition is known as *complementary slackness*.

We can express it as

$$\lambda_i^* > 0 \Longrightarrow f_i(x^*) = 0,$$
$$f_i(x^*) < 0 \Longrightarrow \lambda_i^* = 0.$$

KKT optimality conditions

We now assume that the functions $f_0, \ldots, f_m, h_1, \ldots, h_p$ are differentiable. As above, let x^* and (λ^*, ν^*) be any primal and dual optimal points with zero duality gap.

Since x^* minimizes $L(x, \lambda^*, \nu^*)$ over x, it follows

$$\nabla f_0(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla f_i(x^*) + \sum_{j=1}^{p} \nu_j^* \nabla h_j(x^*) = 0.$$

KKT optimality conditions

Together with constraints and complementary slackness, we have

$$\begin{cases} f_i(x^*) \leqslant 0, \quad i = 1, \ldots, m \\ h_j(x^*) = 0, \quad j = 1, \ldots, p \\ \lambda_i^* \geqslant 0, \quad i = 1, \ldots, m \\ \lambda_i^* f_i(x^*) = 0, \quad i = 1, \ldots, m \\ \nabla f_0(x^*) + \sum_{i=1}^m \lambda_i^* \nabla f_i(x^*) + \sum_{j=1}^p \nu_j^* \nabla h_j(x^*) = 0 \end{cases} \tag{68}$$

which are called the *Karush-Kuhn-Tucker* (KKT) conditions.

KKT optimality conditions

For *any* optimization problem with differentiable objective and constraint functions for which strong duality obtains, any pair of primal and dual optimal points must satisfy the KKT conditions.

When the primal problem is convex, the KKT conditions are also sufficient for the points to be primal and dual optimal.

About optimization algorithm

There is *no* analytical formula for the solution of convex optimization problems, not to mention general nonlinear optimization problems.

Thus we describe numerical methods for solving convex optimization problems in the section.

Recall: descent methods

To solve an unconstrained optimization problem

$$\min f(x)$$

where $f(x)$ is differentiable and convex, we usually employ descent methods.

Recall: descent methods

Given a starting point $x^{(0)}$, a descent method produces a sequence $x^{(k)}, k = 1, \ldots,$ where

$$x^{(k+1)} = x^{(k)} + \alpha_k \delta_x^{(k)}, \quad f(x^{(k+1)}) < f(x^{(k)}). \tag{69}$$

We usually drop the superscripts and use the notation $x := x + \alpha\delta_x$ to focus on one iteration of an algorithm. $\alpha > 0$ is called step size and δ_x called search direction. Different methods differ from choices of α or/and δ_x.

Recall: gradient descent and Newton's method

Given a descent direction δ_x, we usually use line search to determine step size α.

Different search directions:

- Negative gradient:
$$\delta_x = -\nabla f(x).$$
- Normalized steepest descent direction (with respect to the norm $\|\cdot\|$):
$$\delta_{x_{\text{nsd}}} = \arg\min\{\nabla f(x)^\top v \mid \|v\| = 1\}.$$
- Newton step:
$$\delta_{x_{\text{nt}}} = -\nabla^2 f(x)^{-1} \nabla f(x).$$

Equality constrained minimization problems

A convex optimization problem with equality constraints has the form

$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & Ax = b, \end{array} \tag{70}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is convex and twice continuously differentiable, and $A \in \mathbb{R}^{p \times n}$ with $\mathbf{rank} A = p < n$. We assume that an optimal solution x^* exists and $v^* = f(x^*)$.

KKT condition

Recall the KKT conditions for (70): a point $x^* \in \mathbf{dom} f$ is optimal if and only if there is a multiplier $\nu^* \in \mathbb{R}^p$ such that

$$Ax^* = b, \quad \nabla f(x^*) + A^\top \nu^* = 0. \tag{71}$$

The first set of equations, $Ax^* = b$, are called the *primal feasibility equations*.

The second set of equations, $\nabla f(x^*) + A^\top \nu^* = 0$, are called the *dual feasibility equations*.

Newton's method with equality constraints

Newton's method with equality constraints is almost the same as Newton's method without constraints, except for two differences:

- The initial point must be feasible (*i.e.*, $x \in \mathbf{dom} f$ and $Ax = b$).
- The definition of Newton step $\delta_{x_{\mathrm{nt}}}$ is modified to take the equality constraints into account.

The Newton step

To derive the Newton step $\delta_{x_{\text{nt}}}$ for problem (70) at the feasible point x, we replace the objective with its second-order Taylor approximation near x

$$\begin{array}{ll} \text{min} & \hat{f}(x+s) = f(x) + \nabla f(x)^\top s + \frac{1}{2} s^\top \nabla^2 f(x) s \\ \text{s.t.} & A(x+s) = b \end{array} \tag{72}$$

with variable s. Suppose $\delta_{x_{\text{nt}}}$ is optimal for (72). By KKT conditions, there exists associated optimal dual variable $w \in \mathbb{R}^p$ such that

$$\begin{bmatrix} \nabla^2 f(x) & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \delta_{x_{\text{nt}}} \\ w \end{bmatrix} = \begin{bmatrix} -\nabla f(x) \\ 0 \end{bmatrix}. \tag{73}$$

The Newton step

We can also derive the Newton Step $\delta_{x_{\text{nt}}}$ by simply replacing x^* and ν^* in the KKT conditions for problem (70):

$$Ax^* = b, \quad \nabla f(x^*) + A^\top \nu^* = 0$$

with $x + \delta_{x_{\text{nt}}}$ and w, respectively, and replace the gradient term in the second equation by its linearized approximation near x, to obtain the equations

$$A(x + \delta_{x_{\text{nt}}}) = b,$$

$$\nabla f(x + \delta_{x_{\text{nt}}}) + A^\top w \approx \nabla f(x) + \nabla^2 f(x)\delta_{x_{\text{nt}}} + A^\top w = 0.$$

The Newton step

Using $Ax = b$, these become

$$A\delta_{x_{\mathrm{nt}}} = 0, \quad \nabla^2 f(x)\delta_{x_{\mathrm{nt}}} + A^\top w = -\nabla f(x),$$

which are precisely the equations (73).

The Newton decrement

The Newton decrement is defined as

$$\kappa(x) = (\delta_{x_{\mathrm{nt}}}^{\top} \nabla^2 f(x) \delta_{x_{\mathrm{nt}}})^{1/2}.$$

Since

$$\frac{\mathrm{d}}{\mathrm{d}\alpha} f(x + \alpha \delta_{x_{\mathrm{nt}}}) \Big|_{\alpha=0} = \nabla f(x)^{\top} \delta_{x_{\mathrm{nt}}} = -\kappa(x)^2,$$

the algorithm should terminate when $\kappa(x)$ is small.

Newton's method with equality constraints

Algorithm. *Newton's method for equality constrained minimization.*

given starting point $x \in \mathbf{dom} f$ with $Ax = b$, tolerance $\epsilon > 0$.

repeat

1. Compute the Newton step $\delta_{x_{\text{nt}}}$ and the decrement $\kappa(x)$.
2. *Stopping criterion.* **quit** if $\kappa^2/2 \leqslant \epsilon$.
3. *Line search* Choose step size α by backtracking line search.
4. *Update.* $x := x + \alpha\delta_{x_{\text{nt}}}$.

Infeasible start Newton method

Newton's method described above is a feasible descent method. Now we describe a generalization of Newton's method that works with initial points and iterates that are *not* feasible.

Newton step at infeasible points

Let x denote the current point, which we do not assume to be feasible, but we do assume satisfies $x \in \mathbf{dom} f$.

Our goal is to find a step δ_x so that $x + \delta_x$ satisfies the optimality conditions (71), *i.e.*, $x + \delta_x \approx x^*$.

Newton step at infeasible points

Similarly, we substitute $x + \delta_x$ for x^* and μ for ν^* in

$$Ax^* = b, \quad \nabla f(x^*) + A^\top \nu^* = 0$$

and use the first-order approximation for the gradient to obtain

$$A(x + \delta_x) = b,$$

$$\nabla f(x + \delta_x) + A^\top \mu \approx \nabla f(x) + \nabla^2 f(x)\delta_x + A^\top \mu = 0.$$

This is a set of linear equations for δ_x and μ,

$$\begin{bmatrix} \nabla^2 f(x) & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \delta_x \\ \mu \end{bmatrix} = - \begin{bmatrix} \nabla f(x) \\ Ax - b \end{bmatrix}. \tag{74}$$

Interpretation as primal-dual Newton step

We express the optimality conditions (71) as $r(x^*, \nu^*) = 0$, where $r : \mathbb{R}^n \times \mathbb{R}^p \mapsto \mathbb{R}^n \times \mathbb{R}^p$ is defined as

$$r(x, \nu) = (r_{\text{dual}}(x, \nu), r_{\text{pri}}(x, \nu)).$$

Here

$$r_{\text{dual}}(x, \nu) = \nabla f(x) + A^\top \nu, \quad r_{\text{pri}}(x, \nu) = Ax - b$$

are the *dual residual* and *primal residual*, respectively.

Interpretation as primal-dual newton step

The first-order Taylor approximation of r, near our current point $y = (x, \nu)$, is

$$r(y + \delta_y) \approx \hat{r}(y + \delta_y) = r(y) + J[r(y)]\delta_y,$$

where $J[r(y)] \in \mathbb{R}^{(n+p)\times(n+p)}$ is the derivative (Jacobian) of r, evaluated at y.

Interpretation as primal-dual Newton step

We define $\delta_{y_{\text{pd}}}$ as the primal-dual Newton step for which $\hat{r}(y + \delta_y) = 0$, *i.e.*,

$$J[r(y)]\delta_{y_{\text{pd}}} = -r(y). \tag{75}$$

Note that $\delta_{y_{\text{pd}}} = (\delta_{x_{\text{pd}}}, \delta_{\nu_{\text{pd}}})$ gives both a primal and a dual step.

Interpretation as primal-dual Newton step

Equations (75) can be expressed as

$$\begin{bmatrix} \nabla^2 f(x) & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \delta_{x_{\text{pd}}} \\ \delta_{\nu_{\text{pd}}} \end{bmatrix} = -\begin{bmatrix} r_{\text{dual}} \\ r_{\text{pri}} \end{bmatrix} = -\begin{bmatrix} \nabla f(x) + A^\top \nu \\ Ax - b \end{bmatrix}. \quad (76)$$

Writing $\nu + \delta_{\nu_{\text{pd}}}$ as μ, we find it coincide with (74)

$$\begin{bmatrix} \nabla^2 f(x) & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \delta_x \\ \mu \end{bmatrix} = -\begin{bmatrix} \nabla f(x) \\ Ax - b \end{bmatrix}.$$

Residual norm reduction property

The Newton direction at an infeasible point is not necessarily a descent direction for f.

The primal-dual interpretation, however, shows that the norm of the residual decreases in the Newton direction. By calculation we have

$$\frac{\mathrm{d}}{\mathrm{d}\alpha}\|r(y+\alpha\delta_{y_{\mathrm{pd}}})\|_2\Big|_{\alpha=0} = -\|r(y)\|_2.$$

This allows us to use $\|r\|_2$ to measure the progress of the infeasible start Newton method.

Infeasible start Newton method

Algorithm. *Infeasible start Newton method.*

given starting point $x \in \mathbf{dom} f$, tolerance $\epsilon > 0$, $\tau \in (0, 1/2), \gamma \in (0, 1)$.

repeat

1. Compute primal and dual Newton steps $\delta_{x_{\mathrm{nt}}}, \delta_{\nu_{\mathrm{nt}}}$.
2. *Backtracking line search on* $\|r\|_2$.
   $\alpha := 1$.
   while $\|r(x + \alpha\delta_{x_{\mathrm{nt}}}, \nu + \alpha\delta_{\nu_{\mathrm{nt}}})\|_2 > (1 - \tau\alpha)\|r(x, \nu)\|_2$, $\alpha := \gamma\alpha$.
3. *Update.* $x := x + \alpha\delta_{x_{\mathrm{nt}}}, \nu := \nu + \alpha\delta_{\nu_{\mathrm{nt}}}$.

until $Ax = b$ and $\|r(x, \nu)\|_2 \leqslant \epsilon$.

Inequality constrained minimization problems

The convex optimization problems that include inequality constraints:

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \leqslant 0, \quad i = 1, \ldots, m \\ & Ax = b \end{array} \tag{77}$$

where $f_0, \ldots, f_m : \mathbb{R}^n \to \mathbb{R}$ are convex and twice continuously differentiable, and $A \in \mathbb{R}^{p \times n}$ with $\mathbf{rank} A = p < n$.

We assume that an optimal x^* exists and denote the optimal value $v^* = f_0(x^*)$.

Assumptions

We also assume that the problem is strictly feasible, *i.e.*, $\exists x \in \mathcal{D}$ satisfying $Ax = b$ and $f_i(x) < 0$ for $i = 1, \ldots, m$.

This means that Slater's constraint qualification holds, and therefore strong duality holds, so there exists dual optimal $\lambda^* \in \mathbb{R}^m, \nu^* \in \mathbb{R}^p$, which together with x^* satisfy the KKT conditions:

$$\begin{aligned} Ax^* = b, \quad f_i(x^*) &\leqslant 0, \quad i = 1, \ldots, m \\ \lambda^* &\geqslant 0 \\ \nabla f_0(x^*) + \textstyle\sum_{i=1}^m \lambda_i^* \nabla f_i(x^*) + A^\top \nu^* &= 0 \\ \lambda_i^* f_i(x^*) &= 0, \quad i = 1, \ldots, m. \end{aligned} \tag{78}$$

About interior-point method

Interior-point methods solve the problem (77) by applying Newton's method to a sequence of equality constrained problems, or to a sequence of modified versions of the KKT conditions.

We will introduce two particular interior-point algorithms:

- *The barrier method*
- *The primal-dual interior-point method*

Logarithmic barrier function

Rewrite the problem (77) and make the inequality constraints implicit in the objective:

$$\begin{aligned} \min \quad & f_0(x) + \sum_{i=1}^{m} I_-(f_i(x)) \\ \text{s.t.} \quad & Ax = b, \end{aligned} \tag{79}$$

where

$$I_-(u) = \begin{cases} 0 & u \leqslant 0 \\ \infty & u > 0. \end{cases}$$

Logarithmic barrier function

The basic idea of the barrier method is to approximate the indicator function I_- by the function

$$\hat{I}_-(u) = -(1/t)\log(-u), \quad \mathbf{dom}\hat{I}_- = -\mathbb{R}_{++}$$

where t is a parameter that sets the accuracy of the approximation.

Obviously, $\hat{I}_-$ is convex, nondecreasing and differentiable.

Logarithmic barrier function

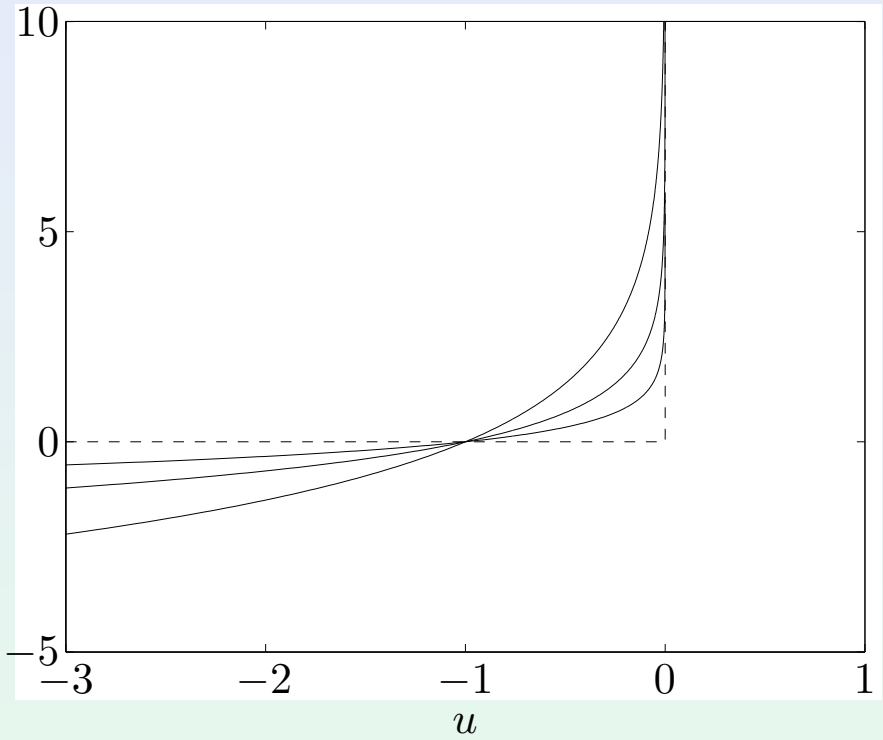


Figure: The dashed lines show the function $I_-(u)$, and the solid curves show $\hat{I}_-(u) = -(1/t)\log(-u)$, for $t = 0.5, 1, 2$. The curve for $t = 2$ gives the best approximation.

Logarithmic barrier function

Substituting $\hat{I}_-$ for I_- in (79) gives the approximation

$$\begin{array}{ll} \min & f_0(x) + \sum_{i=1}^{m} -(1/t)\log(-f_i(x)) \\ \text{s.t.} & Ax = b. \end{array} \tag{80}$$

The function

$$\phi(x) = -\sum_{i=1}^{m} \log(-f_i(x)), \tag{81}$$

is called the *logarithmic barrier* for the problem (77). Its domain is the set of points that satisfy the inequality constraints of (77) strictly:

$$\mathbf{dom}\phi = \{x \in \mathbb{R}^n | f_i(x) < 0, i = 1, \ldots, m\}.$$

Logarithmic barrier function

The gradient and Hessian of ϕ are given by

$$\nabla\phi(x) = \sum_{i=1}^{m} \frac{1}{-f_i(x)} \nabla f_i(x),$$

$$\nabla^2\phi(x) = \sum_{i=1}^{m} \frac{1}{f_i(x)^2} \nabla f_i(x) \nabla f_i(x)^\top + \sum_{i=1}^{m} \frac{1}{-f_i(x)} \nabla^2 f_i(x).$$

Central path

We multiply the objective of (80) by t, and consider the equivalent problem

$$\begin{array}{ll} \min & tf_0(x) + \phi(x) \\ \text{s.t.} & Ax = b. \end{array} \tag{82}$$

We assume problem (82) can be solved via *Newton's method*, and, that it has a unique solution for each $t > 0$.

Central path

For $t > 0$ we define $x^*(t) = \arg\min_x \{tf_0(x) + \phi(x) \text{ s.t. } Ax = b\}$ as the solution of (82).

The *central path* associated with problem (77) is defined as the set of points $\{x^*(t) \mid t > 0\}$, which we call the *central points*.

Central path

Points on the central path are characterized by the following necessary and sufficient conditions: $x^*(t)$ is strictly feasible, *i.e.*, satisfies

$$Ax^*(t) = b, \quad f_i(x^*(t)) < 0, \ i = 1, \ldots, m$$

and $\exists \hat{\nu} \in \mathbb{R}^p$ such that

$$\begin{aligned} 0 &= t\nabla f_0(x^*(t)) + \nabla \phi(x^*(t)) + A^\top \hat{\nu} \\ &= t\nabla f_0(x^*(t)) + \sum_{i=1}^m \frac{1}{-f_i(x^*(t))} \nabla f_i(x^*(t)) + A^\top \hat{\nu} \end{aligned} \tag{83}$$

holds.

Dual points from central path

Every central point yields a dual feasible point.

Define

$$\lambda_i^*(t) = -\frac{1}{tf_i(x^*(t))},\ i = 1, \ldots, m, \quad \nu^*(t) = \frac{\hat{\nu}}{t}. \tag{84}$$

Because $f_i(x^*(t)) < 0, i = 1, \ldots, m$, it's clear that $\lambda^*(t) > 0$.

Dual points from central path

By expressing (83) as

$$\nabla f_0(x^*(t)) + \sum_{i=1}^{m} \lambda_i^*(t) \nabla f_i(x^*(t)) + A^\top \nu^*(t) = 0,$$

we see that $x^*(t)$ minimizes the Lagrangian

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \nu^\top (Ax - b)$$

for $\lambda = \lambda^*(t)$ and $\nu = \nu^*(t)$. Thus $(\lambda^*(t), \nu^*(t))$ is a dual feasible pair.

Dual points from central path

Therefore the dual function $g(\lambda^*(t), \nu^*(t)) = \min_x L(x, \lambda^*(t), \nu^*(t))$ is finite and

$$\begin{aligned} g(\lambda^*(t), \nu^*(t)) &= f_0(x^*(t)) + \sum_{i=1}^{m} \lambda_i^*(t) f_i(x^*(t)) + \nu^*(t)^\top (Ax^*(t) - b) \\ &= f_0(x^*(t)) - m/t. \end{aligned}$$

- As an important consequence, we have

$$f_0(x^*(t)) - v^* \leqslant m/t.$$

- This confirms that $x^*(t)$ converge to an optimal point as $t \to \infty$.

Interpretation via KKT conditions

Since we have assumed that $x^*(t)$ is the unique solution to problem (82) for each $t > 0$, a point is equal to $x^*(t)$ if and only if $\exists \lambda, \nu$ such that

$$\begin{aligned} Ax = b, \quad f_i(x) &\leqslant 0, \qquad i = 1, \ldots, m \\ \lambda &\geqslant 0 \\ \nabla f_0(x) + \textstyle\sum_{i=1}^m \lambda_i \nabla f_i(x) + A^\top \nu &= 0 \\ -\lambda_i f_i(x) &= 1/t, \quad i = 1, \ldots, m. \end{aligned} \tag{85}$$

The only difference between (85) and the KKT condition (78) is that the complementarity condition $-\lambda_i f_i(x) = 0$ is replaced by the condition $-\lambda_i f_i(x) = 1/t$.

In particular, for large t, $x^*(t)$ and $\lambda^*(t), \nu^*(t)$ *'almost' satisfy* the KKT optimality conditions for the problem (77).

The barrier method

Algorithm. *Barrier method*

given strictly feasible x, $t := t^{(0)} > 0$, $\gamma > 1$, tolerance $\epsilon > 0$.
repeat

1. *Centering step.* Starting at x, compute $x^*(t)$ by minimizing $tf_0(x) + \phi(x)$, subject to $Ax = b$.
2. *Update.* $x := x^*(t)$
3. *Stopping criterion.* **quit** if $m/t < \epsilon$.
4. *Increase t.* Let $t := \gamma t$.

An execution of step 1 is called an *outer iteration*. We assume that Newton's method is used in step 1, and we refer to the Newton iterations or steps executed during the centering step as *inner iterations*.

The barrier method

- Computing $x^*(t)$ exactly is not necessary.
- Choice of $t^{(0)}$:
  If $t^{(0)}$ is chosen too large, the first outer iteration will require too many iterations.
  If $t^{(0)}$ is chosen too small, the algorithm will require extra outer iterations.
- The choice of the parameter γ involves a trade-off:
  If γ is small (*i.e.*, near 1) then centering step will be easy since the previous iterate x is a very good starting point but of course there will be a large number of outer iterations.
  On the other hand, a large γ resulting in fewer outer iterations but more inner iterations.

Newton step for modified KKT equations

In the step 1 of the barrier method, the Newton step $\delta_{x_{\text{nt}}}$ and associated dual variable are given by the linear equations

$$\begin{bmatrix} t\nabla^2 f_0(x) + \nabla^2 \phi(x) & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \delta_{x_{\text{nt}}} \\ \nu_{\text{nt}} \end{bmatrix} = - \begin{bmatrix} t\nabla f_0(x) + \nabla \phi(x) \\ 0 \end{bmatrix}. \quad (86)$$

These Newton steps for the centering problem can be interpreted as Newton steps for directly solving the modified KKT equations

$$\begin{aligned} \nabla f_0(x) + \textstyle\sum_{i=1}^m \lambda_i \nabla f_i(x) + A^\top \nu &= 0 \\ -\lambda_i f_i(x) &= 1/t, \quad i = 1, \ldots, m \\ Ax &= b. \end{aligned} \quad (87)$$

Newton step for modified KKT equations

Let $\lambda_i = 1/(-tf_i(x))$. This transforms (87) into

$$\nabla f_0(x) + \sum_{i=1}^m \frac{1}{-tf_i(x)} \nabla f_i(x) + A^\top \nu = 0, \quad Ax = b. \tag{88}$$

For small δ_x,

$$\begin{aligned}
&\nabla f_0(x+\delta_x) + \sum_{i=1}^m \frac{1}{-tf_i(x+\delta_x)} \nabla f_i(x+\delta_x) \\
\approx &\nabla f_0(x) + \nabla^2 f_0(x)\delta_x + \sum_{i=1}^m \frac{1}{-tf_i(x)} \nabla f_i(x) + \sum_{i=1}^m \frac{1}{-tf_i(x)} \nabla^2 f_i(x)\delta_x \\
&+ \sum_{i=1}^m \frac{1}{tf_i(x)^2} \nabla f_i(x) \nabla f_i(x)^\top \delta_x.
\end{aligned}$$

Newton step for modified KKT equations

Let

$$g = \nabla f_0(x) + \sum_{i=1}^{m} \frac{1}{-tf_i(x)} \nabla f_i(x),$$

$$H = \nabla^2 f_0(x) + \sum_{i=1}^{m} \frac{1}{-tf_i(x)} \nabla^2 f_i(x) + \sum_{i=1}^{m} \frac{1}{tf_i(x)^2} \nabla f_i(x) \nabla f_i(x)^\top.$$

Observe that

$$g = \nabla f_0(x) + (1/t)\nabla \phi(x), \quad H = \nabla f_0(x) + (1/t)\nabla^2 \phi(x).$$

The Newton step for (88) is

$$H\delta_x + A^\top \nu = -g, \quad A\delta_x = 0.$$

Comparing this with (86) shows that

$$\delta_x = \delta_{x_{\text{nt}}}, \quad \nu = \frac{\nu_{\text{nt}}}{t}.$$

Feasibility and phase I method

- The barrier method requires a strictly feasible starting point $x^{(0)}$.
- When such a point is not known, the barrier method is preceded by a preliminary stage, called *phase I*, in which a strictly feasible point is computed and used as the starting point for the barrier method.

Basic phase I method

To find a strictly feasible solution of inequalities and equalities

$$f_i(x) < 0, \quad i = 1, \ldots, m, \quad Ax = b, \tag{89}$$

we form and solve the following optimization problem

$$\begin{array}{ll} \min & s \\ \text{s.t.} & f_i(x) \leqslant s, \quad i = 1, \ldots, m \\ & Ax = b \end{array} \tag{90}$$

in the variable $x \in \mathbb{R}^n, s \in \mathbb{R}$. It's always strictly feasible, and called the *phase I optimization problem* associated with the inequality and equality system (89).

Basic phase I method

Let $\bar{v}^*$ be the optimal value of (90).

- If $\bar{v}^* < 0$, then (89) has a strictly feasible solution. In fact, we can terminate solving the problem (90) when $s < 0$.
- If $\bar{v}^* > 0$, then (89) is infeasible. In fact, we can terminate when a central point give a positive lower bound of $\bar{v}^* > 0$.
- If $\bar{v}^* = 0$ and the minimum is attained at x^* and $s^* = 0$, then the set of inequalities is feasible but not strictly feasible. If $\bar{v}^* = 0$ and the minimum is not attained, then the inequalities are infeasible.

Primal-dual search direction

The modified KKT conditions (87) can be expressed as $r_t(x, \lambda, \nu) = 0$, where $t > 0$ and

$$r_t(x, \lambda, \nu) = \begin{bmatrix} \nabla f_0(x) + J[f(x)]^\top \lambda + A^\top \nu \\ -\mathbf{diag}(\lambda) f(x) - (1/t)\mathbf{1} \\ Ax - b \end{bmatrix}. \tag{91}$$

Here $f : \mathbb{R}^n \to \mathbb{R}^m$ and $J[f]$ are given by

$$f(x) = \begin{bmatrix} f_1(x) \\ \vdots \\ f_m(x) \end{bmatrix}, \quad J[f(x)] = \begin{bmatrix} \nabla f_1(x)^\top \\ \vdots \\ \nabla f_m(x)^\top \end{bmatrix}.$$

Primal-dual search direction

If x, λ, ν satisfy $r_t(x, \lambda, \nu) = 0$ (and $f_i(x) < 0$), then $x = x^*(t)$, $\lambda = \lambda^*(t)$ and $\nu = \nu^*(t)$.

- The first block component of r_t,

$$r_{\text{dual}} = \nabla f_0(x) + J[f(x)]^\top \lambda + A^\top \nu$$

  is called the *dual residual*.
- The last block component, $r_{\text{pri}} = Ax - b$, is called the *primal residual*.
- The middle block

$$r_{\text{cent}} = -\mathbf{diag}(\lambda) f(x) - (1/t)\mathbf{1},$$

  is the *centrality residual*, *i.e.*, the residual for the modified complementarity condition.

Primal-dual search direction

Let $y = (x, \lambda, \nu)$ denote the current point and $\delta_y = (\delta_x, \delta_\lambda, \delta_\nu)$ denote the Newton step for solving the equation $r_t(x, \lambda, \nu) = 0$, for fixed t where $f(x) < 0, \lambda > 0$.

The Newton step is characterized by

$$r_t(y + \delta_y) \approx r_t(y) + J[r_t(y)]\delta_y = 0.$$

Primal-dual search direction

In terms of x, λ, ν, we have

$$\begin{bmatrix} \nabla^2 f_0(x) + \sum_{i=1}^m \lambda_i \nabla^2 f_i(x) & J[f(x)]^\top & A^\top \\ -\mathbf{diag}(\lambda) J[f(x)] & -\mathbf{diag}(f(x)) & 0 \\ A & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_x \\ \delta_\lambda \\ \delta_\nu \end{bmatrix} = - \begin{bmatrix} r_{\text{dual}} \\ r_{\text{cent}} \\ r_{\text{pri}} \end{bmatrix} \tag{92}$$

The *primal-dual search direction* $\delta_{y_{\text{pd}}} = (\delta_{x_{\text{pd}}}, \delta_{\lambda_{\text{pd}}}, \delta_{\nu_{\text{pd}}})$ is defined as the solution of (92).

The surrogate duality gap

In the primal-dual interior-point method the iterates $x^{(k)}, \lambda^{(k)}$ and $\nu^{(k)}$ are not necessarily feasible. We cannot easily evaluate a duality gap as we do in the barrier method.

Instead, we define the *surrogate duality gap*, for any x that satisfies $f(x) < 0$ and $\lambda \geqslant 0$, as

$$\hat{\eta}(x, \lambda) = -f(x)^\top \lambda.$$

Remark: The surrogate gap $\hat{\eta}$ would be the duality gap, if x were primal feasible and λ, ν were dual feasible. Note that the value of the parameter t corresponding to the surrogate duality gap $\hat{\eta}$ is $m/\hat{\eta}$.

Primal-dual interior-point method

Algorithm. *Primal-dual interior-point method.*

given x that satisfies
$f_1(x) < 0, \ldots, f_m(x) < 0, \lambda > 0, \gamma > 1, \epsilon_{\text{feas}} > 0, \epsilon > 0.$
repeat

1. *Determine t.* Set $t := \gamma(m/\hat{\eta})$.
2. Compute primal-dual search direction $\delta_{y_{\text{pd}}}$.
3. *Line search and update.*
   Determine step length $\alpha > 0$ and set $y := y + \alpha\delta_{y_{\text{pd}}}$.

until $\|r_{\text{pri}}\|_2 \leqslant \epsilon_{\text{feas}}$, $\|r_{\text{dual}}\|_2 \leqslant \epsilon_{\text{feas}}$, and $\hat{\eta} \leqslant \epsilon$.

Line search in primal-dual interior-point method

The line search in step 3 is a standard backtracking line search.

For a step size α, let

$$y^+ = \begin{bmatrix} x^+ \\ \lambda^+ \\ \nu^+ \end{bmatrix} = \begin{bmatrix} x \\ \lambda \\ \nu \end{bmatrix} + \alpha \begin{bmatrix} \delta_{x_{\mathsf{pd}}} \\ \delta_{\lambda_{\mathsf{pd}}} \\ \delta_{\nu_{\mathsf{pd}}} \end{bmatrix}$$

Let

$$\alpha^{\mathsf{max}} = \mathsf{sup}\{\alpha \in [0,1] \mid \lambda + \alpha\delta_\lambda \geqslant 0\} = \mathsf{min}\left\{1, \mathsf{min}\{\frac{-\lambda_i}{\delta_{\lambda_i}} \mid \delta_{\lambda_i} < 0\}\right\}$$

to be the largest positive step length that gives $\lambda^+ \geqslant 0$.

Line search in primal-dual interior-point method

We start backtracking with $\alpha = 0.99\alpha^{\max}$, and multiply α by $\beta \in (0, 1)$ until we have $f(x^+) < 0$. We continue multiplying α by β until we have

$$\|r_t(x^+, \lambda^+, \nu^+)\|_2 \leqslant (1 - \tau\alpha)\|r_t(x, \lambda, \nu)\|_2.$$

Here τ is typically chosen in the range $[0.01, 0.1]$.

Exercises

Ex 1. Let $C \subseteq \mathbb{R}^n$ be the solution set of a quadratic inequality,

$$C = \{x \in \mathbb{R}^n | x^\top A x + b^\top x + c \leqslant 0\},$$

with $A \in \mathrm{S}^n$, $b \in \mathbb{R}^n$, and $c \in \mathbb{R}$.

(a) Show that C is convex if $A \succeq 0$.

(b) Show that the intersection of C and the hyperplane defined by $g^\top x + h = 0$ (where $g \neq 0$) is convex if $A + \lambda g g^\top \succeq 0$ for some $\lambda \in \mathbb{R}$.

Ex 2. Let $\lambda_1(X) \geqslant \lambda_2(X) \geqslant \ldots \geqslant \lambda_n(X)$ denote the eigenvalues of a matrix $X \in \mathrm{S}^n$. Prove that the maximum eigenvalue $\lambda_1(X)$ is convex. Moreover, Show that $\sum_{i=1}^k \lambda_i(X)$ is convex on S^n. *Hint.* Use the variational characterization

$$\sum_{i=1}^k \lambda_i(X) = \sup\{\mathrm{tr}(V^\top X V) | V \in \mathbb{R}^{n \times k}, V^\top V = \mathrm{I}\}.$$

Exercises

Ex 3. Find the dual function of the LP

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Gx \preceq h \\ & Ax = b. \end{array}$$

Give the dual problem, and make the implicit equality constraints explicit.

Ex 4. Consider the equality constrained least-squares problem

$$\begin{array}{ll} \min & \|Ax - b\|_2^2 \\ \text{s.t.} & Gx = h \end{array}$$

where $A \in \mathbb{R}^{m \times n}$ with $\mathbf{rank} A = n$, and $G \in \mathbb{R}^{p \times n}$ with $\mathbf{rank} G = p$. Give the KKT conditions, and derive expressions for the primal solution x^* and the dual solution ν^*.

Exercises

Ex 5. Suppose $Q \succeq 0$. The problem

$$\begin{array}{ll} \min & f(x) + (Ax - b)^\top Q(Ax - b) \\ \text{s.t.} & Ax = b \end{array}$$

is equivalent to the primal equality constrained optimization problem (70). What is the Newton step for this problem? Is it the same as that for the primal problem?

Exercises

Ex 6. Suppose we use the infeasible start Newton method to minimize $f(x)$ subject to $a_i^\top x = b_i$, $i = 1, \ldots, p$.

(a) Suppose the initial point $x^{(0)}$ satisfies the linear equality $a_i^\top x^{(0)} = b_i$. Show that the linear equality will remain satisfied for future iterates, *i.e.*, $a_i^\top x^{(k)} = b_i$ for all k.

(b) Suppose that one of the equality constraints becomes satisfied at iteration k, *i.e.*, we have $a_i^\top x^{(k-1)} \neq b_i$, $a_i^\top x^{(k)} = b_i$. Show that at iteration k, *all* the equality constraints are satisfied.

Exercises

Ex 7. Suppose we add the constraint $x^\top x \leqslant R^2$ to the problem (77):

$$\begin{array}{ll} \min & f_0(x) \\ \text{s.t.} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & Ax = b \\ & x^T x \leq R^2 \end{array}$$

Let $\tilde{\phi}$ denote the logarithmic barrier function for this modified problem. Find $a > 0$ for which $\nabla^2(tf_0(x) + \phi(x)) \succeq a\mathrm{I}$ holds, for all feasible x.

Exercises

Ex 8. Consider the problem (77), with central path $x^*(t)$ for $t > 0$, defined as the solution of (82).
For $u > p^*$, let $z^*(u)$ denote the solution of

$$\begin{array}{ll} \text{min} & -\log(u - f_0(x)) - \sum_{i=1}^m \log(-f_i(x)) \\ \text{s.t.} & Ax = b \end{array}$$

Show that the curve define by $z^*(u)$, for $u > p^*$, is the central path. (In other words, for each $u > p^*$, there is a $t > 0$ for which $x^*(t) = z^*(u)$, and conversely, for each $t > 0$, there is a $u > p^*$ for which $z^*(u) = x^*(t)$).

Outline I

Outline II

Sparse Optimization

Many problems of recent interest in statistics and related areas can be posed in the framework of sparse optimization. Due to the explosion in size and complexity of modern data analysis (BigData), it is increasingly important to be able to solve problems with a very large number of features, training examples, or both.

0-norm optimization

$$(P_0) \quad \min_{x} \|x\|_0 \quad \text{s.t.} \quad Ax = b. \tag{93}$$

$$(P_0^\epsilon) \quad \min_{x} \|x\|_0 \quad \text{s.t.} \quad \|b - Ax\| \leqslant \epsilon. \tag{94}$$

Greedy algorithms

Greedy strategies are usually adopted in solving the 0-norm problems. The following algorithm is known in the literature of signal processing by the name *Orthogonal Matching Pursuit* (OMP).

Task: Approximate the solution of (P_0): $\min_{\mathbf{x}} \|\mathbf{x}\|_0$ subject to $\mathbf{Ax} = \mathbf{b}$.

Parameters: We are given the matrix $\mathbf{A}$, the vector $\mathbf{b}$, and the error threshold ϵ_0.

Initialization: Initialize $k = 0$, and set

- The initial solution $\mathbf{x}^0 = 0$.
- The initial residual $\mathbf{r}^0 = \mathbf{b} - \mathbf{Ax}^0 = \mathbf{b}$.
- The initial solution support $\mathcal{S}^0 = Support\{\mathbf{x}^0\} = \emptyset$.

Main Iteration: Increment k by 1 and perform the following steps:

- **Sweep:** Compute the errors $\epsilon(j) = \min_{z_j} \|\mathbf{a}_j z_j - \mathbf{r}^{k-1}\|_2^2$ for all j using the optimal choice $z_j^* = \mathbf{a}_j^T \mathbf{r}^{k-1} / \|\mathbf{a}_j\|_2^2$.
- **Update Support:** Find a minimizer j_0 of $\epsilon(j)$: $\forall\ j \notin \mathcal{S}^{k-1}$, $\epsilon(j_0) \leq \epsilon(j)$, and update $\mathcal{S}^k = \mathcal{S}^{k-1} \cup \{j_0\}$.
- **Update Provisional Solution:** Compute $\mathbf{x}^k$, the minimizer of $\|\mathbf{Ax} - \mathbf{b}\|_2^2$ subject to $Support\{\mathbf{x}\} = \mathcal{S}^k$.
- **Update Residual:** Compute $\mathbf{r}^k = \mathbf{b} - \mathbf{Ax}^k$.
- **Stopping Rule:** If $\|\mathbf{r}^k\|_2 < \epsilon_0$, stop. Otherwise, apply another iteration.

Output: The proposed solution is $\mathbf{x}^k$ obtained after k iterations.

Dictionary learning

The optimization model of dictionary learning for sparse and redundant representations:

$$\min_{D,X} \|Y - DX\|_{Frob} \quad \text{s.t.} \quad \|x_j\|_0 \leqslant k_0,\ j = 1, \cdots, N \tag{95}$$

where

$$Y = (y_1, \cdots, y_N) \in \mathbb{R}^{n \times N},$$

$$D = (d_1, \cdots, d_m) \in \mathbb{R}^{n \times m},$$

$$X = (x_1, \cdots, x_N) \in \mathbb{R}^{m \times N}.$$

Dictionary learning

There are two training mechanisms, the first named Method of Optimal Directions (MOD) by Engan et al., and the second named K-SVD, by Aharon et al..

- MOD
- K-SVD
- .....

Convex relaxation

Convex relaxation technique is a way to render 0-norm more tractable.

Convexifying with the ℓ_1 norm, we come to the new optimization problem

$$(P_1) \quad \min_{\mathrm{x}} \|W\mathrm{x}\|_1 \quad \text{s.t.} \quad A\mathrm{x} = \mathrm{b} \tag{96}$$

where W is a diagonal positive-definite matrix that introduces the precompensating weights.

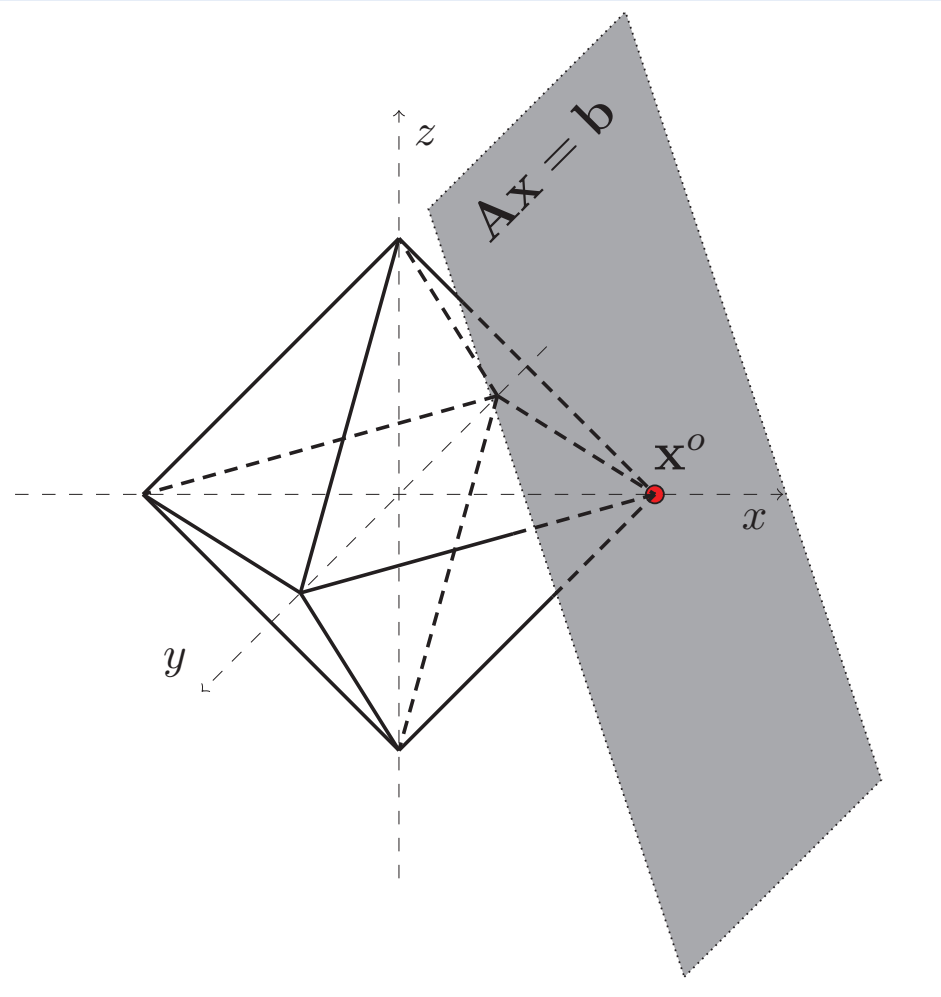
An error-tolerant version of (P_1) is defined by

$$(P_1^\epsilon) \quad \min_{\mathrm{x}} \|W\mathrm{x}\|_1 \quad \text{s.t.} \quad \|\mathrm{b} - A\mathrm{x}\| \leqslant \epsilon. \tag{97}$$

It was named *Basis Pursuit* (BP) when all the columns of A are normalized (and thus $W = I$).

Basic pursuit

$$(BP) \quad \min_{x} \|x\|_1 \quad \text{s.t.} \quad Ax = b.$$



BP denoising and LASSO

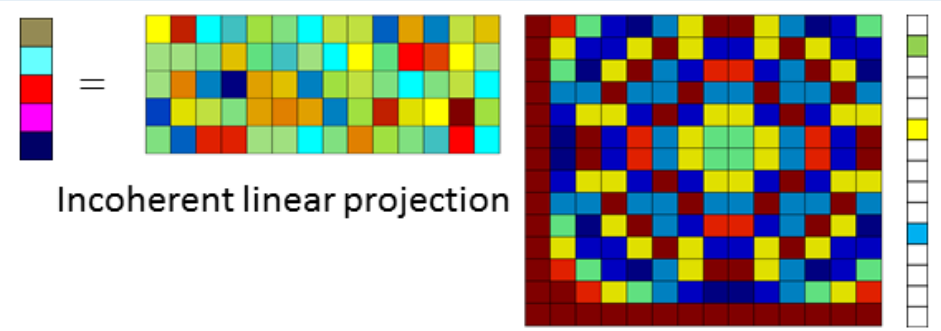
$$
\begin{aligned}
(BP_\tau) \quad & \min_{\mathsf{x}} \|A\mathsf{x}-\mathsf{b}\|_2^2 \quad \text{s.t.} \quad \|\mathsf{x}\|_1 \leqslant \tau, \\
(BP_\mu) \quad & \min_{\mathsf{x}} \|\mathsf{x}\|_1 + \tfrac{\mu}{2}\|A\mathsf{x}-\mathsf{b}\|_2^2, \\
(BP_\delta) \quad & \min_{\mathsf{x}} \|\mathsf{x}\|_1 \quad \text{s.t.} \quad \|A\mathsf{x}-\mathsf{b}\|_2 \leqslant \delta.
\end{aligned}
$$

Questions:

- Are they equivalent? and in what sense?
- How to choose parameters?

Sparse under basis Ψ

$$\min_{s}\{\|s\|_1 : A\Psi s = b\}$$



If Ψ is orthogonal, the problem is equivalent to

$$\min_{x}\{\|\Psi^* x\|_1 : Ax = b\}.$$

Sparse after transform $\mathcal{L}$

$$\min_{x}\{\|\mathcal{L}x\|_1 : Ax = b\}$$

Examples of $\mathcal{L}$:

- DCT, wavelets, curvelets, ridgelets, ...
- tight frames, Gabor, ...
- total (generalized) variation

Ref: E. J. Cands, Y. Eldar, D. Needell and P. Randall. Compressed sensing with coherent and redundant dictionaries. Applied and Computational Harmonic Analysis, 31(1): 59-73, 2011.

Joint/group sparsity

Decompose $\{1, 2, \cdots, n\} = \mathcal{G}_1 \bigcup \mathcal{G}_2 \bigcup \cdots \bigcup \mathcal{G}_S$, and $\mathcal{G}_i \bigcap \mathcal{G}_j = \emptyset, i \neq j$.

Joint/group sparse recovery model:

$$\min_{x} \{\|x\|_{\mathcal{G},2,1} : Ax = b\}$$

where

$$\|x\|_{\mathcal{G},2,1} = \sum_{s=1}^{S} w_s \|x_{\mathcal{G}_s}\|_2.$$

Side constraints

- Nonnegativity: $x \geq 0$
- Box constraints: $lb \leq x \leq ub$
- Linear inequalities: $Qx \leq c$

They generate "corners" and can be very effective in practice.

Shrinkage

- Shrinkage is popular in sparse optimization algorithms
- In optimization, non-smooth functions like ℓ_1 has difficulty using general smooth optimization methods.
- But, ℓ_1 is component-wise separable, so it does get along well with separable (smooth or non-smooth) functions.
- For example,

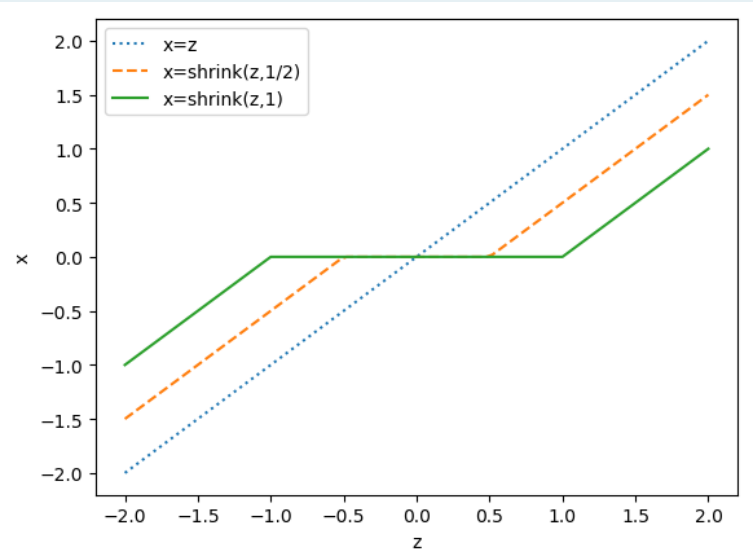
$$\min_{\mathrm{x}} \|\mathrm{x}\|_1 + \frac{1}{2\tau}\|\mathrm{x} - \mathrm{z}\|_2^2$$

is equivalent to solving $\min_{x_i} |x_i| + \frac{1}{2\tau}|x_i - z_i|^2$ over each i.

Soft-thresholding shrinkage

The problem is separable and has an explicit solution

$$(\text{shrink}(z,\tau))_i = \begin{cases} z_i - \tau & z_i > \tau, \\ 0 & -\tau \le z_i \le \tau, \\ z_i + \tau & z_i < -\tau. \end{cases}$$



The shrinkage operator can be written in Matlab code as:

$$\mathbf{x} = \max(\text{abs}(\mathbf{z})\text{-tau},0).*\text{sign}(\mathbf{z}).$$

Soft-thresholding shrinkage

- The following problem is called Moreau-Yosida regularization

$$\min_x r(x) + \frac{1}{2\tau}\|x - z\|_2^2.$$

- For example $r(x) = \|x\|_2$, the solution to

$$\min_x \|x\|_2 + \frac{1}{2\tau}\|x - z\|_2^2$$

  is, if we treat $0/0 = 0$,

$$x_{opt} = \max\{\|z\|_2 - \tau, 0\} \cdot (z/\|z\|_2).$$

- Used in joint/group-sparse recovery algorithms.

Soft-thresholding shrinkage

- Consider the following nuclear norm optimization

$$\min_X \|X\|_* + \frac{1}{2\tau}\|X - Z\|_F^2.$$

Let $Z = U\Sigma V^T$ be the singular value decomposition of Z.

- Let $\hat{\Sigma}$ be the diagonal matrix with diagonal entries

$$\mathrm{diag}(\hat{\Sigma}) = \mathrm{shrink}(\mathrm{diag}(\Sigma), \tau)),$$

then

$$X_{opt} = U\hat{\Sigma}V^T.$$

- In general, matrix problems with only unitary-invariant functions (e.g., $\|\cdot\|_*$, $\|\cdot\|_F$, spectral norm, trace) and constraints (e.g., positive or negative semi-definiteness) typically reduce to vector problems regarding singular values.

Prox-linear algorithm

Consider the general form

$$\min_{x} r(x) + f(x).$$

where r is the regularization function and f is the data fidelity function.

The prox-linear algorithm is:

$$x^{k+1} = \arg\min_{x} r(x) + f(x^k) + < \nabla f(x^k), x - x^k > + \frac{1}{2\delta_k}\|x - x^k\|_2^2.$$

The last term keeps x^{k+1} close to x^k, and the parameter δ_k determines the step size. It is equivalent to

$$x^{k+1} = \arg\min_{x} r(x) + \frac{1}{2\delta_k}\|x - (x^k - \delta_k \nabla f(x^k))\|_2^2.$$

Alternating direction method of multipliers (ADMM)

The Alternating Direction Method of Multipliers (ADMM) was developed in the 1970s, with roots in the 1950s, and is equivalent or closely related to many other algorithms, such as dual decomposition, the method of multipliers, Douglas-Rachford splitting, Spingarns method of partial inverses, Dykstras alternating projections, Bregman iterative algorithms for 1-norm problems, proximal methods, and others.

ADMM

The ADMM can be applied to a wide variety of statistical and machine learning problems of recent interest, including the lasso, sparse logistic regression, basis pursuit, covariance selection, support vector machines, and many others.

ADMM

$$\min_{X \in C^{n \times T}} \mu \|X\|_p + \|AX - B\|_q \tag{98}$$

Let $p := \{2, 1\}, q := \{1, 1\}$ which denote joint convex norm, we have

$$\min_{X \in C^{n \times T}} \mu \|X\|_{2,1} + \|AX - B\|_{1,1}$$

where $\|X\|_{2,1} = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{T} x_{ij}^2}$, $\|X\|_{1,1} = \sum_{i=1}^{n} \sum_{j=1}^{T} |x_{ij}|$.

For example $T = 1$,

$$\min_{x \in C^n} \mu \|x\|_p + \|Ax - b\|_q.$$

$$\min_{X \in C^{n \times T}} \mu \|X\|_p + \|AX - B\|_q \tag{98}$$

Let $p := \{2, 1\}, q := \{1, 1\}$ which denote joint convex norm, we have

$$\min_{X \in C^{n \times T}} \mu \|X\|_{2,1} + \|AX - B\|_{1,1}$$

where $\|X\|_{2,1} = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{T} x_{ij}^2}$, $\|X\|_{1,1} = \sum_{i=1}^{n} \sum_{j=1}^{T} |x_{ij}|$.

For example $T = 1$,

$$\min_{x \in C^n} \mu \|x\|_p + \|Ax - b\|_q.$$

$$\min_{X \in C^{n \times T}} \mu \|X\|_p + \|AX - B\|_q \tag{98}$$

Let $p := \{2, 1\}, q := \{1, 1\}$ which denote joint convex norm, we have

$$\min_{X \in C^{n \times T}} \mu \|X\|_{2,1} + \|AX - B\|_{1,1}$$

where $\|X\|_{2,1} = \sum_{i=1}^{n} \sqrt{\sum_{j=1}^{T} x_{ij}^2}$, $\|X\|_{1,1} = \sum_{i=1}^{n} \sum_{j=1}^{T} |x_{ij}|$.

For example $T = 1$,

$$\min_{x \in C^n} \mu \|x\|_p + \|Ax - b\|_q.$$

ADMM

$$\begin{aligned} \min\ & \mu\|z\|_p + \|y\|_q \\ \text{s.t.}\ & x - z = 0 \\ & Ax - y = b \end{aligned} \tag{99}$$

$$\begin{aligned} L(x, y, z, \lambda_y, \lambda_z, \rho) =& \mu\|z\|_p + \|y\|_q + \mathrm{Re}(\lambda_z^T(x - z) + \lambda_y^T(Ax - y - b)) \\ & + \frac{\rho}{2}(\|x - z\|_2^2 + \|Ax - y - b\|_2^2) \end{aligned} \tag{100}$$

where $\lambda_y \in C^n, \lambda_z \in C^m$ are the Lagrangian multipliers and $\rho > 0$ is a penalty parameter.

ADMM

$$\begin{aligned} \min\ & \mu\|\mathsf{z}\|_p + \|\mathsf{y}\|_q \\ \text{s.t. } & \mathsf{x} - \mathsf{z} = 0 \\ & \mathsf{A}\mathsf{x} - \mathsf{y} = \mathsf{b} \end{aligned} \tag{99}$$

$$\begin{aligned} L(\mathsf{x}, \mathsf{y}, \mathsf{z}, \lambda_y, \lambda_z, \rho) =& \mu\|\mathsf{z}\|_p + \|\mathsf{y}\|_q + \mathrm{Re}(\lambda_z^T(\mathsf{x} - \mathsf{z}) + \lambda_y^T(\mathsf{A}\mathsf{x} - \mathsf{y} - \mathsf{b})) \\ & + \frac{\rho}{2}(\|\mathsf{x} - \mathsf{z}\|_2^2 + \|\mathsf{A}\mathsf{x} - \mathsf{y} - \mathsf{b}\|_2^2) \end{aligned} \tag{100}$$

where $\lambda_y \in C^n, \lambda_z \in C^m$ are the Lagrangian multipliers and $\rho > 0$ is a penalty parameter.

ADMM

$$\begin{cases} x^{k+1} := \arg\min \frac{1}{2}(\|x - z^k + u_z^k\|_2^2 + \|Ax - y^k - b + u_y^k\|_2^2) \\ y^{k+1} := \arg\min \|y\|_q + \frac{\rho}{2}\|y - (Ax^{k+1} - b) - u_y^k\|_2^2 \\ z^{k+1} := \arg\min \mu\|z\|_p + \frac{\rho}{2}\|z - x^{k+1} - u_z^k\|_2^2 \end{cases} \tag{101}$$

After solving three subproblems, we update the Lagrangian multipliers as follows:

$$\begin{cases} u_z^{k+1} = u_z^k + \gamma(x^{k+1} - z^{k+1}) \\ u_y^{k+1} = u_y^k + \gamma(Ax^{k+1} - y^{k+1} - b) \end{cases} \tag{102}$$

where $u_y = \frac{1}{\rho}\lambda_y, u_z = \frac{1}{\rho}\lambda_z, \gamma > 0$ is the step size.

Outline I

Outline II

Overview

Mathematical optimization is one of the pillars of machine learning. Large-scale machine learning, where the amount of both the training data and the parameters is large, represents a distinctive setting in which traditional nonlinear optimization techniques typically falter.

We will briefly introduce some typical optimization problems arising from machine learning and then turn to stochastic algorithms—the main content of this section—and other popular methods together with specific models applicable to them.

Typical Form of Problems

For simplicity, we focus on the problems that arise in the context of *supervised classification*; i.e., we focus on the optimization of prediction functions for labeling unseen data based on information contained in a set of labeled training data.

Such a focus is reasonable as many unsupervised and other learning techniques reduce to optimization problems of comparable form.

Typical Form of Problems

For example:

- Regression. Although the methodology of dealing with regression is quite different from that of classification, regression does share a model similar to supervised classification. Supervised classification and regression are collectively called supervised learning.
- Deep reinforcement learning. In deep Q-learning network (DQN), the samples are attained by interacting with environment, and to train the agent is to solve the Bellman equation in a regression fashion.
- Generative adversarial network. The GAN is composed of a generator and a discriminator, which are usually trained alternately. The training process of each part could be treated as a supervised classification, where the label means whether the sample comes from the data distribution or not.

Fundamentals

Goal determine a prediction function $h : \mathcal{X} \to \mathcal{Y}$ from an input space $\mathcal{X}$ to an output space $\mathcal{Y}$.

Request h should avoid rote memorization and instead generalizes the concepts that can be learned from a given set of examples.

Scheme choose h by attempting to minimize a risk measure over an adequately selected family of prediction functions, call it $\mathcal{H}$.

Suppose that samples are sampled from a joint probability distribution function $Pr(x, y)$.

h to be sought should yield a small *expected risk* of misclassification *over all possible inputs*, i.e., minimize

$$R(h) = Pr\left[h(x) \neq y\right] = E\left[1_{[h(x)\neq y]}\right]. \tag{103}$$

Such a framework is *variational* since we are optimizing over a set of functions, and is *stochastic* since the objective function involves an expectation.

In practice, the expectation is taken on samples $\{(x_i, y_i)\}_{i=1}^n$ and h should minimize the *empirical risk* of misclassification

$$R_n(h) = \frac{1}{n}\sum_{i=1}^{n} 1_{[h(x_i)\neq y_i]}, \quad \text{where } 1_{[A]} = \begin{cases} 1 & \text{if } A \text{ is true} \\ 0 & \text{otherwise} \end{cases}. \tag{104}$$

Choice of Prediction Function Family

The family of function $\mathcal{H}$ should be determined with three potentially competing goals in mind.

1. Adequate capacity: $\mathcal{H}$ should contain prediction functions that are able to achieve a low empirical risk over the training set, so as to avoid underfitting the data.

   This can be achieved by selecting a rich family of functions or by using a *priori* knowledge to select a well-targeted family.

Choice of Prediction Function Family

2. Low generalization error: The gap between expected risk and empirical risk $R(h) - R_n(h)$ should be small over all $h \in \mathcal{H}$.

   Generally this gap decreases when one uses more training examples but it increases when one uses richer families of functions, due to potential overfitting.

3. Efficient training: $\mathcal{H}$ should be selected so that one can efficiently solve the corresponding optimization problem, the difficulty of which may increase when one employs a richer family of functions and/or a larger training set.

Generalization Error

By certain *laws of large numbers*, the Hoeffding inequality guarantees that, with probability $1 - \eta$,

$$|R(h) - R_n(h)| \leqslant \sqrt{\frac{1}{2n} \log\left(\frac{2}{\eta}\right)} \text{ for a given } h \in \mathcal{H}$$

This bound offers the intuitive explanation that the gap decreases as one uses more training examples.

For a uniform generalization error bound, one often turns to *uniform laws of large numbers* and the concept of the Vapnik-Chervonenkis (VC) dimension of $\mathcal{H}$, a measure of the capacity of such a family of functions.

Generalization Error

Roughly speaking, the VC dimension of a family of functions is the minimal size of samples on which all the functions in the family fail.

For the intuition behind this concept, consider, e.g., a binary classification scheme in $\mathbb{R}^2$ where one assigns a label of 1 for points above a polynomial and -1 for points below.Then the set of linear polynomials has a low capacity with VC dimension of 3.

With $d_{\mathcal{H}}$ defined as the VC dimension of $\mathcal{H}$, one has with probability at least $1-\eta$ that

$$\sup_{h\in\mathcal{H}} |R(h) - R_n(h)| \leqslant \mathcal{O}\left(\sqrt{\frac{1}{2n}\log\left(\frac{2}{\eta}\right) + \frac{d_{\mathcal{H}}}{n}\log\left(\frac{n}{d_{\mathcal{H}}}\right)}\right). \tag{105}$$

(105) is one of the most important results in learning theory.

Structural Risk Minimization

Rather than choose a generic family of prediction functions (difficult to optimize and estimate the generalization error) one chooses a *structure*, i.e., a collection of nested function families.

For instance, such a structure can be formed as a collection of subsets of a given family $\mathcal{H}$ in the following manner: given a preference function Ω, choose various values of a *hyperparameter* C, according to each of which one obtains the subset $\mathcal{H}_C := \{h \in \mathcal{H} \mid \Omega(h) \leqslant C\}$.

Given a fixed number of examples, increasing C reduces the empirical risk, but after some point it typically increases the gap between expected and empirical risks, as illustrated in Fig 7.

Other ways to introduce structures are to consider a regularized empirical risk $R_n(h) + \lambda\Omega(h)$.

Structural Risk Minimization

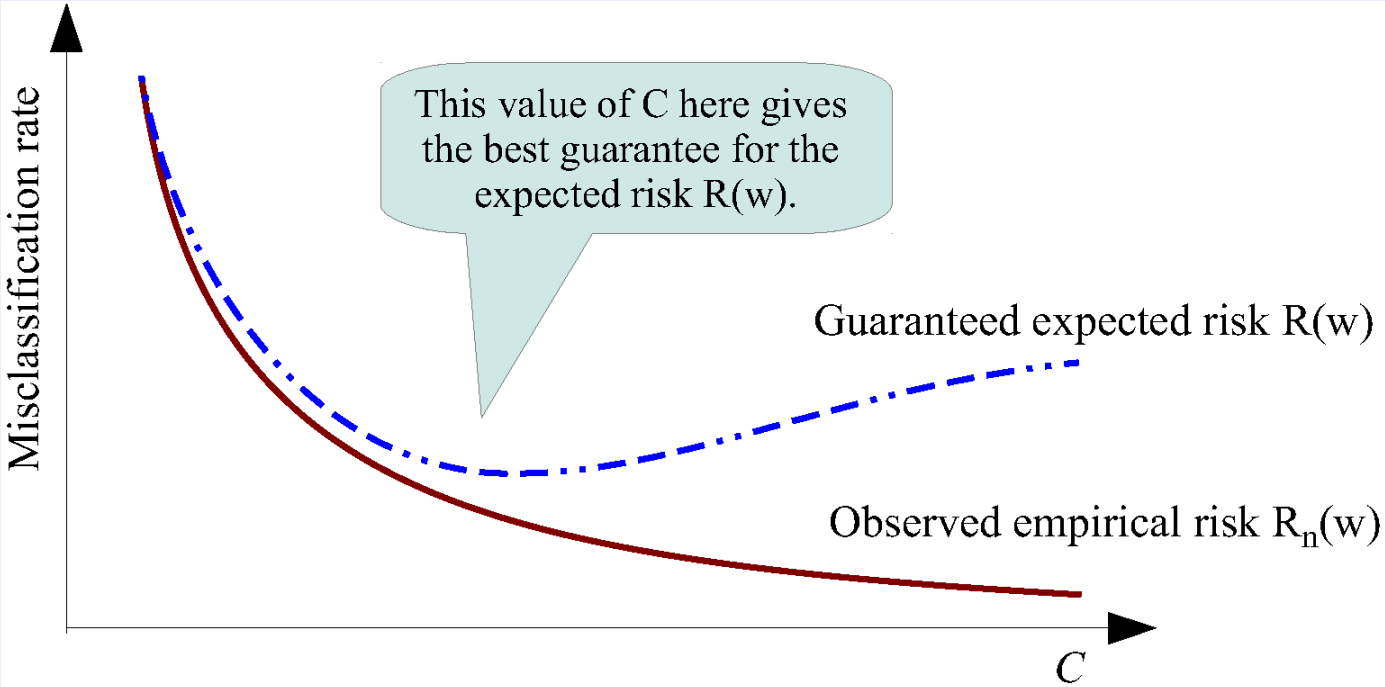


Figure: Illustration of structural risk minimization

Structural Risk Minimization

One can avoid estimating the gap between empirical and expected risk by splitting the available data into three subsets: a *training set*, a *validation set* and a *testing set*.

Specifically, over the training set one minimizes an empirical risk measure R_n over $\mathcal{H}_C$ for various values of C. This results in a handful of candidate functions.

The validation set is then used to estimate the expected risk corresponding to each candidate solution, after which one chooses the function yielding the lowest estimated risk value.

The testing set is used to estimate the expected risk for the candidate that is ultimately chosen.

More Practical Statements

Now we assume that the prediction function h has a fixed form and is parameterized by a real vector $w \in \mathbb{R}^d$ over which the optimization is to be performed.

Formally, for some given $h(\cdot, \cdot) : \mathbb{R}^{d_x} \times \mathbb{R}^d \to \mathbb{R}^{d_y}$, we consider the family of prediction functions

$$\mathcal{H} := \left\{ h(\cdot, w) \mid w \in \mathbb{R}^d \right\}.$$

To measure the losses incurred from inaccurate predictions, we assume a given loss function $\ell : \mathbb{R}^{d_y} \times \mathbb{R}^{d_y} \to \mathbb{R}$. An input-output pair (x, y) yields the predicted output $h(x, w)$ and the loss $\ell(h(x, w), y)$.

More Practical Statements

We have the expected risk

$$R(w) = E_{(x,y)\sim Pr(x,y)}\left[\ell(h(x,w),y)\right], \tag{106}$$

and the empirical risk

$$R_n(w) = \frac{1}{n}\sum_{i=1}^{n}\ell(h(x_i,w),y_i). \tag{107}$$

To simplify the notation, let ξ be a sample (x,y) and $f(w,\xi) = \ell(h(x,w),y)$, then the expected risk is

$$R(w) = E_{\xi}\left[f(w,\xi)\right]. \tag{108}$$

For a set of samples $\{\xi_i\}_{i=1}^{n}$, let us define $f_i(w)$ to be $f(w,\xi_i)$ and then the empirical risk is

$$R_n(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w). \tag{109}$$

Recall the batch (ordinary) gradient descent method. To minimize the empirical risk (as (109)), w is updated by

$$w_{k+1} \leftarrow w_k - \alpha_k \nabla R_n(w_k) = w_k - \frac{\alpha_k}{n} \sum_{i=1}^{n} \nabla f_i(w_k) \tag{110}$$

where $\alpha_k > 0$ is a stepsize. Computing the step $-\alpha_k \nabla R_n(w_k)$ is expensive since it needs accessing all the samples.

Stochastic gradient (SG) meanwhile uses only one sample at each iteration:

$$w_{k+1} \leftarrow w_k - \alpha_k \nabla f_{i_k}(w_k) \tag{111}$$

where i_k is chosen randomly from $\{1, \ldots, n\}$. While $-\nabla f_{i_k}(w_k)$ might not be one of descent from w_k, if it is a descent direction *in expectation*, then the sequence $\{w_k\}$ can be guided toward a minimizer of R_n.

A Brief Introduction

To generalize SG method, we consider two ways:

- reduce the noise (variance) of each iteration by generating a batch of samples instead of a single sample.
- make use of second-order information and compute a stochastic Newton or quasi-Newton direction rather than a gradient direction.

A Brief Introduction

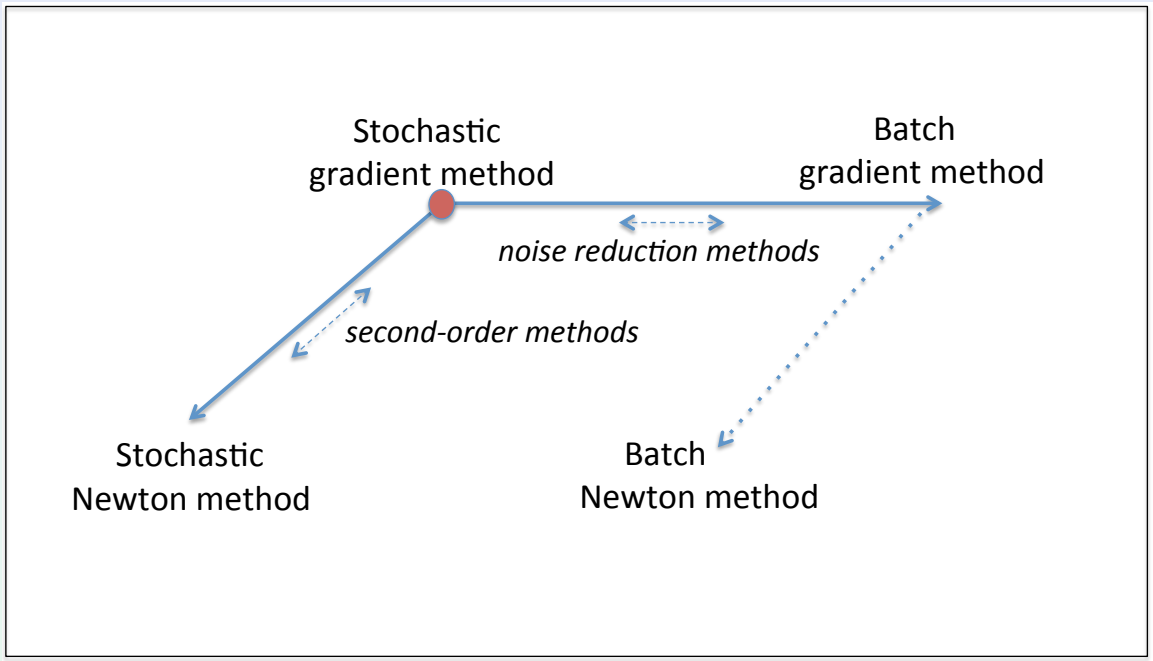


Figure: Schematic of a two dimensional spectrum of optimization methods for machine learning.

Analyses of SG Method

Here we give a general framework of stochastic gradient methods by introducing a general ξ_k and a general direction $g(w_k, \xi_k)$:

Algorithm 1 Stochastic Gradient

1: Choose an initial iterate w_1.
2: **for** $k = 1, 2, \ldots$ **do**
3: Generate a realization of the random variable ξ_k.
4: Compute a direction $g(w_k, \xi_k)$.
5: Choose a stepsize $\alpha_k > 0$.
6: Set the new iterate as $w_{k+1} \leftarrow w_k - \alpha_k g(w_k, \xi_k)$.
7: **end for**

Analyses of SG Method

ξ_k could be either one sample or a set of samples, and our analysis cover the following choices of $g(w_k, \xi_k)$:

$$g(w_k, \xi_k) = \begin{cases} \nabla_w f(w_k, \xi_k) \\ \dfrac{1}{n_k} \displaystyle\sum_{i=1}^{n_k} \nabla_w f(w_k, \xi_{k,i}) \\ H_k \dfrac{1}{n_k} \displaystyle\sum_{i=1}^{n_k} \nabla_w f(w_k, \xi_{k,i}) \end{cases} \tag{112}$$

where H_k is a symmetric positive definite scaling matrix and the third choice represents a stochastic Newton or quasi-Newton direction.

Two Fundamental Lemmas

Before establishing the convergence guarantees for SG, we need to make an assumption of smoothness of the objective function:

Assumption (Lipschitz-continuous objective gradients)

The objective function $F : \mathbb{R}^d \to \mathbb{R}$ is continuously differentiable and the gradient $\nabla F : \mathbb{R}^d \to \mathbb{R}^d$, is Lipschitz continuous with Lipschitz constant $L > 0$, i.e.,

$$\|\nabla F(w) - \nabla F(\bar{w})\|_2 \leqslant L\|w - \bar{w}\|_2, \quad \forall w, \bar{w} \in \mathbb{R}^d. \tag{113}$$

Two Fundamental Lemmas

Under the above assumption, we obtain the following lemma.

引理

Under Assumption (113)*, the iterates of SG (Algorithm 1) satisfy the following inequality for all* $k \in \mathbb{N}$*:*

$$\begin{aligned} E_{\xi_k}\left[F(w_{k+1})\right] - F(w_k) \leqslant & -\alpha_k \nabla F(w_k)^\top E_{\xi_k}\left[g(w_k, \xi_k)\right] \\ & + \frac{\alpha_k^2 L}{2} E_{\xi_k}\left[\|g(w_k, \xi_k)\|_2^2\right]. \end{aligned} \tag{114}$$

Noting that w_{k+1} *but not* w_k *depends on* ξ_k*, we can derive this equation immediately by simply applying the second-order expansion of* $F(w_{k+1}) - F(w_k)$ *and the assumption* (113) *then taking expectations.*

Two Fundamental Lemmas

To get further, we need another assumption about the first and second moments of the stochastic vectors $\{g(w_k, \xi_k)\}$.

Assumption (First and second moment limits)

The objective function and SG satisfy the following:

(a) *The sequence of iterates $\{w_k\}$ is contained in an open set over which F is bounded below by a scalar F_{inf}.*

(b) *There exist scalars $\mu_G \geqslant \mu > 0$ such that, for all $k \in \mathbb{N}$,*

$$\nabla F(w_k)^\top E_{\xi_k}\left[g(w_k, \xi_k)\right] \geqslant \mu \left\|\nabla F(w_k)\right\|_2^2 \text{ and} \tag{115a}$$

$$\left\|E_{\xi_k}\left[g(w_k, \xi_k)\right]\right\|_2 \leqslant \mu_G \left\|\nabla F(w_k)\right\|_2. \tag{115b}$$

(c) *There exist scalars $M \geqslant 0$ and $M_V \geqslant 0$ such that, for all $k \in \mathbb{N}$,*

$$Var_{\xi_k}\left[\|g(w_k, \xi_k)\|\right] \leqslant M + M_V \|\nabla F(w_k)\|_2^2. \tag{116}$$

Two Fundamental Lemmas

By the definition of variance, it requires that the second moment of $g(w_k, \xi_k)$ satisfies

$$E_{\xi_k}\left[\|g(w_k,\xi_k)\|_2^2\right] \leqslant M + M_G\|\nabla F(w_k)\|_2^2 \text{ with } M_G := M_V + \mu_G^2 \geqslant \mu^2 > 0. \tag{117}$$

引理

Under the above two assumptions, the iterates of SG satisfy the following inequalities for all $k \in \mathbb{N}$:

$$E_{\xi_k}\left[F(w_{k+1})\right] - F(w_k) \leqslant -\mu\alpha_k\|\nabla F(w_k)\|_2^2 + \frac{\alpha_k^2 L}{2} E_{\xi_k}[\|g(w_k,\xi_k)\|_2^2] \tag{118a}$$

$$\leqslant -(\mu\alpha_k - \frac{\alpha_k^2 L}{2} M_G)\|\nabla F(w_k)\|_2^2 + \frac{\alpha_k^2 L}{2} M. \tag{118b}$$

SG for Strongly Convex Objectives

The most benign setting for analyzing the SG method is in the context of minimizing a strongly convex objective function. We formalize a strong convexity assumption as the following.

Assumption (Strong convexity)

The objective function $F : \mathbb{R}^d \to \mathbb{R}$ *is strongly convex in that there exists a constant* $c > 0$ *such that*

$$F(\bar{w}) \geqslant F(w) + \nabla F(w)^\top (\bar{w} - w) + \frac{c}{2}\|\bar{w} - w\|_2^2, \ \forall (\bar{w}, w) \in \mathbb{R}^d \times \mathbb{R}^d \quad (119)$$

Hence, F *has a unique minimizer, denoted as* $w^* \in \mathbb{R}^d$ *with* $F_* := F(w^*)$.

SG for Strongly Convex Objectives

A useful fact is that, under the above assumption, we can bound the optimality gap at a given point:

$$F(w) - F_* \leqslant \frac{1}{2c} \|\nabla F(w)\|_2^2, \ \ \forall w \in \mathbb{R}^d. \tag{120}$$

Noting that

$$\begin{aligned} F(w) - F_* &\leqslant -\nabla F(w)^\top (w^* - w) - \tfrac{c}{2}\|w^* - w\|_2^2 \\ &= -\|\sqrt{\tfrac{1}{2c}}\nabla F(w) + \sqrt{\tfrac{c}{2}}(w^* - w)\|_2^2 + \tfrac{1}{2c}\|\nabla F(w)\|_2^2 \\ &\leqslant \tfrac{1}{2c}\|\nabla F(w)\|_2^2. \end{aligned}$$

SG for Strongly Convex Objectives

We now state our first convergence theorem for SG.

- We use $E[\cdot]$ to denote an expected value taken with respect to the joint distribution of all random variables.
- For example, since w_k is determined by $\{\xi_1, \xi_2, \ldots, \xi_{k-1}\}$, the *total expectation* of $F(w_k)$ for any $k \in \mathbb{N}$ can be taken as

$$E\left[F(w_k)\right] = E_{\xi_1} E_{\xi_2} \ldots E_{\xi_{k-1}} \left[F(w_k)\right]$$

SG for Strongly Convex Objectives

定理 (Strongly Convex Objective, Fixed Stepsize)

Under the above three assumptions (with $F_{\text{inf}} = F_*$), suppose that the SG method is run with a fixed stepsize, $\alpha_k = \bar{\alpha}$ for all $k \in \mathbb{N}$, satisfying

$$0 < \bar{\alpha} \leqslant \frac{\mu}{LM_G}. \tag{121}$$

Then the expected optimality gap satisfies the following inequality for all $k \in \mathbb{N}$:

$$\begin{aligned} E[F(w_k) - F_*] \leqslant & \frac{\bar{\alpha}LM}{2c\mu} + (1 - \bar{\alpha}c\mu)^k \left(F(w_1) - F_* - \frac{\bar{\alpha}LM}{2c\mu} \right) \\ & \xrightarrow{k\to\infty} \frac{\bar{\alpha}LM}{2c\mu}. \end{aligned} \tag{122}$$

SG for Strongly Convex Objectives

This theorem illustrates the interplay between the stepsizes and bound on the variance of the stochastic directions.

- If the variance of $g(w_k, \xi_k)$ is 0 or if noise is to decay with $\|\nabla F(w_k)\|_2^2$, then we can obtain linear convergence to the optimal value.
- On the other hand, when the gradient computation is noisy, a fixed and small enough stepsize can assure the expected objective values will converge linearly to a neighborhood of the optimal value, but the noise in the gradient estimates prevent further progress.

It's natural to ask if diminishing stepsizes will bring a better result.

SG for Strongly Convex Objectives

定理 (Strongly Convex Objective, Diminishing Stepsizes)

Under the assumptions of Lipschitz-continuous objective gradients, first and second moment limits and strong convexity, suppose that SG method is run with a step size sequence such that, for all $k \in \mathbb{N}$,

$$\alpha_k = \frac{\beta}{\gamma + k} \text{ for some } \beta > \frac{1}{c\mu} \text{ and } \gamma > 0 \text{ such that } \alpha_1 \leqslant \frac{\mu}{LM_G}. \quad (123)$$

Then, for all $k \in \mathbb{N}$, *the expected optimality gap satisfies*

$$E\left[F(w_k) - F_*\right] \leqslant \frac{\nu}{\gamma + k}, \quad (124)$$

where

$$\nu := \max\left\{\frac{\beta^2 LM}{2(\beta c\mu - 1)}, (\gamma + 1)(F(w_1) - F_*)\right\}. \quad (125)$$

SG for General Objectives

Many important machine learning models lead to nonconvex optimization problems. Analyzing the SG method when minimizing nonconvex objectives is more challenging since such functions may possess multiple local minima and other stationary points.

Still, one can provide meaningful guarantees for the SG method in nonconvex settings.

While one cannot bound the expected optimality gap as in the convex case, we can bound the average norm of the gradient of the objective function observed on $\{w_k\}$ visited during the first K iterations.

定理 (Nonconvex Objective, Fixed Stepsize)

Under the assumptions of Lipschitz-continuous objective gradients (113) *and first and second moment limits* (115)-(116), *suppose the SG method is run with a fixed stepsize* $\alpha_k = \bar{\alpha}$ *satisfying*

$$0 < \bar{\alpha} \leqslant \frac{\mu}{LM_G}. \tag{126}$$

Then the expected sum-of-squares and average-squared gradients of F *corresponding to the SG iterates satisfy the following inequalities for all* $K \in \mathbb{N}$*:*

$$E\left[\sum_{k=1}^{K} \|\nabla F(w_k)\|_2^2\right] \leqslant \frac{K\bar{\alpha}LM}{\mu} + \frac{2(F(w_1) - F_{\text{inf}})}{\mu\bar{\alpha}} \tag{127a}$$

$$\text{and therefore } E\left[\frac{1}{K}\sum_{k=1}^{K} \|\nabla F(w_k)\|_2^2\right] \leqslant \frac{\bar{\alpha}LM}{\mu} + \frac{2(F(w_1) - F_{\text{inf}})}{K\mu\bar{\alpha}} \tag{127b}$$

$$\xrightarrow{K\to\infty} \frac{\bar{\alpha}LM}{\mu}.$$

Proof.

Taking the total expectation of (118b) and from (126),

$$E\left[F(w_{k+1})\right] - E[F(w_k)] \leqslant -(\mu - \frac{\bar{\alpha}LM_G}{2})\bar{\alpha}E\left[\|\nabla F(x_k)\|_2^2\right] + \frac{\bar{\alpha}^2 LM}{2}$$
$$\leqslant -\frac{\mu\bar{\alpha}}{2}E\left[\|\nabla F(w_k)\|_2^2\right] + \frac{\bar{\alpha}^2 LM}{2}.$$

Summing both sides of this inequality for $k \in \{1, \ldots, K\}$ and recalling (a) of the assumption on first and second moment limits gives

$$F_{\text{inf}} - F(w_1) \leqslant E\left[F(w_{K+1})\right] - F(w_1) \leqslant -\frac{\mu\bar{\alpha}}{2}\sum_{k=1}^{K} E\left[\|\nabla F(w_k)\|_2^2\right] + \frac{K\bar{\alpha}^2 LM}{2}.$$

Rearranging yields (127a), and dividing further by K yields (127b). □

SG for General Objectives

定理 (Nonconvex Objective, Diminishing Stepsize)

Under the assumptions of Lipschitz-continuous objective gradients (113) *and first and second moment limits* (115)-(116), *suppose that the SG method is run with a stepsize sequence satisfying*

$$\sum_{k=1}^{\infty} \alpha_k = \infty, \quad \sum_{k=1}^{\infty} \alpha_k^2 < \infty. \tag{128}$$

Then

$$\liminf_{k \to \infty} \left(E \left[\|\nabla F(w_k)\|_2^2 \right] \right) = 0. \tag{129}$$

While not the strongest result in this context, this theorem is perhaps the easiest to interpret and remember. The proof of this theorem follows based on the stronger results given in the next theorem.

SG for General Objectives

定理 (Nonconvex Objective, Diminishing Stepsize)

Under the assumptions of Lipschitz-continuous objective gradients (113) *and first and second moment limits* (115)-(116), *suppose that the SG method is run with a stepsize sequence satisfying* (128). *Then, with* $A_K := \sum_{k=1}^K \alpha_k$,

$$E\left[\sum_{k=1}^{K} \alpha_k \|\nabla F(w_k)\|_2^2\right] < \infty \tag{130a}$$

$$\text{and therefore } E\left[\frac{1}{A_K}\sum_{k=1}^{K} \alpha_k \|\nabla F(w_k)\|_2^2\right] \xrightarrow{K\to\infty} 0. \tag{130b}$$

推论

Suppose the conditions of the last theorem hold. For any $K \in \mathbb{N}$, *let* $k(K) \in \{1, \ldots, K\}$ *represent a random index chosen with probabilities proportional to* $\{\alpha_k\}_{k=1}^K$. *Then* $\|\nabla F(w_{k(K)})\|_2 \xrightarrow{K\to\infty} 0$ *in probability.*

推论

Under the conditions of the last theorem, if we further assume that the objective function F *is twice differentiable, and that the mapping* $w \mapsto \|\nabla F(w)\|_2^2$ *has Lipschitz-continuous derivatives, then*

$$\lim_{k\to\infty} E\left[\|\nabla F(w_k)\|_2^2\right] = 0.$$

Noise Reduction Methods

SG suffers from the adverse effect of noisy gradient estimates. To address this limitation, methods endowed with *noise reduction* capabilities have been developed.

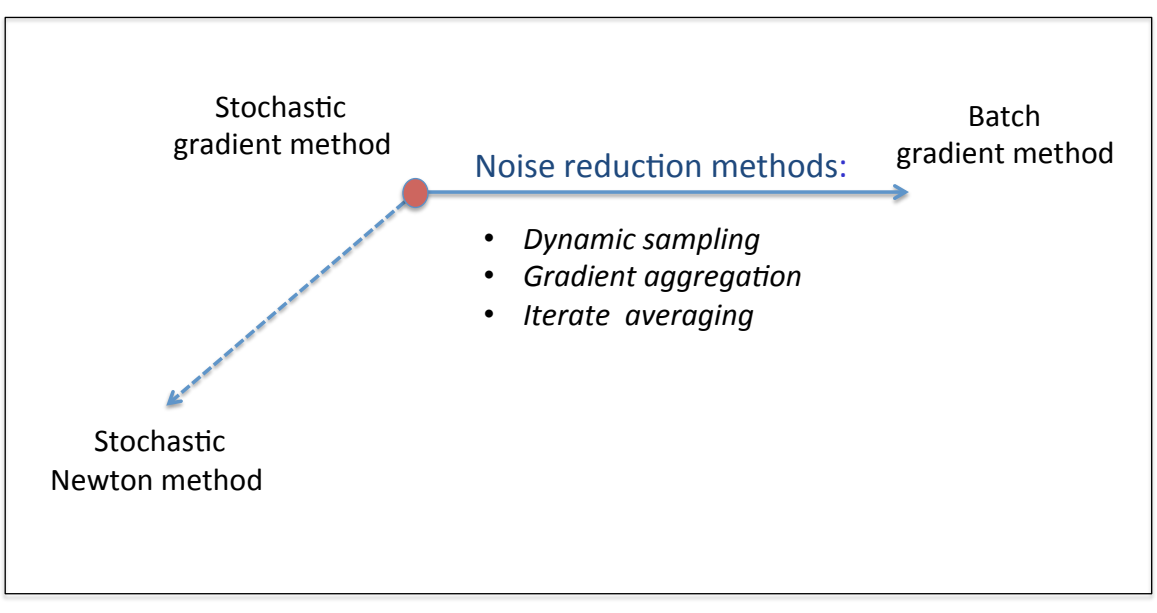


Figure: View of the schematic with a focus on noise reduction methods.

Noise Reduction Methods

The first two classes of methods achieve noise reduction in a manner that allows them to possess a linear rate of convergence to the optimal value using a fixed stepsize. The third class of methods employing a stepsize sequence of order $\mathcal{O}(1/\sqrt{k})$ rather than $\mathcal{O}(1/k)$.

- **Dynamic sampling methods** achieve noise reduction by gradually increasing the mini-batch size used in the gradient computation.
- **Gradient aggregation methods** improve the quality of the search directions by storing gradient estimates in previous iterations, updating one (or some) of these estimates in each iteration, and defining the search direction as a weighted average of these estimates.
- **Iterate averaging methods** accomplish noise reduction by maintaining an average of iterates computed during the optimization process.

Dynamic Sampling Methods

Recall the first lemma in this section

$$E_{\xi_k}\left[F(w_{k+1})\right] - F(w_k) \leqslant -\alpha_k \nabla F(w_k)^\top E_{\xi_k}\left[g(w_k, \xi_k)\right] + \frac{\alpha_k^2 L}{2} E_{\xi_k}\left[\|g(w_k, \xi_k)\|_2^2\right].$$

If we are able to decrease $E_{\xi_k}[\|g(w_k, \xi_k)\|_2^2]$ fast enough, then the noise will not prevent the convergence.

We'll show that the sequence of expected optimality gaps vanishes at a linear rate as long as the variance of the stochastic vectors, denoted by $Var_{\xi_k}\left[g(w_k, \xi_k)\right]$, decreases geometrically.

定理 (Strongly Convex Objective, Noise Reduction)

Under the assumptions of Lipschitz-continuous objective gradients and first and second moment limits and strong convexity, but with (116) *refined to the existence of constants* $M \geqslant 0$ *and* $\zeta \in (0, 1)$ *such that*

$$Var_{\xi_k}[g(w_k, \xi_k)] \leqslant M\zeta^{k-1}, \quad \forall k \in \mathbb{N}. \tag{131}$$

In addition, suppose that the SG method is run with a fixed stepsize, $\alpha_k = \bar{\alpha}$ *satisfying*

$$0 < \bar{\alpha} \leqslant \min\left\{\frac{\mu}{L\mu_G^2}, \frac{1}{c\mu}\right\}. \tag{132}$$

定理 (Strongly Convex Objective, Noise Reduction)

Then the expected optimality gap satisfies

$$E\left[F(w_k) - F_*\right] \leqslant \omega \rho^{k-1}, \tag{133}$$

where

$$\omega := \max\{\frac{\bar{\alpha} LM}{c\mu}, F(w_1) - F_*\} \text{ and } \rho := \max\{1 - \frac{\bar{\alpha} c\mu}{2}, \zeta\} < 1.$$

The restriction on the stepsize $\bar{\alpha}$ is not unrealistic in practical situations, considering the typical magnitudes of the constants μ, L, μ_G and c.

Now a natural question is how to design efficient optimization methods for attaining the critical bound (131) on the variance of the stochastic directions.

Dynamic Sampling Methods

Consider the iteration

$$w_{k+1} \leftarrow w_k - \bar{\alpha} g(w_k, \xi_k), \tag{134}$$

where the stochastic directions are computed for some $\tau > 1$ as

$$g(w_k, \xi_k) := \frac{1}{n_k} \sum_{i \in \mathcal{S}_k} \nabla f(w_k; \xi_{k,i}) \text{ with } n_k := |\mathcal{S}_k| = \lceil \tau^{k-1} \rceil. \tag{135}$$

That is, a mini-batch SG iteration with a fixed stepsize in which the mini-batch size increases geometrically as a function of the iteration counter k.

Dynamic Sampling Methods

If we assume that each stochastic gradient $\nabla f(w_k; \xi_{k,i})$ has an expectation equal to the true gradient $\nabla F(w_k)$, then (115) holds with $\mu_G = \mu = 1$. If, in addition, the variance of each such stochastic gradient is equal and is bounded by $M \geqslant 0$, then for arbitrary $i \in \mathcal{S}_k$ we have

$$Var_{\xi_k}\left[g(w_k, \xi_k)\right] \leqslant \frac{Var_{\xi_k}\left[\nabla f(w_k; \xi_{k,i})\right]}{n_k} \leqslant \frac{M}{n_k}. \tag{136}$$

This bound combined with the rate of increase in n_k given in (135) yields (131). We state these formally as the following corollary.

推论

Let $\{w_k\}$ be the iterates generated by (134)-(135) *with $E_{\xi_{k,i}}[\nabla f(w_k; \xi_{k,i})] = \nabla F(w_k)$, $\forall k \in \mathbb{N}, i \in \mathcal{S}_k$. Then, the variance condition* (131) *is satisfied and if all other assumptions of the theorem of noise reduction for strongly convex objective holds, then the expected optimality gap vanishes linearly in the sense of* (133).

Dynamic Sampling Methods

But, comparing to classical SG approach, is it meaningful to describe a method as linearly convergent if the per-iteration cost increases without bound?

To address this question, let's estimate the number of evaluations of the individual gradients $\nabla f(w_k, \xi_{k,i})$ required to compute an ϵ-optimal solution, i.e., to achieve

$$E\left[F(w_k) - F_*\right] \leqslant \epsilon. \tag{137}$$

As previously mentioned, the number of stochastic gradient evaluations required by the SG method to guarantee (137) is $\mathcal{O}(\epsilon^{-1})$.

Dynamic Sampling Methods

定理

Suppose the dynamic sampling SG method (134)-(135) *is run with a stepsize* $\bar{\alpha}$ *satisfying* (132) *and some* $\tau \in (1, (1 - \frac{\bar{\alpha} c \mu}{2})^{-1}]$. *In addition, suppose that the three assumptions hold. Then the total number of evaluations of a stochastic gradient of the form* $\nabla f(w_k, \xi_{k,i})$ *required to obtain* (137) *is* $\mathcal{O}(\epsilon^{-1})$.

Gradient Aggregation Methods

Rather than compute increasingly more *new* stochastic gradient information in each iteration, *gradient aggregation methods* achieve a lower variance by *reusing* and/or *revising* previously computed information.

If the current iterate has not been displaced too far from previous iterates, then stochastic gradient information from previous iterates may still be useful.

Gradient Aggregation Methods

The first method we consider is composed of outer and inner iterations.

At each step of outer iteration, an iterate w_k is available at which the algorithm computes a batch gradient $\nabla R_n(w_k) = \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(w_k)$.

Then, after initializing $\tilde{w}_1 \leftarrow w_k$, m inner iterations indexed by j are performed:

$$\tilde{w}_{j+1} \leftarrow \tilde{w}_j - \alpha \tilde{g}_j$$

where

$$\tilde{g}_j \leftarrow \nabla f_{i_j}(\tilde{w}_j) - \left(\nabla f_{i_j}(w_k) - \nabla R_n(w_k)\right) \tag{138}$$

and $i_j \in \{1, \ldots, n\}$ is chosen at random.

Gradient Aggregation Methods

Interpretation:

Since $E_{i_j}\left[\nabla f_{i_j}(w_k)\right] = \nabla R_n(w_k)$, we can view $\nabla f_{i_j}(w_k) - \nabla R_n(w_k)$ as the bias in the gradient estimate $\nabla f_{i_j}(w_k)$. Thus the stochastic gradient $\nabla f_{i_j}(\tilde{w}_j)$ evaluated at the current inner iterate $\tilde{w}_j$ is corrected based on a perceived bias.

Overall, $\tilde{g}_j$ represents an unbiased estimator of $\nabla R_n(\tilde{w}_j)$, with a smaller variance than simply choosing $\nabla f_{i_j}(\tilde{w}_j)$ (as in simple SG). This is the reason why the method is referred to as the *stochastic variance reduced gradient* (SVRG) method.

Gradient Aggregation Methods

Algorithm 2 SVRG Methods for Minimizing an Empirical Risk R_n

Choose an initial iterate $w_1 \in \mathbb{R}^d$, stepsize $\alpha > 0$, positive integer m.
for $k = 1, 2, \ldots$ **do**
  Compute the batch gradient $\nabla R_n(w_k)$.
  Initialize $\tilde{w}_1 \leftarrow w_k$.
  for $j = 1, \ldots, m$ **do**
    Choose i_j uniformly from $\{1, \ldots, n\}$.
    $\tilde{g}_j \leftarrow \nabla f_{i_j}(\tilde{w}_j) - (\nabla f_{i_j}(w_k) - \nabla R_n(w_k))$.
    Set $\tilde{w}_{j+1} \leftarrow \tilde{w}_j - \alpha \tilde{g}_j$.
  end for
  Option (a): Set $w_{k+1} = \tilde{w}_{m+1}$
  Option (b): Set $w_{k+1} = \frac{1}{m} \sum_{j=1}^{m} \tilde{w}_{j+1}$
  Option (c): Choose j uniformly from $\{1, \ldots, m\}$ and set $w_{k+1} = \tilde{w}_{j+1}$.
end for

Gradient Aggregation Methods

For both options (b) and (c), it can achieve a linear rate of convergence when R_n is strongly convex.

> More precisely, if the stepsize α and the length of the inner cycle m are chosen so that
>
> $$\rho := \frac{1}{1-2\alpha L}\left(\frac{1}{mc\alpha} + 2L\alpha\right) < 1,$$
>
> then, given that the algorithm has reached w_k, one obtains
>
> $$E_{i_j}\left[R_n(w_{k+1}) - R_n(w_*)\right] \leqslant \rho E_{i_j}\left[R_n(w_\kappa) - R_n(w_*)\right].$$

Each step (of outer iteration) requires $2m + n$ evaluations of component gradients, which is much more expensive than one in SG, and in fact is comparable to a full gradient iteration.

Gradient Aggregation Methods

The second method does not include inner loop nor does it compute batch gradients (except possibly at the initial point).

Instead, in each iteration, it computes a stochastic vector g_k as the average of stochastic gradients evaluated at previous iterates.

Gradient Aggregation Methods

Specifically, in iteration k, the method will have stored $\nabla f_i(w_{[i]})$ for all $i \in \{1, \ldots, n\}$ where $w_{[i]}$ represents the latest iterate at which ∇f_i was evaluated. An integer $j \in \{1, \ldots, n\}$ is then chosen at random and the stochastic vector is set by

$$g_k \leftarrow \nabla f_j(w_k) - \nabla f_j(w_{[j]}) + \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(w_{[i]}). \tag{139}$$

Taking the expectation of g_k w.r.t. all choices of $j \in \{1, \ldots, n\}$, we have $E[g_k] = \nabla R_n(w_k)$. Thus the gradient estimates is unbiased with variances that are expected to be less than the stochastic gradients in a basic SG.

Algorithm 3 SAGA Methods for Minimizing an Empirical Risk R_n

Choose an initial iterate $w_1 \in \mathbb{R}^d$ and stepsize $\alpha > 0$.
for $k = 1, 2, \ldots$ **do**
  Compute $\nabla f_i(w_1)$.
  Store $\nabla f_i(w_{[i]}) \leftarrow \nabla f_i(w_1)$.
end for
for $k = 1, 2, \ldots$ **do**
  Choose j uniformly in $\{1, \ldots, n\}$.
  Compute $\nabla f_j(w_k)$.
  Set $g_k \leftarrow \nabla f_j(w_k) - \nabla f_j(w_{[j]}) + \frac{1}{n} \sum_{i=1}^n \nabla f_i(w_{[i]})$.
  Store $\nabla f_j(w_{[j]}) \leftarrow \nabla f_j(w_k)$.
  Set $w_{k+1} \leftarrow w_k - \alpha g_k$.
end for

Gradient Aggregation Methods

Beyond its initialization phase, the per-iteration cost of it is the same as in a basic SG method. However, it can achieve a linear rate of convergence when R_n is strongly convex. With $\alpha = 1/(2(cn + L))$, we have

$$E\left[\|w_k - w_*\|_2^2\right] \leqslant \left(1 - \frac{c}{2(cn+L)}\right)^k \left(\|w_1 - w_*\|_2^2 + \frac{nD}{cn+L}\right)$$

where $D := R_n(w_1) - R_n(w_*) - \nabla R_n(w_*)^\top (w_1 - w_*)$.

Alternative initialization techniques could be used in practice. For example, one could perform one epoch of simple SG, or assimilate iterates one-by-one and compute g_k only using the gradients available up to that point.

Gradient Aggregation Methods

One important drawback of Algorithm 3 is the need to store n stochastic gradient vectors. Note, however, that if the component functions are of the form $f_i(w_k) = \hat{f}(x_i^\top w_k)$, then

$$\nabla f_i(w_k) = \hat{f}'(x_i^\top w_k) x_i.$$

That is, when the feature vectors $\{x_i\}$ are already available in storage, one need only store the scalar $\hat{f}'(x_i^\top w_k)$ to construct $\nabla f_i(w_k)$ at a later iteration. This occurs in logistic and least squares regression.

Gradient Aggregation Methods

Although the gradient aggregation methods above enjoy a faster rate of convergence than SG, they should not be regarded as clearly superior to SG.

Following similar analysis as before, the computing time for SG can be shown to be $\mathcal{T}(n, \epsilon) \sim \kappa^2/\epsilon$ with $\kappa := L/c$. On the other hand, the computing times for SVRG and SAGA are $\mathcal{T}(n, \epsilon) \sim (n + \kappa) \log(1/\epsilon)$.

For very large n, gradient aggregation methods are comparable to batch algorithms and therefore cannot beat SG in this regime.

Iterate Averaging Methods

SG generates noisy iterate sequences that tend to oscillate around minimizers. Hence, a natural idea is to compute a corresponding sequence of *iterate averages* that would automatically possess less noisy behavior.

Specifically, for minimizing a continuously differentiable F with unbiased gradient estimates, it employs the iteration

$$w_{k+1} \leftarrow w_k - \alpha_k g(w_k, \xi_k)$$
$$\text{and } \tilde{w}_{k+1} \leftarrow \frac{1}{k+1} \sum_{j=1}^{k+1} w_j. \tag{140}$$

Iterate Averaging Methods

However, convergence properties better than SG of this method is elusive when using classical stepsize sequences that diminish with a rate of $\mathcal{O}(1/k)$.

An idea is to employ the iteration (140) but with stepsizes diminishing at a slower rate of $\mathcal{O}(1/(k^a))$ for some $a \in (\frac{1}{2}, 1)$. When minimizing strongly convex objectives, it follows from this choice that

$$E[\|w_k - w_*\|_2^2] = \mathcal{O}(1/(k^a)) \text{ while } E[\|\tilde{w}_k - w_*\|_2^2] = \mathcal{O}(1/k).$$

Second-Order Methods

Besides reducing the noise in the stochastic directions, another manner to move beyond classical SG is to address the adverse effects of high nonlinearity and ill-conditioning of the objective function through the use of second-order information.

Deterministic methods are known to benefit from the use of second-order information, e.g., Newton's method achieves a locally quadratic convergence.

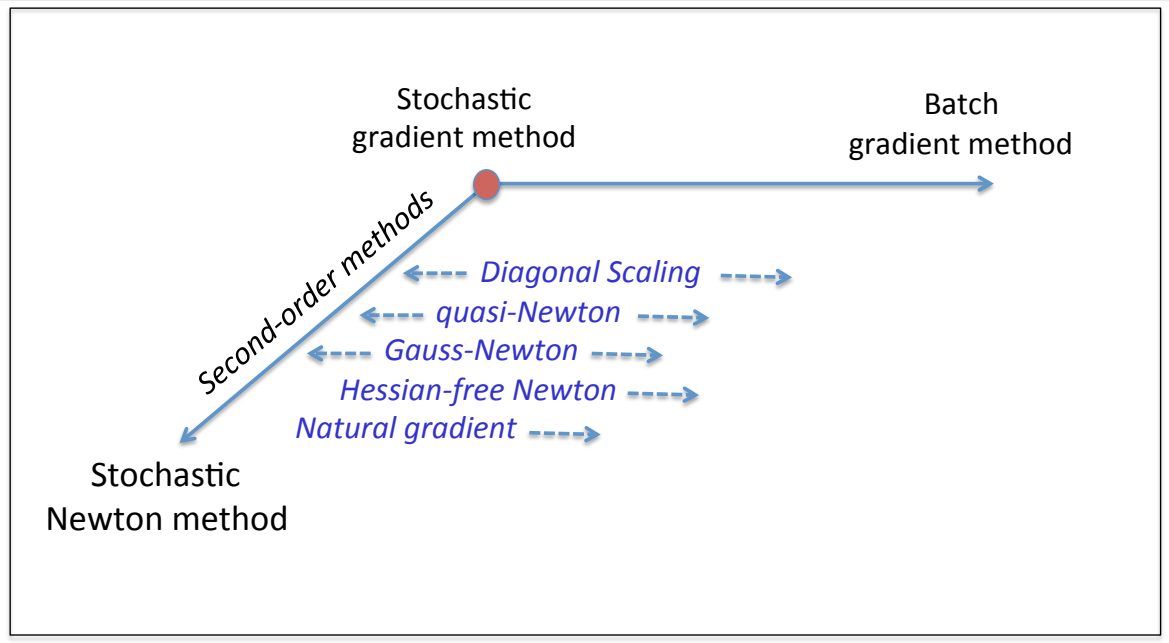
Second-Order Methods

We start by considering a *Hessian-free Newton* method that employs exact second-order information in a judicious manner that exploits the stochastic nature of the objective function.

Then we describe methods that attempt to mimic the behavior of a Newton algorithm through first-order information computed over sequences of iterates, including *quasi-Newton, Gauss-Newton* and related algorithms that employ only diagonal re-scalings.

Finally we will sketch the *natural gradient* method.

Second-Order Methods



We use double-sided arrows for the methods that can be effective throughout the spectrum between the stochastic and batch regimes. Single-sides arrows are used for those methods that are effective only with at least a moderate batch size in the stochastic gradient estimates.

Hessian-Free Inexact Newton Methods

When minimizing a twice-continuously differentiable F, a Newton iteration is

$$w_{k+1} \leftarrow w_k + \alpha_k s_k \tag{141a}$$

$$\text{where } \nabla^2 F(w_k) s_k = -\nabla F(w_k). \tag{141b}$$

This iteration demands much in terms of computation and storage. However, we can instead only solve (141b) inexactly through an iterative approach such as the conjugate gradient (CG) method.

> By ensuring that the linear solves are accurate enough, such an *inexact Newton-CG* method can enjoy a superlinear convergence.

CG applied to (141b) does not require access to the Hessian itself, only Hessian-vector products. Such a method may be called *Hessian-free*.

Subsampled Hessian-Free Newton Methods

In inexact Newton methods, the Hessian matrix need not be as accurate as the gradient to yield an effective iteration. It means that the iteration is more tolerant to noise in the Hessian estimate than it is to noise in the gradient estimate.

We employ a smaller sample for defining the Hessian than for the stochastic gradient estimate. Let the stochastic gradient estimate be

$$\nabla f_{\mathcal{S}_k}(w_k, \xi_k) = \frac{1}{|\mathcal{S}_k|} \sum_{i \in \mathcal{S}_k} \nabla f(w_k, \xi_{k,i})$$

and let the stochastic Hessian estimate be

$$\nabla^2 f_{\mathcal{S}_k^H}(w_k, \xi_k^H) = \frac{1}{|\mathcal{S}_k^H|} \sum_{i \in \mathcal{S}_k^H} \nabla^2 f(w_k, \xi_{k,i}) \tag{142}$$

where $\mathcal{S}_k^H \subseteq \mathcal{S}_k$.

Subsampled Hessian-Free Newton Methods

If the subsample size $|\mathcal{S}_k^H|$ small enough, then the cost of each product involving the Hessian approximation can be reduced significantly, thus reducing the cost of each CG iteration.

On the other hand, one should choose $|\mathcal{S}_k^H|$ large enough so that the curvature information captured through the Hessian-vector products is productive.

Subsampled Hessian-Free Newton Methods

Algorithm 4 Subsampled Hessian-Free Inexact Newton Method

1: Choose an initial iterate w_1.
2: Choose constants $\rho \in (0,1), \eta \in (0,1)$, and $\max_{cg} \in \mathbb{N}$.
3: **for** $k = 1, 2, \ldots$ **do**
4: Generate a realizations of ξ_k and ξ_k^H corresponding to $\mathcal{S}_k^H \subseteq \mathcal{S}_k$.
5: Compute s_k by applying Hessian-free CG to solve

$$\nabla^2 f_{\mathcal{S}_k^H}(w_k, \xi_k^H) s = -\nabla f_{\mathcal{S}_k}(w_k, \xi_k) \tag{143}$$

until $\max_{cg}$ iterations have been performed or a trial solution yields

$$\|r_k\|_2 := \left\| \nabla^2 f_{\mathcal{S}_k^H}(w_k, \xi_k^H) s + \nabla f_{\mathcal{S}_k}(w_k, \xi_k) \right\|_2 \leqslant \rho \left\| \nabla f_{\mathcal{S}_k}(w_k, \xi_k) \right\|_2 .$$

6: Set $w_{k+1} \leftarrow w_k + \alpha_k s_k$, where $\alpha_k \in \{\gamma^0, \gamma^1, \gamma^2, \ldots\}$ is the largest element with

$$f_{\mathcal{S}_k}(w_{k+1}, \xi_k) \leqslant f_{\mathcal{S}_k}(w_k, \xi_k) + \eta \alpha_k \nabla f_{\mathcal{S}_k}(w_k, \xi_k)^\top s_k. \tag{144}$$

7: **end for**

Subsampled Hessian-Free Newton Methods

If the algorithm were to operate in the stochastic regime of SG where $|\mathcal{S}_k|$ is small and gradients are very noisy, then it may be necessary to choose $|\mathcal{S}_k^H| > |\mathcal{S}_k|$ so that Hessian approximations do not corrupt the step.

> Therefore, the subsampled Hessian-free Newton method outlined here is only recommended when $\mathcal{S}_k$ is large.

When full gradients are always used, it's easy to establish the convergence of Algorithm 4 for minimizing a strongly convex empirical risk measure $F = R_n$ with $\mathcal{S}_k^H = \mathcal{S}_k = \{1, \ldots, n\}$.

When the Hessians are subsampled, it has not been shown that the rate of convergence is faster than linear.

Dealing with Nonconvexity

When Hessian-free Newton methods are applied for the solution of nonconvex problems, it's common to employ a *trust region* instead of a line search and to add an additional condition in Step 5 of Algorithm 4: terminate CG if a candidate solution s_k is a direction of negative curvature, i.e., $s_k^\top \nabla^2 f_{\mathcal{S}_k^H}(w_k; \xi_k^H) s_k < 0$.

Instead of coping with indefiniteness, one can focus on strategies for ensuring positive (semi)definite Hessian approximations. One of the most attractive ways of doing this in the context of machine learning is to employ a (subsampled) Gauss-Newton approximation to the Hessian, which we will explain later.

Stochastic Quasi-Newton Methods

The quasi-Newton iteration for minimizing a twice continuously differentiable function F has the form

$$w_{k+1} \leftarrow w_k - \alpha_k H_k \nabla F(w_k), \tag{145}$$

where H_k is a approximation of $\left(\nabla^2 F(w_k)\right)^{-1}$. The most popular quasi-Newton scheme is BFGS.

In BFGS, the sequence $\{H_k\}$ is updated dynamically, without the need for second-order derivative computations nor any linear system solves. It enjoys a local superlinear convergence with only first-order information.

But H_k is often a dense matrix, even when the exact Hessian is sparse, restricting its use to small and midsize problems. A common solution for this is to employ a *limited memory scheme*, leading to a method such as L-BFGS. In this case, H_k need not be formed explicitly.

Stochastic Quasi-Newton Methods

Now we consider the iterations taking the form

$$w_{k+1} \leftarrow w_k - \alpha_k H_k g(w_k, \xi_k). \tag{146}$$

Since we are interested in large-scale problems, we assume that (146) implements an L-BFGS scheme. A number of questions arise when considering (146), and we list them now with some proposed solutions:

Stochastic Quasi-Newton Methods

Theoretical Limitations The convergence rate of a stochastic iteration such as (146) cannot be faster than sublinear. Since SG also has a sublinear rate of convergence, what benefit could come from incorporating H_k in (146)?

Benefit: The constant that appears in the sublinear rate.

For SG, the constant depends on the conditioning of $\{\nabla^2 F(w_k)\}$. This is typical of first-order methods. In contrast, if the sequence of Hessian approximations in (146) satisfies $\{H_k\} \to \nabla^2 F(w_*)^{-1}$, then the constant is independent of the conditioning of the Hessian.

Stochastic Quasi-Newton Methods

Additional Per-Iteration Costs The product $H_k g(w_k, \xi_k)$ requires $4md$ operations where m is the memory in the L-BFGS updating scheme. Assuming the cost of evaluating $g(w_k, \xi_k)$ is exactly d operations (using only one sample) and m is set to the typical value of 5, then the stochastic quasi-Newton method is 20 times more expensive than SG. Can we offset this additional per-iteration cost?

When employing mini-batch gradient estimates, the additional cost of the iteration (146) is only marginal. The use of mini-batches may be considered essential. Mini-batch should not be less than, say, 20 or 50, and mini-batches of size 256 are common in practice.

Stochastic Quasi-Newton Methods

Conditioning of the Scaling Matrices Updating H_k involves differences in gradient estimates computed in consecutive iterations. $\{g(w_k, \xi_k)\}$ are noisy estimates of $\{\nabla F(w_k)\}$, which can cause the updating process to yield poor curvature estimates. How could such effects be avoided in the stochastic regime?

One possibility is to employ the same sample when computing gradient differences. An alternative approach is to *decouple* the step computation and the Hessian update.

Stochastic Quasi-Newton Methods

Replacing deterministic gradients with stochastic gradients, we have

$$s_k := w_{k+1} - w_k \text{ and } v_k := \nabla f_{\mathcal{S}_k}(w_{k+1}, \xi_k) - \nabla f_{\mathcal{S}_k}(w_k, \xi_k). \tag{147}$$

and H_k is defined recursively by

$$H_{k+1} \leftarrow \left(I - \frac{v_k s_k^\top}{s_k^\top v_k}\right)^\top H_k \left(I - \frac{v_k s_k^\top}{s_k^\top v_k}\right) + \frac{s_k s_k^\top}{s_k^\top v_k}.$$

Note that the use of the same realization ξ_k in the two gradient estimates, in order to address the issues related to noise mentioned above.

Stochastic Quasi-Newton Methods

A worrisome feature is that updating the inverse Hessian approximation with *every* step may not be warranted and could easily represent a poor approximation of the action of the true Hessian of F.

Here's an alternative strategy for this issue. Since $\nabla f(w_{k+1}) - \nabla F(w_k) \approx \nabla^2 F(w_k)(w_{k+1} - w_k)$, we can define

$$v_k := \nabla^2 f_{\mathcal{S}_k^H}(w_k, \xi_k^H) s_k, \tag{148}$$

where $\nabla^2 f_{\mathcal{S}_k^H}(w_k; \xi_k^H)$ is a subsampled Hessian and $|\mathcal{S}_k^H|$ is large enough to provide useful curvature information.

When $|\mathcal{S}_k^H|$ is much larger than $|\mathcal{S}_k|$, the computation of v_k can be performed only after a sequence of iterations, to amortize the cost of quasi-Newton updating.

Stochastic Quasi-Newton Methods

This leads to the idea of decoupling the step computation from the quasi-Newton update. This approach, which we refer to as SQN, performs a sequence of iterations of (146) with H_k fixed, then computes a new displacement pair (s_k, v_k) with s_k defined as in (147) and v_k set using one of the strategies outlined above.

To formalize all of these alternatives, we state the general stochastic quasi-Newton method presented as Algorithm 5.

Algorithm 5 Stochastic Quasi-Newton Framework

Choose an initial iterate w_1 and initialize $\mathcal{P} \leftarrow \emptyset$.
Choose a constant $m \in \mathbb{N}$.
Choose a stepsize sequence $\{\alpha_k\} \subset \mathbb{R}_{++}$.
for $k = 1, 2, \ldots,$ **do**
  Generate realizations of ξ_k and ξ_k^H corresponding to $\mathcal{S}_k^H \subseteq \mathcal{S}_k$
  Compute $\hat{s}_k = H_k g(w_k, \xi_k)$ using the two-loop recursion based on the set $\mathcal{P}$.
  Set $s_k \leftarrow -\alpha_k \hat{s}_k$.
  Set $w_{k+1} \leftarrow w_k + s_k$.
  if update pairs **then**
    Compute s_k and v_k (based on the sample $\mathcal{S}_k^H$).
    Add the new displacement pair (s_k, v_k) to $\mathcal{P}$.
    If $|\mathcal{P}| > m$, then remove eldest pair from $\mathcal{P}$.
  end if
end for

Gauss-Newton Methods

The Gauss-Newton method is a classical approach for nonlinear least squares. It constructs an approximation to the Hessian using only first-order information, and this approximation is guaranteed to be positive semidefinite, even when the full Hessian itself may be indefinite.

Gauss-Newton Methods

Given an input-output pair (x_ξ, y_ξ), the loss incurred by a parameter vector w is measured via a squared norm discrepancy between $h(x_\xi, w) \in \mathbb{R}^d$ and $y \in \mathbb{R}^d$:

$$f(w, \xi) = \ell(h(x_\xi, w), y_\xi) = \frac{1}{2} \|h(x_\xi, w) - y_\xi\|_2^2 .$$

Let $J_h(\cdot, \xi)$ represent the Jacobian of $h(x_\xi, \cdot)$ with respect to w. The affine approximation of $h(x_\xi, w)$ is

$$h(x_\xi, w) \approx h(x_\xi, w_k) + J_h(w_k, \xi)(w - w_k),$$

which leads to

$$\begin{aligned} f(w, \xi) \approx & \frac{1}{2} \|h(x_\xi, w_k) + J_h(w_k, \xi)(w - w_k) - y_\xi\|_2^2 \\ = & \frac{1}{2} \|h(x_\xi, w_k) - y_\xi\|_2^2 + (h(x_\xi, w_k) - y_\xi)^\top J_h(w_k, \xi)(w - w_k) \\ & + \frac{1}{2}(w - w_k)^\top J_h(w_k, \xi)^\top J_h(w_k, \xi)(w - w_k). \end{aligned}$$

Gauss-Newton Methods

It is similar to a second-oder Taylor series model, except that the terms involving the second derivatives of h with respect to w have been dropped, and the remaining second-order terms are resulting from the positive curvature of the quadratic loss ℓ.

This leads to replacing the subsample Hessian matrix by the Gauss-Newton matrix

$$G_{\mathcal{S}_k^H}(w_k, \xi_k^H) = \frac{1}{|\mathcal{S}_k^H|} \sum_{i \in \mathcal{S}_k^H} J_h(w_k, \xi_{k,i})^\top J_h(w_k, \xi_{k,i}). \tag{149}$$

Gauss-Newton Methods

A challenge of Gauss-Newton method is that Gauss-Newton matrix is often singular or nearly singular. In practice, this is handled by regularizing it by adding to it a positive multiple of the identity matrix.

The computational cost of the Gauss-Newton method depends on the dimensionality of the prediction function. It should be remarked that in machine learning, computing the stochastic gradient vector $\nabla f(w, \xi)$ does not usually require the explicit computation of all rows of the Jacobian matrix. And there are some new ways to solve a Gauss-Newton iterate at a low cost.

Generalized Gauss-Newton

Consider a slightly more general situation in which loss between a prediction function h and output y is measured by an arbitrary convex loss function $\ell(h, y)$. Combining the affine approximation of the prediction function $h(x_\xi, w)$ with a second order Taylor expansion of the loss function ℓ leads to the generalized Gauss-Newton matrix

$$G_{\mathcal{S}_k^H}(w_k, \xi_k^H) = \frac{1}{|\mathcal{S}_k^H|} \sum_{i \in \mathcal{S}_k^H} J_h(w_k, \xi_{k,i})^\top H_\ell(w_k, \xi_{k,i}) J_h(w_k, \xi_{k,i}) \qquad (150)$$

where $H_\ell(w_k, \xi) = \frac{\partial^2 \ell}{\partial h^2}(h(x_\xi, w_k), y_\xi)$ captures the curvature of the loss function ℓ.

Diagonal Scalings

We have seen that the added per-iteration costs of second-order methods (such as L-BFGS) can be as little as $4md$ operations. A strategy to further reduce this multiplicative factor is to restrict attention to *diagonal* or *block-diagonal* scaling matrices.

The incorporation of a diagonal scaling matrix will only scale the individual search direction components. This can be efficiently achieved by multiplying the individual search direction components.

Computing Diagonal Curvature

A first family of algorithms directly computes the diagonal terms of the Hessian or Gauss-Newton matrix, then divides each coefficient of the stochastic gradient vector $g(w_k, \xi_k)$ by the corresponding diagonal term.

For instance, each iteration of the proposed algorithm picks a training example, computes $g(w_k, \xi_k)$, updates a running estimate of the diagonal coefficients of the Gauss-Newton matrix by

$$[G_k]_i = (1-\lambda)[G_{k-1}]_i + \lambda \left[J_h(w_k, \xi_k)^\top J_h(w_k, \xi_k) \right]_{ii} \text{ for some } 0 < \lambda < 1,$$

Computing Diagonal Curvature

then performs the scaled stochastic weight update

$$[w_{k+1}]_i = [w_k]_i - \left(\frac{\alpha}{[G_k]_i + \mu} \right) [g(w_k, \xi_k)]_i.$$

The small regularization constant $\mu > 0$ is introduced to deal with a singular or nearly singular Gauss-Newton matrix.

> It's more enlightening to view such an algorithm as a scheme to periodically retune a first-order SG approach rather than as a complete second-order method.

Estimating Diagonal Curvature

Instead of explicitly computing the diagonal terms of the curvature matrix, one can follow the template of quasi-Newton method and directly estimate the diagonal $[H_k]_i$ of the inverse Hessian using displacement pairs $\{(s_k, v_k)\}$.

For instance, $[H_k]_i$ can be computed with the running average

$$[H_{k+1}]_i = (1-\lambda)[H_k]_i + \lambda \, Proj\left(\frac{[s_k]_i}{[v_k]_i}\right),$$

where $Proj(\cdot)$ represents a projection onto a predefined positive interval. But a direct application of (147) after a parameter update introduces a correlated noise that ruins the curvature estimate, which is hard to correct because of the chaotic behavior of the rescaling factors $[H_k]_i$.

Estimating Diagonal Curvature

These problems can be addressed with a combination of two ideas.

First, estimate the diagonal of the Hessian instead of its inverse.

Second, ensure the effective stepsizes are monotonically decreasing by replacing the running average by the sum

$$[G_{k+1}]_i = [G_k]_i + Proj\left(\frac{[v_k]_i}{[s_k]_i}\right).$$

Keeping the curvature estimates in a fixed positive interval ensures the effective stepsizes decrease at the rate $\mathcal{O}(\frac{1}{k})$.

Natural Gradient Method

The essential idea of natural gradient method consists of formulating the gradient descent algorithm in the space of prediction functions rather than specific parameters. The actual computation of course takes place with respect to the parameters, but the algorithm will move the parameters more quickly along directions that have a small impact on the decision function.

The space $\mathcal{H}$ of prediction functions is a family of densities $h_w(x)$ parametrized by $w \in \mathcal{W}$ and satisfying the normalization condition

$$\int h_w(x)dx = 1, \quad \forall w \in \mathcal{W}.$$

And we assume sufficient regularity, i.e.,

$$\forall t > 0, \quad \int \frac{\partial^t h_w(x)}{\partial w^t} dx = \frac{\partial^t}{\partial w^t} \int h_w(x)dx = \frac{\partial^t 1}{\partial w^t} = 0. \tag{151}$$

Natural Gradient Method

To quantify how the density h_w changes when adding a small quantity δw to its paramter, we use the Kullback-Leibler (KL) divergence

$$D_{KL}(h_w \| h_{w+\delta w}) = E_{h_W}\left[\log\left(\frac{h_w(x)}{h_{w+\delta w}(x)}\right)\right]. \tag{152}$$

Approximating the divergence with a second-order Taylor expansion, we have

$$\begin{aligned} D_{KL}(h_w \| h_{w+\delta w}) =& E_{h_W}[\log(h_w(x)) - \log(h_{w+\delta w}(x))] \\ \approx& -\delta w^\top E_{h_W}\left[\frac{\partial \log(h_w(x))}{\partial w}\right] - \frac{1}{2}\delta w^\top E_{h_w}\left[\frac{\partial^2 \log(h_w(x))}{\partial w^2}\right]\delta w. \end{aligned}$$

By (151),

$$D_{KL}(h_w \| h_{w+\delta w}) \approx \frac{1}{2}\delta w^\top G(w)\delta w. \tag{153}$$

Natural Gradient Method

Natural gradient method minimizes a functional $F : h_w \in \mathcal{H} \mapsto F(h_w) = F(w) \in \mathbb{R}$. A greedy strategy is

$$h_{w_{k+1}} = \arg\min_{h \in \mathcal{H}} F(h) \quad \text{s.t.} \quad D_{KL}(h_{w_k} \| h) \leqslant \eta_k^2. \tag{154}$$

Use (153) we can reformulate it in terms of the parameters:

$$w_{k+1} = \arg\min_{w \in \mathcal{W}} F(w) \quad \text{s.t.} \quad \frac{1}{2}(w - w_k)^\top G(w_k)(w - w_k) \leqslant \eta_k^2. \tag{155}$$

Natural Gradient Method

Lagrangian formulation is customarily used to handle this problem. Assuming η_k small, we can replace $F(w)$ with $F(w_k) + \nabla F(w_k)^\top (w - w_k)$. These two choices lead to

$$w_{k+1} = \arg\min_{w \in \mathcal{W}} \nabla F(w_k)^\top (w - w_k) + \frac{1}{2\alpha_k} (w - w_k)^\top G(w_k)(w - w_k),$$

and the optimization of the right-hand side leads to the natural gradient iteration

$$w_{k+1} = w_k - \alpha_k G^{-1}(w_k) \nabla F(w_k). \tag{156}$$

Natural Gradient Method

$G(w)$ is a called *Fisher information* matrix, with expression

$$\begin{aligned} G(w) :=& - E_{h_w}\left[\frac{\partial^2 \log(h_w(x))}{\partial w^2}\right] \\ =& E_{h_w}\left[\left(\frac{\partial \log(h_w(x))}{\partial w}\right)\left(\frac{\partial \log(h_w(x))}{\partial w}\right)^\top\right], \end{aligned} \tag{157}$$

where the latter equality follows from (151).

A sampled version of $G(w_k)$ is

$$\tilde{G}(w_k) = \frac{1}{|\mathcal{S}_k|}\sum_{i\in\mathcal{S}_k}\left(\left.\frac{\partial \log(h_w(x_i))}{\partial w}\right|_{w_k}\right)\left(\left.\frac{\partial \log(h_w(x_i))}{\partial w}\right|_{w_k}\right)^\top.$$

Gradient Methods with Momentum

With an initial point $w_1 = w_0$, scalar sequences $\{\alpha_k\}$ and $\{\beta_k\}$, the iteration of gradient methods with momentum is

$$w_{k+1} \leftarrow w_k - \alpha_k \nabla F(w_k) + \beta_k (w_k - w_{k-1}). \tag{158}$$

The latter is referred to as the *momentum* term. It is named after the fact that it represents a discretization of a certain second-order ordinary differential equation with friction.

When $\beta_k = 0$ for all $k \in \mathbb{N}$, it reduces to the steepest descent method.

When $\alpha_k = \alpha$ and $\beta_k = \beta$ for some constants $\alpha > 0$ and $\beta > 0$, it is referred to as the heavy ball method, which yields a linear convergence with a superior rate compared to steepest descent with a fixed stepsize for certain functions.

Gradient Methods with Momentum

Additional connection with (158) can be made when F is a strictly convex quadratic. If (α_k, β_k) is chosen optimally in the sense that

$$(\alpha_k, \beta_k) = \arg\min_{(\alpha,\beta)} F(w_k - \alpha \nabla F(w_k) + \beta(w_k - w_{k-1})), \tag{159}$$

then (158) is exactly the linear conjugate gradient (CG) algorithm.

An alternative view of the heavy ball method is obtained by expanding (158) as:

$$w_{k+1} \leftarrow w_k - \alpha \sum_{j=1}^{k} \beta^{k-j} \nabla F(w_k);$$

thus, each step can be viewed as an exponentially average of past gradients.

Accelerated Gradient Methods

Nesterov accelerated gradient method is similar to (158) but with its own unique properties. It involves the updates

$$\begin{aligned} \tilde{w}_k &\leftarrow w_k + \beta_k(w_k - w_{k-1}) \\ \text{and } w_{k+1} &\leftarrow \tilde{w}_k - \alpha_k \nabla F(\tilde{w}_k), \end{aligned} \tag{160}$$

which leads to the condensed form

$$w_{k+1} \leftarrow w_k - \alpha_k \nabla F(w_k + \beta_k(w_k - w_{k-1})) + \beta_k(w_k - w_{k-1}). \tag{161}$$

Compared with gradient method with momentum, it applies the momentum term first, then takes a steepest descent step at $\tilde{w}_k$.

Accelerated Gradient Methods

When F is convex and continuously differentiable with a Lipschitz continuous gradient, with appropriately chosen $\alpha_k = \alpha > 0$ for all $k \in \mathbb{N}$ and $\{\beta_k\} \nearrow 1$ leads to an *optimal* iteration complexity.

While the convergence rate of steepest descent method is $\mathcal{O}(\frac{1}{k})$, the iteration (161) converges with a rate $\mathcal{O}(\frac{1}{k^2})$, which is provably the best rate that can be achieved by a gradient method.

Unfortunately, no intuitive explanation as to how Nesterov's method achieves this optimal rate has been widely accepted.

Coordinate Descent Methods

Coordinate descent (CD) methods operate to a single variable while all others are kept fixed, then other variables are updated similarly.

The CD method for minimizing $F : \mathbb{R}^d \to \mathbb{R}$ is given by the iteration

$$w_{k+1} \leftarrow w_k - \alpha_k \nabla_{i_k} F(w_k) e_{i_k}, \tag{162}$$

where $\nabla_{i_k} F(w_k) := \frac{\partial F}{\partial w^{i_k}}(w_k)$, w^{i_k} represents the i_k-th element of the parameter vector, and e_{i_k} represents the i_k-th coordinate vector for some $i_k \in \{1, \ldots, d\}$.

Coordinate Descent Methods

Specific versions of the CD method are defined by the manner in which the sequences $\{\alpha_k\}$ and $\{i_k\}$ are chosen.

$\{\alpha_k\}$:

- Choose α_k as the global minimizer of F from w_k along the i_k-th coordinate.
- Choose α_k yielding a sufficient reduction in F from w_k.
- Compute α_k as the minimizer of a quadratic model of F along the i_k-th coordinate direction. (so-called second-order CD methods)

$\{i_k\}$:

- Cycle through $\{1, \ldots, d\}$.
- Cycle through a random reordering of $\{1, \ldots, d\}$, with the indexes reordered after each set of d steps.
- Simply choose an index randomly with replacement in each iteration.

The latter two strategies for $\{i_k\}$ have superior theoretical properties than the first strategy.

Convergence Properties

A CD method is not guaranteed to converge when applied to minimize any given continuously differentiable function. This is in contrast with the full gradient method, which guarantees convergence to stationarity even when the objective is nonconvex.

However, if the objective F is strongly convex, the CD method will not fail. The analysis is very simple when using a constant stepsize. Assume that ∇F is coordinate-wise Lipschitz continuous in the sense that for all $w \in \mathbb{R}^d, i \in \{1, \ldots, d\}$, and $\Delta w^i \in \mathbb{R}$, there exists a constant $L_i > 0$ such that

$$|\nabla_i F(w + \Delta w^i e_i) - \nabla_i F(w)| \leqslant L_i |\Delta w^i|. \tag{163}$$

And we define $\hat{L} := \max_{i \in \{1,\ldots,d\}} L_i$.

Convergence Properties

定理

Suppose that the objective function $F : \mathbb{R}^d \to \mathbb{R}$ *is continuously differentiable, strongly convex with constant* $c > 0$, *and has a gradient that is coordinate-wise Lipshcitz continuous with constants* $\{L_1, \ldots, L_d\}$. *In addition, suppose that* $\alpha_k = 1/\hat{L}$ *and* i_k *is chosen independently and uniformly from* $\{1, \ldots, d\}$ *for all* $k \in \mathbb{N}$. *Then for all* $k \in \mathbb{N}$, *the iteration* (162) *yields*

$$E\left[F(w_{k+1})\right] - F_* \leqslant \left(1 - \frac{c}{d\hat{L}}\right)^k (F(w_1) - F_*). \tag{164}$$

Favorable Problem Structures

A simple randomized CD method is linearly convergent with constant dependent on the parameter dimension d. If d coordinate updates can be performed at a cost similar to the evaluation of one full gradient, the method is competitive with a full gradient method both theoretically and in practice.

This kind of problems include those in which the objective function is

$$F(w) = \frac{1}{n}\sum_{j=1}^{n} \tilde{F}_j(x_j^\top w) + \sum_{i=1}^{d} \hat{F}_i(w^i), \tag{165}$$

where $\forall j \in \{1, \ldots, n\}$, $\tilde{F}_j$ is continuously differentiable and dependent on the *sparse* data vector x_j, and $\forall i \in \{1, \ldots, d\}$, $\hat{F}_i$ is a regularization function (potentially nonsmooth).

Favorable Problem Structures

For example, consider an objective function of the form

$$f(w) = \frac{1}{2}\|Xw - y\|_2^2 + \sum_{i=1}^{d} \hat{F}_i(w^i) \text{ with } X = [x_1 \ \ldots \ x_n].$$

In this setting,

$$\nabla_{i_k} f(w_{k+1}) = x_{i_k}^\top r_{k+1} + \hat{F}'_{i_k}(w_{k+1}^{i_k}) \text{ with } r_{k+1} := Aw_{k+1} - b,$$

where, with $w_{k+1} = w_k + \beta_k e_{i_k}$, we have $r_{k+1} = r_k + \beta_k x_{i_k}$.

Since the residuals $\{r_k\}$ can be updated with cost proportional to the number of nonzeros in x_{i_k}, call it $nnz(x_{i_k})$, the overall cost of computing the search direction in iteration $k+1$ is also $\mathcal{O}(nnz(x_{i_k}))$. On the other hand, an evaluation of the entire gradient requires a cost of $\mathcal{O}(\sum_{j=1}^{n} nnz(x_j))$.

Stochastic Dual Coordinate Ascent

Consider minimizing a convex objective function of the form (165) by maximizing its dual.

Defining the convex conjugate of $\tilde{F}_j$ as $\tilde{F}_j^*(u) := \max_w(w^\top u - \tilde{F}_j(w))$ when $\hat{F}_i(\cdot) = \frac{\lambda}{2}(\cdot)^2$ for all $i \in \{1, \ldots, d\}$ is given by

$$F_{dual}(v) = \frac{1}{n}\sum_{j=1}^{n}\left[-\tilde{F}_j^*(-v_j)\right] - \frac{\lambda}{2}\left\|\frac{1}{\lambda n}\sum_{j=1}^{n} v_j x_j\right\|_2^2.$$

The stochastic dual coordinate ascent (SDCA) method applied to a function of this form has an iteration similar to (162), except that negative gradient steps are replaced by gradient steps.

When the algorithm terminates, the corresponding primal solution can be obtained as $w \leftarrow \frac{1}{\lambda n}\sum_{j=1}^{n} v_j x_j$.

Parallel CD Methods

Consider a multicore system in which the parameter vector w is stored in shared memory.

Each core can then execute a CD iteration independently and in an asynchronous manner, where if d is large compared to the number of cores, then it is unlikely that two cores are attempting to update the same variable at the same time.

Each update is being made based on slightly stale information. However, convergence of the method can be proved, and improves when one can bound the degree of staleness of each update.

Methods for Regularized Models

The discussion of structural risk minimization highlighted the key role played by regularization functions.

The optimization methods we have presented in this section are all applicable for objectives involving smooth regularizers, such as the squared ℓ_2-norm. And we expand our investigation by considering optimization methods that handle the regularization as a distinct entity, in particular when the function is *nonsmooth*, for example, ℓ_1-norm, which induces sparsity in the optimal solution vector.

For machine learning, sparsity can be seen as a form of *feature selection*.

Methods for Regularized Models

We focuses on the nonsmooth optimization problem

$$\min_{w \in \mathbb{R}^d} \Phi(w) := F(w) + \lambda \Omega(w), \tag{166}$$

where $F : \mathbb{R}^d \to \mathbb{R}$ includes the composition of a loss and prediction function, $\lambda > 0$ is a regularization parameter, and $\Omega : \mathbb{R}^d \to \mathbb{R}$ is a convex, nonsmooth regularization function.

Specifically, we pay special attention to methods for solving the problem

$$\min_{w \in \mathbb{R}^d} \phi(w) := F(w) + \lambda \|w\|_1. \tag{167}$$

First-order Methods for Generic Convex Regularizers

For solving problem (166), the *proximal gradient* method represents a fundamental approach.

Given an iterate w_k, a generic proximal gradient iteration with $\alpha_k > 0$ is given by

$$w_{k+1} \leftarrow \arg\min_{w \in \mathbb{R}^d} \left(F(w_k) + \nabla F(w_k)^\top (w - w_k) + \frac{1}{2\alpha_k} \|w - w_k\|_2^2 + \lambda \Omega(w) \right) \tag{168}$$

The term *proximal* refers to the presence of the third term in the minimization problem on the right-hand side, which encourage the new iterate to be close to w_k. If the last term were not present, then (168) exactly recovers the gradient method update $w_{k+1} \leftarrow w_k - \alpha_k \nabla F(w_k)$; hence we refer to α_k as the stepsize parameter.

First-order Methods for Generic Convex Regularizers

定理

Suppose that $F : \mathbb{R}^d \to \mathbb{R}$ *is continuously differentiable, strongly convex with constant* $c > 0$*, and has a gradient that is Lipschitz continuous with constant* $L > 0$*. In addition, suppose that* $\alpha_k = \alpha \in (0, 1/L)$ *for all* $k \in \mathbb{N}$*. Then, for all* $k \in \mathbb{N}$*, the iteration* (168) *yields*

$$\Phi(w_{k+1}) - \Phi(w_*) \leqslant (1 - \alpha c)^k (\Phi(w_1) - \Phi(w_*)),$$

where $w_* \in \mathbb{R}^d$ *is the unique global minimizer of* Φ *in* (166)*.*

First-order Methods for Generic Convex Regularizers

The proximal gradient iteration (168) is practical only when the proximal mapping

$$\underset{\lambda\Omega,\alpha_k}{prox}(\tilde{w}) := \arg\min_{w\in\mathbb{R}^n} \left(\lambda\Omega(w) + \frac{1}{2\alpha_k}\|w-\tilde{w}\|_2^2 \right)$$

can be computed efficiently. Situations when the proximal mapping is inexpensive to compute include when Ω is the indicator function for a simple set, when it is the ℓ_1-norm, or when it is separable.

A stochastic version of the proximal gradient method can be obtained by replacing $\nabla F(w_k)$ in (168) by a stochastic approximation $g(w_k, \xi_k)$. The resulting method attains similar behavior as a stochastic gradient method.

Iterative Soft Thresholding Algorithm (ISTA)

For solving the ℓ_1-norm regularized problem (167), the proximal gradient method is

$$w_{k+1} \leftarrow \arg\min_{w \in \mathbb{R}^d} \left(F(w_k) + \nabla F(w_k)^\top (w - w_k) + \frac{1}{2\alpha_k} \|w - w_k\|_2^2 + \lambda \|w\|_1 \right) \tag{169}$$

The solution can be written component-wise in closed form, with $(\cdot)_+ := \max\{\cdot, 0\}$, as

$$w_{k+1} \leftarrow \mathcal{T}_{\alpha_k \lambda}(w_k - \alpha_k \nabla F(w_k)), \text{ where } [\mathcal{T}_{\alpha_k \lambda}]_i = (|\tilde{w}_i| - \alpha_k \lambda)_+ \, sgn(\tilde{w}_i). \tag{170}$$

$\mathcal{T}_{\alpha_k \lambda}$ is referred to as the soft-thresholding operator, which leads to the name *iterative soft-thresholding algorithm* (ISTA). It is clear from (170) that the ISTA iteration induces sparsity in the iterates.

Bound-constrained Methods for ℓ_1-norm Regularized Problems

An equivalent *smooth* reformulation of problem (167) is easily derived, by writing $w = u - v$ where u and v play the *positive part* and *negative part* of w respectively:

$$\min_{(u,v)\in\mathbb{R}^d\times\mathbb{R}^d} \tilde{\phi}(u,v) \quad \text{s.t.} \quad (u,v) \geqslant 0 \tag{171}$$

where $\tilde{\phi}(u,v) = F(u-v) + \lambda \sum_{i=1}^{d}(u_i + v_i)$.

The fundamental iteration for solving bound-constrained optimization problems is the *gradient projection* method. In the context of (171), the iteration reduces to

$$\begin{bmatrix} u_{k+1} \\ v_{k+1} \end{bmatrix} \leftarrow P_+\left(\begin{bmatrix} u_k \\ v_k \end{bmatrix} - \alpha_k \begin{bmatrix} \nabla_u \tilde{\phi}(u_k, v_k) \\ \nabla_v \tilde{\phi}(u_k, v_k) \end{bmatrix}\right) = P_+\left(\begin{bmatrix} u_k - \alpha_k \nabla F(u_k - v_k) - \alpha_k \lambda e \\ v_k + \alpha_k \nabla F(u_k - v_k) - \alpha_k \lambda e \end{bmatrix}\right) \tag{172}$$

where P_+ projects onto the nonnegative orthant and $e \in \mathbb{R}^d$ is a vector of ones.

Bound-Constrained Methods for ℓ_1-norm Regularized Problems

The iteration (172) is expected to inherit the property of being globally linearly convergent when F satisfies the assumptions of the last theorem. However, since the variables in (171) have been split into positive and negative parts, this property is maintained *only if* the iteration maintains complementarity of each iterate pair, i.e., if $[u_k]_i[v_k]_i = 0,\ \forall k \in \mathbb{N}, i \in \{1, \ldots, d\}$.

A stochastic projected gradient method, with $\nabla F(w_k)$ replaced by $g(w_k, \xi_k)$, has similar convergence properties as a standard SG method.

Second-order Methods

For solving problem (167), a *proximal Newton* method is one that constructs, at each $k \in \mathbb{N}$, a model

$$q_k(w) = F(w_k) + \nabla F(w_k)^\top (w - w_k) + \frac{1}{2}(w - w_k)^\top H_k (w - w_k) + \lambda \|w\|_1, \tag{173}$$

where H_k represents $\nabla^2 F(w_k)$ or a quasi-Newton approximation of it.

A proximal Newton method would involve (approximately) minimizing this model to compute a trial iterate $\tilde{w}_k$, then a step size $\alpha_k > 0$ would be taken from a predetermined sequence or chosen by a line search to ensure that the new iterate $w_{k+1} \leftarrow w_k + \alpha_k(\tilde{w}_k - w_k)$ yields $\phi(w_{k+1}) < \phi(w_k)$.

Proximal Newton Methods

Proximal Newton methods are more challenging to design, analyze and implement than proximal gradient methods. Assuming H_k has been chosen to be positive definite, here are three essential ingredients in proximal Newton method:

Choice of Subproblem Solver q_k is nonsmooth and is challenging to minimize. One choice is coordinate descent, since the global minimizer of q_k along a coordinate descent direction can be computed analytically.

Inaccurate Subproblem Solves It's impractical to minimize q_k accurately for all $k \in \mathbb{N}$. Thus we need a practical and theoretically sufficient termination criteria.

Proximal Newton Methods

Interestingly, the norm of an ISTA step is an appropriate measure. Let $ista_k(w)$ represent the result of an ISTA step applied to q_k from w. A trial point $\tilde{w}_k$ represents a sufficiently accurate minimizer of q_k if, for some $\eta \in [0, 1)$, one finds

$$\| ista_k(\tilde{w}_k) - \tilde{w}_k \|_2 \leqslant \eta \| ista_k(w_k) - w_k \|_2 \text{ and } q_k(\tilde{w}_k) < q_k(w_k).$$

Elimination of Variables Due to the structure created by the ℓ_1-norm regularizer, it can be effective in some applications to first identify a set of *active* variables then compute an approximate minimizer of q_k over the remaining *free* variables.

Orthant-based Methods

Our second class of second-order methods is based on the observation that ℓ_1-norm regularized objective ϕ in problem (167) is smooth in any orthant in $\mathbb{R}^d$.

In every iteration, orthant-based methods construct a smooth quadratic model of the objective, then produce a search direction by minimizing this model.

After performing a line search designed to reduce the objective function, a new orthant is selected and the process is repeated.

Orthant-based Methods

With the minimum norm subgradient of ϕ at $w \in \mathbb{R}^d$, which is given component-wise for all $i \in \{1, \ldots, d\}$ by

$$\hat{g}_i(w) = \begin{cases} [\nabla F(w)]_i + \lambda & \text{if } w_i > 0 \text{ or } \{w_i = 0 \text{ and } [\nabla F(w)]_i + \lambda < 0\} \\ [\nabla F(w)]_i - \lambda & \text{if } w_i < 0 \text{ or } \{w_i = 0 \text{ and } [\nabla F(w)]_i - \lambda > 0\} \\ 0 & \text{otherwise,} \end{cases} \tag{174}$$

the active orthant for an iterate w_k is characterized by the sign vector

$$\zeta_{k,i} = \begin{cases} sgn([w_k]_i) & \text{if } [w_k]_i \neq 0 \\ sgn(-[\hat{g}(w_k)]_i) & \text{if } [w_k]_i = 0. \end{cases} \tag{175}$$

Along these lines, define the subsets of $\{1, \ldots, d\}$ given by

$$\mathcal{A}_k = \{i : [w_k]_i = 0 \text{ and } |[\nabla F(w_k)]_i| \leqslant \lambda\} \tag{176}$$

$$\text{and } \mathcal{F}_k = \{i : [w_k]_i \neq 0\} \cup \{i : [w_k]_i = 0 \text{ and } |[\nabla F(w_k)]_i| > \lambda\}, \tag{177}$$

where $\mathcal{A}_k$ represents the indices of variables that are active and kept at zero while $\mathcal{F}_k$ represents those that are free to move.

Orthant-based Methods

Given these quantities, an orthant-based method proceeds as follows. First, compute the (approximate) solution d_k of the (smooth) quadratic problem

$$\min_{d \in \mathbb{R}^n} \hat{g}(w_k)^\top d + \frac{1}{2} d^\top H_k d$$
$$\text{s.t. } d_i = 0, \quad i \in \mathcal{A}_k,$$

where H_k represents $\nabla^2 F(w_k)$ or an approximation of it.

Then the algorithm performs a line search—over a path contained in the current orthant—to compute the next iterate.

> One option is a projected backtracking line search along d_k, computing the largest α_k in a decreasing geometric sequence so
>
> $$F(P_k(w_k + \alpha_k d_k)) < F(w_k),$$
>
> where $P_k(w)$ projects $w \in \mathbb{R}^d$ onto the orthant defined by ζ_k.

Outline I

Outline II

Thanks for your attention!