Optimization theory

My learning notes

Kaikai Zhao

First draft: September 1, 2021 Last update: March 23, 2023

Contents

To-do list

This document is updated almost every day. I will write proofs about the convergence results for all algorithms presented in this document. Some have been done and the rest will be done later. Also, I will write an article that only focuses on convergence analysis in the future by extracting the related parts from this document and presenting them in greater detail. My goal is to make convergence analysis less intimidating for optimization beginners.

1 Notations

In this document, $\|\cdot\|$ denotes the Euclidean norm $\|\cdot\|_2$, i.e., $\|\mathbf{v}\|_2 = \sqrt{\sum_i v_i^2}$, but it tends to be used for general norm for vectors when other norms appear.

2 Proof on saddle point

See Ex. 3.14 on Page 115 of B&V's Convex Optimization book and the proof of Proposition 2.35 on Page 75 of Hoang Tuy's Convex Analysis and Global Optimization book.

Suppose that $f : \mathbf{R}^n \times \mathbf{R}^m \to \mathbf{R}$ is convex-concave and differentiable, with $\nabla f(\tilde{x}, \tilde{z}) = 0$. Show that the saddle-point property holds: for all x, z, we have

$$f(\tilde{x}, z) \le f(\tilde{x}, \tilde{z}) \le f(x, \tilde{z}). \tag{1}$$

Show that this implies that f satisfies the strong max-min property:

$$\sup_z \inf_x f(x, z) = \inf_x \sup_z f(x, z)$$

(and their common value is $f(\tilde{x},\tilde{z})$).

Proof. Since f is convex over x for a given z and $\nabla f(\tilde{x},\tilde{z})=0$, $\tilde{x}$ minimizes $f(x,\tilde{z})$ over all x. Thus, we have $f(\tilde{x},\tilde{z})\leq f(x,\tilde{z})$. Similarly, Since f is concave over z for a given x and $\nabla f(\tilde{x},\tilde{z})=0$, $\tilde{z}$ maximizes $f(\tilde{x},z)$ over all z. Thus, we have $f(\tilde{x},z)\leq f(\tilde{x},\tilde{z})$. Hence, we get $(\tilde{x},z)\leq f(\tilde{x},\tilde{z})\leq f(x,\tilde{z})$ for all x, z. Based on this result, we have

$$\begin{aligned}(1) &\Leftrightarrow \sup_z f(\tilde{x},z)\leq f(\tilde{x},\tilde{z})\leq \inf_x f(x,\tilde{z})\\ &\Leftrightarrow \inf_x\sup_z f(x,z)\leq \sup_z f(\tilde{x},z)\leq f(\tilde{x},\tilde{z})\\ &\qquad\leq \inf_x f(x,\tilde{z})\leq \sup_z\inf_x f(x,z)\end{aligned}\tag{2}$$

From (2), we get $\inf_x\sup_z f(x,z)\leq \sup_z\inf_x f(x,z)$. More straightforwardly, the following derivation does not depend on any properties of f and always holds,

$$\begin{aligned}&f(x,z)\leq \sup_z f(x,z)\\ &\Leftrightarrow \inf_x f(x,z)\leq \inf_x\sup_z f(x,z)\\ &\Leftrightarrow \sup_z\inf_x f(x,z)\leq \inf_x\sup_z f(x,z)\end{aligned}\tag{3}$$

From the results of (2) and (3), we must have $\sup_z\inf_x f(x,z)=\inf_x\sup_z f(x,z)$. To show they have the common value $f(\tilde{x},\tilde{z})$, we combine (2) and (3) to get $f(\tilde{x},\tilde{z})\leq\sup_z\inf_x f(x,z)\leq\inf_x\sup_z f(x,z)$ and $\sup_z\inf_x f(x,z)\leq\inf_x\sup_z f(x,z)\leq f(\tilde{x},\tilde{z})$, which results in $\sup_z\inf_x f(x,z)=\inf_x\sup_z f(x,z)=f(\tilde{x},\tilde{z})$. □

3 Basics for convergence analysis

3.1 Lipschitz condition and Lipschitz constant

Definition 1. A function $f:[a,b]\to\mathbb{R}$, is said to satisfy the Lipschitz condition if there is a constant $L>0$ such that[1]

$$|f(y)-f(x)|\leq L|y-x|,\quad \forall x,y\in[a,b].$$

The smallest constant M satisfying the Lipschitz condition is called Lipschitz constant[2], i.e.,

$$\sup_{x\neq y}\frac{|f(y)-f(x)|}{|y-x|}.$$

The definition can be readily extended to vector-valued functions on subsets of any Euclidean space and, more in general, to mappings between metric spaces.

3.2 Lipschitz gradient and its implications

Let f be twice continuously differentiable on $S=\mathbf{dom}(f)$. If ∇f is Lipschitz with constant L, we have the following results.

(1) $\|\nabla f(x)-\nabla f(y)\|\leq L\|x-y\|$ for all $x,y\in S$;

(2) $\nabla^2 f(x)\preceq LI$ for all $x\in\mathbf{int}S$, where $\nabla^2 f(x)$ is the Hessian matrix of f at $x\in\mathbf{int}S$ and defined as

$$\nabla^2 f(x)=\frac{\partial^2 f(x)}{\partial x_i\partial x_j},\quad i,j=1,\dots,n;$$

(3) $f(y)\leq f(x)+\nabla f(x)^T(y-x)+\frac{L}{2}\|y-x\|^2$ for all x, y;

(4) $(\nabla f(y)-\nabla f(x))^T(y-x)\leq L\|y-x\|^2$ for all x, y;

(5) $\frac{L}{2}\|x\|_2^2-f(x)$ is convex. Note that here is the Euclidean norm.

- (1) is the definition of Lipschitz gradients.

[1] https://encyclopediaofmath.org/wiki/Lipschitz_condition
[2] https://encyclopediaofmath.org/wiki/Lipschitz_constant

- (1) $\Longrightarrow$ (2): By the mean value theorem for $f : \mathbb{R}^n \to \mathbb{R}$, we have $\nabla f(x) - \nabla f(y) = \nabla^2 f(z)(x-y)$ with $z = (1-\lambda)x + \lambda y$ and $\lambda \in (0,1)$. Thus,

$$\|\nabla f(x) - \nabla f(y)\|_2 = \|\nabla^2 f(z)(x-y)\|_2 \le \|\nabla^2 f(z)\|_2 \|x-y\|_2$$

~~Since $\|\nabla f(x) - \nabla f(y)\| \le L\|x-y\|$ holds for all x, y, then $\|\nabla^2 f(z)\|_2 \|x-y\|_2 \le L\|x-y\|_2$ which gives $\|\nabla^2 f(z)\|_2 \le L$ and implies that $\nabla^2 f(x) \preceq LI$.~~

Note. We striked out the wrong argument since we could not arrive at $\|\nabla^2 f(z)\|_2 \|x-y\|_2 \le L\|x-y\|_2$ directly.

Now we present the correct proof which is largely taken from [Beck, 2014]. Given that f is twice continuously differentiable and ∇f is Lipschitz with constant L, according to the fundamental theorem of calculus, we have, for any $x, x + \alpha\Delta x \in S$ and $\alpha > 0$,

$$\begin{aligned}
&\int_0^\alpha \nabla^2 f(x + t\Delta x)\Delta x \mathrm{d}t = \nabla f(x + \alpha\Delta x) - \nabla f(x) \\
&\left\| \int_0^\alpha \nabla^2 f(x + t\Delta x)\Delta x \mathrm{d}t \right\|_2 = \|\nabla f(x + \alpha\Delta x) - \nabla f(x)\|_2 \le \alpha L \|\Delta x\|_2 \\
&\left\| \left(\int_0^\alpha \nabla^2 f(x + t\Delta x)\mathrm{d}t \right) \Delta x \right\|_2 \le \alpha L \|\Delta x\|_2 \\
&\left\| \frac{\int_0^\alpha \nabla^2 f(x + t\Delta x)\mathrm{d}t}{\alpha} \Delta x \right\|_2 \le L \|\Delta x\|_2 \\
&\left\| \lim_{\alpha \to 0^+} \frac{\int_0^\alpha \nabla^2 f(x + t\Delta x)\mathrm{d}t}{\alpha} \Delta x \right\|_2 \le L \|\Delta x\|_2 \\
&\|\nabla^2 f(x)\Delta x\|_2 \le L \|\Delta x\|_2
\end{aligned}$$

which implies $\|\nabla^2 f(x)\|_2 \le L$.

- (2) $\Longrightarrow$ (1):

$$\begin{aligned}
\int_0^1 \nabla^2 f(x + t(y-x))(y-x)\mathrm{d}t &= \nabla f(y) - \nabla f(x) \\
\left\| \int_0^1 \nabla^2 f(x + t(y-x))\mathrm{d}t \cdot (y-x) \right\| &= \|\nabla f(y) - \nabla f(x)\| \\
\int_0^1 \left\| \nabla^2 f(x + t(y-x)) \right\| \mathrm{d}t \cdot \|y-x\| &\ge \|\nabla f(y) - \nabla f(x)\| \\
L\|y-x\| &\ge \|\nabla f(y) - \nabla f(x)\|,
\end{aligned}$$

as desired.

- (2) $\Longrightarrow$ (3): From Taylor mean value theorem, we have, for some $z = (1-\lambda)x + \lambda y$ with $\lambda \in (0,1)$,

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T \nabla^2 f(z)(y-x) \le f(x) + \nabla f(x)^T(y-x) + \frac{L}{2}\|y-x\|^2$$

The inequality follows from (2).

- (1) $\Longrightarrow$ (4):

$$(\nabla f(y) - \nabla f(x))^T(y-x) \le \|\nabla f(y) - \nabla f(x)\| \|y-x\| \le L\|y-x\|^2$$

- (3) $\Longrightarrow$ (5):

$$\begin{gathered}
(3) \Longleftrightarrow -f(y) + \frac{L}{2}\|y-x\|^2 \ge -f(x) - \langle \nabla f(x), y-x \rangle \\
\Updownarrow \\
\frac{L}{2}\|y\|^2 - f(y) \ge -f(x) - \langle \nabla f(x), y-x \rangle - \frac{L}{2}\|x\|^2 + L\langle x, y \rangle \\
\frac{L}{2}\|y\|^2 - f(y) \ge \frac{L}{2}\|x\|^2 - f(x) - \langle \nabla f(x), y-x \rangle - L\langle x, x \rangle + L\langle x, y \rangle \\
\frac{L}{2}\|y\|^2 - f(y) \ge \frac{L}{2}\|x\|^2 - f(x) + \langle -\nabla f(x), y-x \rangle + \langle Lx, (y-x) \rangle \\
\frac{L}{2}\|y\|^2 - f(y) \ge \frac{L}{2}\|x\|^2 - f(x) + \langle Lx - \nabla f(x), y-x \rangle
\end{gathered}$$

3.3 A useful result from Lipschitz gradients

Let $y = x^+ = x - t\nabla f(x)$.

$$\begin{aligned} f(x^+) &\le f(x) + \nabla f(x)^T(-t\nabla f(x)) + \frac{L}{2}\|-t\nabla f(x)\|_2^2 \\ &= f(x) - t\|\nabla f(x)\|_2^2 + \frac{Lt^2}{2}\|\nabla f(x)\|_2^2 \\ &= f(x) - (1 - \frac{Lt}{2})t\|\nabla f(x)\|_2^2 \\ &\le f(x) - \frac{t}{2}\|\nabla f(x)\|_2^2 \quad (0 < t \le 1/L) \end{aligned} \quad \Rightarrow f(x^+) \le f(x) - \frac{t}{2}\|\nabla f(x)\|_2^2 \quad (0 < t \le 1/L) \tag{4}$$

The left column is the derivation and the right column is the result which guarantees "the descent in objective values", i.e., $f(x^{(k)}) \le f(x^{(k-1)}) \le \cdots \le f(x^{(0)})$.

3.4 Lipschitz continuous softmax function

The softmax function is defined by

$$\sigma(\mathbf{z}) = \frac{\exp(\mathbf{z})}{\mathbf{1}^T \exp(\mathbf{z})},$$

where $\mathbf{z}$ is a column vector and $\mathbf{1}$ is a vector with all entries being 1. Then we know that the gradient of the log-sum-exp function $f(\mathbf{z}) = \log(\sum_{i=1}^n \exp(z_i))$ is a softmax function. To see this,

$$\nabla f(\mathbf{z}) = \nabla \log(\sum_{i=1}^n \exp(z_i)) = \frac{1}{\sum_{i=1}^n \exp(z_i)} \begin{bmatrix} \exp(z_1) \\ \exp(z_2) \\ \vdots \\ \exp(z_n) \end{bmatrix} = \frac{\exp(\mathbf{z})}{\mathbf{1}^T \exp(\mathbf{z})} = \sigma(\mathbf{z}).$$

Then the Jacobian matrix of $\sigma(\mathbf{z})$ w.r.t. $\mathbf{z}$ is given by $D[\sigma(\mathbf{z})]$. The diagonal entries of $D[\sigma(\mathbf{z})]$ are

$$\frac{\partial[\sigma(\mathbf{z})]_i}{\partial z_i} = \frac{\partial^2 \log(\sum_{i=1}^n \exp(z_i))}{\partial z_i^2} = \frac{\exp(z_i)\mathbf{1}^T \exp(\mathbf{z}) - (\exp(z_i))^2}{(\mathbf{1}^T \exp(\mathbf{z}))^2} = \frac{\exp(z_i)}{\mathbf{1}^T \exp(\mathbf{z})} - \frac{(\exp(z_i))^2}{(\mathbf{1}^T \exp(\mathbf{z}))^2}$$

The off-diagonal entries are

$$\frac{\partial[\sigma(\mathbf{z})]_i}{\partial z_j} = \frac{\partial^2 \log(\sum_{i=1}^n \exp(z_i))}{\partial z_i \partial z_j} = \frac{0 - \exp(z_i)\exp(z_j)}{(\mathbf{1}^T \exp(\mathbf{z}))^2} = -\frac{\exp(z_i)\exp(z_j)}{(\mathbf{1}^T \exp(\mathbf{z}))^2}$$

Hence,

$$D[\sigma(\mathbf{z})] = \mathrm{diag}(\sigma(\mathbf{z})) - \sigma(\mathbf{z})\sigma(\mathbf{z})^T. \tag{5}$$

Let $\mathbf{v}$ be an arbitrary vector in $\mathbb{R}^n$. Then we have

$$\begin{aligned} \mathbf{v}^T D[\sigma(\mathbf{z})]\mathbf{v} =& \mathbf{v}^T(\mathrm{diag}(\sigma(\mathbf{z})) - \sigma(\mathbf{z})\sigma(\mathbf{z})^T)\mathbf{v} \\ =& \mathbf{v}^T \mathrm{diag}(\sigma(\mathbf{z}))\mathbf{v} - \mathbf{v}^T\sigma(\mathbf{z})\sigma(\mathbf{z})^T\mathbf{v} \\ =& \mathbf{v}^T \mathrm{diag}(\sigma(\mathbf{z}))\mathbf{v} - \|\sigma(\mathbf{z})^T\mathbf{v}\|_2^2 \\ \le& \mathbf{v}^T \mathrm{diag}(\sigma(\mathbf{z}))\mathbf{v} = \|\mathrm{diag}(\sqrt{\sigma(\mathbf{z})})\mathbf{v}\|_2^2 \\ \le& \mathbf{1}^T(\sqrt{\sigma(\mathbf{z})})^2\|\mathbf{v}\|_2^2 = 1 \cdot \|\mathbf{v}\|_2^2 = \|\mathbf{v}\|_2^2 \end{aligned}$$

which implies the Lipschitz constant is 1 according to the second result in Section (3.2). The last line follows from Cauchy inequality.

Since the cross-entropy loss employs the combination of the logarithm function and the softmax function, we will introduce the gradients of the cross-entropy function in the next subsection.

3.5 Cross-entropy function and its gradients

We first introduce some notations to facilitate the derivations of subsequent formulae.

- $\mathbf{x} \in \mathbb{R}^{d\times 1}$ is an input vector, where d is the number of features;
- $\mathbf{W} \in \mathbb{R}^{c\times d}$ is the weight matrix, where c is the number of classes;

- $\mathbf{b} \in \mathbb{R}^{c\times 1}$ is the bias vector;
- $\tilde{\mathbf{y}} \in \mathbb{R}^{c\times 1}$ is the output of the softmax function;
- $\mathbf{e}_j = [0, \cdots, 1, \cdots, 0]^T$, where only the j-th entry is 1;
- $\mathbf{y} \in \mathbb{R}^{c\times 1}$ is the target vector and a one-hot column vector.

Then we have the following relations.

$$\mathbf{z} = \mathbf{W}\mathbf{x} + \mathbf{b}$$
$$\tilde{\mathbf{y}} = \sigma(\mathbf{z}) = \frac{\exp(\mathbf{z})}{\mathbf{1}^T \exp(\mathbf{z})}$$

The definition of cross-entropy loss is

$$\mathrm{CE}(\mathbf{z}) = -\mathbf{y}^T \log \tilde{\mathbf{y}}.$$

Since $\mathbf{y}$ is a one-hot vector, we suppose $y_k = 1$. Then the cross-entropy reduces to the following form.

$$\mathrm{CE}(\mathbf{z}) = -y_k \log \frac{\exp(z_k)}{\mathbf{1}^T \exp(\mathbf{z})} = -z_k + \log(\sum_{i=1}^{c} \exp(z_i))$$

By the chain rule, we have

$$\frac{\partial\,\mathrm{CE}}{\partial W_{ij}} = (\frac{\partial\,\mathrm{CE}}{\partial \mathbf{z}})^T \frac{\partial \mathbf{z}}{\partial W_{ij}} = \left(\frac{\partial \tilde{\mathbf{y}}}{\partial \mathbf{z}} \frac{\partial\,\mathrm{CE}}{\partial \tilde{\mathbf{y}}}\right)^T \frac{\partial \mathbf{z}}{\partial W_{ij}}$$

where we adopt the denominator layout, so the chain rule is performed from right to left in the second equality. Now we deal with the above three terms separately. The first term $\frac{\partial \tilde{\mathbf{y}}}{\partial \mathbf{z}}$ has been calculated in the preceding subsection, namely,

$$\begin{aligned}
\frac{\partial \tilde{\mathbf{y}}}{\partial \mathbf{z}} = \frac{\partial \sigma(\mathbf{z})}{\partial \mathbf{z}} &= \frac{\partial \frac{\exp(\mathbf{z})}{\mathbf{1}^T \exp(\mathbf{z})}}{\partial \mathbf{z}} \\
&= \frac{\frac{\partial \exp(\mathbf{z})}{\partial \mathbf{z}} \cdot \mathbf{1}^T \exp(\mathbf{z}) - \exp(\mathbf{z}) \cdot \frac{\partial(\mathbf{1}^T \exp(\mathbf{z}))}{\partial \mathbf{z}}}{(\mathbf{1}^T \exp(\mathbf{z}))^2} \\
&= \frac{\mathrm{diag}(\exp(\mathbf{z}))}{\mathbf{1}^T \exp(\mathbf{z})} - \frac{\exp(\mathbf{z}) \cdot \mathrm{diag}(\exp(\mathbf{z})) \cdot \mathbf{1}}{(\mathbf{1}^T \exp(\mathbf{z}))^2} \\
&= \mathrm{diag}(\frac{\exp(\mathbf{z})}{\mathbf{1}^T \exp(\mathbf{z})}) - \frac{\exp(\mathbf{z}) \cdot \exp(\mathbf{z}^T)}{(\mathbf{1}^T \exp(\mathbf{z}))^2} \\
&= \mathrm{diag}(\sigma(\mathbf{z})) - \sigma(\mathbf{z})\sigma(\mathbf{z})^T.
\end{aligned}$$

Here, $a = a(\boldsymbol{x}), \boldsymbol{u} = \boldsymbol{u}(\boldsymbol{x})$, where $\boldsymbol{x}$ is a vector, a is a scalar function, $\boldsymbol{u}$ is a valued function. Under the numerator layout layout, $\frac{\partial a\boldsymbol{u}}{\boldsymbol{x}} = a\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} + \boldsymbol{u}\frac{\partial a}{\partial \boldsymbol{x}}$. However, under the the denominator layout, it will be $\frac{\partial a\boldsymbol{u}}{\partial \boldsymbol{x}} = a\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}} + \frac{\partial a}{\partial \boldsymbol{x}}\boldsymbol{u}^T$.

The second term

$$\frac{\partial\,\mathrm{CE}(\tilde{\mathbf{y}})}{\partial \tilde{\mathbf{y}}} = \frac{\partial \log \tilde{\mathbf{y}}}{\partial \tilde{\mathbf{y}}} \frac{\partial(-\mathbf{y}^T \log \tilde{\mathbf{y}})}{\partial \log(\tilde{\mathbf{y}})} = (\mathrm{diag}(\tilde{\mathbf{y}}))^{-1}(-\mathbf{y}),$$

Since $z_k = \sum_{j=1}^{d} W_{kj}x_j + b_k$, then the third term is

$$\frac{\partial \mathbf{z}}{\partial W_{ij}} = \begin{bmatrix} \frac{\partial z_1}{\partial W_{ij}} \\ \vdots \\ \frac{\partial z_i}{\partial W_{ij}} \\ \vdots \\ \frac{\partial z_c}{\partial W_{ij}} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ x_j \\ \vdots \\ 0 \end{bmatrix} = x_j \mathbf{e}_i$$

Putting them together gives,

$$\begin{aligned}
\left(\frac{\partial \tilde{\mathbf{y}}}{\partial \mathbf{z}} \frac{\partial\,\mathrm{CE}}{\partial \tilde{\mathbf{y}}}\right)^T \frac{\partial \mathbf{z}}{\partial W_{ij}} &= \left((\mathrm{diag}(\tilde{\mathbf{y}}) - \tilde{\mathbf{y}}\tilde{\mathbf{y}}^T)(\mathrm{diag}(\tilde{\mathbf{y}}))^{-1}(-\mathbf{y})\right)^T x_j \mathbf{e}_i \\
&= \left(\mathbf{I} \cdot (-\mathbf{y}) - \tilde{\mathbf{y}}\tilde{\mathbf{y}}^T (\mathrm{diag}(\tilde{\mathbf{y}}))^{-1}(-\mathbf{y})\right)^T x_j \mathbf{e}_i \\
&= \left(\mathbf{I} \cdot (-\mathbf{y}) + \tilde{\mathbf{y}}\mathbf{1}^T \mathbf{y})\right)^T x_j \mathbf{e}_i \\
&= x_j (\tilde{\mathbf{y}} - \mathbf{y})^T \mathbf{e}_i \\
&= x_j \mathrm{err_i}
\end{aligned}$$

where $\mathrm{err_i} = [\tilde{\mathbf{y}} - \mathbf{y}]_i$ denotes the i-th entry of the residual $(\tilde{\mathbf{y}} - \mathbf{y})$. Thus,

$$\frac{\partial \mathrm{CE}}{\partial \mathbf{W}} = \left(\frac{\partial \tilde{\mathbf{y}}}{\partial \mathbf{z}} \frac{\partial\, \mathrm{CE}}{\partial \tilde{\mathbf{y}}}\right)^T \frac{\partial \mathbf{z}}{\partial W} = (\tilde{\mathbf{y}} - \mathbf{y})\mathbf{x}^T$$

Similarly,

$$\frac{\partial \mathbf{z}}{\partial b_i} = \begin{bmatrix} \frac{\partial z_1}{\partial b_i} \\ \vdots \\ \frac{\partial z_i}{\partial b_i} \\ \vdots \\ \frac{\partial z_c}{\partial b_i} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ x_j \\ \vdots \\ 0 \end{bmatrix} = x_j \mathbf{e}_i$$

Then,

$$\begin{aligned}\left(\frac{\partial \tilde{\mathbf{y}}}{\partial \mathbf{z}} \frac{\partial\, \mathrm{CE}}{\partial \tilde{\mathbf{y}}}\right)^T \frac{\partial \mathbf{z}}{\partial b_i} &= \left((\mathrm{diag}(\tilde{\mathbf{y}}) - \tilde{\mathbf{y}}\tilde{\mathbf{y}}^T)(\mathrm{diag}(\tilde{\mathbf{y}}))^{-1}(-\mathbf{y})\right)^T \mathbf{e}_i \\ &= (\tilde{\mathbf{y}} - \mathbf{y})^T \mathbf{e}_i \\ &= \mathrm{err_i}.\end{aligned}$$

Thus,

$$\frac{\partial \mathrm{CE}}{\partial \mathbf{b}} = \left(\frac{\partial \tilde{\mathbf{y}}}{\partial \mathbf{z}} \frac{\partial\, \mathrm{CE}}{\partial \tilde{\mathbf{y}}}\right)^T \frac{\partial \mathbf{z}}{\partial \mathbf{b}} = \tilde{\mathbf{y}} - \mathbf{y}$$

Hence, the update for $\mathbf{W}, \mathbf{b}$ via SGD is

$$\begin{aligned}\mathbf{W} &\longleftarrow \mathbf{W} - \lambda(\tilde{\mathbf{y}} - \mathbf{y})\mathbf{x}^T \\ \mathbf{b} &\longleftarrow \mathbf{b} - \lambda(\tilde{\mathbf{y}} - \mathbf{y})\end{aligned}$$

3.6 Strong convexity and its implications

The definitions, the fourth and the fifth results follow [Garber and Hazan, 2015] which proposes a faster Frank-Wolfe method over strongly-convex sets. For the following definitions let $\mathbf{E}$ be a finite vector space and $\|\cdot\|, \|\cdot\|_*$ be a pair of dual norms over $\mathbf{E}$.

3.6.1 Definition on strongly convex sets

Definition 2 (strongly convex sets)**.** We say that a convex set $\mathcal{K} \subset \mathbf{E}$ is α-strongly convex with respect to $\|\cdot\|$ if for any $x, y \in \mathcal{K}$, any $\gamma \in [0,1]$ and any vector $z \in \mathbf{E}$ such that $\|\cdot\| = 1$, it holds that[Garber and Hazan, 2015]

$$\gamma x + (1-\gamma)y + \gamma(1-\gamma)\frac{\alpha}{2}\|x-y\|^2 z \in \mathcal{K}.$$

That is, $\mathcal{K}$ contains a ball of radius $\gamma(1-\gamma)\frac{\alpha}{2}\|x-y\|^2$ induced by the norm $\|\cdot\|$ centered at $\gamma x + (1-\gamma)y$.

3.6.2 Definition on strongly convex functions

Definition 3 (strongly convex functions)**.** We say that a function $f\colon \mathbf{dom}f \to \mathbb{R}$ is α-strongly convex over a convex set $\mathcal{K} \subset \mathbf{dom}f$ with respect to $\|\cdot\|$ if it satisfies the following two equivalent conditions[Garber and Hazan, 2015]

1. $\forall x, y \in \mathcal{K}$:
$$f(y) \geq f(x) + \nabla f(x)^T(y-x) + \frac{\alpha}{2}\|y-x\|^2. \tag{6}$$

2. $\forall x, y \in \mathcal{K}, \gamma \in [0,1]$:
$$f(\gamma x + (1-\gamma)y) \leq \gamma f(x) + (1-\gamma)f(y) - \frac{\alpha}{2}\gamma(1-\gamma)\|y-x\|^2.$$

3.6.3 Implications from strong convexity

We assume that f is strongly convex on a convex set S with a constant $\alpha > 0$. We have the following results,

(1) $f(x) - \frac{\alpha}{2}\|x\|_2^2$ is convex;

(2) $\nabla^2 f(x) \succeq \alpha I$ for all $x \in S$;

(3) $(\nabla f(y) - \nabla f(x))^T(y-x) \geq \alpha\|y-x\|_2^2$ for all x, y;

(4) If $x^* = \text{argmin}_{x\in S}$, then it holds that

$$f(x) - f(x^*) \geq \frac{\alpha}{2}\|x-x^*\|^2.$$

Proof. (1) By definition,

$$\begin{aligned}
f(y) &\geq f(x) + \nabla f(x)^T(y-x) + \frac{\alpha}{2}\|y-x\|_2^2 \\
&= f(x) + \nabla f(x)^T(y-x) + \frac{\alpha}{2}\|y\|_2^2 + \frac{\alpha}{2}\|x\|_2^2 - \alpha x^T y \\
f(y) - \frac{\alpha}{2}\|y\|_2^2 &\geq f(x) - \frac{\alpha}{2}\|x\|_2^2 + \alpha x^T x + \nabla f(x)^T(y-x) - \alpha x^T y \\
&= f(x) - \frac{\alpha}{2}\|x\|_2^2 + \nabla f(x)^T(y-x) - \alpha x^T(y-x) \\
&= f(x) - \frac{\alpha}{2}\|x\|_2^2 + (\nabla f(x) - \alpha x)^T(y-x),
\end{aligned}$$

which is exactly the definition of the convexity of $f(x) - \frac{\alpha}{2}\|x\|_2^2$. Given the convexity of $f(x) - \frac{\alpha}{2}\|x\|_2^2$, the above derivations still hold if we do from the end to the beginning. Thus, the strong convexity of $f(x)$ is equivalent to the convexity of $f(x) - \frac{\alpha}{2}\|x\|_2^2$.

(2) By the Taylor mean-value theorem, for all $x, y \in S$, we have

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

for some z on the line segment $[x, y]$. Since f is strongly convex on S, for any $x, y \in S$, we have

$$f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x) \geq f(x) + \nabla f(x)^T(y-x) + \frac{\alpha}{2}\|y-x\|_2^2$$

$$\Updownarrow$$

$$\frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x) \geq \frac{\alpha}{2}\|y-x\|_2^2 = \frac{1}{2}(y-x)^T\alpha I(y-x)$$

$$\Updownarrow$$

$$\nabla^2 f(z) \succeq \alpha I,$$

as desired.

(3) By the mean value theorem, we have

$$\nabla f(y) - \nabla f(x) = \nabla^2 f(z)(y-x)$$

for some z on the line segment $[x, y]$. Thus,

$$\begin{aligned}
(\nabla f(y) - \nabla f(x))^T(y-x) &= (y-x)^T\nabla^2 f(z)(y-x) \\
&\geq \alpha\|y-x\|_2^2
\end{aligned}$$

where the second line follows from the second result, i.e., $\nabla^2 f(z) \succeq \alpha I$ for all $z \in S$.

(4) By definition, we have

$$\begin{aligned}
f(x) &\geq f(x^*) + \nabla f(x^*)^T(x-x^*) + \frac{\alpha}{2}\|x-x^*\|^2 \\
&= f(x^*) + \mathbf{0}^T(x-x^*) + \frac{\alpha}{2}\|x-x^*\|^2 \qquad \Longrightarrow f(x) - f(x^*) \geq \frac{\alpha}{2}\|x-x^*\|^2 \\
&= f(x^*) + \frac{\alpha}{2}\|x-x^*\|^2
\end{aligned}$$

where the second line follows from the first-order optimality condition and the $\|\cdot\|$ is not restricted to $\|\cdot\|_2$. This completes our proof.

□

Remark 1. Due to the updown arrows in the proof of the second result, the first three implications are equivalent to the definition of strongly convex functions w.r.t. $\|\cdot\|_2$.

3.6.4 Strong convexity of quadratic functions

A direct consequence of the first result from the above subsection is the following corollary.

Corollary 1 (Strongly convex quadratic functions)**.** Given $Q \in \mathbb{R}^{n\times n}, b \in \mathbb{R}^n, c \in \mathbb{R}$, the following quadratic function

$$f(x) = \frac{1}{2}x^TQx + b^Tx + c$$

is α strongly convex if and only if

$$Q \succeq \alpha I$$

Proof. We have known that $f(x)$ is α strongly convex if and only if $f(x) - \frac{\alpha}{2}x^Tx$ is convex. Thus, $f(x) = \frac{1}{2}x^TQx + b^Tx + c$ is α strongly convex if and only if

$$\frac{1}{2}x^T(Q - \alpha I)x + b^Tx + c$$

is convex. In other words, $Q - \alpha I \succeq 0$, i.e., $Q \succeq \alpha I$, which means $\lambda_{\min}(Q) \geq \alpha$. □

Remark 2. In a nutshell, $f(x) = \frac{1}{2}x^TQx + b^Tx + c$ is strongly convex if and only if Q is positive definite in which case $\lambda_{\min}(Q)$ is its largest strong convexity constant.

3.6.5 An important result from strong convexity

This subsection is mostly taken from Section 9.1.2 of B & V's Convex Optimization.

The righthand side of the first inequality in the definition of strongly convex functions, i.e., Eq (6), is a convex quadratic function of y (for fixed x). Setting the gradient with respect to y equal to zero, we find that $\tilde{y} = x - (1/m)\nabla f(x)$ minimizes the righthand side. Therefore, we have

$$\begin{aligned} f(y) &\geq f(x) + \nabla f(x)^T(y - x) + \frac{m}{2}\|y - x\|^2 \\ &\geq f(x) + \nabla f(x)^T(\tilde{y} - x) + \frac{m}{2}\|\tilde{y} - x\|^2 \\ &= f(x) - \frac{1}{2m}\|\nabla f(x)\|_2^2 \end{aligned} \tag{7}$$

Since this holds for any $y \in S$, we get

$$p^* \geq f(x) - \frac{1}{2m}\|\nabla f(x)\|_2^2 \Leftrightarrow f(x) - p^* \leq \frac{1}{2m}\|\nabla f(x)\|_2^2 \tag{8}$$

We denote the optimal value, $\inf_x f(x) = f(x^*)$, as p^*. This inequality shows that if the gradient is small at a point, then the point is nearly optimal. The inequality (8) can also be interpreted as a condition for suboptimality which generalizes the optimality condition $\nabla f(x^*) = 0$:

$$\|\nabla f(x)\|_2 \leq \sqrt{2m\epsilon} \Rightarrow f(x) - p^* \leq \epsilon \tag{9}$$

We can also derive a bound on $\|x - x^*\|_2$, the distance between x and any optimal point x^*, in terms of $\|\nabla f(x)\|_2$:

$$\|x - x^*\|_2 \leq \frac{2}{m}\|\nabla f(x)\|_2 \tag{10}$$

To see this, we apply (6) with $y = x^*$ to obtain

$$\begin{aligned} p^* = f(x^*) &\geq f(x) + \nabla f(x)^T(x^* - x) + \frac{m}{2}\|x^* - x\|^2 \\ &\geq f(x) - \|\nabla f(x)\|_2\|x^* - x\|_2 + \frac{m}{2}\|x^* - x\|^2 \end{aligned}$$

where we use the Cauchy-Schwarz inequality in the second inequality. Since $p^* \leq f(x)$, we must have

$$-\|\nabla f(x)\|_2\|x^* - x\|_2 + \frac{m}{2}\|x^* - x\|^2 \leq 0,$$

from which (10) follows. One consequence of (10) is that the optimal point x^* is unique.

3.6.6 Connection with Newton decrement

We can connect this result with Newton decrement defined as

$$\lambda(x) = \left(\nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x)\right)^{1/2}.$$

From this, we have

$$\lambda(x)^2 = \nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x) \geq \frac{1}{M} \|\nabla f(x)\|_2^2.$$

Combining this with (8), we have

$$f(x) - p^* \leq \frac{M}{2m} \lambda(x)^2.$$

3.6.7 A result on $\|\nabla f(x)\|_*$ from quadratic functional growth (weaker than strong convexity)

Follow [Garber and Hazan, 2015] and derive the result by hand when I have time.

3.7 Lower and upper bounds on the Hessian $\nabla^2 f(x)$

First of all, we need to emphasize that the gradient-based optimization methods require a suitable starting point $x^{(0)}$. We follow Section 9.1 of [Boyd and Vandenberghe, 2004] to present the description about $x^{(0)}$.

The starting point must lie in $\textbf{dom} f$, and in addition the sublevel set

$$S = \{x \in \textbf{dom} f \mid f(x) \leq f(x^{(0)})\}$$

must be closed. This condition is satisfied for all $x^{(0)} \in \textbf{dom} f$ if f is closed, i.e., all its sublevel sets are closed. Continuous functions with $\textbf{dom} f = \mathbb{R}^n$ are closed, so if $\textbf{dom} f = \mathbb{R}^n$, the initial sublevel set condition is satisfied by any $x^{(0)}$. Another important class of closed functions are continuous functions with open domains, for which $f(x)$ tends to infinity as x approaches $\textbf{bd}\,\textbf{dom} f$.

For the lower bound on $\nabla^2 f(x)$, from Section 3.6.3, we know that strong convexity of f is equivalent to $\nabla^2 f(x) \succeq \alpha I$ for all $x \in S$.

For the upper bound on $\nabla^2 f(x)$, we follow Section 9.1.2 of [Boyd and Vandenberghe, 2004]. The inequality (6) implies that the sublevel sets contained in S are bounded, so in particular, S is bounded. To see this[3], for $x = x^*$, plugging this into (6) yields

$$f(y) \geq f(x^*) + \nabla f(x^*)^T (y - x^*) + \frac{m}{2} \|y - x^*\|^2$$

Since x^* is a global minimizer, then $\nabla f(x^*) = 0$. Thus,

$$f(y) \geq f(x^*) + \frac{m}{2} \|y - x^*\|^2$$

Combining this with the definition of $x^{(0)}$, we get

$$f(x^{(0)}) \geq f(y) \geq f(x^*) + \frac{m}{2} \|y - x^*\|^2$$

which gives

$$\|y - x^*\|^2 \leq \frac{2}{m} (f(x^{(0)}) - f(x^*)), \quad \forall y \in S$$

which shows the boundedness of S.

Therefore the maximum eigenvalue of $\nabla^2 f(x)$, which is a continuous function of x on S, is bounded above on S, i.e., there exists a constant M such that

$$\nabla^2 f(x) \preceq MI$$

for all $x \in S$. To see this[4], let $g(x) = \lambda_{\max}(\nabla^2 f(x)) = \|\nabla^2 f(x)\|_2$. Clearly, $g(x)$ is continuous on S. Since S is closed and bounded, then S is compact. By the extreme value theorem, the maximum value of $g(x)$ on S can be attained and we denote it by M. Thus, $\|\nabla^2 f(x)\|_2 \leq M$.

[3] https://math.stackexchange.com/questions/993357/boundedness-of-sublevel-sets-of-convex-function?rq=1

[4] https://math.stackexchange.com/questions/2909563/boundedness-of-sublevelsets-of-strongly-convex-functions-implies-boundedn

Actually, the least M is the largest eigenvalue of $\nabla^2 f(x)$. Now a question comes naturally: is this M equivalent to the Lipschitz constant L for $\nabla f(x)$ in section 3.2? I think the answer is yes. (see Theorem 4.20 in Amir Beck's Introduction to nonlinear optimization) This upper bound on the Hessian implies for any $x, y \in S$,

$$f(y) \leq f(x) + \nabla f(x)^T (y - x) + \frac{M}{2}\|y - x\|^2 \tag{11}$$

which is analogous to (6). Minimizing each side over y yields

$$p^* \leq f(x) - \frac{1}{2M}\|\nabla f(x)\|_2^2$$

the counterpart of (8). Altogether,

$$f(x) - \frac{1}{2m}\|\nabla f(x)\|_2^2 \leq p^* \leq f(x) - \frac{1}{2M}\|\nabla f(x)\|_2^2$$

or

$$\sqrt{2m(f(x) - p^*)} \leq \|\nabla f(x)\|_2 \leq \sqrt{2M(f(x) - p^*)}$$

3.8 A useful equality

Let the update be $x^{(k)} = x^{(k-1)} - t\nabla f(x^{(k-1)})$. We substitute the update into the LHS of the following equality and expand, then we have

$$\|x^{(k-1)} - x^*\|_2^2 - \|x^{(k)} - x^*\|_2^2 = 2t\left(\nabla f(x^{(k-1)})^T (x^{(k-1)} - x^*) - \frac{t}{2}\|\nabla f(x^{(k-1)})\|_2^2\right) \tag{12}$$

3.9 A useful inequality

Let f be convex. Using the above equality and the convexity of f, we have

$$\|x^{(k)} - x^*\|_2^2 \leq \|x^{(k-1)} - x^*\|_2^2 - 2t(f(x^{(k-1)}) - f(x^*)) + t^2\|\partial f(x^{(k-1)})\|_2^2 \tag{13}$$

where $\partial f(x)$ denotes the subdifferential of $f(x)$ at x. If $f(x)$ is differential, $\partial f(x)$ will be $\nabla f(x)$.

4 Gradient descent

The following is taken from CMU's Convex optimization course in which Ryan Tibshirani is the instructor.

Consider unconstrained, smooth convex optimization

$$\min_x f(x)$$

i.e., f is convex and differentiable with $\text{dom}(f) = \mathbf{R}^n$. Denote the optimal criterion value by $f^* = \min_x f(x)$, and a solution by x^*?

Gradient descent: choose initial point $x^{(0)} \in \mathbf{R}^n$, repeat:

$$x^{(k)} = x^{(k-1)} - t_k \cdot \nabla f(x^{(k-1)}), \quad k = 1, 2, 3, \cdots$$

Stop at some point.

Gradient descent interpretation: At each iteration, consider the expansion

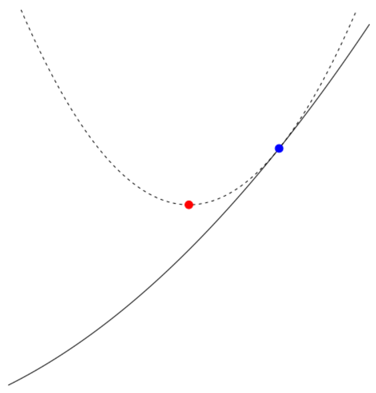
$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2t}\|y - x\|_2^2$$

As shown in Fig 1, We can think of the above as quadratic approximation by replacing the usual Hessian by $\frac{1}{2t}I$. $f(x) + \nabla f(x)^T (y - x)$ is the linear approximation. $\frac{1}{2t}\|y - x\|_2^2$ is a proximity term to x with a weight $\frac{1}{2t}$.

4.1 Backtracking line search

One way to adaptively choose the step size is to use backtracking line search:

- Fix parameters $0 < \beta \leq 1$ and $0 < \alpha \leq 0.5$



Blue point is x, red point is

$$x^+ = \underset{y}{\text{argmin}}\ f(x) + \nabla f(x)^T(y-x) + \frac{1}{2t}\|y-x\|_2^2$$

Figure 1: Quadratic approximation

- At each iteration, start with $t = 1$, and while

  $$f(x - t\nabla f(x)) > f(x) - \alpha t\|\nabla f(x)\|_2^2$$

  shrink $t = \beta t$. Else perform gradient descent update

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

Backtracking line search is simple and tends to work well. We can take further simplification by taking $\alpha = 0.5$. Backtracking line search interpretation is presented in Fig 2 in which $\Delta x = -\nabla f(x)$. As t moves from right to left, the backtracking procedure stops when $f(x + t\Delta x)$ is under the line $f(x) + \alpha t\nabla f(x)^T\Delta x$, i.e., $f(x - t\nabla f(x) \leq f(x) - \alpha t\|\nabla\|_2^2$, which guarantees the descent in $f(x)$.

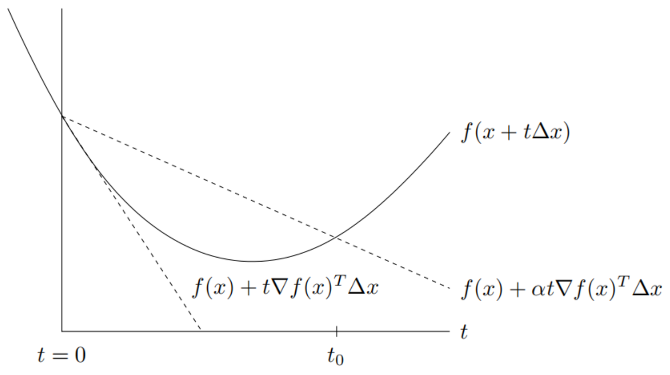

$f(x + t\Delta x)$
$f(x) + t\nabla f(x)^T\Delta x$
$f(x) + \alpha t\nabla f(x)^T\Delta x$
t
$t = 0$
t_0

Figure 2: Backtracking line search

4.2 Convergence analysis

Assume that f convex and differentiable, with $\text{dom}(f) = \mathbf{R}^n$, and additionally $\|f(x) - f(y)\|_2 \leq L\|x - y\|_2$ for any x,y i.e., ∇f is Lipschitz continuous with constant $L > 0$.

Theorem 1. When f is convex and differentiable, and ∇f is Lipschitz continuous with constant $L > 0$, gradient descent with fixed step size $t \leq 1/L$ satisfies

$$f(x^{(k)}) - f(x^*) \leq \frac{\|x^{(0)} - x^*\|_2^2}{2tk}$$

and same result holds for backtracking, with t replaced by β/L

Proof. Since $\nabla f(x)$ is Lipschitz continuous with constant $L > 0$, we have the result from (4)

$$f(x^+) \leq f(x) - \frac{t}{2}\|\nabla f(x)\|_2^2 \quad (0 < t \leq 1/L) \tag{14}$$

which means $f(x^{(k)}) \leq f(x^{(k-1)}) \leq \cdots \leq f(x^{(0)})$. By the convexity of f, we have

$$f(x^*) \geq f(x) + \nabla f(x)^T(x^* - x)) \Leftrightarrow f(x) \leq f(x^*) + \nabla f(x)^T(x - x^*) \tag{15}$$

Combine (14) and (15),

$$f(x^+) \leq f(x^*) + \nabla f(x)^T(x - x^*) - \frac{t}{2}\|\nabla f(x)\|_2^2 \tag{16}$$

Using (12), we get

$$\frac{1}{2t}(\|x - x^*\|_2^2 - \|x^+ - x^*\|_2^2) = \nabla f(x)^T(x - x^*) - \frac{t}{2}\|\nabla f(x)\|_2^2 \tag{17}$$

Substituting this result into (16).

$$\begin{aligned} f(x^+) - f(x^*) &\leq \nabla f(x)^T(x - x^*) - \frac{t}{2}\|\nabla f(x)\|_2^2 \\ &= \frac{1}{2t}(\|x - x^*\|_2^2 - \|x^+ - x^*\|_2^2) \end{aligned} \tag{18}$$

By specifying the indices, we have $f(x^{(i)}) - f(x^*) \leq \frac{1}{2t}(\|x^{(i-1)} - x^*\|_2^2 - \|x^{(i)} - x^*\|_2^2)$ and sum both sides over i.

$$\sum_{i=1}^{k}\Big(f(x^{(i)}) - f(x^*)\Big) \leq \frac{1}{2t}\sum_{i=1}^{k}(\|x^{(i-1)} - x^*\|_2^2 - \|x^{(i)} - x^*\|_2^2) = \frac{1}{2t}(\|x^{(0)} - x^*\|_2^2 - \|x^{(k)} - x^*\|_2^2) \leq \frac{1}{2t}\|x^{(0)} - x^*\|_2^2 \tag{19}$$

From (4), we have $f(x^{(k)}) \leq f(x^{(i)})$ for any $i \leq k$. Thus, $f(x^{(k)}) - f(x^*) \leq f(x^{(i)}) - f(x^*)$. We combine this with (19) to get

$$k(f(x^{(k)}) - f(x^*)) \leq \sum_{i=1}^{k}(f(x^{(i)}) - f(x^*)) \leq \frac{1}{2t}\|x^{(0)} - x^*\|_2^2$$

Therefore,

$$f(x^{(k)}) - f(x^*) \leq \frac{\|x^{(0)} - x^*\|_2^2}{2tk}$$

□

We say gradient descent has convergence rate $O(1/k)$, i.e., it finds ϵ-suboptimal point in $O(1/\epsilon)$ iterations.

Theorem 2. When f is convex and differentiable, and ∇f is Lipschitz continuous with constant $L > 0$, gradient descent with backtracking line search satisfies

$$f(x^{(k)}) - f(x^*) \leq \frac{\|x^{(0)} - x^*\|_2^2}{2t_{min}k}$$

where $t_{min} = \min\{1, \beta/L\}$.

Proof. Using the result from Lipschitz gradients, i.e., (4), we have,

$$\begin{aligned} f(x^+) &\leq f(x) - \frac{t}{2}\|\nabla f(x)\|_2^2 \\ &\leq f(x) - \alpha t\|\nabla f(x)\|_2^2 \end{aligned}$$

since α. Hence, the backtracking exit condition will be satisfied whenever $0 \leq t \leq 1/L$. Therefore the backtracking line search terminates either with $t = 1$ or with a value $t \geq \beta/L$. This provides a lower bound on the decrease in the objective function. In the first case we have

$$f(x^+) \leq f(x) - \alpha\|\nabla f(x)\|_2^2$$

and in the second case we have

$$f(x^+) \leq f(x) - (\beta\alpha/L)\|\nabla f(x)\|_2^2$$

Putting these together, we always have

$$f(x^+) \leq f(x) - \alpha\min\{1, \beta/L\}\|\nabla f(x)\|_2^2 \tag{20}$$

Hence, the minimum of t is $\min\{1, \beta/L\}$ in the backtracking line search case. Using this result and (18) obtained in the proof of the fixed step size case, we have

$$f(x^{(i)}) - f(x^*) \leq \frac{1}{2t_i}(\|x^{(i-1)} - x^*\|_2^2 - \|x^{(i)} - x^*\|_2^2) \leq \frac{1}{2\min\{1, \beta/L\}}(\|x^{(i-1)} - x^*\|_2^2 - \|x^{(i)} - x^*\|_2^2)$$

Then the rest of this proof follows the fixed step size case. □

Theorem 3. When f is strongly convex with $\nabla^2 f(x) \succeq mI$, and ∇f is Lipschitz continuous with constant $L > 0$, gradient descent with fixed step size t or with backtracking line search search satisfies $t \leq 2/(m + L)$

$$f(x^{(k)}) - f^* \leq c^k \frac{L}{2}\|x^{(0)} - x^*\|_2^2$$

where $0 < c < 1$

Proof. From (4)(a result from Lipschitz gradients), i.e., $f(x^+) \leq f(x) - \frac{t}{2}\|\nabla f(x)\|_2^2$, we have

$$\begin{aligned} &f(x^{(k+1)}) - f(x^{(k)}) \leq -\frac{t}{2}\|\nabla f(x^{(k)})\|_2^2 \\ \Rightarrow &f(x^{(k+1)}) - f(x^*) - (f(x^{(k)}) - f(x^*)) \leq -\frac{t}{2}\|\nabla f(x^{(k)})\|_2^2 \end{aligned} \tag{21}$$

From the strong convexity of f, we have

$$\begin{aligned} &f(x^*) \geq f(x^{(k)}) + \nabla f(x^{(k)})^T(x^* - x^{(k)}) + \frac{m}{2}\|x^* - x^{(k)}\|_2^2 \\ \Rightarrow &f(x^{(k)}) - f(x^*) \leq -\nabla f(x^{(k)})^T(x^* - x^{(k)}) - \frac{m}{2}\|x^* - x^{(k)}\|_2^2 \end{aligned} \tag{22}$$

Using (17), we get

$$-\nabla f(x^{(k)})^T(x^* - x^{(k)}) = \frac{1}{2t}(\|x^{(k)} - x^*\|_2^2 - \|x^{(k+1)} - x^*\|_2^2) + \frac{t}{2}\|\nabla f(x^{(k)})\|_2^2 \tag{23}$$

Substitute this into (22),

$$f(x^{(k)}) - f(x^*) \leq \frac{1}{2t}(\|x^{(k)} - x^*\|_2^2 - \|x^{(k+1)} - x^*\|_2^2) + \frac{t}{2}\|\nabla f(x^{(k)})\|_2^2 - \frac{m}{2}\|x^* - x^{(k)}\|_2^2 \tag{24}$$

Add (21) and (24) together,

$$\begin{aligned} f(x^{(k+1)}) - f(x^*) &\leq \frac{1}{2t}(\|x^{(k)} - x^*\|_2^2 - \|x^{(k+1)} - x^*\|_2^2) - \frac{m}{2}\|x^* - x^{(k)}\|_2^2 \\ &= (\frac{1}{2t} - \frac{m}{2})\|x^* - x^{(k)}\|_2^2 - \frac{1}{2t}\|x^{(k+1)} - x^*\|_2^2 \end{aligned} \tag{25}$$

Rearrange the above,

$$\begin{aligned} \|x^{(k+1)} - x^*\|_2^2 &\leq (1 - mt)\|x^* - x^{(k)}\|_2^2 - 2t(f(x^{(k+1)}) - f(x^*)) \\ &\leq (1 - mt)\|x^* - x^{(k)}\|_2^2 \\ &\leq (1 - mt)^{k+1}\|x^* - x^{(0)}\|_2^2 \end{aligned} \tag{26}$$

which means $\|x^{(k)} - x^*\|_2^2 \leq (1 - mt)^k\|x^* - x^{(0)}\|_2^2$. From (25) we get

$$\begin{aligned} f(x^{(k)}) - f(x^*) &\leq (\frac{1}{2t} - \frac{m}{2})\|x^{(k-1)} - x^*\|_2^2 \\ &\leq \frac{(1 - mt)}{2t}(1 - mt)^{k-1}\|x^* - x^{(0)}\|_2^2 \\ &= (1 - mt)^k \frac{1}{2t}\|x^* - x^{(0)}\|_2^2 \end{aligned} \tag{27}$$

where $t \leq 1/L$.??When $t = 1/L$, we have

$$f(x^{(k)}) - f(x^*) \leq (1 - \frac{m}{L})^k \frac{L}{2}\|x^* - x^{(0)}\|_2^2 = c^k \frac{L}{2}\|x^* - x^{(0)}\|_2^2$$

where $c = (1 - \frac{m}{L})^k$. □

The convergence rate under strong convexity is $O(c^k)$, exponentially fast. That is to say, we find an ϵ-suboptimal point in $O(\log(\frac{1}{\epsilon}))$ iterations. This is called linear convergence, because looks linear on a semi-log plot. Contraction factor c in rate depends adversely on condition number L/m: higher condition number $\Rightarrow$ slower rate.

What about nonconvex functions? Assume f is differentiable with Lipschitz gradient as before, but now nonconvex. Asking for optimality is too much. So we'll settle for x such that $\|\nabla f(x)\|_2 \leq \epsilon$, called ϵ-stationarity.

Theorem 4. If f is differentiable with Lipschitz gradient, gradient descent with fixed step size $t \leq 1/L$ satisfies

$$\min_{i=0,\cdots,k} \|\nabla f(x^{(i)})\|_2 \leq \sqrt{\frac{2(f(x^{(0)}) - f(x^*))}{t(k+1)}}$$

Thus gradient descent has rate $O(1/\sqrt{k})$, or $O(1/\epsilon^2)$, even in the nonconvex case for finding stationary points. This rate cannot be improved (over class of differentiable functions with Lipschitz gradients) by any deterministic algorithm[5].

Proof. Using the result about Lipschitz gradients, i.e., we have

$$\|\nabla f(x^{(i)})\|_2^2 \leq \frac{2}{t}(f(x^{(i)}) - f(x^{(i+1)})) \quad (0 < t \leq 1/L) \tag{28}$$

Summing over i,

$$\begin{aligned}
&\sum_{i=0}^{k} \|\nabla f(x^{(i)})\|_2^2 \leq \sum_{i=0}^{k} \frac{2}{t}(f(x^{(i)}) - f(x^{(i+1)})) \\
\Rightarrow &(k+1) \min_i \|\nabla f(x^{(i)})\|_2^2 \leq \frac{2}{t}(f(x^0) - f(x^{(k)})) \leq \frac{2}{t}(f(x^0) - f(x^*)) \\
\Rightarrow &\min_i \|\nabla f(x^{(i)})\|_2 \leq \sqrt{\frac{2(f(x^0) - f(x^*)}{t(k+1)}}
\end{aligned} \tag{29}$$

□

5 Subgradient methods

5.1 Subgradients

Subgradients are alternatives to gradients when the function f is non-smooth or non-differentiable. A subgradient of a convex function f at x is any $g \in \mathbf{R}^n$ such that, for all x, y:

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

- g always exists on the relative interior of $\text{dom} f$.
- If f differentiable at x, then $g = \nabla f(x)$ uniquely.
- The same definition works for nonconvex f (however, subgradients need not exist).

5.1.1 Subgradients and sublevel sets

If g is a subgradient of f at x, then

$$f(y) \leq f(x) \Longrightarrow g^T(y - x) \leq 0 \tag{30}$$

which follows from the definition of subgradients,

$$\begin{gathered}
f(y) \geq f(x) + g^T(y - x) \\
g^T(y - x) \leq f(y) - f(x) \leq 0 \quad (\because f(y) \leq f(x))
\end{gathered}$$

(30) shows that the nonzero subgradients at x define supporting hyperplanes to the sublevel set[6] as illustrated in Fig. 3

$$\{y \mid f(y) \leq f(x)\}$$

[5] Carmon et al. (2017), "Lower bounds for finding stationary points I"
[6] http://www.seas.ucla.edu/~vandenbe/236C/lectures/subgradients.pdf

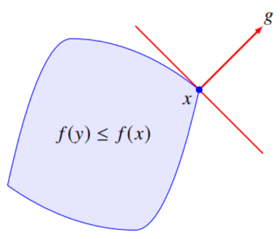


Figure 3: Subgradient vs sublevel set.

5.2 Subdifferential

The set of all subgradients of convex f is called the subdifferential:

$$\partial f = \{g \in \mathbf{R}^n : g \text{ is a subgradient of } f \text{ at } x\}$$

- Nonempty (only for convex f)
- $\partial f(x)$ is closed, convex, and bounded (even for nonconvex f).
- If f is differentiable at x, then $\partial f(x) = \{\nabla f(x)\}$.
- If $\partial f(x) = \{g\}$, then f is differentiable at x and $\{\nabla f(x)\} = g$.

Proposition 1. If f is convex, $\partial f(x)$ is nonempty when $x \in \operatorname{intdom} f$.

Proof. Refer to page 5 of `http://www.seas.ucla.edu/~vandenbe/236C/lectures/subgradients.pdf`. This will be done later. □

Proposition 2. Given a function $f(x)$ and $x \in \operatorname{intdom} f$, even if it is nonconvex, $\partial f(x)$ is closed, convex, and bounded. [7]

Proof. By the definition of subdifferential,

$$\partial f(x) := \{g \in \mathbb{R}^n \mid f(y) \geq f(x) + g^T(y-x), \forall y\}$$

It is straightforward to see that $D_y = \{g \mid f(y) - f(x) \geq g^T(y-x)\}$ is a closed half space. Since $\partial f(x) = \cap D_y$, then $\partial f(x)$ is closed and convex.

If f is finite on some open set U containing x, then there is some open $V \subset U$ containing x and some L such that f is Lipschitz of rank L on V. In particular, we have $|f(y) - f(x)| \leq L\|y - x\|$. Thus, let $B = \{g \mid g^{\mathrm{T}}(y-x) \leq f(y) - f(x) \leq L\|y-x\|, \forall y\}$, that is to say, $B = \{g \mid g^{\mathrm{T}}d \leq L\|d\|, \forall d\}$ which implies $\|g\| \leq L$. Hence, $\partial f(x)$ is bounded. □

There is a different proof at page 6 of `http://www.seas.ucla.edu/~vandenbe/236C/lectures/subgradients.pdf`.

5.2.1 Monotonicity of the subdifferential

[8] The subdifferential of a convex function is a monotone operator: given $u = \partial f(x)$ and $v = \partial f(y)$, the following holds

$$(x-y)^T(u-v) \geq 0$$

Proof. By the definition of subgradient, we have

$$f(y) \geq f(x) + u^T(y-x), \quad f(x) \geq f(y) + v^T(x-y)$$

Adding up these two inequalities, we have

$$f(y) + f(x) \geq f(x) + f(y) + u^T(y-x) + v^T(x-y) \iff 0 \geq u^T(y-x) + v^T(x-y) \iff (u-v)^T(x-y) \geq 0.$$

□

[7] https://math.stackexchange.com/questions/1113693/subdifferential-is-closed-convex-and-bounded
[8] `http://www.seas.ucla.edu/~vandenbe/236C/lectures/subgradients.pdf`

5.3 Optimality conditions

This subsection follows L. Vandenburghe's lecture slides on subgradients[8] with my own understandings.

5.3.1 Optimality conditions for unconstrained problems

For any f (convex or not),

$$f(x^*) = \min_x f(x) \Leftrightarrow 0 \in \partial f(x^*) \tag{31}$$

I.e., x^* is a minimizer if and only if 0 is a subgradient of f at x^*. This is called the subgradient optimality condition. This follows directly from the definition of subgradient:

Proof. If x^* is a minimizer, by the definition of subdifferential, $0 \in \partial f(x^*)$. For the converse, if $0 \in \partial f(x^*)$, $f(y) \geq f(x^*)$, $\forall y$, which still follows from the definition of subdifferential.

$$\begin{array}{l} f(y) \geq f(x^*), \quad \forall y \\ f(y) \geq f(x^*) + 0^T(y - x^*), \quad \forall y \end{array} \iff \quad 0 \in \partial f(x^*)$$

□

5.3.2 Optimality conditions for unconstrained problems

$$\begin{array}{ll} \text{minimize}_x & f_0(x) \\ \text{subject to} & f_i(x), \quad i = 1, 2, \ldots, m \end{array}$$

Assume $\text{intdom} f_i \in \mathbb{R}^n$, so f_i is subdifferentiable everywhere.

Karush–Kuhn–Tucker conditions

If strong duality holds, then x^*, λ^* are primal, dual optimal if and only if

1. x^* is primal feasible
2. $\lambda^* \succeq 0$
3. $\lambda_i^* f_i(x^*) = 0$, for $i = 1, 2, \ldots, m$
4. x^* is a minimizer of $L(x, \lambda^*) = f_0(x) + \sum_{i=1}^m \lambda_i^* f_i(x)$:

$$0 \in \partial f_0(x) + \sum_{i=1}^m \lambda_i^* \partial f_i(x)$$

5.4 Subgradient calculus

Basic rules for convex functions:

1. Scaling: $\partial(af) = a \cdot \partial f$ provided $a > 0$
2. Addition: $\partial(f_1 + f_2) = \partial f_1 + \partial f_2$
3. Affine composition: if $g(x) = f(Ax + b)$, then $\partial g(x) = A^{\mathrm{T}} \partial f(Ax + b)$
4. Finite pointwise maximum: if $f(x) = \max_{i=1,\ldots,m} f_i(x)$, then

$$\partial f(x) = \text{conv}\left(\bigcup_{i: f_i(x) = f(x)} \partial f_i(x) \right)$$

   convex hull of union of subdifferentials of active functions at x.

5. General pointwise maximum: if $f(x) = \max_{s \in S} f_s(x)$, then

$$\partial f(x) \supseteq \text{cl}\left\{ \text{conv}\left(\bigcup_{i: f_i(x) = f(x)} \partial f_i(x) \right) \right\}$$

   under some regularity conditions (on S, f_s), we get equality.

6. Norms: let $\|\cdot\|$ be a norm on $\mathbb{R}^n$. The associated **dual norm**, denoted $\|\cdot\|_\star$, is defined as

$$\|x\|_\star = \sup\{x^{\mathrm{T}} z \mid \|z\| \leq 1\}$$

where max and sup are used interchangeably. Then we have the following important result,

$$\partial\|x\|_\star = \underset{\|z\|\leq 1}{\operatorname{argmax}}\ x^{\mathrm{T}} z$$

Proof. Suppose that the given norm is $\|x\|_p$ with $p \geq 1$, then its dual norm is

$$\|x\|_\star = \|x\|_q = \max_{\|z\|_p\leq 1}\ x^T z$$

where $\frac{1}{p} + \frac{1}{q} = 1$. This is a known result. We also know the following famous Hölder inequality.

$$x^T z \leq \|x\|_p \|z\|_q.$$

Thus, we have $x^T z \leq \|x\|_\star \|z\|$. We first show $\partial\|x\|_\star \subset \operatorname{argmax}_{\|z\|\leq 1}\ x^T z$. If $g \in \partial\|x\|_\star$, we substitute $y = 0$ and $y = 2x$ into the definition of $\partial\|x\|_\star$, i.e. $\partial\|x\|_\star = \{g \in \mathbb{R}^n \mid \|y\|_\star \geq \|x\|_\star + g^T(y - x)\}$, respectively. We get $g^T x \geq \|x\|_\star$ and $g^T x \leq \|x\|_\star$, which gives $g^T x = \|x\|_\star$. Substituting this into the definition of $\partial\|x\|_\star$, we get

$$\begin{aligned}\|y\|_\star &\geq \|x\|_\star + g^T y - g^T x\} \\ &= \|x\|_\star + g^T y - \|x\|_\star \\ &= g^T y\end{aligned}$$

By Hölder inequality, we have $\|y\|_\star \|g\| \geq g^T y$ for all y. To ensure $\|y\|_\star \geq g^T y$ for all y, we need to let $\|y\|_\star \|g\| \leq \|y\|_\star$ which gives $\|g\| \leq 1$. Now $\|g\| \leq 1$ and $g^T x = \|x\|_\star$ are obtained, which demonstrates $g \in \operatorname{argmax}_{\|z\|\leq 1}\ x^T z$ by definition, i.e. $\partial\|x\|_\star \subset \operatorname{argmax}_{\|z\|\leq 1}\ x^T z$.
Now we show $\operatorname{argmax}_{\|z\|\leq 1}\ x^T z \subset \partial\|x\|_\star$. Suppose $z_+ \in \operatorname{argmax}_{\|z\|\leq 1}\ x^T z$, then $\|x\|_\star = x^T z_+$ and $\|z_+\| \leq 1$. We have

$$\begin{aligned}\|x\|_\star + z_+^T(y - x) &= \|x\|_\star + z_+^T y - x^T z_+ \\ &= \|x\|_\star + z_+^T y - \|x\|_\star \\ &= z_+^T y \\ &\leq \|z_+^T\| \|y\|_\star \text{ (use Hölder inequality)} \\ &\leq \|y\|_\star \quad (\because \|z_+\| \leq 1)\end{aligned}$$

which gives $\|y\|_\star \geq \|x\|_\star + z_+^T(y - x)$, i.e., $z_+ \in \partial\|x\|_\star$. This shows $\operatorname{argmax}_{\|z\|\leq 1}\ x^T z \subset \partial\|x\|_\star$. Hence, $\partial\|x\|_\star = \operatorname{argmax}_{\|z\|\leq 1}\ x^T z$. □

The following is my own way to prove $\operatorname{argmax}_{\|z\|\leq 1}\ x^T z \subset \partial\|x\|_\star$.

Proof. Suppose the maximizer of the above maximization problem is z_+, then we get $\|x\|_\star = x^{\mathrm{T}} z_+$. Thus,

$$\partial_x\|x\|_\star = z_+ = \underset{\|z\|\leq 1}{\operatorname{argmax}}\ x^{\mathrm{T}} z$$

□

An important special case, $f(x) = \|x\|_p$. Let q be such that $1/p + 1/q = 1$, then

$$\|x\|_p = \max_{\|z\|_q\leq 1}\ x^{\mathrm{T}} z$$

And

$$\partial f(x) = \underset{\|z\|_q\leq 1}{\operatorname{argmax}}\, x^{\mathrm{T}} z$$

5.5 Subgradient method

Now consider f convex, having $\mathrm{dom}(f) = \mathbf{R}^n$, but not necessarily differentiable.

Subgradient method: like gradient descent, but replacing gradients with subgradients, i.e., initialize $x^{(0)}$, repeat:

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

where $g^{(k-1)} \in \partial f(x^{(k-1)})$, any subgradient of f at $x^{(k-1)}$.

Subgradient method is not necessarily a descent method, so we keep track of best iterate $x^{(k)}$ best among $x^{(0)}, \cdots, x^{(k)}$ so far, i.e.,

$$f(x^{(k)}_{\text{best}}) = \min_{i=0,\cdots,k} f(x^{(i)})$$

5.6 Step size choices

- Fixed step size;
- Diminishing step sizes: choose to meet conditions

$$\sum_{k=1}^{\infty} t_k^2 < \infty, \quad \sum_{k=1}^{\infty} t_k = \infty$$

  i.e., square summable but not summable. Important here that step sizes go to zero, but not too fast.
- Polyak step sizes: when the optimal value f is known, take

$$t_k = \frac{f(x^{(k-1)}) - f(x^{(*)})}{\|g^{(k-1)}\|_2^2}, \quad k = 1, 2, 3, \cdots$$

  this can be motivated from minimizing the RHS of (12). In this case, the convergence rate is $O(1/\epsilon^2)$.

5.7 Convergence analysis

Assume that f convex, $\mathrm{dom}(f) = \mathbf{R}^n$, and also that f is Lipschitz continuous with constant $G > 0$, i.e., $\|f(x) - f(y)\| \leq G\|x - y\|_2$ for all x, y.

Theorem 5. For a fixed step size t, subgradient method satisfies

$$\lim_{k\to\infty} f(x^{(k)}_{\text{best}}) \leq f^* + G^2 t/2$$

Theorem 6. For diminishing step sizes, subgradient method satisfies

$$\lim_{k\to\infty} f(x^{(k)}_{\text{best}}) = f^*$$

Proof. By iterating over the useful inequality (13), we get

$$\|x^{(k)} - x^*\|_2^2 \leq \|x^{(0)} - x^*\|_2^2 - 2\sum_{i=1}^{k} t_i \left(f(x^{(i-1)}) - f(x^*)\right) + \sum_{i=1}^{k} t_i^2 \|\partial f(x^{(i-1)})\|_2^2$$

where $R = \|x^{(0)} - x^*\|_2$. Since f is Lipschitz with constant $G > 0$, then $\|\partial f\|_2 \leq G$. So,

$$\begin{aligned}
0 \leq \|x^{(k)} - x^*\|_2^2 &\leq R^2 - 2\sum_{i=1}^{k} t_i(f(x^{(i-1)}) - f(x^*)) + G^2 \sum_{i=1}^{k} t_i^2 \\
(f(x^{(k)}_{\text{best}}) - f(x^*))2\sum_{i=1}^{k} t_i &\leq 2\sum_{i=1}^{k} t_i(f(x^{(i-1)}) - f(x^*)) \leq R^2 + G^2 \sum_{i=1}^{k} t_i^2 \\
f(x^{(k)}_{\text{best}}) - f(x^*) &\leq \frac{R^2 + G^2 \sum_{i=1}^{k} t_i^2}{2\sum_{i=1}^{k} t_i} \\
f(x^{(k)}_{\text{best}}) &\leq f(x^*) + \frac{R^2}{2\sum_{i=1}^{k} t_i} + \frac{G^2 \sum_{i=1}^{k} t_i^2}{2\sum_{i=1}^{k} t_i} \qquad (32) \\
(\text{For fixed } t) \quad f(x^{(k)}_{\text{best}}) &\leq f(x^*) + \frac{R^2}{2kt} + \frac{G^2 t}{2} \\
(\text{As } k \to \infty) \quad f(x^{(k)}_{\text{best}}) &\leq f(x^*) + \frac{G^2 t}{2} \\
(\text{For diminishing } t, \sum_{i=1}^{k} t_i^2 < \infty, \sum_{i=1}^{k} t_i = \infty) \quad f(x^{(k)}_{\text{best}}) &\leq f(x^*) + 0 + 0 = f(x^*)
\end{aligned}$$

where $f(x_{\text{best}}^{(k)}) = \min_{i=1,\cdots,k} f(x^{(i-1)})$. □

With fixed step size t, the sixth line of the above proof gives

$$f(x_{\text{best}}^{(k)}) - f(x^*) \leq \frac{R^2}{2kt} + \frac{G^2 t}{2} \tag{33}$$

For this to be $\leq \epsilon$, let's make each term $\leq \epsilon/2$. So we get $t \leq \epsilon/G^2$ and $k \geq R^2G^2/\epsilon^2$. Hence, subgradient method has convergence rate of $O(1/\epsilon^2)$ which is slower than $O(1/\epsilon)$ for the gradient descent method.

5.8 Projected subgradient method

To optimize a convex function f over a convex set C,

$$\min_x f(x), \text{ subject to } x \in C$$

we can use the projected subgradient method. Just like the usual subgradient method, except we project onto C at each iteration:

$$x^{(k)} = P_C(x^{(k-1)} - t_k \cdot g(x^{(k-1)})) \quad k = 1, 2, 3, \cdots$$

Assuming we can do this projection, we get the same convergence guarantees as the usual subgradient method, with the same step size choices.

5.9 Can we do better?

Upside of the subgradient method: broad applicability. Downside: $O(1/\epsilon^2)$ convergence rate over problem class of convex, Lipschitz functions is really slow. Nonsmooth first-order methods: iterative methods updating $x^{(k)}$ in

$$x^{(0)} + span\{g^{(0), g^{(1)}, \cdots, g^{(k-1)}}\}$$

where subgradients $g^{(0)}, g^{(1)}, \cdots, g^{(k-1)}$ come from weak oracle.

Theorem 7 (Nesterov)**.** For any $k \leq n-1$ and starting point $x^{(0)}$, there is a function in the problem class such that any nonsmooth first-order method satisfies

$$f(x^{(k)}) - f(x^*) \geq \frac{RG}{2(1 + \sqrt{k+1})}$$

As a result we cannot do better than $O(1/\epsilon^2)$.

6 Proximal gradient descent

6.1 Composite functions

Suppose

$$f(x) = g(x) + h(x)$$

- g is convex, differentiable, $\text{dom}(g) = \mathbf{R}^n$
- h is convex, not necessarily differentiable

If f were differentiable, then gradient descent update would be:

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

Recall motivation: minimize quadratic approximation to f around x, replace $\nabla^2 f(x)$ by $\frac{1}{t}I$,

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

In our case f is not differentiable, but $f = g + h$, g differentiable. Why don't we make quadratic approximation to g, leave h alone?

That is, update

$$\begin{aligned} x^+ &= \operatorname*{argmin}_z \bar{g}(z) + h(z) \\ &= \operatorname*{argmin}_z g(x) + \nabla g(x)^T(z-x) + \frac{1}{2t}\|z-x\|_2^2 + h(z) \\ &= \operatorname*{argmin}_z \frac{1}{2t}\|z-(x-t\nabla g(x))\|_2^2 + h(z). \end{aligned} \tag{34}$$

In the above derivation, $\frac{1}{2t}\|z-(x-t\nabla g(x))\|_2^2$ makes z stay close to gradient update for g and $h(z)$ also makes h small.

6.2 Motivation for choosing stepsize

This subsection will answer the question why $t \le \frac{1}{L(\nabla g)}$ is required for the stepsize t in proximal GD methods.

Recall that the implication of Lipschitz gradients of a function $g \in C^{1,1}$

$$g(y) \le g(x) + \nabla g(x)^T(y-x) + \frac{L(\nabla g)}{2}\|y-x\|^2, \quad \forall x, y \tag{35}$$

where ∇g is Lipschitz continuous with constant $L(\nabla g)$. Note that (35) **holds for any $L \ge L(\nabla g)$.** Then, with such an L the quadratic approximation model will be

$$\begin{aligned} f(y) = g(y) + h(y) &\le g(x) + \nabla g(x)^T(y-x) + \frac{L}{2}\|y-x\|^2 + h(y) \\ &\le g(x) + \frac{L}{2}\left(\|y-x\|^2 + \frac{2}{L}\nabla g(x)^T(y-x) + \frac{\|\nabla g(x)\|_2^2}{L^2} - \frac{\|\nabla g(x)\|_2^2}{L^2}\right) + h(y) \\ &\le g(x) + \frac{L}{2}\left(\|y-(x-\frac{\nabla g(x)}{L})\|_2^2\right) - \frac{\|\nabla g(x)\|_2^2}{2L} + h(y) \end{aligned}$$

Taking min on both sides,

$$\min_y f(y) \le \min_y \frac{L}{2}\left(\|y-(x-\frac{\nabla g(x)}{L})\|_2^2\right) + h(y) + g(x) - \frac{\|\nabla g(x)\|_2^2}{2L}$$

Now we find the upper bound for the optimal objective value. We only need to optimize the upper bound until reach the optimal value of the original objective. In section 6.5, we will show the least upper bound is exactly the minimum of f when gradient map defined there vanishes.

Taking argmin and dropping the irrelevant terms,

$$x^+ = \operatorname*{argmin}_y \frac{L}{2}\left(\|y-(x-\frac{\nabla g(x)}{L})\|_2^2\right) + h(y)$$

Let $L = \frac{1}{t} \ge L(\nabla g)$, i.e., $0 < t \le \frac{1}{L(\nabla g)}$,

$$x^+ = \operatorname*{argmin}_y \frac{1}{2t}\left(\|y-(x-t\nabla g(x))\|_2^2\right) + h(y)$$

where t plays the role of stepsize in proximal gradient descent algorithms. The condition $t \le \frac{1}{L(\nabla g)}$ is critical to support all of derivations in this subsection.

6.3 Proximal mapping

Define proximal mapping:

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

If h is convex and closed (has a closed epigraph), then $\operatorname{prox}_{h,t}$ exists and is unique for all x. From the optimality conditions of minimization in the above definition, we have

$$\begin{aligned} x^+ = \operatorname{prox}_{h,t}(x) &\Longleftrightarrow \frac{x-x^+}{t} \in \partial h(x^+) \\ &\Longleftrightarrow h(z) \ge h(x^+) + \frac{1}{t}(x-x^+)^T(z-x^+), \quad \forall z \end{aligned} \tag{36}$$

where the second line follows from the definition of subdifferential.

6.4 Nonexpansiveness

[9] If $u = \text{prox}_{h,t}(x)$, $v = \text{prox}_{h,t}(y)$, then

$$(u-v)^T(x-y) \geq \|u-v\|_2^2$$

$\text{prox}_{h,t}$ is firmly nonexpansive, or co-coercive with constant 1, which follows from (36) and the monotonicity of the subdifferential (see Section 5.2.1)

$$\begin{aligned}\frac{x-u}{t} \in \partial h(u),\ \frac{y-v}{t} \in \partial h(v) &\Longrightarrow \frac{1}{t}(x-u-y+v)^T(u-v) \geq 0 \\ &\Longrightarrow (x-y)^T(u-v) - (u-v)^T(u-v) \geq 0 \\ &\Longrightarrow (x-y)^T(u-v) \geq \|u-v\|^2.\end{aligned}$$

By Cauchy-Schwarz inequality, we get

$$\|\text{prox}_{h,t}(x) - \text{prox}_{h,t}(y)\|_2 \leq \|x-y\|_2$$

which also demonstrates that $\text{prox}_{h,t}$ is nonexpansive, or Lipschitz continuous with constant 1.

6.5 Gradient map

Note that $x = x^{(k-1)}$ from this section on.

$$\begin{aligned} x^+ &= \text{prox}_{h,t}\left(x - t\nabla g(x)\right), \\ &= x - t \cdot G_t(x), \end{aligned}$$

where G_t is the "generalized" gradient of f,

$$G_t(x) = \frac{x - \text{prox}_{h,t}\left(x - t\nabla g(x)\right)}{t} = \frac{x - x^+}{t}.$$

- $G_t(x)$ is not a gradient or subgradient of $f = g + h$.
- From the definition of prox-operator,

$$\begin{aligned}\frac{x^+ - (x - t\nabla g(x))}{t} \in -\partial h(x^+) &\Longrightarrow \frac{x^+ - x}{t} + \nabla g(x) \in -\partial h(x^+) \\ &\Longrightarrow G_t(x) - \nabla g(x) \in \partial h(x^+) \\ &\Longrightarrow G_t(x) \in \nabla g(x) + \partial h(x^+) \\ &\Longrightarrow G_t(x) \in \nabla g(x) + \partial h(x - tG_t(x))\end{aligned}$$

- $G_t(x) = 0$ if and only if x minimizes $f(x) = g(x) + h(x)$.

Now we show the last result based on the second result.

Proof. We first the necessity. From the second result, if $G_t(x) = 0$, we have

$$G_t(x) \in \nabla g(x) + \partial h(x - tG_t(x)) \overset{G_t(x)=0}{\Longrightarrow} 0 \in \nabla g(x) + \partial h(x)$$

Now we show the sufficiency. If x^+ minimizes $f(x) = g(x) + h(x)$, then

$$\begin{aligned} 0 \in \nabla g(x^+) + \partial h(x^+) &\Longrightarrow 0 = \nabla g(x^+) + G_t(x) - \nabla g(x) \quad (\because G_t(x) - \nabla g(x) \in \partial h(x^+)) \\ &\Longrightarrow \nabla g(x^+) - \nabla g(x) = \frac{x - x^+}{t} \quad (\because G_t(x) = \frac{x - x^+}{t}) \\ &\Longrightarrow x = x^+ \quad (\because t \text{ is arbitrary on } (0, \frac{1}{L(\nabla g)}]) \\ &\Longrightarrow G_t(x) = G_t(x^+) = 0 \end{aligned}$$

□

Note that from the second item, we have $G_t(x) \in \nabla g(x) + \partial h(x - tG_t(x))$. If $G_t(x) = 0$, i.e., $x^+ = x$, we obtain $0 \in \nabla g(x) + \partial h(x)$, which is the optimality condition for unconstrained subgradient methods. In this case, $x^+ = x$ and x is exactly the solution we want to find. If we take derivatives on $f(x)$ and let it be 0, we will get $0 \in \nabla g(x) + \partial h(x)$, but there is no analytical form for $\partial h(x)$. However, the proximal GD method nicely circumvent this difficulty by leaving h alone and making use of the proximal operator of h which is normally readily to get.

[9] http://www.seas.ucla.edu/~vandenbe/236C/lectures/proxgrad.pdf

6.6 Proximal gradient descent method

Proximal gradient descent: choose initialize $x^{(0)}$, repeat:

$$\begin{aligned} x^{(k)} &= \text{prox}_{h,t}\left(x^{(k-1)} - t_k \nabla g(x^{(k-1)})\right), \quad k = 1,2,3,\cdots \\ &= x^{(k-1)} - t_k \cdot G_{t_k}(x^{(k-1)}), \end{aligned}$$

The key point is that $\text{prox}_{h,t}(\cdot)$ has a closed-form for many important functions h. Note:

- Mapping $\text{prox}_{h,t}(\cdot)$ doesn't depend on g at all, only on h;
- Smooth part g can be complicated, we only need to compute its gradients.

Convergence analysis will be in terms of the number of iterations, and each iteration evaluates $\text{prox}_{h,t}(\cdot)$ once (this can be cheap or expensive, depending on h).

6.7 Backtracking line search

Backtracking for prox gradient descent works similar as before (in gradient descent), but operates on g and not f. Choose parameter $0 < \beta < 1$. At each iteration, start at $t = t_{\text{init}}$, and while

$$g\left(x - tG_t(x)\right) > g(x) - t\nabla g(x)^T G_t(x) + \frac{t}{2}\|G_t(x)\|_2^2 \tag{37}$$

shrink $t = \beta t$, for some $0 < \beta < 1$. Otherwise, perform proximal gradient update. Why is (37) the condition of the backtracking line search for proximal GD?

6.8 Consequences of Lipschitz assumption

Recall that for convex g with Lipschitz continuous gradient

$$g(y) \le f(x) + \nabla g(x)^T (y - x) + \frac{L}{2}\|y - x\|_2^2, \quad \forall x, y$$

Note that even for nonconvex g this bound still holds since the proof of this bound does not require the convexity of g.

Substituting $x^+ = x - tG_t(x)$ into this bound, we get

$$g\left(x - tG_t(x)\right) \le g(x) - t\nabla g(x)^T G_t(x) + \frac{t^2 L}{2}\|G_t(x)\|_2^2, \tag{38}$$

where the RHS is a quadratic polynomial w.r.t. t which passes the origin regardless of the term $g(x)$. If $0 < t \le \frac{1}{L}$, then the cord connecting the origin with the point $(\frac{1}{L}, -\frac{1}{L}\nabla g(x)^T G_t(x) + \frac{1}{2L}\|G_t(x)\|_2^2)$ always lies above the parabola on the interval of $(0, \frac{1}{L}]$. Since the slope of the cord is $\frac{-\frac{1}{L}\nabla g(x)^T G_t(x) + \frac{1}{2L}\|G_t(x)\|_2^2}{\frac{1}{L}} = -\nabla g(x)^T G_t(x) + \frac{1}{2}\|G_t(x)\|_2^2$, the cord is $-t\nabla g(x)^T G_t(x) + \frac{t}{2}\|G_t(x)\|_2^2$. Thus, we obtain

$$g\left(x - tG_t(x)\right) \le g(x) - t\nabla g(x)^T G_t(x) + \frac{t}{2}\|G_t(x)\|_2^2.$$

which is the condition for backtracking line search, i.e., (37).

6.9 Convergence analysis

For criterion $f(x) = g(x) + h(x)$, we assume:

- g is convex, differentiable, $\text{dom}(g) = \mathbf{R}^n$, and ∇g is Lipschitz continuous with constant $L > 0$;
- h is convex, $\text{prox}_t(x) = \text{argmin}_z \{\frac{1}{2t}\|z - x\|_2^2 + h(z)\}$ can be evaluated.

Theorem 8. Proximal gradient descent with fixed step size $t \le 1/L$ satisfies

$$f(x^k) - f^* = \frac{\|x^0 - x^*\|}{2tk}$$

and same result holds for backtracking, with t replaced by β/L.

Proof. Adding $h(x^+)$ to both sides of (38),

$$f(x) = g(x^+) + h(x^+) \leq g(x) + h(x^+) - t\nabla g(x)^T G_t(x) + \frac{t^2 L}{2}\|G_t(x)\|_2^2 \tag{39}$$

By the convexity of g and h, we have

$$g(x) \leq g(z) + \nabla g(x)^T(x-z), \quad h(x^+) \leq h(z) + v^T(x^+ - z)$$

where $v = G_t(x) - \nabla g(x) \in \partial h(x^+)$. Substituting these two inequalities into the RHS of (39),

$$\begin{aligned}
f(x^+) &\leq g(z) + \nabla g(x)^T(x-z) + h(z) + (G_t(x) - \nabla g(x))^T(x^+ - z) - t\nabla g(x)^T G_t(x) + \frac{t^2 L}{2}\|G_t(x)\|_2^2 \\
&\leq f(z) + \nabla g(x)^T(x-z) + (G_t(x) - \nabla g(x))^T(x^+ - z) - t\nabla g(x)^T G_t(x) + \frac{t^2 L}{2}\|G_t(x)\|_2^2 \\
&\leq f(z) + \nabla g(x)^T(x-z) + (G_t(x) - \nabla g(x))^T(x^+ - z) + \nabla g(x)^T(x^+ - x) + \frac{t^2 L}{2}\|G_t(x)\|_2^2 \\
&\leq f(z) + \nabla g(x)^T(x^+-z) + (G_t(x) - \nabla g(x))^T(x^+ - z) + \frac{t^2 L}{2}\|G_t(x)\|_2^2 \\
&\leq f(z) + G_t(x)^T(x^+ - z) + \frac{t^2 L}{2}\|G_t(x)\|_2^2 \\
&\leq f(z) + G_t(x)^T(x - z) + (\frac{Lt^2}{2} - t)\|G_t(x)\|_2^2
\end{aligned}$$

If $0 < t \leq \frac{1}{L}$, $k = \frac{\frac{L}{2L^2} - \frac{1}{L}}{\frac{1}{L}} = -\frac{1}{2}$. So, the cord connecting the origin with the minimizer $t^* = \frac{1}{L}$ of $\frac{Lt^2}{2} - t$ is always above $\frac{Lt^2}{2} - t$ on $(0, \frac{1}{L}]$, namely, $-\frac{t}{2} \geq \frac{Lt^2}{2} - t$. Thus,

$$f(x^+) - f(z) \leq G_t(x)^T(x-z) - \frac{t}{2}\|G_t(x)\|_2^2$$

Setting $z = x^*$ and substituting $G_t(x) = \frac{x - x^+}{t}$ into the above inequality,

$$\begin{aligned}
f(x^+) - f(x^*) &\leq \frac{(x-x^+)^T(x-x^*)}{t} - \frac{1}{2t}\|x - x^+\|_2^2 \\
&= \frac{2(x-x^+)^T(x-x^*) - \|x-x^+\|_2^2}{2t} \\
&= -\frac{\|x-x^+\|_2^2 - 2(x-x^+)^T(x-x^*) + \|x-x^*\|_2^2 - \|x-x^*\|_2^2}{2t} \\
&= \frac{1}{2t}(\|x-x^*\|_2^2 - \|x^+ - x^*\|_2^2)
\end{aligned}$$

where $x^+ = x^{(k)}, x = x^{(k-1)}$ and x^* denotes the optimal solution. Summing both sides from 1 to k,

$$k(f(x^i) - f(x^*)) \leq \sum_{i=1}^{k}(f(x^i) - f(x^*)) \leq \frac{1}{2t}(\|x^{(0)} - x^*\|_2^2 - \|x^{(k)} - x^*\|_2^2) \leq \frac{\|x^{(0)} - x^*\|_2^2}{2t}$$

where the first inequality follows from that $f(x^i)$ is nonincreasing. Thus,

$$f(x^i) - f(x^*) \leq \frac{\|x^{(0)} - x^*\|_2^2}{2kt}.$$

This proves the sublinear convergence of proximal GD method. □

Proximal gradient descent has convergence rate $O(1/k)$ or $O(1/\epsilon)$ which matches gradient descent rate! (But remember prox cost ...).

6.10 Example: ISTA

Given $y \in \mathbf{R}^n$, $X \in \mathbf{R}^{n \times p}$, recall the lasso criterion:

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

Proximal mapping is now

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

where $S_\lambda(\beta)$ is the soft-thresholding operator,

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

Recall $g(\beta) = -X^T(y - X\beta)$, hence proximal gradient update is:

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

this is often called the iterative soft-thresholding algorithm (ISTA).

6.11 Example: matrix completion

Reference: This section is taken from Ryan Tibshirani's slides and the corresponding scribe written by students who took his course.

Given a matrix $Y \in \mathbf{R}^{m\times n}$, and only observe entries Y_{ij}, $(i,j) \in \Omega$. Suppose we want to fill in missing entries (e.g., for a recommender system), so we solve a matrix completion problem:

$$\min_B \frac{1}{2}\sum_{i,j\in\Omega}(Y_{ij} - B_{ij})^2 + \lambda\|B\|_{\text{tr}}$$

where $\|B\|_{\text{tr}}$ is the trace (or nuclear) norm of B,

$$\|B\|_{tr} = \sum_{i=1}^r \sigma_i(B)$$

where $r = \text{rank}(B)$ and $\sigma_1(X) \ge \cdots \ge \sigma_r(X) \ge 0$ are the singular values.

Define P_Ω, projection operator onto observed set:

$$[P_\Omega(B)]_{ij} = \begin{cases}B_{ij} & (i,j)\in\Omega\\ 0 & (i,j)\notin\Omega\end{cases}$$

Then the criterion is

$$f(B) = \underbrace{\frac{1}{2}\|P_\Omega(Y) - P_\Omega(B)\|_F^2}_{g(B)} + \underbrace{\lambda\|B\|_{\text{tr}}}_{h(B)}$$

$\|B\|_{\text{tr}}$ can be thought of as a convex but non-smooth relaxation of rank(B) (analogous to the L_1 norm being a relaxation of L_0). The low-rank provision comes from thinking about applications where there is an assumed low dimensional latent factor model that explains Y.

Two ingredients are needed for proximal gradient descent:

- Gradient calculation: $\nabla g(B) = -(P_\Omega(Y) - P_\Omega(B))$;
- Prox function:

$$\text{prox}_t(B) = \underset{Z}{\text{argmin}}\, \frac{1}{2t}\|B - Z\|_F^2 + \lambda\|Z\|_{\text{tr}}$$

Claim: $\text{prox}_t(B) = S_{\lambda t}(B)$, matrix soft-thresholding at the level λ. Here $S_\lambda(B)$ is defined by

$$S_\lambda(B) = U\Sigma_\lambda V^T$$

where $B = U\Sigma V^T$ is an SVD, and Σ_λ is diagonal with

$$(\Sigma_\lambda)_{ii} = \max\{\Sigma_{ii}, 0\}$$

The following proof is taken from Cai et al. A Singular Value Thresholding Algorithm for Matrix Completion.

Proof. Note that $\text{prox}_t(B) = Z$, and the subgradient optimality (31) tells us that Z satisfies

$$0 \in Z - B + \lambda t \partial \|Z\|_{\text{tr}}.$$

We know the fact that if $Z = U\Sigma V^T$, then

$$\partial \|Z\|_{\text{tr}} = \{UV^T + W : \|W\|_{\text{op}} \leq 1, U^T W = 0, WV = 0\}.$$

We can decompose the SVD of B as

$$B = U_0 \Sigma_0 V_0^T + U_1 \Sigma_1 V_1^T.$$

where U_0, V_0 (resp. U_1, V_1) are the singular vectors associated with singular values greater than λt(resp. smaller than or equal to λt). With these notations, we have

$$Z = U_0(\Sigma_0 - \lambda t I)V_0^T.$$

and, therefore,

$$\begin{aligned} B - Z &= U_0 \Sigma_0 V_0^T + U_1 \Sigma_1 V_1^T - U_0(\Sigma_0 - \lambda t I)V_0^T \\ &= \lambda t U_0 V_0^T + U_1 \Sigma_1 V_1^T \\ &= \lambda t (U_0 V_0^T + W) \end{aligned} .$$

where $W = (\lambda t)^{-1} U_1 \Sigma_1 V_1^T$.

By the definition of SVD, $U_0^T W = 0$ and $WV_0 = 0$. Since the diagonal elements of Σ_1 have magnitudes bounded by λt, we also have $\|W\|_2 \leq 1$. Hence, $B - Z \in \partial \|Z\|_{\text{tr}}$, which concludes the proof. □

Hence proximal gradient update step is:

$$B^+ = S_{\lambda t}(B + t(P_\Omega(Y) - P_\Omega(B))).$$

Note that $\nabla g(B)$ is Lipschitz continuous with $L = 1$, so we can choose fixed step size $t = 1$. Update step is now:

$$B^+ = S_\lambda \left(P_\Omega(Y) + P_\Omega^\perp(B)\right).$$

where $P_\Omega^\perp$ projects onto unobserved set. $P_\Omega(Y) + P_\Omega^\perp(B) = B$.

This is the soft-impute algorithm[10], simple and effective method for matrix completion.

6.12 Special cases

Proximal gradient descent also called composite gradient descent, or generalized gradient descent. Why "generalized"? This refers to the several special cases, when minimizing $f = g + h$:

- $h = 0$: gradient descent;
- $h = I_C$: projected gradient descent;
- $g = 0$: proximal minimization algorithm.

The first case is obvious. For the second case, we have talked about projected gradient descent in 5.8. Specifically,

$$\min_{x \in C} g(x) \Leftrightarrow \min g(x) + I_C.$$

where $I_C(x) = \begin{cases} 0, & \text{if } x \in C \\ \infty, & \text{otherwise.} \end{cases}$ is an indicator function of C.

Hence,

$$\begin{aligned} \text{prox}_t(x) &= \underset{z}{\text{argmin}} \frac{1}{2t}\|z - x\|_2^2 + I_C(z) \\ &= \underset{z \in C}{\text{argmin}} \|z - x\|_2^2. \end{aligned}$$

That is, $\text{prox}_t(x) = P_C(x)$, projection operator onto C.

Proximal minimization algorithm. Consider for h convex (not necessarily differentiable),

$$\min_x h(x)$$

[10]Mazumder et al. (2011), "Spectral regularization algorithms for learning large incomplete matrices"

Proximal gradient update step is just:

$$x^+ = \operatorname*{argmin}_z \frac{1}{2t}\|x-z\|_2^2 + h(z)$$

since $\nabla g(x) = 0$. This is called proximal minimization algorithm which is faster than subgradient method, but not implementable unless we know prox in closed form. This method predates proximal gradient descent and under certain assumptions, guarantees $O(1/\epsilon)$ convergence rate, which is faster than the $O(1/\epsilon^2)$ convergence of subgradient methods.

6.13 Accelerated proximal gradient method

It turns out that we can accelerate proximal gradient descent in order to achieve the optimal $O(1/\sqrt{\epsilon})$ convergence rate. Four ideas (three acceleration methods) by Nesterov:

- 1983: original acceleration idea for smooth functions;
- 1988: another acceleration idea for smooth functions;
- 2005: smoothing techniques for nonsmooth functions, coupled with original acceleration idea;
- 2007: acceleration idea for composite functions[11].

We will follow Beck and Teboulle (2008), an extension of Nesterov (1983) to composite functions[12].
As before, consider:

$$\min_x g(x) + h(x)$$

where g convex, differentiable, and h convex. Accelerated proximal gradient method: choose initial point $x^{(0)} = x^{(-1)} \in \mathbf{R}^n$, repeat:

$$v = x^{(k-1)} + \frac{k-2}{k+1}(x^{(k-1)} - x^{(k-2)})$$

$$x^{(k)} = \text{prox}_{h,t}\left(v - t\nabla g(v)\right)$$

for $k = 1, 2, 3, \cdots$.

- First step k = 1 is just usual proximal gradient update;
- After that, $v = x^{(k-1)} + \frac{k-2}{k+1}(x^{(k-1)} - x^{(k-2)})$ carries some "momentum" from previous iterations;
- When $h = 0$ we get accelerated gradient method.

Backtracking with acceleration is in different ways. Simple approach: fix $\beta < 1, t_0 = 1$. At iteration k, start with $t = t_{k-1}$, and while

$$g(x^+) > g(x) + \nabla g(x)^T(x^+ - x) + \frac{1}{2t}\|x^+ - x\|_2^2$$

shrink $t = \beta t$, and let $x^+ = \text{prox}_{h,t}\left(v - t\nabla g(v)\right)$. Otherwise, we keep x^+.
Note that this strategy forces us to take decreasing step sizes.

6.14 Convergence analysis

For criterion $f(x) = g(x) + h(x)$, we assume as before:

- g is convex, differentiable, $\text{dom}(g) = \mathbf{R}^n$, and ∇g is Lipschitz continuous with constant $L > 0$;
- h is convex, $\text{prox}_t(x) = \text{argmin}_z\{\|z - x\|_2^2/(2t) + h(z)\}$ can be evaluated.

Theorem 9. Accelerated proximal gradient method with fixed step size $t \leq 1/L$ satisfies

$$f(x^k) - f^* = \frac{2\|x^{(0)} - x^*\|_2^2}{t(k+1)^2}$$

and the same result holds for backtracking, with t replaced by β/L.

[11]Each step uses entire history of previous steps and makes two prox calls.
[12]Each step uses information from two last steps and makes one prox call.

The main idea behind this method is that as we approach the solution, the gradient is vanishing, making the gradient step sizes smaller. Given that you are approaching the solution in a smooth path, the history (and directions taken) tell you something about which direction is "good", and using this information makes a notable difference when the gradient is vanishing. This achieves optimal rate $O(1/k^2)$ or $O(1/\sqrt{\epsilon})$ for first-order methods.

However, acceleration is not always useful since combining the last two steps together may destroy the possible low rank property. For instance, the low rank property makes SVD less computationally expensive in calculating the proximal operator of the matrix completion problem.

7 Stochastic gradient descent

Consider minimizing an average of functions:

$$\min_x \frac{1}{m}\sum_{i=1}^m f_i(x).$$

As $\nabla \sum_{i=1}^m f_i(x) = \sum_{i=1}^m \nabla f_i(x)$ gradient descent would repeat:

$$x^{(k)} = x^{(k-1)} - t_k \cdot \frac{1}{m}\sum_{i=1}^m \nabla f_i(x), \quad k = 1, 2, 3, \cdots.$$

In comparison, stochastic gradient descent (SGD) (or incremental gradient descent) repeats:

$$x^{(k)} = x^{(k-1)} - t_k \cdot \frac{1}{m}\nabla f_{i_k}(x), \quad k = 1, 2, 3, \cdots.$$

where $i_k \in 1, \cdots, m$ is some chosen index at iteration k. In practice, we choose i_k randomly at iteration k. Note that

$$\mathbb{E}[\nabla f_{i_k}(x)] = \nabla f(x)$$

so we can view SGD as using an unbiased estimate of the gradient at each step. The main appeal of SGD:

- Iteration cost is independent of m (number of functions);
- Can also be a big savings in terms of memory useage.

7.1 Example: stochastic logistic regression

Given $(x_i, y_i) \in \mathbf{R}^p \times \{0, 1\}, i = 1, \cdots, n$, recall logistic regression:

$$\min_{\beta\in\mathbf{R}^p} f(\beta) = \frac{1}{n}\sum_{i=1}^n \underbrace{\left(-y_i\beta^T x_i + \ln\left(1 + \exp(\beta^T x_i)\right)\right)}_{f_i(\beta)}$$

Gradient computation $\nabla f(\beta) = -\frac{1}{n}\sum_{i=1}^n (y_i - p_i(\beta))$ is doable when n is moderate, but not when n is huge. Here $p_i = \exp(\beta^T x_i)/\left(1 + \exp(\beta^T x_i)\right)$.

Full gradient (also called batch) versus stochastic gradient:

- One batch update costs $O(np)$
- One stochastic update costs $O(p)$

As shown in Fig. 4, small example with $n = 10, p = 2$ to show the "classic picture" for batch versus stochastic methods:

Blue: batch steps, $O(np)$.

Red: stochastic steps, $O(p)$.

Rule of thumb for stochastic methods:

- generally thrive far from optimum;
- generally struggle close to optimum.

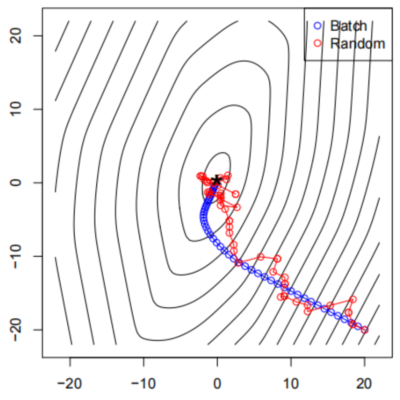


Figure 4: Batch GD vs. SGD

It is standard in SGD to use diminishing step sizes, e.g., $t_k = 1/k$, for $k = 1, 2, 3, \cdots$. Why not fixed step sizes? Here's some intuition. Suppose we take cyclic rule for simplicity. Set $t_k = t$ for m updates in a row, we get:

$$x^{(k+m)} = x^{(k)} - t \cdot \frac{1}{m} \sum_{i=1}^{m} \nabla f_i(x^{(k+i-1)})$$

Meanwhile, full gradient with step size t would give:

$$x^{(k+1)} = x^{(k)} - t \cdot \frac{1}{m} \sum_{i=1}^{m} \nabla f_i(x^{(k)})$$

The difference here: $t \sum_{i=1}^{m} [\nabla f_i(x^{(k+i-1)}) - \nabla f_i(x^{(k)})]$, and if we hold t constant, this difference will not generally be going to zero.

7.2 Convergence rates

In section 5.7, we got that if f is convex and Lipschitz continuous with constant $G > 0$, gradient descent with diminishing step sizes satisfies

$$f(x^{(k)}) - f^* = O(1/\sqrt{k})$$

When f is differentiable with Lipschitz gradient, we get for suitable fixed step sizes (see section 4.2)

$$f(x^{(k)}) - f^* = O(1/k)$$

What about SGD? For convex f, SGD with diminishing step sizes satisfies[13]

$$\mathbb{E}[f(x^{(k)}) - f^*] = O(1/\sqrt{k})$$

Unfortunately this does not improve when we further assume f has Lipschitz gradient.
It is even worse. Let's see the following discrepancy!
When f is strongly convex and has a Lipschitz gradient, gradient descent satisfies

$$f(x^{(k)}) - f^* = O(c^k)$$

where $c < 1$. But under same conditions, SGD gives us[14]

$$\mathbb{E}[f(x^{(k)}) - f^*] = O(1/k)$$

So stochastic methods do not enjoy the linear convergence rate of gradient descent under strong convexity.
What can we do to improve SGD?

[13] E.g., Nemirosvki et al. (2009), "Robust stochastic optimization approach to stochastic programming"
[14] E.g., Nemirosvki et al. (2009), "Robust stochastic optimization approach to stochastic programming"

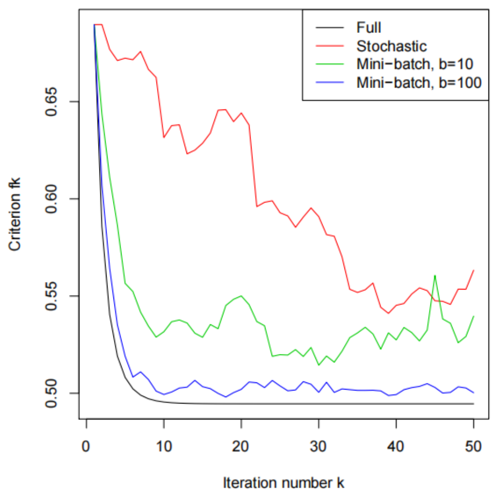


Figure 5: Objective values vs. iterations

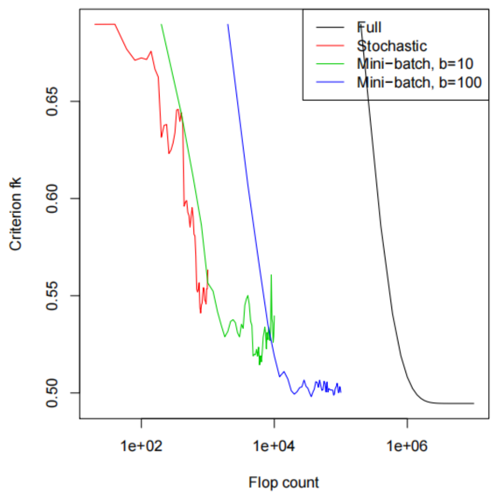


Figure 6: Objective values vs. flops

7.3 Mini-batch stochastic gradient descent

Mini-batch stochastic gradient descent is also common, where we choose a random subset $I_k \subseteq \{1, \cdots, m\}$ of size $|I_k| = b \ll m$, and repeat:

$$x^{(k+1)} = x^{(k)} - t \cdot \frac{1}{b} \sum_{i \in I_k}^{m} \nabla f_i(x^{(k)}), \quad k = 1, 2, 3, \cdots$$

Again, we are approximating full graident by an unbiased estimate:

$$\mathbb{E}[\frac{1}{b} \sum_{i \in I_k}^{m} \nabla f_i(x)] = \nabla f(x)$$

Using mini-batches reduces the variance of our gradient estimate by a factor $1/b$, but is also b times more expensive. One mini-batch update costs $O(bp)$ which is in between.

Let's see an example of regularized logistic regression with $n = 10,000$, $p = 20$. All methods in Fig. 5-7 use fixed step sizes:

7.4 Recap

- SGD can be super effective in terms of iteration cost, memory;
- But SGD is slow to converge, can't adapt to strong convexity;
- And mini-batches seem to be a wash in terms of flops (though they can still be useful in practice).

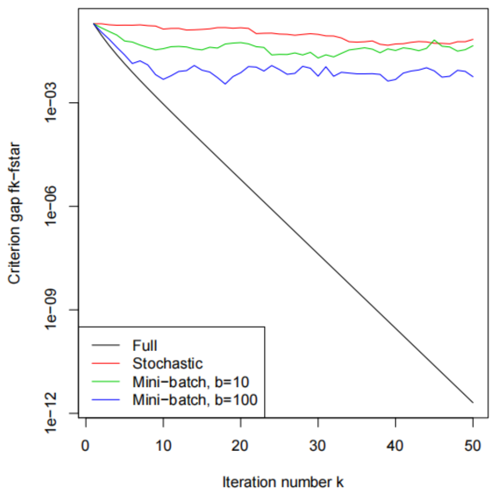


Figure 7: Objective value gap vs. iterations

Is this the end of the story for SGD?

For a while, the answer was believed to be yes. Slow convergence for strongly convex functions was believed inevitable, as Nemirovski and others established matching lower bounds. . . but this was for a more general stochastic problem, where $f(x) = \int F(x, \xi) dP(\xi)$.

New wave of "variance reduction" work shows we can modify SGD to converge much faster for finite sums.

7.5 SGD in large-scale ML

SGD has really taken off in large-scale machine learning.

- In many ML problems we do not care about optimizing to high accuracy, it does not pay off in terms of statistical performance;
- Many variants provide better practical stability, convergence: momentum, acceleration, averaging, coordinate-adapted step sizes, variance reduction. . . ;
- See AdaGrad, Adam, AdaMax, SVRG, SAG, SAGA. . . .

7.6 Early stopping

Suppose p is large and we wanted to fit (say) a logistic regression model to data $(xi, yi) \in \mathbf{R}^p \in 0, 1, i = 1, \cdots, n$ We could solve (say) ℓ_2 regularized logistic regression:

$$\min_{\beta \in \mathbf{R}^p} f(\beta) = \frac{1}{n} \sum_{i=1}^{n} \left(-y_i \beta^T x_i + \ln\left(1 + \exp(\beta^T x_i)\right)\right) \text{ s.t. } \|\beta\|_2 \leq t.$$

We could also run gradient descent on the unregularized problem:

$$\min_{\beta \in \mathbf{R}^p} f(\beta) = \frac{1}{n} \sum_{i=1}^{n} \left(-y_i \beta^T x_i + \ln\left(1 + \exp(\beta^T x_i)\right)\right)$$

and stop early, i.e., terminate gradient descent well-short of the global minimum.

Consider the following, for a very small constant step size ϵ:

- Start at $\beta^{(0)} = 0$, solution to regularized problem at $t = 0$;
- Perform gradient descent on unregularized criterion

$$\beta^{(k)} = \beta^{(k-1)} - \frac{1}{n} \sum_{i=1}^{n} (y_i - p_i(\beta^{k-1})) x_i, \quad k = 1, 2, 3, \cdots$$

  (we could equally well consider SGD);
- Treat β^k as an approximate solution to regularized problem with $t = \|\beta^{(k)}\|_2$.

This is called early stopping for gradient descent. Why would we ever do this? It's both more convenient and potentially much more efficient than using explicit regularization.

8 Duality

8.1 Duality for general form linear programs (LP)

This subsection is taken from scribed notes which are available at http://www.stat.cmu.edu/~ryantibs/convexopt/scribes/dual-lps-scribed.pdf. We use u and v to denote dual variables for inequality constraints and equality constraints, respectively.

Given $c \in \mathbf{R}^n$, $A \in \mathbf{R}^{m\times n}$, $b \in \mathbf{R}^m$, $G \in \mathbf{R}^{r\times n}$, and $h \in \mathbf{R}^r$, the primal LP problem is

$$\begin{aligned} \min_{x\in\mathbf{R}^n} \quad & c^T x \\ \text{subject to } & Ax = b \\ & Gx \le h \end{aligned}$$

Let C and f^* denote the primal feasible set and the optimal primal value, respectively. For any v and $u \ge 0$, we have

$$c^T x \ge c^T x + \underbrace{v^T(Ax-b)}_{=0} + \underbrace{u^T(Gx-h)}_{\le 0} := \mathcal{L}(x,u,v)$$

Taking minimum on both sides, we get

$$f^* = \min_{x\in C} c^T x \ge \min_{x\in C} L(x,u,v) \ge \min L(x,u,v) := g(u,v)$$

Thus, $g(u,v)$ is a lower bound on f^* for any v and $u \ge 0$. We can maximize $g(u,v)$ over v and $u \ge 0$ to get the tightest lower bound.

Note that

$$g(u,v) = \begin{cases} -b^T v - h^T u, & \text{if } c = -A^T v - G^T u \\ -\infty, & \text{otherwise.} \end{cases}$$

which is exactly the dual LP. More specifically,

$$\begin{aligned} \max_{u,v} \quad & -b^T v - h^T u \\ \text{s.t.} \quad & c = -A^T v - G^T u \\ & v \ge 0. \end{aligned}$$

Another explanation: for any v and $u \ge 0$, and x primal feasible,

$$v^T(Ax-b) + u^T(Gx-h) \le 0$$

i.e.,

$$(-A^T v - G^T u)^T x \ge -b^T v - h^T u$$

So, if $c = -A^T v - G^T u$, we get a bound on primal optimal value.

8.2 Lagrangian

Consider a general minimization problem

$$\begin{aligned} \min_{x} \quad & f(x) \\ \text{subject to} \quad & h_i(x) \le 0, i = 1, \cdots, m \\ & l_j(x) = 0, j = 1, \cdots, r. \end{aligned}$$

We define the Lagrangian as

$$L(x,u,v) = f(x) + \sum_{i=1}^{m} u_i h_i(x) + \sum_{j=1}^{r} v_j l_j(x)$$

with $u \in \mathbf{R}_+{}^m$ and $v \in \mathbf{R}^r$.

An important property: for any $u \ge 0$ and v

$$f(x) \ge L(x,u,v) \quad \text{at each feasible } x.$$

Why? For any feasible x,

$$L(x,u,v) = f(x) + \sum_{i=1}^{m} u_i \underbrace{h_i(x)}_{\leq 0} + \sum_{j=1}^{r} v_j \underbrace{l_j(x)}_{=0} \leq f(x)$$

Minimizing $L(x,u,v)$ over all x gives a lower bound:

$$f^* \geq \min_{x\in C} L(x,u,v) \geq \min_{x} L(x,u,v) := g(u,v)$$

We call $g(u,v)$ the Lagrange dual function, and it gives a lower bound on f^* for any $u \geq 0$ and v, called dual feasible u, v.

8.3 Example: quadratic program

Consider quadratic program:

$$\begin{aligned} \min_{x} \quad & \frac{1}{2}x^TQx + c^Tx \\ \text{subject to} \quad & Ax = b, x \geq 0. \end{aligned}$$

where $Q \geq 0$.

Lagrangian:

$$L(x,u,v) = \frac{1}{2}x^TQx + c^Tx - u^Tx + v^T(Ax-b)$$

Lagrange dual function:

$$g(u,v) = \min_{x} L(x,u,v) = -\frac{1}{2}(c-u+A^Tv)^TQ^{-1}(c-u+A^Tv) - b^Tv$$

For any $u \geq 0$ and any v, this is a lower bound on the primal optimal f^*.

If $Q \succeq 0$, then it is possible that $c-u+A^Tv \notin \text{col}(Q)$. In this case, then $L(x,u,v)$ may be negative infinity. When $c-u+A^Tv \in \text{col}(Q)$, then $x = -Q^\dagger(c-u+A^Tv)$ where $Q^\dagger$ is the generalized inverse of Q. In this case, the Lagrange dual function is

$$g(u,v) = \begin{cases} -\frac{1}{2}(c-u+A^Tv)^TQ^\dagger(c-u+A^Tv) - b^Tv, & \text{if } c-u+A^Tv \perp \text{null}(Q^T) \\ -\infty, & \text{otherwise.} \end{cases}$$

where it is the null space of Q^T instead of Q in the original slides(see its page 9) since Q is supposed to be symmetric.

8.4 Weak duality and strong duality

By maximizing $g(u,v)$ over all dual feasible u, v, yielding Lagrange dual problem:

$$\begin{aligned} \max_{u,v} \quad & g(u,v) \\ \text{subject to} \quad & u \geq 0. \end{aligned}$$

if the dual optimal value is g^*, then

$$f^* \geq g^*$$

this is called weak duality. Note that this always holds (even if primal problem is nonconvex).

The dual problem is a convex optimization problem (as written, it is a concave maximization problem). Again, this is always true (even when primal problem is not convex).

By definition:

$$\begin{aligned} g(u,v) &= \min_{x} f(x) + \sum_{i=1}^{m} u_ih_i(x) + \sum_{j=1}^{r} v_jl_j(x) \\ &= -\max_{x} \quad \underbrace{-f(x) - \sum_{i=1}^{m} u_ih_i(x) - \sum_{j=1}^{r} v_jl_j(x)}_{\text{pointwise maximum of affine functions in } (u,v)} \end{aligned}$$

i.e., g is concave in (u,v), and $u \geq 0$ is a convex constraint, hence dual problem is a concave maximization problem.

Recall that we always have $f^* \geq g^*$(weak duality). On the other hand, in some problems we have observed that actually

$$f^* = g^*$$

which is called strong duality.

Slater's condition: if the primal is a convex problem (i.e., f and $h_1, \cdots, h_m$ are convex, $l_1, \cdots, l_r$ are affine), and there exists at least one strictly feasible $x \in \mathbf{R}^n$, meaning

$$h_1(x) \leq 0, \cdots, h_1(x) \leq 0 \text{ and } l_1(x) = 0, \cdots, l_r(x) = 0$$

then strong duality holds.

This is a pretty weak condition. An important refinement: strict inequalities only need to hold over functions h_i that are not affine.

8.5 Duality gap

Given primal feasible x and dual feasible u, v, the quantity

$$f(x) - g(u, v)$$

is called the duality gap between x and u, v. Note that

$$f(x) - f^* \leq f(x) - g(u, v), \quad g^* - g(u, v) \leq f(x) - g(u, v) \tag{40}$$

so if the duality gap is zero, then x is primal optimal (and similarly, u, v are dual optimal).

From an algorithmic viewpoint, provides a stopping criterion: if $f(x) - g(u, v) \leq \epsilon$, then we are guaranteed that $f(x) - f^* \leq \epsilon$. This is very useful, especially in conjunction with iterative methods.

9 Karush-Kuhn-Tucker conditions

9.1 KKT conditions

Given general problem

$$\begin{aligned} \min_x \quad & f(x) \\ \text{subject to} \quad & h_i(x) \leq 0, i = 1, \cdots, m \\ & l_j(x) = 0, j = 1, \cdots, r. \end{aligned}$$

The Karush-Kuhn-Tucker conditions or KKT conditions are:

Stationarity $0 \in \partial \left(f(x) + \sum_{i=1}^m u_i h_i(x) + \sum_{j=1}^r v_j l_j(x) \right)$

Complementary slackness $u_i h_i(x) = 0$ for all i

Primal feasibility $h_i(x) \leq 0, l_j(x) = 0$ for all i, j

Dual feasibility $u_i \geq 0$ for all i

9.2 Necessity

Let x^* and u^*, v^* be primal and dual solutions with zero duality gap (strong duality holds, e.g., under Slater's condition). Then

$$\begin{aligned} f(x^*) &= g(u^*, v^*) \\ &= \min_x f(x) + \sum_{i=1}^m u_i^* h_i(x) + \sum_{j=1}^r v_j^* l_j(x) \\ &\leq f(x^*) + \sum_{i=1}^m u_i^* h_i(x^*) + \sum_{j=1}^r v_j^* l_j(x^*) \\ &\leq f(x^*) \end{aligned}$$

In other words, all these inequalities are actually equalities.

- The point x^* minimizes $L(x, u^*, v^*)$ over $x \in \mathbf{R}^n$. Hence the subdifferential of $L(x, u^*, v^*)$ must contain 0 at $x = x^*$, which is exactly the stationarity condition;
- We must have $\sum_{i=1}^m u_i^* h_i(x^*) = 0$, and since each term here is nonpositive, this implies $u_i^* h_i(x^*) = 0$ for every i, which is exactly the complementary slackness;
- Primal and dual feasibility hold by virtue of optimality.

Therefore:
If x^* and u^*, v^* are primal and dual solutions, with zero duality gap, then x^*, u^*, v^* satisfy the KKT conditions.
Note that this statement assumes nothing a priori about convexity of our problem, i.e., of f, h_i, l_j.

9.3 Sufficiency

If there exists x^*, u^*, v^* that satisfy the KKT conditions, then

$$\begin{aligned}
g(u^*, v^*) &= L(x^*, u^*, v^*) \quad \text{(stationarity)} \\
&= f(x^*) + \sum_{i=1}^m u_i^* h_i(x^*) + \sum_{j=1}^r v_j^* l_j(x^*) \\
&= f(x^*) \quad \text{(complementary slackness and primal feasibility)} \\
&\le L(x, u^*, v^*) = f(x) + \sum_{i=1}^m u_i^* h_i(x) + \sum_{j=1}^r v_j^* l_j(x), \quad \forall x \in \mathbf{R}^n \\
&\le f(x) + \sum_{i=1}^m u_i^* h_i(x) + \sum_{j=1}^r v_j^* l_j(x), \quad \forall x \in C \\
&\le f(x)
\end{aligned}$$

Thus,

$$g(u^*, v^*) = f(x^*) \le L(x, u^*, v^*) \le f(x)$$

which proves sufficiency, i.e., x^* and u^*, v^* are the primal solution(or solutions if not unique) and dual solutions.
Note that this sufficiency always holds since the subdifferential of $L(x^*, u^*, v^*)$ contains 0 which means $L(x, u^*, v^*) \ge L(x^*, u^*, v^*) + \langle 0, x - x^* \rangle$ (see this discussion).
Therefore the duality gap is zero (and x^* and u^*, v^* are primal and dual feasible) so x^* and u^*, v^* are primal and dual optimal. Hence, we've shown:
If x^* and u^*, v^* satisfy the KKT conditions, then x^* and u^*, v^* are primal and dual solutions.

9.4 Summary about KKT conditions

In summary, KKT conditions:

- always sufficient;
- necessary under strong duality.

For a problem with strong duality (e.g., assume Slater's condition: convex problem and there exists x strictly satisfying non-affine inequality contraints),
x^* and u^*, v^* are primal and dual solutions $\Leftrightarrow$ x^* and u^*, v^* satisfy the KKT conditions
Warning: concerning the stationarity condition: for a differentiable function f, we cannot use $\partial f(x) = \{\nabla f(x)\}$ unless f is convex!
For unconstrained problems, the KKT conditions are nothing more than the subgradient optimality condition.
For general convex problems, the KKT conditions could have been derived entirely from studying optimality via subgradients

$$0 \in \partial f(x^*) + \sum_{i=1}^m \mathcal{N}_{\{h_i(x^*) \le 0\}}(x^*) + \sum_{j=1}^r \mathcal{N}_{\{l_j(x^*) = 0\}}(x^*)$$

where $\mathcal{N}_C(x)$ is the normal cone of C at x.

9.5 Example: quadratic with equality constraints

Consider for $Q \succeq 0$,

$$\begin{aligned} \min_x \quad & \frac{1}{2}x^TQx + c^Tx \\ \text{subject to} \quad & Ax = 0. \end{aligned}$$

This is a convex problem that contains no inequality constraints, so by KKT conditions: x is a solution if and only if

$$\begin{bmatrix} Q & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} x^* \\ v^* \end{bmatrix} = \begin{bmatrix} -c \\ 0 \end{bmatrix}.$$

for some v^*. This corresponds to Newton step for equality-constrained problem $\min_x f(x)$ subject to $Ax = b$ after replacing the RHS with $\begin{bmatrix} -c \\ b \end{bmatrix}$. The 2×2 matrix on the LHS is also called KKT matrix.

Linear system combines stationarity, primal feasibility (complementary slackness and dual feasibility are vacuous).

9.6 Constrained and Lagrange forms

Often in statistics and machine learning we'll switch back and forth between constrained form, where $t \in \mathbf{R}$ is a tuning parameter,

$$\min_x f(x) \text{ subject to } h(x) \le t \tag{C}$$

and Lagrange form, where $\lambda \ge 0$ is a tuning parameter,

$$\min_x f(x) + \lambda \cdot h(x) \tag{L}$$

(C) to (L): if problem (C) is strictly feasible, then strong duality holds, and there exists some $\lambda \ge 0$ (dual solution) such that any solution x^* in (C) minimizes

$$f(x) + \lambda(h(x) - t)$$

so x^* is also a solution to (L).

(L) to (C): if x^* is a solution to (L), then the KKT conditions for (C) are satisfied, so x^* is a solution to (C).

Note that when the only value of t that leads to a feasible but not strictly feasible constraint set is $t = 0$, then this is a perfect equivalence.

9.7 Uses of duality

From (40), one important use is that the duality gap can be used as a stopping criterion in algorithms. Another use is solving the primal via the dual.

Under strong duality, given a dual solution u^*, v^* any primal solution x^* solves

$$\min_x f(x) + \sum_{i=1}^m u_i^* h_i(x) + \sum_{j=1}^r v_j^* l_j(x)$$

Often, by using stationarity, solutions of this unconstrained problem can be expressed explicitly, giving an explicit characterization of primal solutions from dual solutions. Suppose the solution of this problem is unique, then it must be the primal solution. This can be very helpful when the dual is easier to solve than the primal. For example,

$$\min_x \sum_{i=1}^n f_i(x_i) \text{ subject to } a^Tx = b$$

where each f_i is smooth and strictly convex. The dual function is

$$\begin{aligned} g(v) &= \min_x \sum_{i=1}^n f_i(x_i) + v(b - a^Tx) \\ &= vb + \sum_{i=1}^n \min_x \{f_i(x_i) - va_ix_i\} \\ &= vb - \sum_{i=1}^n \max_x \{va_ix_i - f_i(x_i)\} \\ &= vb - \sum_{i=1}^n f_i^*(va_i) \end{aligned}$$

where f_i^* is the conjugate of f_i. Therefore the dual problem is

$$\max_v vb - \sum_{i=1}^n f_i^*(va_i) \Longleftrightarrow \min_v \sum_{i=1}^n f_i^*(va_i) + vb$$

This is a convex minimization problem with a scalar variable—much easier to solve than the primal problem. Given v^*, the primal solution x^* solves

$$\sum_{i=1}^n \min_x \{f_i(x_i) - v^* a_i x_i\}$$

Strict convexity of each f_i implies that this has a unique solution, namely x^*, which we compute by solving $\nabla f_i(x_i) = a_i v^*$ for each i.

10 Dual norms

Let $\|x\|$ or $\|X\|$ be a norm, we define its dual norm $\|x\|$ as

$$\|x\|_* = \max_{\|z\|\leq 1} \{z^T x\}$$

which gives us the inequality $z^T x \leq \|z\| \|x\|_*$ (like generalized Holder). Given a vector y, we always have $\|\frac{y}{\|y\|}\| = 1$, then

$$\frac{y^T}{\|y\|} x \leq \max_{\|z\|\leq 1} \{z^T x\} = \|x\|_* \Leftrightarrow y^T x \leq \max_{\|z\|\leq 1} \{z^T x\} = \|y\| \|x\|_* \tag{41}$$

Some examples:

- l_p norm dual: $(\|x\|_p)_* = \|x\|_q$, where $\frac{1}{p} + \frac{1}{q} = 1$, e.g. $p = 2, q = 2$ and $p = 1, q = \infty$ are dual pairs.
- Trace norm dual: $(\|X\|_{\text{tr}})_* = \|X\|_{\text{op}} = \sigma_1(X)$

Here $\|X\|_{\text{op}} = \|X\|_2$. Often times, we use $\|X\|_{\text{op}}$ and $\|X\|_2$ interchangeably. Strictly speaking, the spectral norm is a special case of the operator norm.

Dual norm of dual norm: $\|x\|_{**} = \|x\|$. We can prove it by solving the dual problem of the following trivial-looking problem and using strong duality.

$$\min_y \|y\| \text{ s.t. } y = x$$

11 Conjugate function

Given a function $f : \mathbf{R}^n \to \mathbf{R}$, we define its conjugate as $f^* : \mathbf{R}^n \to \mathbf{R}$,

$$f^*(y) = \max_x \{y^T x - f(x)\}$$

In other words, $f^*(y)$ is the maximum gap between the linear function $y^T x$ and $f(x)$.

Note that f^* is always convex, since it is the pointwise maximum of convex (affine) functions in y (here f need not be convex).

In physics, for differentiable f, conjugation is called Legendre transform[15].

Properties:

- Fenchel's inequality: $f(x) + f(y^*) \geq x^T y \ \forall x, y$.

  Proof. By the definition of conjugate,

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

  □

- Conjugate of conjugate $f^{**} \leq f$
- If f is closed and convex, then $f^{**} = f$

[15]See the footnote of page 3 in http://www.stat.cmu.edu/~ryantibs/convexopt-F18/scribes/Lecture_13.pdf

- If f is closed and convex, then for any x, y,

$$\begin{aligned} x \in \partial f^*(y) &\iff y \in \partial f(x) \\ &\iff f(x) + f^*(y) = x^T y \end{aligned}$$

For example, if f and f^* are differentiable, then $(\nabla f)^{-1} = \nabla f^*$. Said another way,

$$x \in \nabla f^*(y) \iff y \in \nabla f(x)$$

- If $f(u, v) = f_1(u) + f_2(v)$, then

$$f^*(w, z) = f_1^*(w) + f_2^*(z)$$

Examples:

- Simple quadratic: let $f(x) = \frac{1}{2}x^T Q x$, where $Q \succ 0$. Then $y^T x - \frac{1}{2}x^T Q x$ is strictly concave in y and is maximized at $x = Q^{-1}y$, so

$$f^* = \frac{1}{2}y^T Q y$$

- Indicator function: if $f(x) = I_C(x)$, its conjugate is

$$f^*(y) = \max_x y^T x - f(x) = \max_{x \in C} y^T x = I_C^*(y)$$

called the support function of C

- Norm: if $f(x) = \|x\|$, then its conjugate is

$$f^*(y) = I_{\|z\|_* \leq 1}(y)$$

Proof. By the definition of conjugate,

$$f^*(y) = \max_x \{y^T x - \|x\|\}$$

if $\|y\|_* > 1$, then there exists a z with $\|z\| \leq 1$ such that $y^T z > 1$ which implies $\frac{y^T z}{\|z\|} \geq y^T z > 1$. Furthermore, $y^T z - \|z\| > 1$ Let $x = tz$ with $t \geq 0$, then

$$y^T x - \|x\| = t(y^T z - \|z\|) \to +\infty \text{ as } t \to +\infty.$$

From (41), $y^T x \leq \|x\| \|y\|_*$ always holds. If $\|y\|_* \leq 1$, then $y^T x \leq \|x\|$, i.e., $y^T x - \|x\| \leq 0$. When $x = 0$, the maximum is achieved, i.e. 0. Hence, $f^*(y) = I_{\{\|z\|_* \leq 1\}}(y)$. □

11.1 Lasso dual

Given $y \in \mathbf{R}^n, X \in \mathbf{R}^{n \times p}$, recall the lasso problem:

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

We transform the primal to

$$\min_{\beta, z} \frac{1}{2}\|y - z\|_2^2 + \lambda\|\beta\|_1 \text{ s.t. } z = X\beta$$

so now the dual function is

$$\begin{aligned} g(v) &= \min_{\beta, z} \frac{1}{2}\|y - z\|_2^2 + \lambda\|\beta\|_1 + v^T(z - X\beta) \\ &= \min_z \{\frac{1}{2}\|y - z\|_2^2 + v^T z\} + \min_\beta \{\lambda\|\beta\|_1 - (X^T v)^T \beta)\} \\ &= -\lambda \max_\beta \{\frac{(X^T v)^T \beta}{\lambda}) - \|\beta\|_1\} \\ &= v^T y - \frac{1}{2}\|v\|^2 - \lambda I_{\{u: \|u\|_\infty \leq 1\}}(\frac{X^T v}{\lambda}) \\ &= \frac{1}{2}\|y\|_2^2 - \frac{1}{2}\|y - v\|_2^2 - I_{\{u: \|u\|_\infty \leq 1\}}(\frac{X^T v}{\lambda}) \end{aligned}$$

Therefore the lasso dual problem is

$$\max_v \{\frac{1}{2}\|y\|_2^2 - \frac{1}{2}\|y-v\|_2^2\} \text{ s.t. } \|X^T v\|_\infty \leq \lambda \Longleftrightarrow \min_v \{\frac{1}{2}\|y-v\|_2^2\} \text{ s.t. } \|X^T v\|_\infty \leq \lambda$$

Check: Slater's condition holds, and hence so does strong duality. Actually, this is a projection problem, i.e. projecting y onto a l_∞ norm ball. But note: the optimal value of the last problem is not the optimal lasso objective value because we dropped $\frac{1}{2}\|y\|_2^2$ on the RHS of $\Longleftrightarrow$.

Further, note that given the dual solution u, any lasso solution β satisfies

$$X\beta = y - v$$

This is from KKT stationarity condition for z (i.e., $z - y + v = 0$). So the lasso fit is just the dual residual.

11.2 Conjugates and dual problems

Conjugates appear frequently in derivation of dual problems, via

$$-f^*(u) = \min_x \{f(x) - u^T x\}$$

in minimization of the Lagrangian. E.g., consider

$$\min_x f(x) + h(x)$$

Equivalently: $\min_{x,z}\{f(x) + h(z)\}$ s.t. $z = x$. Dual function:

$$\begin{aligned} g(u) &= \min_{x,z}\{f(x) + h(z) + v^T(z - x)\} \\ &= \min_x\{f(x) - v^T x\} + \min_z\{h(z) + v^T z\} \\ &= -f^*(v) - h^*(-v) \end{aligned}$$

Hence, the dual problem is

$$\max_v \; -f^*(v) - h^*(-v).$$

Examples of this last calculation:

- Indicator function:

$$\text{Primal: } \min_x \; f(x) + I_C(x)$$

$$\text{Dual: } \max_u \; -f^*(u) - I_C^*(-u)$$

  where I_C^* is the support function of C.

- Norms: the dual of

$$\text{Primal: } \min_x \; f(x) + \|x\|$$

$$\text{Dual: } \max_u \; -f^*(u) \text{ s.t. } \|u\|_* \leq 1$$

  where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$.

11.3 Shifting linear transformations

Dual formulations can help us by "shifting" a linear transformation between one part of the objective and another. Consider

$$\min_x f(x) + g(Ax)$$

Equivalently: $\min_{x,z}\{f(x) + g(z)\}$ s.t. $z = Ax$. As before, the dual function:

$$\max_v \; -f^*(A^T v) - g^*(-v).$$

Example: for a norm and its dual norm, $\|\cdot\|, \|\cdot\|_*$,

$$\text{Primal: } min_x \; f(x) + \|Ax\|$$

$$\text{Dual: } max_u \; -f^*(A^T u) \text{ s.t. } \|u\|_* \leq 1$$

The dual can often be a helpful transformation here.

11.4 Dual tricks and subtleties

- Often, we will transform the dual into an equivalent problem and still call this the dual. Under strong duality, we can use solutions of the (transformed) dual problem to characterize or compute primal solutions. **Warning**: the optimal value of this transformed dual problem is not necessarily the optimal primal value because of simplification.
- A common trick in deriving duals for unconstrained problems is to first transform the primal by adding a dummy variable and an equality constraint. Usually there is ambiguity in how to do this. Different choices can lead to different dual problems!

Consider general minimization problem with linear constraints:

$$\begin{aligned} \min_x \quad & f(x) \\ \text{subject to} \quad & Ax \le b,\ Cx = d. \end{aligned}$$

The Lagrangian is

$$L(x,u,v) = f(x) + (A^T u + C^T v)^T x - b^T u - d^T v$$

and hence the dual problem is

$$\begin{aligned} \max_{u,v} \ & -f^*(-A^T u - C^T v) - b^T u - d^T v \\ \text{s.t. } & u \ge 0 \end{aligned}$$

Recall property: $f^{**} = f$ if f is closed and convex.

12 Newton's method

Given unconstrained, smooth convex optimization

$$\min_x f(x)$$

where f is convex, twice differentable, and $\mathrm{dom}(f) = \mathbf{R}^n$. Recall that gradient descent chooses initial $x^{(0)} \in \mathbf{R}^n$, and repeats

$$x^{(k)} = x^{(k-1)} - t_k \cdot \nabla f(x^{(k-1)}), k = 1, 2, 3, \dots$$

In comparison, Newton's method repeats

$$x^{(k)} = x^{(k-1)} - \left(\nabla^2 f(x^{(k-1)})\right)^{-1} \nabla f(x^{(k-1)}), k = 1, 2, 3, \dots$$

where $\nabla^2 f(x^{(k-1)})$ is the Hessian matrix of f at $x^{(k-1)}$.

12.1 Newton's method interpretation

Recall the motivation for gradient descent step at x: we minimize the quadratic approximation

$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2t}\|y - x\|_2^2$$

over y, and this yields the update $x^+ = x - t\nabla f(x)$.

Newton's method uses in a sense a better quadratic approximation

$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(x)(y - x)$$

and minimizes over y to yield $x^+ = x - \left(\nabla^2 f(x)\right)^{-1} \nabla f(x)$

Linearized optimality condition

Aternative interpretation of Newton step at x: we seek a direction v so that $\nabla f(x+v) = 0$. Let $F(x) = \nabla f(x)$. Consider linearizing F around x, via approximation $F(y) = F(x) + DF(x)(y - x)$, i.e.,

$$0 = \nabla f(x + v) \approx \nabla f(x) + \nabla^2 f(x) v$$

Solving for v yields $v = \left(\nabla^2 f(x^{(k-1)})\right)^{-1} \nabla f(x)$. Note that the transpose of the derivative (or Jacobian) of f is called the gradient of the function, i.e., $\nabla f(x) = Df(x)^T$. History: work of Newton (1685)and Raphson (1690) originally focused on finding roots of polynomials. Simpson (1740) applied this idea to general nonlinear equations, and minimization by setting the gradient to zero.

12.2 Affine invariance of Newton's method

Given f, nonsingular $A \in \mathbf{R}^{n\times n}$. Let $x = Ay$, and $g(y) = f(Ay)$. Newton steps on g are

$$\begin{aligned} y^+ &= y - \left(\nabla^2 g(y)\right)^{-1} \nabla g(y) \\ &= y - \left(A^T \nabla^2 f(Ay) A\right)^{-1} (A^T)^{-1} A^T \nabla f(Ay) \\ &= y - A^{-1} \left(\nabla^2 f(Ay)\right)^{-1} \nabla f(Ay) \end{aligned}$$

Hence,

$$Ay^+ = A\left(y - A^{-1}\left(\nabla^2 f(Ay)\right)^{-1} \nabla f(Ay)\right)$$

i.e.,

$$x^+ = x - \left(\nabla^2 f(x)\right)^{-1} \nabla f(x)$$

So progress is independent of problem scaling; recall that this is not true of gradient descent.

12.3 Newton decrement

At a point x, we define the Newton decrement as

$$\lambda(x) = \left(\nabla f(x)^T \left(\nabla^2 f(x)\right)^{-1} \nabla f(x)\right)^{\frac{1}{2}}$$

This relates to the difference between $f(x)$ and the minimum of its quadratic approximation:

$$\begin{aligned} f(x) - \min_y &\left\{ f(x) + \nabla f(x)^T (y-x) + \frac{1}{2}(y-x)^T \nabla^2 f(x)(y-x)\right\} \\ &= f(x) - \left(f(x) - \frac{1}{2}\left(\nabla f(x)^T \left(\nabla^2 f(x)\right)^{-1} \nabla f(x)\right)\right) \\ &= \frac{1}{2}\lambda(x)^2 \end{aligned}$$

Therefore we can think of $\frac{1}{2}\lambda(x)^2$ as an approximate upper bound on the suboptimality gap $f(x) - f^*$.

Another interpretation of Newton decrement: if Newton direction is $v = -\left(\nabla^2 f(x)\right)^{-1} \nabla f(x)$, then

$$\lambda(x) = \left(v^T \nabla^2 f(x) v\right)^{\frac{1}{2}} = \|v\|_{\nabla^2 f(x)}$$

i.e., $\lambda(x)$ is the length of the Newton step in the norm defined by the Hessian $\nabla^2 f(x)$.

Note that the Newton decrement, like the Newton steps, are affine invariant; i.e., if we defined $g(y) = f(Ay)$ for nonsingular A, then $\lambda_g(y)$ would match $\lambda_f(x)$ at $x = Ay$. Since

$$\begin{aligned} \lambda_g(y) &= \left(\nabla g(y)^T \left(\nabla^2 g(y)\right)^{-1} \nabla g(y)\right)^{\frac{1}{2}} \\ &= \left(\nabla f(Ay)^T A \left(A^T \nabla^2 f(Ay) A\right)^{-1} A^T \nabla f(Ay)\right)^{\frac{1}{2}} \\ &= \left(\nabla f(Ay)^T \left(\nabla^2 f(Ay)\right)^{-1} \nabla f(Ay)\right)^{\frac{1}{2}} \\ &= \lambda_f(x) \end{aligned}$$

12.4 Backtracking line search

So far what we've seen is called pure Newton's method. This need not converge. In practice, we use damped Newton's method (i.e., Newton's method), which repeats

$$x^+ = x - t\left(\nabla^2 f(x)\right)^{-1} \nabla f(x)$$

Note that the pure method uses $t = 1$.

Step sizes here typically are chosen by backtracking search, with parameters $0 < \alpha \le 1/2, 0 < \beta < 1$. At each iteration, we start with $t = 1$ and while

$$f(x+v) > f(x) + \alpha t \nabla f(x)^T v$$

we shrink $t = \beta t$, else we perform the Newton update. Note that here $v = -\left(\nabla^2 f(x)\right)^{-1} \nabla f(x)$, so $\nabla^2 f(x) v = -\lambda(x)^2$.

12.5 Convergence analysis

Assume that f convex, twice differentiable, having $\mathrm{dom} f = \mathbf{R}^n$

- $\nabla f(x)$ is Lipschitz with parameter M, i.e., $\nabla^2 f(x) \preceq MI$
- f is strongly convex with parameter m
- $\nabla^2 f(x)$ is Lipschitz with parameter L

Theorem 10. Newton's method with backtracking line search satisfies the following two-stage convergence bounds

$$f(x^k) - f^* \leq \begin{cases} f(x^k) - f^* - \gamma k, & \text{if } k \leq k_0 \\ \frac{2m^3}{L^2}(\frac{1}{2})^{k-k_0+1}, & \text{if } k > k_0. \end{cases}$$

where $\gamma = \alpha\beta^2\eta^2 m/M^2$, $\eta = \min\{1, 3(1-2\alpha)\}m^2/L$, and k_0 is the number of steps until $\|\nabla f(x^{(k_0+1)})\|_2 < \eta$.

The following analysis w.r.t the convergence proof of Newton's method follows from section 9.5.3 of B & V's Convex Optimization book??.

We first present an idea and outline of convergence proof. We will show that there are numbers γ and η with $\gamma > 0$ and $0 < \eta \leq m^2/L$.

- Damped phase: if $\|\nabla f(x^{(k)})\| \geq \eta$, then
$$f(x^{(k+1)}) - f(x^{(k)}) \leq -\gamma \tag{42}$$
- Pure phase: if $\|\nabla f(x^{(k)})\| < 0$, then the backtracking line search selects $t_k = 1$ and
$$\frac{L}{2m^2}\|\nabla f(x^{(k+1)})\|_2 \leq \left(\frac{L}{2m^2}\|\nabla f(x^{(k)})\|_2\right)^2 \tag{43}$$

Let us analyze the implications of the second condition. Suppose that it is satisfied for iteration k, i.e., $\|\nabla f(x^{(k)})\|_2 < \eta$. Since $0 < \eta \leq m^2/L$, we have

$$\begin{aligned}
\|\nabla f(x^{(k+1)})\|_2 &\leq \frac{2m^2}{L}\left(\frac{L}{2m^2}\|\nabla f(x^{(k)})\|_2\right)^2 \\
&< \frac{2m^2}{L}\left(\frac{L}{2m^2}\eta\right)^2 \\
&= \frac{2m^2}{L}\left(\frac{L}{2m^2}\right)^2 \eta \cdot \eta \\
&\leq \frac{2m^2}{L}\left(\frac{L}{2m^2}\right)^2 \frac{2m^2}{L} \cdot \eta \\
&= \frac{1}{2}\eta < \eta
\end{aligned}$$

Thus, the second condition is also satisfied at iteration $k+1$. Continuing recursively, we conclude that once the second condition holds, it will hold for all future iterates, i.e., for all $l \geq k$, we have $\|\nabla f(x^{(l)})\|_2 < \eta$.

Proof. Strong convexity implies that $\nabla^2 f(x) \preceq MI$, so we have

$$\begin{aligned}
f(x + tv) &\leq f(x) + t\nabla f(x)^T v + \frac{Mt^2}{2}\|v\|_2^2 \\
&= f(x) - t\nabla f(x)^T \nabla^2 f(x)^{-1}\nabla f(x) + \frac{Mt^2}{2}\|v\|_2^2 \\
&\leq f(x) - t\lambda(x)^2 + \frac{M}{2m}t^2\lambda(x)^2 \\
&= f(x) + (\frac{M}{2m}t^2 - t)\lambda(x)^2 \\
&= f(x) - \frac{1}{2}t\lambda(x)^2 \quad (\text{when } 0 < t \leq \frac{m}{M}) \\
&\leq f(x) - \alpha t\lambda(x)^2 \quad (\text{since } 0 < \alpha \leq \frac{1}{2})
\end{aligned}$$

where we use $\lambda(x)^2 = \nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x) = v^T \nabla^2 f(x) v \geq m\|v\|_2^2$ in the third inequality. From the last two inequalities, we know that if $t \leq \frac{m}{M}$, the exit condition of the line search is always satisfied. Thus, the line search returns a step size $t \geq \beta m/M$ because if $t < \beta m/M$, then the stepsize in the previous iteration $t/\beta < m/M$ which implies the exit condition was already satisfied in the previous iteration, a contradiction. This step size results in a decrease of the objective function

$$\begin{aligned} f(x^+) - f(x) &\leq -\alpha t \lambda(x)^2 \\ &\leq -\alpha\beta \frac{m}{M} \lambda(x)^2 \\ &\leq -\alpha\beta \frac{m}{M}\frac{1}{M} \|\nabla f(x)\|_2^2 \\ &\leq -\alpha\beta\eta^2 \frac{m}{M^2} \quad (\text{since } \|\nabla f(x)\|_2 \geq \eta) \end{aligned}$$

where the third inequality use $\lambda(x)^2 = \nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x) \geq \frac{1}{M}\|\nabla f(x)\|_2^2$. Therefore, (42) is satisfied with

$$\gamma = \alpha\beta\eta^2 \frac{m}{M^2}.$$

We now establish the inequality (43). Assume $\|\nabla f(x)\|_2 < \eta$. We first show that the backtracking line search selects unit steps, provided

$$\eta = 3(1-2\alpha)m^2/L$$

By the Lipschitz condition of $\nabla^2 f(x)$, we have, for $t \geq 0$,

$$\|\nabla^2 f(x+tv) - \nabla^2 f(x)\|_2 \leq tL\|v\|_2$$

then

$$|v^T \left(\nabla^2 f(x+tv) - \nabla^2 f(x)\right) v| \leq tL\|v\|_2^3$$

With $\tilde{f}(t) = f(x+tv)$, we have $\tilde{f}'(t) = \nabla f(x+tv)^T v$ and $\tilde{f}''(t) = v^T \nabla^2 f(x+tv)^T v$, so the inequality above is

$$|\tilde{f}''(t) - \tilde{f}''(0)| \leq tL\|v\|_2^3$$

thus

$$\tilde{f}''(t) \leq \tilde{f}''(0) + tL\|v\|_2^3 \leq \lambda(x)^2 + \frac{tL}{m^{3/2}}\lambda(x)^3$$

where we use $\lambda(x)^2 \geq m\|v\|_2^2$ in the last inequality. Integrating over t, we get

$$\tilde{f}'(t) \leq \tilde{f}'(0) + \lambda(x)^2 t + \lambda(x)^3 \frac{t^2 L}{m^{3/2}} = -\lambda(x)^2 + \lambda(x)^2 t + \lambda(x)^3 \frac{t^2 L}{m^{3/2}}$$

Integrating over t again,

$$\tilde{f}(t) \leq \tilde{f}(0) - \lambda(x)^2 t + \lambda(x)^2 \frac{t^2}{2} + \lambda(x)^3 \frac{t^3 L}{6m^{3/2}}$$

letting $t = 1$,

$$\begin{aligned} f(x+v) &\leq f(x) - \lambda(x)^2 + \lambda(x)^2 \frac{1}{2} + \lambda(x)^3 \frac{L}{6m^{3/2}} \\ &\leq f(x) - \lambda(x)^2 \left(\frac{1}{2} - \lambda(x)\frac{L}{6m^{3/2}}\right) \end{aligned} \tag{44}$$

By strong convexity of f, we have

$$\lambda(x)^2 = \nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x) \leq \frac{1}{m}\|\nabla f(x)\|_2^2$$

with the condition $\|\nabla f(x)\|_2 < \eta$, if $\eta \leq 3(1-2\alpha)\frac{m^2}{L}$,

$$\lambda(x) \leq m^{\frac{1}{2}} \|\nabla f(x)\|_2 \leq 3(1-2\alpha)\frac{m^{3/2}}{L}$$

which implies the following

$$-\lambda(x)\frac{L}{6m^{3/2}} \geq -3(1-2\alpha)\frac{m^{3/2}}{L}\frac{L}{6m^{3/2}} = -\frac{1-2\alpha}{2} \Longleftrightarrow \frac{1}{2} - \lambda(x)\frac{L}{6m^{3/2}} \geq \alpha \Longleftrightarrow -\lambda(x)^2\left(\frac{1}{2} - \lambda(x)\frac{L}{6m^{3/2}}\right) \leq -\alpha\lambda(x)^2$$

combining this with (44), we obtain

$$f(x+v) \leq f(x) - \alpha\lambda(x)^2 = f(x) + \alpha\nabla f(x)^T v$$

which shows that the unit step $t = 1$ is accepted by the backtracking line search.

Let us now examine the rate of convergence. Applying the Lipschitz condition of $\nabla^2 f(x)$, we have

$$\begin{aligned}
\|\nabla f(x^+)\|_2 &\leq \|\nabla f(x+tv) - \nabla f(x) - \nabla^2 f(x)v\|_2 \\
&= \|\int_0^1 \left(\nabla^2 f(x+tv) - \nabla^2 f(x)\right) v\|_2 dt\|_2 \\
&\leq \int_0^1 \|\left(\nabla^2 f(x+tv) - \nabla^2 f(x)\right)\| \|v\|_2 dt \\
&\leq \int_0^1 tL\|v\|_2^2 \\
&= \frac{L}{2}\|v\|_2^2 \\
&\leq \frac{L}{2m^2}\|\nabla f(x)\|_2^2
\end{aligned}$$

where we use $\|v\|_2^2 \leq 1/m^2\|\nabla f(x)\|_2^2$ in the last inequality. Thus,

$$\frac{L}{2m^2}\|\nabla f(x^{(k+1)})\|_2 \leq \frac{L}{2m^2} \cdot \frac{L}{2m^2}\|\nabla f(x^{(k)})\|_2^2 \leq \left(\frac{L}{2m^2}\|\nabla f(x^{(k)})\|_2\right)^2$$

i.e., the inequality (43).

In conclusion, the algorithm selects unit steps and satisfies the condition (43) if $\|\nabla f(x)\|_2 < \eta$, where

$$\eta = \min\{1, 3(1-2\alpha)\}m^2/L$$

Applying (43) recursively, we find that for all $l \geq k$

$$\frac{L}{2m^2}\|\nabla f(x^{(l)})\|_2 \leq \left(\frac{L}{2m^2}\|\nabla f(x^{(k)})\|_2\right)^{2^{l-k}} < \left(\frac{L}{2m^2}\eta\right)^{2^{l-k}} \leq \left(\frac{L}{2m^2} \cdot \frac{m^2}{L}\right)^{2^{l-k}} = \left(\frac{1}{2}\right)^{2^{l-k}}$$

Combining this with (8), we have

$$\begin{aligned}
f(x^l) - p^* &\leq \frac{1}{2m}\|\nabla f(x^l)\|_2^2 \\
&\leq \frac{1}{2m} \cdot \left(\frac{2m^2}{L} \cdot \left(\frac{1}{2}\right)^{2^{l-k}}\right)^2 \\
&= \frac{1}{2m} \cdot \left(\frac{2m^2}{L}\right)^2 \cdot \left(\frac{1}{2}\right)^{2 \times 2^{l-k}} \\
&= \frac{2m^3}{L^2}\left(\frac{1}{2}\right)^{2^{l-k+1}}
\end{aligned}$$

This last inequality shows that convergence is extremely rapid once the second condition is satisfied. This phenomenon is called quadratic convergence. □

Now we can estimate the total complexity. First we derive an upper bound on the number of iterations in the damped Newton phase. Since f decreases by at least γ at each iteration, the number of damped Newton steps cannot exceed

$$\frac{f(x^0) - p^*}{\gamma},$$

since if it did, f would be less than p^*, which is impossible.

We can bound the number of iterations in the quadratically convergent phase using the inequality (**??**). It implies that we must have $f(x) - p^* \leq \epsilon$ after no more than

$$\log_2 \log_2(\frac{\epsilon_0}{\epsilon})$$

iterations in the quadratically convergent phase, where $\epsilon_0 = \frac{2m^3}{L^2}$.

Overall, then, the number of iterations until $f(x) - p^* \leq \epsilon$ is bounded above by

$$\frac{f(x^0) - p^*}{\gamma} + \log_2 \log_2(\frac{\epsilon_0}{\epsilon})$$

The second term grows extremely slowly with required accuracy ϵ, and can be considered a constant for practical purposes, say five or six. (Six iterations of the quadratically convergent stage gives an accuracy of about $\epsilon \approx 5 \cdot 10^{-20}\epsilon_0$.)

Not quite accurately, then, we can say that the number of Newton iterations required to minimize f is bounded above by

$$\frac{f(x^0) - p^*}{\gamma} + 6 \tag{45}$$

A more precise statement is that (45) is a bound on the number of iterations to compute an extremely good approximation of the solution.

Substituting $\gamma = \alpha\beta\eta^2 \frac{m}{M^2}$ into (45), we get the final bound

$$\frac{L^2 M^2}{\alpha\beta m^5 \min\{1, 9(1-2\alpha)^2\}} \left(f(x^0) - p^*\right) + 6$$

12.6 Self-concordance

A scale-free analysis is possible for self-concordant functions: on R, a convex function f is called self-concordant if

$$|f'''(x)| \leq 2f''(x)^{3/2} \text{ for all } x$$

and on $\mathbf{R}^n$ is called self-concordant if its projection onto every line segment is so.

Theorem 11 (Nesterov and Nemirovskii)**.** Newton's method with backtracking line search requires at most

$$C(\alpha, \beta)\left(f(x^{(0)}) - f^*\right) + \log\log(1/\epsilon)$$

iterations to reach $f(x^{(0)}) - f^* \leq \epsilon$, where $C(\alpha, \beta)$ is a constant that only depends on α and β.

What kind of functions are self-concordant?

- $f(x) = -\sum_{i=1}^n \log(x_i)$ on $\mathbf{R}^n$
- $f(X) = -\log\det(X)$ on $\mathbf{S}^n_{++}$
- If g is self-concordant, then so is $f(x) = g(Ax + b)$
- In the definition of self-concordance, we can replace factor of 2 by a general k
- If g is k-self-concordant, then we can rescale: $f(x) = k^2/4g(x)$ is self-concordant (2-self-concordant).

For the last item, given $g(x)$ is k-self-concordant, we suppose that $f(x) = ag(x)$ with $a > 0$ is 2-self-concordant. We can get the specific value of a by the following derivation.

$$\begin{aligned}
|f'''(x)| &= a|g'''(x)| \\
&\leq akg''(x)^{3/2} \quad \text{(since } g \text{ is } k\text{-self-concordant)} \\
&= ak \cdot a^{-3/2} f''(x)^{3/2} \quad \text{(since } f''(x)^{3/2} = a^{3/2} g''(x)^{3/2}\text{)} \\
&= 2\left(\frac{ka^{-1/2}}{2} f''(x)^{3/2}\right) \\
&= 2f''(x)^{3/2} \quad \text{(if } \frac{ka^{-1/2}}{2} = 1, \text{ i.e., } a = k^2/4\text{)}
\end{aligned}$$

12.7 Comparison to first-order methods

At a high-level:

- Memory: each iteration of Newton's method requires $O(n^2)$ storage ($n \times n$ Hessian); each gradient iteration requires $O(n)$ storage (n-dimensional gradient);
- Computation: each Newton iteration requires $O(n^3)$ flops (solving a dense $n \times n$ linear system); each gradient iteration requires $O(n)$ flops (scaling/adding n-dimensional vectors);

- Backtracking: backtracking line search has roughly the same cost, both use O(n) flops per inner backtracking step;
- Conditioning: Newton's method is not affected by a problem's conditioning, but gradient descent can seriously degrade;
- Fragility: Newton's method may be empirically more sensitive to bugs/numerical errors, gradient descent is more robust.

12.8 Sparse and structured problems

When the inner linear systems (in Hessian) can be solved efficiently and reliably, if $\nabla^2 f(x)$ is sparse and structured for all x, say banded, then both memory and computation are $O(n)$ with Newton iterations.

13 Equality-constrained Newton's method

Consider now a problem with equality constraints, as in

$$\min_x f(x) \text{ s.t. } Ax = b$$

Several options:

- Eliminating equality constraints: write $x = Fy + x_0$, where F spans null space of A, and $Ax_0 = b$. Solve in terms of y;
- Deriving the dual: can check that the Lagrange dual function is $-f^*(-A^T v) - b^T v$, and strong duality holds. With luck, we can express x^* in terms of v^*;
- Equality-constrained Newton: in many cases, this is the most straightforward option.

To derive the Newton step Δx_{nt} for the above equality constrained problem at the feasible point x, we replace the objective with its second-order Taylor approximation near x, to form the problem

$$\min_v \hat{f}(x+v) = f(x) + \nabla f(x)^T v + (1/2) v^T \nabla^2 f(x) v \quad \text{s.t. } A(x+v) = b.$$

We know from KKT conditions that the Newton step is characterized by Δx_{nt} satisfies

$$\begin{bmatrix} \nabla^2 f(x) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x_{\text{nt}} \\ w \end{bmatrix} = \begin{bmatrix} -\nabla f(x) \\ 0 \end{bmatrix}$$

14 Barrier method

The intuition behind the barrier method is that we are going to push the inequality constraints in the problem in a smooth way, reducing the problem to an equality-constrained problem.

14.1 Log barrier function

Consider the convex optimization problem

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & h_i(x) \le 0, \ i = 1, \cdots, m \\ & Ax = b \end{aligned}$$

We will assume that $f, h_1, \cdots, h_m$ are convex, twice differentiable, each with domain $\mathbf{R}^n$. The function

$$\phi(x) = -\sum_{i=1}^m \log\left(-h_i(x)\right)$$

is called the log barrier for the above problem. Its domain is the set of strictly feasible points, $\{x : h_i(x) < 0, \ i = 1, \cdots, m\}$ which we assume is nonempty. (Note this implies strong duality holds)

Its gradient:

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

Its Hessian:

$$\nabla^2\phi(x) = \sum_{i=1}^{m} \frac{1}{h_i(x)^2}\nabla h_i(x)\nabla h_i(x)^T - \sum_{i=1}^{m} \frac{1}{h_i(x)}\nabla^2 h_i(x).$$

Ignoring equality constraints for now, our problem can be written as

$$\min_x \; f(x) + \sum_{i=1}^{m} I_{h_i(x)\le 0}(x)$$

We can approximate the sum of indicators by the log barrier:

$$\min_x \; f(x) - 1/t\sum_{i=1}^{m} \log\left(-h_i(x)\right)$$

where $t > 0$ is a large number.

As shown in Fig. 8, the larger t is, the more accurate the approximation will be. But for any value of t, the log barrier approaches ∞ if any $h_i(x) \to 0$.

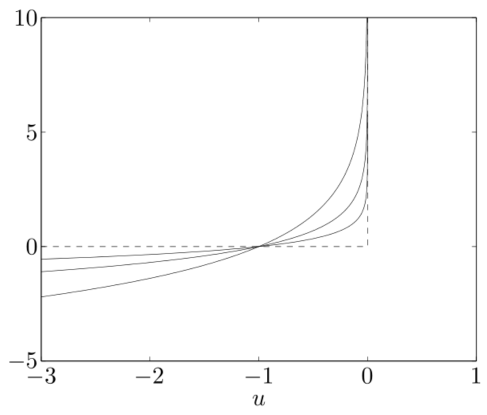


Figure 11.1 The dashed lines show the function $I_-(u)$, and the solid curves show $\widehat{I}_-(u) = -(1/t)\log(-u)$, for $t = 0.5,\ 1,\ 2$. The curve for $t = 2$ gives the best approximation.

Figure 8: Log-barrier-function(from B&V page 563 (Fig. 11.1))

14.2 Central path

Consider barrier problem:

$$\begin{aligned} \min_x \quad & tf(x) + \phi(x) \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

The central path is defined as the solution $x^*(t)$ as a function of $t > 0$

Hope is that, as $t \to \infty$, we will have $x^*(t) \to x^*(t)$, solution to our original problem.

Why don't we just set t to be some huge value, and solve the above problem? Directly seek solution at end of central path? Problem is that this is seriously inefficient in practice. It takes a long time to enter the pure phase, that is to say, quadratic convergence rate. Instead, it is better to take a sequence of increasing t values and use each solution $x^*(t)$ as a warm start for the next t. If this is done carefully, Newton's method can be kept in the pure phase.

14.3 KKT conditions and duality

Central path is characterized by its KKT conditions:

$$t\nabla f\left(x^*(t)\right) - \sum_{i=1}^{m} \frac{1}{h_i(x)}\nabla h_i(x) + A^T w = 0,$$

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

for some $w \in \mathbf{R}^m$. But we don't really care about dual variable for the barrier method.

From central path points, we can derive feasible dual points for our original problem. Given x^* and corresponding w, we define

$$u_i^*(t) = -\frac{1}{th_i(x^*)},\ i = 1, \cdots, m,\ v^*(t) = -\frac{w}{t}$$

We claim $u^*(t), v^*(t)$ are dual feasible for original problem, whose Lagrangian is

$$L(x, u, v) = f(x) + \sum_{i=1}^{m} u_i h_i(x) + v^T(Ax - b)$$

Why?

- Note that $u_i^*(t) > 0$ since $h_i(x^*(t)) < 0$ for all $i = 1, \cdots, m$
- Further, the point $(u_i^*(t), v_i^*(t))$ lies in domain of Lagrange dual function $g(u, v)$, since by definition

  $$\nabla f\left(x^*(t)\right) + \sum_{i=1}^{m} u_i^*(t) \nabla h_i(x^*(t)) + A^T v^*(t) = 0,$$

  i.e., $x^*(t)$ minimizes Lagrangian $L(x, u_i^*(t), v_i^*(t))$ over x, so $g(u^*(t), v^*(t)) > -\infty$.

14.4 Duality gap

This allows us to bound suboptimality of $f(x^*(t))$, with respect to original problem, via the duality gap. We compute

$$\begin{aligned} g(u^*(t), v^*(t)) &= f\left(x^*(t)\right) + \sum_{i=1}^{m} u_i^*(t) h_i(x^*(t)) + v^*(t)^T(Ax^*(t) - b) \\ &= f\left(x^*(t)\right) - m/t \end{aligned}$$

That is, we know that

$$f\left(x^*(t)\right) - f^* \leq f\left(x^*(t)\right) - g(u^*(t), v^*(t)) = m/t$$

where in the inequality we assume that $f^* \geq g(u^*(t), v^*(t))$. This confirms the intuitive idea that $x^*(t)$ converges to an optimal point as $t \to \infty$.

14.5 Perturbed KKT conditions

We can now reinterpret central path $(x^*(t), u^*(t), v^*(t))$ as solving the perturbed KKT conditions:

$$\nabla f(x) + \sum_{i=1}^{m} u_i \nabla h_i(x) + A^T v = 0,$$

$$u_i h_i(x) = -1/t,\ i = 1, \cdots, m,$$

$$h_i(x) \leq 0,\ i = 1, \cdots, m, Ax = b$$

$$u_i \geq 0,\ i = 1, \cdots, m.$$

Only difference between these and actual KKT conditions for our original problem is the second line: these are replaced by

$$u_i h_i(x) = 0,\ i = 1, \cdots, m$$

i.e., complementary slackness, in actual KKT conditions.

14.6 Barrier method

The barrier method solves a sequence of problems

$$\begin{aligned} \min_{x} \quad & tf(x) + \phi(x) \\ \text{s.t.} \quad & Ax = b \end{aligned}$$

for increasing values of $t > 0$, until duality gap satisfies $m/t \leq \epsilon$.

We fix $t^{(0)} > 0, \mu > 1$. We use Newton to compute $x^{(0)} = x^*(t)$, a solution to barrier problem at $t = t^{(0)}$. For $k = 1, 2, 3, \cdots$

- Solve the barrier problem at $t = t^{(k)}$, using Newton initialized at $x^{(k-1)}$, to yield $x^{(k)} = x^*(t)$
- Stop if $m/t \leq \epsilon$, else update $t^{(k+1)} = \mu t$

The first step above is called a centering step (since it brings $x^{(k)}$ onto the central path).
Considerations:

- Choice of μ: if μ is too small, then many outer iterations might be needed; if μ is too big, then Newton's method (each centering step) might take many iterations
- Choice of $t^{(0)}$: if $t^{(0)}$ is too small, then many outer iterations might be needed; if $t^{(0)}$ is too big, then the first Newton solve (first centering step) might require many iterations.

Fortunately, the performance of the barrier method is often quite robust to the choice of μ and $t^{(0)}$ in practice. (However, note that the appropriate range for these parameters is scale dependent.)

14.7 Convergence analysis

Assume that we solve the centering steps exactly. The following result is immediate.

Theorem 12. The barrier method after k centering steps satisfies

$$f(x^{(k)}) - f^* \leq \frac{m}{\mu^k t^{(0)}}$$

In other words, to reach a desired accuracy level of ϵ, we require

$$\frac{\log(m/(t^{(0)}\epsilon))}{\log \mu}$$

centering steps with the barrier method (plus initial centering step).

14.8 How many Newton iterations?

Informally, due to careful central path traversal, in each centering step, Newton is already in quadratic convergence phase, so takes nearly constant number of iterations.

This can be formalized under self-concordance. Suppose:

- The function $tf + \phi$ is self-concordant
- Our original problem has bounded sublevel sets

Then we can terminate each Newton solve at appropriate accuracy, and the total number of Newton iterations is still $\log(m/(t^{(0)}\epsilon))$ (where constants do not depend on problem-specific conditioning).

Importantly, $tf + \phi = tf - \sum_{i=1}^m \log(-h_i)$ is self-concordant when f, h_i are all linear or quadratic. So this covers LPs, QPs, QCQPs.

14.9 Feasibility methods

We have implicitly assumed that we have a strictly feasible point for the first centering step, i.e., for computing $x^{(0)} = x^*$, solution of barrier problem at $t = t^{(0)}$. This is x such that

$$h_i(x) < 0,\ i = 1, \cdots, m,\ Ax = b$$

How to find such a feasible x? By solving

$$\begin{aligned} \min_{x,s} \quad & s \\ \text{s.t.} \quad & h_i(x) \leq s, i = 1, \cdots, m \\ & Ax = b \end{aligned}$$

The goal is for s to be negative at the solution. This is known as a feasibility method. We can apply the barrier method to the above problem, since it is easy to find a strictly feasible starting point.

Note that we do not need to solve this problem to high accuracy. Once we find a feasible (x, s) with $s < 0$, we can terminate early.

An alternative is to solve the problem

$$\begin{aligned}\min_{x,s} \quad & \mathbf{1}^T s \\ \text{s.t.} \quad & h_i(x) \le s_i, i = 1, \cdots, m \\ & Ax = b, \;\; s \ge 0\end{aligned}$$

Previously s was the maximum infeasibility across all inequalities. Now each inequality has own infeasibility variable $s_i, i = 1, \cdots, m$.

One advantage: when the original system is infeasible, the solution of the above problem will be informative. The nonzero entries of s will tell us which of the constraints cannot be satisfied.

15 Primal-Dual Interior-Point Methods

15.1 Perturbed KKT as nonlinear system

The perturbed KKT conditions in section 14.5 can be viewed as a nonlinear system of equations, written as

$$r(x, u, v) = \begin{pmatrix} \nabla f(x) + Dh(x)^T u + A^T v \\ -diag(u)h(x) - 1/t\mathbf{1} \\ Ax - b \end{pmatrix} = 0$$

where

$$h(x) = \begin{pmatrix} h_1(x) \\ \cdots \\ h_m(x) \end{pmatrix}, Dh(x) = \begin{bmatrix} \nabla h_1(x)^T \\ \cdots \\ \nabla h_m(x)^T \end{bmatrix}$$

Recall that Newton's method is generally a root-finder for a nonlinear system $F(y) = 0$. Approximating $F(y + \Delta y) \approx F(y) + DF(y)\Delta y$ leads to

$$\Delta y = -\left(DF(y)^{-1} F(y)\right)$$

What happens if we apply this to $r(x, u, v) = 0$ above?

15.2 Newton on perturbed KKT, v1

Approach 1: from middle equation (relaxed complementary slackness), note that $u_i = -1/(th_i(x)), i = 1, \ldots, m$. So after eliminating u, we get

$$r(x, v) = \begin{pmatrix} \nabla f(x) + \sum_{i=1}^m \frac{1}{-th_i(x)} \nabla h_i(x) + A^T v \\ Ax - b \end{pmatrix} = 0$$

Thus the Newton root-finding update $(\Delta x, \Delta v)$ is determined by

$$\begin{bmatrix} H_{\text{bar}} & A^T \\ A & 0 \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta v \end{pmatrix} = -r(x, v)$$

where $H_{\text{bar}} = \nabla^2 f(x) + \sum_{i=1}^m \frac{1}{h_i(x)^2} \nabla h_i(x) \nabla h_i(x)^T - \sum_{i=1}^m \frac{1}{h_i(x)} \nabla^2 h_i(x)$.

This is just the KKT system solved by one iteration of Newton's method for minimizing the barrier problem.

15.3 Newton on perturbed KKT, v2

Approach 2: directly apply Newton root-finding update, without eliminating u. Introduce notation called the dual, central, and primal residuals at $y = (x, u, v)$. Now root-finding update $\Delta y = (\Delta x, \Delta u, \Delta v)$ is given by

$$\begin{bmatrix} H_{\text{pd}} & Dh(x)^T & A^T \\ -diag(u)Dh(x) & -diag(h(x)) & 0 \\ A & 0 & 0 \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta u \\ \Delta v \end{pmatrix} = -\begin{pmatrix} r_{\text{dual}} \\ r_{\text{cent}} \\ r_{\text{prim}} \end{pmatrix}$$

where $H_{\text{pd}} = \Delta^2 f(x) + \sum_{i=1}^m u_i \nabla^2 h_i(x)$.

Some notes:

- In v2, update directions for the primal and dual variables are inexorably linked together
- Also, v2 and v1 leads to different (nonequivalent) updates
- As we saw, one iteration of v1 is equivalent to inner iteration in the barrier method
- And v2 defines a new method called primal-dual interior-point method
- One complication: in v2, the dual iterates are not necessarily feasible for the original dual problem

15.4 Surrogate duality gap

For barrier method, we have simple duality gap: m/t, since we set $u_i = -1/(th_i(x)), i = 1, \cdots, m$ and saw this was dual feasible.

For primal-dual interior-point method, we can construct surrogate duality gap:

$$\eta = -h(x)^T u = -\sum_{i=1}^{m} u_i h_i(x)$$

This would be a bonafide duality gap if we had feasible points, i.e., $r_{\text{prim}} = 0$ and $r_{\text{dual}} = 0$, but we don't, so it's not.

What value of parameter t does this correspond to in perturbed KKT conditions? This is $t = m/\eta$.

15.5 Primal-dual interior-point method

Putting it all together, we now have our primal-dual interior-point method. Start with $x^{(0)}$ such that $h_i(x^{(0)}) < 0, i = 1, \cdots, m$, and $u^{(0)} > 0, u^{(0)}$. Define $\eta^{(0)} = -h(x^{(0)})^T u^{(0)}$. We fix $\mu > 1$, repeat for $k = 1, 2, 3, \cdots$

- Define $t = \mu m/\eta^{(k-1)}$
- Compute primal-dual update direction Δy
- Use backtracking to determine step size s
- Update $y^{(k)} = y^{(k-1)} + s\Delta y$
- Compute $\eta^{(k)} = -h(x^{(k)})^T u^{(k)}$
- Stop if $\eta^{(k)} \leq \epsilon$ and $(\|r_{\text{dual}}\|_2^2 + \|r_{\text{prim}}\|_2^2)^{1/2} \leq \epsilon$

Note that we stop based on surrogate duality gap, and approximate feasibility. (Line search maintains $h_i(x) < 0, u_i > 0, i = 1, \cdots, m$.

15.6 Backtracking line search

At each step, we must ensure that we arrive at $y^+ = y + s\Delta y$, i.e.,

$$x^+ = x + s\Delta x, u^+ = u + s\Delta u, v^+ = v + s\Delta v$$

that maintains both $h_i(x) < 0$, and $u_i > 0, i = 1, \cdots, m$.

A multi-stage backtracking line search for this purpose: start with largest step size $s_{\max} \leq 1$ that makes $u + s\Delta u \geq 0$:

$$s_{\max} = \min\{1, \min -u_i/\Delta u_i : \Delta u_i < 0\}$$

Then, with parameters $\alpha, \beta \in (0, 1)$, we set $s = 0.99 s_{\max}$, and

- Update $s = \beta s$, until $h_i(x) < 0, i = 1, \cdots, m$
- Update $s = \beta s$, until $\|r(x^+, u^+, v^+)\|_2 \leq (1 - \alpha s)\|r(x, u, v)\|_2$

15.7 Highlight: standard LP

Recall the standard form LP:

$$\begin{aligned} &\min c^T x \\ &\text{s.t. } Ax = b \\ &\quad x \geq 0 \end{aligned}$$

for $c \in \mathbf{R}^n, A \in \mathbf{R}^{m \times n}, b \in \mathbf{R}^m$. Its dual is:

$$\begin{aligned} &\min_{u,v} b^T v \\ &\text{s.t. } A^T v + u = c \\ &\quad u \geq 0 \end{aligned}$$

KKT conditions

The points x^* and $u^*, v^*)$ are respectively primal and dual optimal LP solutions if and only if they solve:

$$A^T v + u = c$$

$$u_i x_i = 0, i = 1, \cdots, n$$

$$Ax = b$$

$$x, u \geq 0$$

Neat fact: the simplex method maintains the first three conditions and aims for the fourth one; interior-point methods maintain the first and last two, and aim for the second.

The perturbed KKT conditions for standard form LP are hence:

$$A^T v + u = c$$

$$u_i x_i = 1/t, i = 1, \cdots, n$$

$$Ax = b$$

$$x, u \geq 0$$

What do our interior point methods do?

15.8 Interior point methods in LP

After eliminating u, Barrier method gives us

$$0 = r_{\text{br}}(x, v) = \begin{pmatrix} A^T v + \text{diag}(x)^{-1} \cdot 1/t\mathbf{1} - c \\ Ax - b \end{pmatrix}$$

The primal-dual method gives us

$$0 = r_{\text{pd}}(x, u, v) = \begin{pmatrix} A^T v + u - c \\ \text{diag}(x)u - 1/t\mathbf{1} \\ Ax - b \end{pmatrix}$$

Barrier method: set $0 = r_{\text{br}}(y + \Delta y) \approx r_{\text{br}}(y) + Dr_{\text{br}}(y)\Delta y$, i.e., solve

$$\begin{bmatrix} -\text{diag}(x)^{-2} & A^T \\ A & 0 \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta v \end{pmatrix} = -r_{\text{br}}(x, v)$$

and take a step $y^{(k)} = y^{(k-1)} + s\Delta y$ with line search $s > 0$, and iterate until convergence. Then update $t = \mu t$.

Primal-dual method: set $0 = r_{\text{pd}}(y + \Delta y) \approx r_{\text{pd}}(y) + Dr_{\text{pd}}(y)\Delta y$, i.e., solve

$$\begin{bmatrix} -\text{diag}(x)^{-2} & I & A^T \\ \text{diag}(u) & \text{diag}(x) & 0 \\ A & 0 & 0 \end{bmatrix} \begin{pmatrix} \Delta x \\ \Delta u \\ \Delta v \end{pmatrix} = -r_{\text{pd}}(x, u, v)$$

and take a step $y^{(k)} = y^{(k-1)} + s\Delta y$ with line search $s > 0$, but only once. Then update $t = \mu t$.

16 Quasi-Newton Methods

Two main steps in Newton iteration:

- Compute Hessian $\nabla^2 f(x)$
- Solve the system $\nabla^2 f(x)\Delta x = -\nabla f(x)$

Each of these two steps could be expensive. Quasi-Newton methods repeat updates of the form

$$x^+ = x + t\Delta x$$

where direction Δx is defined by linear system

$$B\Delta x = -\nabla f(x)$$

for some approximation B of $\nabla^2 f(x)$. We want B to be easy to compute, and $B\Delta = g$ to be easy to solve.

16.1 Quasi-Newton template

Let $x^{(0)} \in \mathbf{R}^n, B^{(0)} \preceq 0$. For $k = 1, 2, 3, \ldots,$, repeat:

(1) Solve $B^{(k-1)} \Delta x^{(k-1)} = -\nabla f(x^{(k-1)})$

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

(3) Compute $B^{(k)}$ from $B^{(k-1)}$

Different quasi-Newton methods implement Step 3 differently. As we will see, commonly we can compute $(B^{(k)})^{-1}$ from $(B^{(k-1)})^{-1}$.

Basic idea: as $B^{(k-1)}$ already contains info about the Hessian, use suitable matrix update to form $B^{(k)}$.

Reasonable requirement for $B^{(k)}$:

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

16.2 Secant equation

We can equivalently write the above condition as

$$\nabla f(x^+) = \nabla f(x) + B^+(x^+ - x)$$

Letting $\nabla f(x^+) - \nabla f(x)$, and $s = x^+ - x$ this becomes

$$B^+ s = y$$

This is called secant equation. In addition to secant equation, we want:

- B^+ to be symmetric
- B^+ to be close to B
- $B \preceq 0 \Rightarrow B^+ \preceq$

16.3 Symmetric rank-one update

Let's try an update of the form

$$B^+ = B + auu^T$$

The secant equation $B^+ s = y$ yields

$$(au^T s)u = y - Bs$$

This only holds if u is a multiple of $y - Bs$. Putting $u = y - Bs$, we solve the above, $a = 1/(y - Bs)^T s$, which leads to

$$B^+ = B + \frac{(y - Bs)(y - Bs)^T}{(y - Bs)^T s}$$

called the symmetric rank-one (SR1) update.

How can we solve $B^+ \Delta x^+ = -\nabla f(x^+)$, in order to take next step? In addition to propagating B to B^+, let's propagate inverses, i.e., $C = B^{-1}$ to $C^+ = (B^+)^{-1}$.

Sherman-Morrison formula:

$$(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^T A^{-1}}{1 + v^T A^{-1} u}$$

Thus for the SR1 update the inverse is also easily updated:

$$\begin{aligned}
C^+ = (B^+)^{-1} &= \big(B + a(y - Bs)(y - Bs)^T\big)^{-1} \quad (\text{let } a = 1/(y - Bs)^T s) \\
&= B^{-1} - a\frac{B^{-1}(y - Bs)(y - Bs)^T B^{-1}}{1 + a(y - Bs)^T B^{-1}(y - Bs)} \\
&= C - a\frac{(Cy - s)(y^T C - s^T)}{1 + a(y - Bs)^T (Cy - s)} \\
&= C - \frac{1}{(y - Bs)^T s} \cdot \frac{(Cy - s)(y^T C - s^T)}{1 + \frac{(y-Bs)^T(Cy-s)}{(y-Bs)^T s}} \\
&= C - \frac{(Cy - s)(y^T C - s^T)}{(y - Bs)^T s + (y - Bs)^T (Cy - s)} \\
&= C - \frac{(Cy - s)(Cy - s)^T}{(y - Bs)^T Cy} \quad (B \text{ is symmetric}) \\
&= C - \frac{(Cy - s)(Cy - s)^T}{(y^T Cy - s^T y)} \quad (B \text{ is symmetric}) \\
&= C + \frac{(s - Cy)(s - Cy)^T}{(s - Cy)^T y}
\end{aligned}$$

In general, SR1 is simple and cheap, but has key shortcoming: it does not preserve positive definiteness.

16.4 Broyden-Fletcher-Goldfarb-Shanno (BFGS) update

Instead of a rank-one update to B, let's try a rank-two update

$$B^+ = B + auu^T + bvv^T$$

Using secant equation $B^+ s = y$ gives

$$y - Bs = (au^T s)u + (bv^T s)v$$

Setting $u = y, v = Bs$ and solving for a, b we get $a = 1/y^T s, b = -1/s^T Bs$

$$B^+ = B - \frac{Bss^T B}{s^T Bs} + \frac{yy^T}{y^T s}$$

called the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update. Woodbury formula (generalization of Sherman-Morrison):

$$(A + UDV)^{-1} = A^{-1} - A^{-1}U(D^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

Applied to our case, with

$$U = V^T = \begin{bmatrix} Bs & y \end{bmatrix}, \quad D = \begin{bmatrix} -\frac{1}{s^T Bs} & 0 \\ 0 & \frac{1}{y^T s} \end{bmatrix}$$

Substituting this into the Woodbury formula we get a rank-two update on C:

$$\begin{aligned}
C^+ = (B^+)^{-1} &= \left(B - \frac{Bss^TB}{s^TBs} + \frac{yy^T}{y^Ts}\right)^{-1} \\
&= B^{-1} - B^{-1}\begin{bmatrix} Bs & y \end{bmatrix}\left(\begin{bmatrix} -s^TBs & 0 \\ 0 & y^Ts \end{bmatrix} + \begin{bmatrix} s^TB \\ y^T \end{bmatrix} B^{-1} \begin{bmatrix} Bs & y \end{bmatrix}\right)^{-1} \begin{bmatrix} s^TB \\ y^T \end{bmatrix} B^{-1} \\
&= C - \begin{bmatrix} s & Cy \end{bmatrix}\left(\begin{bmatrix} -s^TBs & 0 \\ 0 & y^Ts \end{bmatrix} + \begin{bmatrix} s^TBs & y^Ts \\ y^Ts & y^TCy \end{bmatrix}\right)^{-1} \begin{bmatrix} s^T \\ y^TC \end{bmatrix} \\
&= C - \begin{bmatrix} s & Cy \end{bmatrix}\begin{bmatrix} 0 & y^Ts \\ y^Ts & y^Ts + y^TCy \end{bmatrix}^{-1} \begin{bmatrix} s^T \\ y^TC \end{bmatrix} \\
&= C - \begin{bmatrix} s & Cy \end{bmatrix}\frac{-1}{(y^Ts)^2}\begin{bmatrix} y^Ts + y^TCy & -y^Ts \\ -y^Ts & 0 \end{bmatrix} \begin{bmatrix} s^T \\ y^TC \end{bmatrix} \\
&= C + \frac{1}{(y^Ts)^2}\begin{bmatrix} (y^Ts + y^TCy)s + (-y^Ts)Cy & (-y^Ts)s \end{bmatrix} \begin{bmatrix} s^T \\ y^TC \end{bmatrix} \\
&= C + \frac{1}{(y^Ts)^2}\left((y^Ts + y^TCy)ss^T + (-y^Ts)Cys^T + (-y^Ts)sy^TC\right) \\
&= C + \frac{ss^T}{y^Ts} + \frac{y^TCyss^T}{(y^Ts)^2} - \frac{Cys^T}{y^Ts} - \frac{sy^TC}{y^Ts} \\
&= C + \frac{(ys^T)^TCys^T}{(y^Ts)^2} - \frac{Cys^T}{y^Ts} - \frac{sy^TC}{y^Ts} + \frac{ss^T}{y^Ts} \\
&= \left(I - \frac{sy^T}{y^Ts}\right) C \left(I - \frac{ys^T}{y^Ts}\right) + \frac{ss^T}{y^Ts}
\end{aligned}$$

The BFGS update is thus still quite cheap, $O(n^2)$ per update.

16.5 Positive definiteness of BFGS update

Importantly, unlike SR1, the BFGS update preserves positive definiteness under appropriate conditions. Assume $y^Ts = (\nabla f(x^+) - \nabla f(x))(x^+ - x) > 0$ (recall that e.g. strict convexity will imply this condition) and $C \succ 0$. Then consider the term

$$\begin{aligned}
x^TC^+x &= x^T\left(\left(I - \frac{sy^T}{y^Ts}\right) C \left(I - \frac{ys^T}{y^Ts}\right) + \frac{ss^T}{y^Ts}\right)x \\
&= (x - \frac{s^Tx}{y^Ts}y)^TC(x - \frac{s^Tx}{y^Ts}y) + \frac{(s^Tx)^2}{y^Ts}
\end{aligned}$$

Both terms are nonnegative: the second term is zero only if $s^Tx = 0$, in that case the first term is 0 only when $x = 0$.

16.6 Davidon-Fletcher-Powell update

Alternatively, compute a rank-two update directly on inverse C

$$C^+ = C + auu^T + bvv^T$$

Using secant equation $s = C^+y$, setting $u = s, v = Cy$, and solving for a, b gives

$$C^+y = Cy + auu^Ty + bvv^Ty \iff s = v + auu^Ty + bvv^Ty \iff u = v + a(u^Ty)u + b(v^Ty)v$$

Thus,

$$a = \frac{1}{u^Ty}, \quad b = -\frac{1}{v^Ty}$$

Hence,

$$C^+ = C + \frac{uu^T}{u^Ty} - \frac{vv^T}{v^Ty} \iff C^+ = C - \frac{CyyC^T}{y^TCy} + \frac{ss^T}{s^Ty}$$

called the Davidon-Fletcher-Powell (DFP) update which pre-dates BFGS, with same beneficial properties (preserves positive definiteness of Hessian, $O(n^2)$ computation), but not often used anymore.

16.7 Convergence analysis

Assume that f convex, twice differentiable, having $\mathrm{dom}(f) = \mathbf{R}^n$, and additionally

- $\nabla f(x)$ is Lipschitz with parameter M, i.e., $\nabla^2 f(x) \preceq MI$
- f is strongly convex with parameter m
- $\nabla^2 f(x)$ is Lipschitz with parameter L

(same conditions as in the analysis of Newton's method)

Theorem 13. Both BFGS and DFP, with backtracking line search, converge globally. Furthermore, for all $k \geq k_0$,

$$\|x^{(k)} - x^*\|_2 \leq c_k \|x^{(k-1)} - x^*\|_2$$

where $c_k \to 0$ as $k \to \infty$. Here k_0, c_k depend on L, m, M.

This is called local superlinear convergence.

Linear convergence:

$$\|x^{(k)} - x^*\|_2 \leq c \|x^{(k-1)} - x^*\|_2$$

where c is a constant.

Superlinear convergence:

$$\|x^{(k)} - x^*\|_2 \leq c_k \|x^{(k-1)} - x^*\|_2$$

where $c_k \to 0$ as $k \to \infty$.

Quadratic convergence:

$$\|x^{(k)} - x^*\|_2 \leq c \|x^{(k-1)} - x^*\|_2^2$$

where c is a constant.

16.8 Implicit-form quasi-Newton

For large problems, quasi-Newton updates can become too costly.

Basic idea: instead of explicitly computing and storing C, compute an implicit version of C by maintaining all pairs (y, s). Recall BFGS updates C via

$$C^+ = \left(I - \frac{sy^T}{y^T s}\right) C \left(I - \frac{ys^T}{y^T s}\right) + \frac{ss^T}{y^T s}$$

We observe that this leads to

$$\begin{aligned}
C^+ g &= \left(I - \frac{sy^T}{y^T s}\right) C \left(I - \frac{ys^T}{y^T s}\right) g + \frac{ss^T}{y^T s} g \\
&= \left(I - \frac{sy^T}{y^T s}\right) C \left(g - \frac{s^T g}{y^T s} y\right) + \frac{s^T g}{y^T s} s \\
&= \left(I - \frac{sy^T}{y^T s}\right) C \left(g - \alpha y\right) + \alpha s \quad (\text{let } \alpha = \frac{s^T g}{y^T s}) \\
&= \left(I - \frac{sy^T}{y^T s}\right) p + \alpha s \quad (\text{let } q = g - \alpha y \text{ and } p = Cq) \\
&= p - \frac{y^T p}{y^T s} s + \alpha s \\
&= p + (\alpha - \beta) s \quad (\text{let } \beta = \frac{y^T p}{y^T s})
\end{aligned}$$

We see that $C^+ g$ can be computed via two loops of length k (if C^+ is the approximation to the inverse Hessian after k iterations):

(1) Let $q = -\nabla f(x^{(k)})$

(2) For $i = k-1, \cdots, 0$:

  (a) Compute $\alpha_i = \frac{(s^{(i)})^T q}{(y^{(i)})^T s^{(i)})}$)

  (b) Update $q = q - \alpha y^{(i)}$

(3) Let $p = C^{(0)}q$

(4) For $i = 0, \cdots, k-1$:

 (a) Compute $\beta = \frac{(y^{(i)})^T p}{(y^{(i)})^T s^{(i)})}$

 (b) Update $p = p + (\alpha_i - \beta)s^{(i)}$

(5) Return

16.9 Limited memory BFGS

Limited memory BFGS (LBFGS) simply limits each of these loops to be length m:

(1) Let $q = -\nabla f(x^{(k)})$

(2) For $i = k-1, \cdots, k-m$:

 (a) Compute $\alpha_i = \frac{(s^{(i)})^T q}{(y^{(i)})^T s^{(i)})}$

 (b) Update $q = q - \alpha y^{(i)}$

(3) Let $p = \bar{C}^{(k-m)}q$

(4) For $i = k-m, \cdots, k-1$:

 (a) Compute $\beta = \frac{(y^{(i)})^T p}{(y^{(i)})^T s^{(i)})}$

 (b) Update $p = p + (\alpha_i - \beta)s^{(i)}$

(5) Return

In Step 3, $\bar{C}^{(k-m)}$ is our guess at $C^{(k-m)}$ (which is not stored). A popular choice is $\bar{C}^{(k-m)} = I$, more sophisticated choices exist.

17 Proximal Newton method

Recall motivation for prox gradient: iteratively minimize quadratic expansion in g, plus original h

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

Quadratic approximation uses $\frac{1}{t}I$ (spherical curvature), as in pure gradient descent when $h = 0$. A fundamental difference between gradient descent and Newton's method: latter also iteratively minimizes quadratic approximations, but these use the local Hessian of the function in question. So what happens if we replace $\frac{1}{t}I$ in the above with $\nabla^2 g(x)$?

This leads us to proximal Newton method: we repeat

$$\begin{aligned}
v^{(k)} &= \underset{v}{\operatorname{argmin}}\, \nabla g(x^{(k-1)})^T v + \frac{1}{2}v^T H^{(k-1)} v + h(x^{(k-1)} + v) \\
x^{(k)} &= x^{(k)} + t_k v^{(k)}
\end{aligned}$$

Here $H^{(k-1)} = \nabla^2 g(x^{(k-1)})$ is the Hessian of g at $x^{(k-1)}$, and t_k is the step size. Equivalent formulation:

$$\begin{aligned}
z^{(k)} &= \underset{z}{\operatorname{argmin}}\, \nabla g(x^{(k-1)})^T (z - x^{(k-1)}) + \frac{1}{2}(z - x^{(k-1)})^T H^{(k-1)} (z - x^{(k-1)}) + h(z) \\
x^{(k)} &= x^{(k)} + t_k(z^{(k)} - x^{(k-1)})
\end{aligned}$$

Proximal gradient	**Proximal Newton**
• Iteratively minimize $\|b-x\|_2^2+h(x)$	• Iteratively minimize $b^Tx+x^TAx^T+h(x)$
• Often closed-form prox	• Almost never closed-form prox
• Iterations are cheap	• Iterations are very very expensive
• Convergence of gradient descent	• Convergence of Newton's method

17.1 Scaled proximal map

Given $H \succ 0$, define

$$\text{prox}_H = \underset{x}{\text{argmin}} \frac{1}{2}\|x-z\|_H^2+h(z)$$

where $\|x\|_H^2 = x^T Hx$. This is called a scaled proximal map.

With $H = \frac{1}{t}I$, we get back usual (unscaled) definition. In general, the scaled prox shares retains many of the nice properties of usual prox (e.g., uniqueness, nonexpansiveness).

Now consider

$$\begin{aligned} z^+ &= \underset{z}{\text{argmin}}\, \nabla g(x)^T(z-x) + \frac{1}{2}(z-x)^T H^{(k-1)}(z-x) + h(z) \\ &= \underset{z}{\text{argmin}} \frac{1}{2}\|x - H^{-1}\nabla g(x) - z\|_H^2+h(z) \end{aligned}$$

Thus another equivalent form for proximal Newton update:

$$\begin{aligned} z^{(k)} &= \text{prox}_{H^{(k-1)}}\left(x^{(k-1)} - (H^{(k-1)})^{-1}\nabla g(x^{(k-1)})\right) \\ x^{(k)} &= x^{(k)} + t_k(z^{(k)} - x^{(k-1)}) \end{aligned}$$

Notes:

- When $h(z) = 0$, we get back the usual Newton update
- If we replaced $H^{(k-1)}$ by $\frac{1}{r_k}I$, and set $t_k = 1$, we get proximal gradient update, with step size r_k
- Difficulty of prox depends strongly on h. However, now it also depends on the structure of the Hessian of g
- E.g., having a diagonal or banded Hessian generally makes a big difference compared to a dense Hessian

17.2 Backtracking line search

As with Newton's method in fully smooth problems, pure step sizes $t_k = 1, k = 1, 2, 3, \cdots$ need not converge. We apply backtracking line search: fix $0 < \alpha \le \frac{1}{2}$, $< 0\beta < 1$, and

$$v = \text{prox}_H\left(x - H^{-1}\nabla g(x)\right) - x$$

be the proximal Newton direction at a given iteration. Start with $t = 1$, and while

$$f(x+tv) > f(x) + \alpha t \nabla g(x)^T v + \alpha\left(h(x+tv) - h(x)\right)$$

we shrink $t = \beta t$. (Here $f = g + h$).

Note: this scheme is actually of a different spirit than the one we studied for proximal gradient descent, as it avoids recomputing the prox at each inner backtracking iteration.

17.3 When would we use proximal Newton?

High-level picture, for problem: $\min_x \quad g(x) + h(x)$.

Proximal gradient:

So we use proximal Newton when we have an fast inner optimizer for scaled prox (quadratic plus h), expect a few iterations.

17.4 Convergence analysis

Following [Lee et al., 2014], assume that $f = g + h$, where g, h are convex and g is twice smooth. Assume further:

- $mI \preceq H \preceq LI$, and $\nabla^2 g(x)$ Lipschitz with parameter M
- $\text{prox}_H(\cdot)$ is exactly evaluable

Theorem 14. Proximal Newton method with backtracking line search converges globally. Furthermore, for all $k \geq k_0$,

$$\|x^{(k)} - x^*\|_2 \leq \frac{M}{2m}\|x^{(k-1)} - x^*\|_2^2$$

Recall that this is called local quadratic convergence. After $k \geq k_0$, to get within $f(x^{(k)}) - f^* \leq \epsilon$, we need $O(\log\log(1/\epsilon))$ iterations. Note: each iteration uses scaled prox evaluation!

18 Numerical Linear Algebra Primer

18.1 Complexity of basic operations

Flop (floating point operation):

- One addition, subtraction, multiplication, division of floating point numbers
- Serves as a basic unit of computation
- We are interested in rough, not exact flop counts

Vector-vector operations: given $a, b\mathbf{R}^n$:

- Addition, $a + b$: costs n flops
- Scalar multiplication, $c \cdot a$: costs n flops
- Inner product, $a^T b$: costs $2n$ flops

Flops do not tell the whole story: setting every element of a to 1 costs 0 flops.
Matrix-vector product: given $A \in \mathbf{R}^{m\times n}$, $b \in \mathbf{R}^n$, consider Ab:

- In general: costs $2mn$ flops
- For s-sparse A: costs $2s$ flops
- For k-banded $A \in \mathbf{R}^{n\times n}$: costs $2nk$ flops
- For $A = \sum_{i=1}^r u_i v_i^T \in \mathbf{R}^{m\times n}$: costs $2r(m+n)$ flops
- For $A \in \mathbf{R}^{n\times n}$ a permutation matrix: costs 0 flops

Matrix-matrix product: for $A \in \mathbf{R}^{m\times n}$, $B \in \mathbf{R}^{n\times p}$, consider AB:

- In general: costs $2mnp$ flops
- For s-sparse A: costs $2s$ flops (less if B is also sparse)

Matrix-matrix-vector product: for $A \in \mathbf{R}^{m\times n}$, $B \in \mathbf{R}^{n\times p}$, $c \in \mathbf{R}^p$, consider ABc:

- Costs $2np + 2mn$ flops if done properly (or $2mnp + 2mp$ if done improperly!)

18.2 Solving linear systems

For nonsingular $A \in \mathbf{R}^{n\times n}$, consider solving linear system $Ax = b$:

- In general: costs about n^3 flops
- For diagonal A: costs n flops

$$x = (b_1/a_1, \cdots, b_n/a_n)$$

- For lower triangular $A (A_{ij} = 0, j > i)$: costs about n^2 flops

$$\begin{aligned} x_1 &= b_1/A_{11} \\ x_2 &= (b_2 - A_{21}x_1)/A_{22} \\ &\vdots \\ x_n &= (b_n - A_{n,n-1}x_{n-1} - A_{n,n-2}x_{n-2}, \cdots, -A_{21}x_1)/A_{nn} \end{aligned}$$

  This is called forward substitution

- For upper triangular A: costs about n^2, by back substitution
- For s-sparse A, often costs $\ll n^3$ flops, but exact (worse-case) flop counts are not known for abitrary sparsity structures
- For k-banded A: costs about nk^2 flops
- For orthogonal A: we have $A^{-1} = A^T$, so $x = A^T b$ costs $2n^2$ flops
- For permutation A: again $A^{-1} = A^T$, so $x = A^T b$ costs 0 flops.

18.3 Matrix factorizations

As you've probably learned, we can solve $Ax = b$ by, e.g., Gaussian elimination. More typically, it is useful to instead factorize A:

$$A = A_1 A_2 \dots A_k$$

and then compute $x = A_k^{-1} \dots A_1^{-1} A_2^{-1} b$. Usually $k = 2$ or 3, and:

- Computing the factorization is expensive, about n^3 flops
- Applying $A_1^{-1}, \dots, A_k^{-1}$ is cheaper, about n^2 flops
- This is because $A_1 A_2 \dots A_k$ are structured: either orthogonal, triangular, diagonal, or permutation matrices

18.4 QR decomposition

Any $A \in \mathbf{R}^{m \times n}$, with $m \geq n$, has a QR decomposition:

$$A = QR$$

with $Q \in \mathbf{R}^{m \times n}$ orthogonal (i.e., $Q^T Q = I$), and $R \in \mathbf{R}^{n \times n}$ upper triangular. We can compute this in $2mn^2 - n^3/3$ flops.

Property: number of nonzero diagonal elements of R is the rank of A, corresponding columns of Q span col(A).

Assuming A is nonsingular and square, we can now solve $Ax = b$:

- Compute $y = Q^T b$, in $2n^2$ flops
- Solve $Rx = y$, in n^2 flops (back substitution)

So solving costs $3n^2$ flops.

18.5 Least squares problems and QR

Key identity:

$$\|x\|_2^2 = \|P^T x\|_2^2 = \|Q^T x\|_2^2 + \|\tilde{Q}^T x\|_2^2$$

where $P = [Q\ \tilde{Q}]$ is orthogonal. Applied to $x = y - X\beta$:

$$\|y - X\beta\|_2^2 = \|P^T(y - QR\beta)\|_2^2 = \|\begin{pmatrix} Q^T \\ \tilde{Q}^T \end{pmatrix} (y - QR\beta)\|_2^2 = \|Q^T y - R\beta\|_2^2 + \|\tilde{Q}^T y\|_2^2$$

The second term does not depend on β. So for least squares solution:

- Compute $X = QR$, in $2np^2 - p^3/3$ flops
- Compute $Q^T y$, in $2pn$ flops
- Solve $R\beta = Q^T y$, in p^2 flops (back substitution)

Thus in total, about $2np^2 - p^3/3$ flops (or $2np^2$ flops if $n \gg p$).

18.6 Cholesky decomposition

More specialized, any $A \in \mathbf{S}^n_{++}$, has a Cholesky decomposition:

$$A = LL^T$$

with $L \in \mathbf{R}^{n \times n}$ lower triangular. We can compute this in $n^3/3$ flops. (We could compute Cholesky from QR, but this would be inefficient)

Proof. Since A is positive definite, there exist a nonsingular matrix P such that $A = P^T P$. Since A is symmetric, A is square. Furthermore, there exists a QR factorization for P, i.e. $P = QR$. Hence, $A = (QR)^T QR = R^T Q^T QR = R^T R = LL^T$ with $L = R^T$ which is lower triangular. □

From Cholesky factors, we can solve $Ax = b$:

- Compute $y = L^{-1}b$, in n^2 flops (forward substitution)
- Solve $x = (L^T)^{-1}y$, in n^2 flops (back substitution)

So solving costs $2n^2$ flops.
An important extension for k-banded A: computing Cholesky takes $nk^2/4$ flops, and solving takes $2nk$ flops.

18.7 Least squares problems and Cholesky

Given $y \in \mathbf{R}^n$, $X \in \mathbf{R}^{n \times p}$, consider the least squares problem:

$$\min_{\beta \in \mathbf{R}^p} \|y - X\beta\|_2^2$$

Assuming X has full column rank, solution is $\hat{\beta} = (X^T X)^{-1} X^T y$. How expensive?

- Compute $X^T y$, in $2pn$ flops
- Compute $X^T X$, in $p^2 n$ flops
- Compute Cholesky of $X^T X$, in $p^3/3$ flops (back substitution)
- Solve $(X^T X)\beta = X^T y$, in $2p^2$ flops

Thus in total, about $2np^2 + p^3/3$ flops (or $2np^2$ flops if $n \gg p$).

18.8 Linear systems and stability

Consider first the linear system $Ax = b$, for nonsingular $A \in \mathbf{R}^{n \times n}$. The singular value decomposition (SVD) of A:

$$A = U \Sigma V^T$$

where $U, V \in \mathbf{R}^{n \times n}$ are orthogonal, and $\Sigma \in \mathbf{R}^{n \times n}$ is diagonal with elements $\sigma_1 \geq \ldots \sigma_n > 0$. Even if A is full rank, it could be "near" a singular matrix B, i.e.,

$$\text{dist}(A, \mathcal{R}_k) = \min_{\text{rank}(B) = k} \|A - B\|_{\text{op}}$$

could be small, for some $k < n$. By the Eckhart-Young Theorem, we have $\text{dist}(A, \mathcal{R}_k) = \sigma_{k+1}$. If this is small, then solving $x = A^{-1}b$ could pose problems.

18.9 Sensitivity analysis

Precise sensitivity analysis: fix some $F \in \mathbf{R}^{n \times n}$, $f \in \mathbf{R}^n$. Solve the "perturbed linear system":

$$(A + \epsilon F)x(\epsilon) = (b + \epsilon f)$$

Theorem 15. The solution to the perturbed system, abbreviating $x = x(0)$, satisfies

$$\frac{\|x(\epsilon) - x\|_2}{\|x\|_2} \leq \kappa(A)(\rho_A + \rho_b) + O(\epsilon^2)$$

where $\kappa(A) = \sigma_1/\sigma_n$ is the condition number of A, and ρ_A, ρ_b are the relative errors $\rho_A = |\epsilon| \|F\|_{\text{op}} / \|A\|_{\text{op}}, \rho_b = |\epsilon| \|f\|_2 / \|b\|_2$.

Proof. By implicit differentiation,

$$\begin{aligned}
&Fx(\epsilon) + (A + \epsilon F)x'(\epsilon) = f \\
\Rightarrow &(A + \epsilon F)x'(\epsilon) = f - Fx(\epsilon) \\
\Rightarrow &x'(\epsilon) = (A + \epsilon F)^{-1}(f - Fx(\epsilon))
\end{aligned}$$

setting $\epsilon = 0$, we get

$$x'(0) = A^{-1}(f - Fx)$$

Using a Taylor expansion around 0,

$$x(\epsilon) = x + \epsilon A^{-1}(f - Fx) + O(\epsilon^2)$$

Rearranging gives

$$\begin{aligned}
&x(\epsilon) - x = \epsilon A^{-1}(f - Fx) + O(\epsilon^2) \\
\Rightarrow &\|x(\epsilon) - x\|_2 = \|\epsilon A^{-1}(f - Fx) + O(\epsilon^2)\|_2 \\
&\leq |\epsilon|\|A^{-1}\|_{\text{op}}(\|f\|_2 + \|F\|_{\text{op}}\|x\|_2) + O(\epsilon^2) \\
\Rightarrow &\frac{\|x(\epsilon) - x\|_2}{\|x\|_2} \leq |\epsilon|\|A^{-1}\|_{\text{op}}(\frac{\|f\|_2}{\|x\|_2} + \|F\|_{\text{op}}) + O(\epsilon^2) \\
&\leq \kappa(A)(\frac{|\epsilon|\|f\|_2}{\|A\|_{\text{op}}\|x\|_2} + |\epsilon|\frac{\|F\|_{\text{op}}}{\|A\|_{\text{op}}}) + O(\epsilon^2) \\
&\leq \kappa(A)(\frac{|\epsilon|\|f\|_2}{\|b\|_2} + \frac{|\epsilon|\|F\|_{\text{op}}}{\|A\|_{\text{op}}}) + O(\epsilon^2) \quad (\|b\|_2 = \|Ax\|_2 \leq \|A\|_{\text{op}}\|x\|_2) \\
&\leq \kappa(A)(\rho_A + \rho_b) + O(\epsilon^2)
\end{aligned}$$

□

18.10 Cholesky versus QR for least squares

Linear systems: worse conditioning means great sensitivity. What about for least squares problems?

$$\min_{\beta \in \mathbf{R}^p} \|y - X\beta\|_2^2$$

- Recall Cholesky solves $(X^TX)\beta = X^Ty$. Hence we know that sensitivity scales with $\kappa(X^TX) = \kappa(X)^2$
- Meanwhile, QR operates on X, never forms X^TX, and can show that sensitivity scales with $\kappa(X) + \rho_{\text{LS}} \cdot \kappa(X)^2$, where $\rho_{\text{LS}} = \|y - X\hat{\beta}\|_2^2$

Summary: Cholesky is cheaper (and uses less memory), but QR is more stable when ρ_{LS} is small and $\kappa(X)$ is large.

18.11 Jacobi and Gauss-Seidl

So far we've been talking about direct methods for linear systems. These return the exact solution (in perfect computing environment). Indirect methods (iterative methods) produce $x(k), k = 1, 2, 3, \ldots$ converging to a solution x. Most often used for very large, sparse systems.

Given $A \in \mathbf{S}^n_{++}$, two basic iterative approaches for solving $Ax = b$:

- Jacobi iterations: initialize $x^{(0)} \in \mathbf{R}^n$, repeat

  $$x_i^{(k)} = \left(b_i - \Sigma_{j \neq i} A_{ij} x_j^{(k-1)}\right) / A_{ii}, i = 1, \cdots, n$$

  for $k = 1, 2, 3, \ldots$

- Gauss-Seidl iterations: initialize $x^{(0)} \in \mathbf{R}^n$, repeat

  $$x_i^{(k)} = \left(b_i - \Sigma_{j<i} A_{ij} x_j^{(k)} - \Sigma_{j>i} A_{ij} x_j^{(k-1)}\right) / A_{ii}, i = 1, \cdots, n$$

  for $k = 1, 2, 3, \ldots$ Gauss-Seidl uses most recent info possible

- Gauss-Seidl iterations always converge, but Jacobi iterations do not.

19 Coordinate Descent

19.1 Coordinatewise minima

Q: Given convex, differentiable $f : \mathbf{R}^n \to \mathbf{R}$, if we are at a point x such that $f(x)$ is minimized along each coordinate axis, then have we found a global minimizer? In other words, does $f(x + \delta e_i) \geq f(x)$ for all δ, e_i imply $f(x) = \min_z f(z)$? (Here $e_i = (0, \ldots, 1, \ldots, 0) \in \mathbf{R}^n$, the ith standard basis vector)

A: Yes.

Proof. Since $f(x+\delta e_i) \geq f(x)$ holds for all δ, e_i, we have $\frac{\partial f}{\partial x_i}(x) = 0$ for all i. Thus, $\nabla f(x) = \left(\frac{\partial f}{\partial x_1}(x), \cdots, \frac{\partial f}{\partial x_n}(x)\right) = 0$. Since f is convex and differentiable, x is a global minimizer. $\square$

Q: Same question, but now for f convex, and not differentiable?

A: No! Refer to Fig. 9 for a counterexample. As we can see in Fig. 9, the criterion will increase when we move the intersection of two dotted red lines along either axis. However, the global minimizer is the blue point.

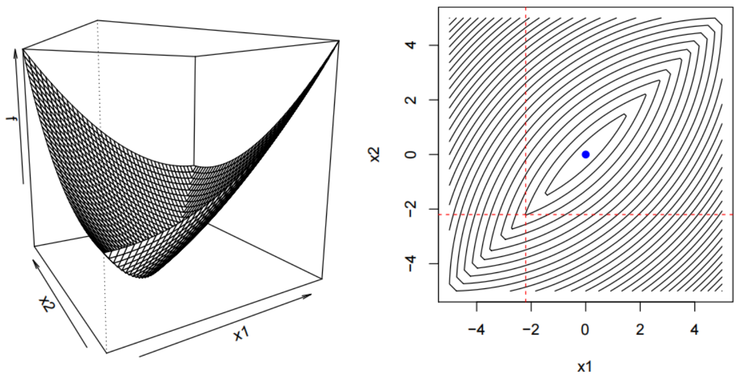


Figure 9: A counterexample of global optimality for the coordinate descent method when f is convex but not differentiable.

Q: Same question again, but now $f(x) = g(x) + \sum_{i=1}^n h_i(x_i)$, with g convex, differentiable and each h_i convex? (Here the nonsmooth part is called separable).

A: Yes!

Proof. By the subgradient optimality, we have

$$0 \in \nabla g_i(x) + \partial h_i(x_i) \Rightarrow -\nabla g_i(x) \in \partial h_i(x_i)$$

Thus, by the convexity of h, we have

$$\begin{aligned} h(y_i) &\geq h(x_i) + \partial h_i(x)(y_i - x_i) = h(x_i) - \nabla g_i(x)(y_i - x_i) \\ \Rightarrow& \nabla_i g(x)(y_i - x_i) + h(y_i) - h(x_i) \geq 0 \end{aligned}$$

By the convexity of f, we have

$$\begin{aligned} f(y) - f(x) &= g(y) - g(x) + \sum_{i=1}^n [h(y_i) - h(x_i)] \\ &\geq \nabla g(x)^T (y - x) + \sum_{i=1}^n [h(y_i) - h(x_i)] \\ &= \sum_{i=1}^n \underbrace{[\nabla_i g(x)(y_i - x_i) + h(y_i) - h(x_i)]}_{\geq 0} \geq 0 \end{aligned}$$

$\square$

19.2 Coordinate descent method

This suggests that for the problem

$$\min_x f(x)$$

where $f(x) = g(x) + \sum_i^n h_i(x_i)$, with g convex and differentiable and each h_i convex, we can use coordinate descent: let $x^{(0)} \in \mathbf{R}^n$, and repeat

$$x_i^{(k)} = \operatorname*{argmin}_{x_i} f(x_1^{(k)}, \dots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k-1)}, \dots, x_n^{(k-1)},), i = 1, \dots, n$$

for $k = 1, 2, 3, \dots$

Important note: we always use the most recent information possible.

Tseng (2001) proves that for such f (provided f is continuous on compact set $x : f(x) \leq f(x^{(0)})$ and f attains its minimum), any limit point of $x^{(k)}, k = 1, 2, 3, \dots$ is a minimizer of f[16].

Notes:

- Order of cycle through coordinates is arbitrary, can use any permutation of $1, 2, \dots, n$
- Can everywhere replace individual coordinates with blocks of coordinates
- "One-at-a-time" update scheme is critical, and "all-at-once" scheme does not necessarily converge
- The analogy for solving linear systems: Gauss-Seidel versus Jacobi method

19.3 Examples

19.3.1 Linear regression

Given $y \in \mathbf{R}^n$, and $X \in \mathbf{R}^{n\times p}$ with columns $X_1, \dots, X_p$, consider the linear regression problem:

$$\min_\beta \frac{1}{2}\|y - X\beta\|_2^2$$

Minimizing over β_i, with all $\beta_j, j \neq i$ fixed:

$$0 = \nabla_i f(\beta) = X_i^T(X\beta - y) = X_i^T(X_i\beta_i + X_{-i}\beta_{-i} - y)$$

i.e., we take

$$\beta_i = \frac{X_i^T(y - X_{-i}\beta_{-i})}{X_i^T X_i}$$

Coordinate descent repeats this update for $i = 1, 2, \dots, p, 1, 2, \dots$. Note that this is exactly Gauss-Seidl for the system $X^T X = X^T y$.

Is it fair to compare 1 cycle of coordinate descent to 1 iteration of gradient descent? Yes,

- Gradient descent: $\beta \leftarrow \beta + tX^T(y - X\beta))$, costs $O(np)$ flops
- Coordinate descent, one coordinate update:

  $$\beta_i \leftarrow \frac{X_i^T(y - X_{-i}\beta_{-i})}{X_i^T X_i} = \frac{X_i^T(y - X_{-i}\beta_{-i} - X_i\beta_i + X_i\beta_i)}{X_i^T X_i} = \frac{X_i^T r}{X_i^T X_i} + \beta_i$$

  where $r = y - X\beta$
- Each coordinate costs $O(n)$ flops: $O(n)$ to update r by using $r = r_{old} + X_i^T\beta_i^{k-1} - X_i^T\beta_i^k$, $O(n)$ to compute $X_i^T r$
- One cycle of coordinate descent costs $O(np)$ operations, same as gradient descent

[16]Using real analysis, we know that $x^{(k)}$ has subsequence converging to $x^\star$ (Bolzano-Weierstrass), and $f(x^{(k)})$ converges to $f^\star$ (monotone convergence)

19.3.2 Lasso regression

Consider the lasso problem:

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

Note that the nonsmooth part is separable: $\|\beta\|_1= \sum_{i=1}^p|\beta_i|$.
Minimizing over β_i, with all $\beta_j, j \neq i$ fixed:

$$0 = X_i^T X_i\beta_i + X_i^T(X_{-i}\beta_{-i} - y) + \lambda s_i \Rightarrow 0 = \beta_i + \frac{X_i^T(X_{-i}\beta_{-i} - y)}{X_i^T X_i} + \frac{\lambda}{X_i^T X_i}s_i$$

where $s_i \in \partial|\beta_i|$. If $\beta_i > 0$, $\beta_i = \frac{X_i^T(y-X_{-i}\beta_{-i})}{X_i^T X_i} - \frac{\lambda}{X_i^T X_i}$. If $\beta_i < 0$, $\beta_i = \frac{X_i^T(y-X_{-i}\beta_{-i})}{X_i^T X_i} + \frac{\lambda}{X_i^T X_i}$. When $\beta_i = 0$, $|\frac{X_i^T(y-X_{-i}\beta_{-i})}{X_i^T X_i}|< \frac{\lambda}{X_i^T X_i}$. Thus, the solution is simply given by soft-thresholding

$$\beta_i = S_{\lambda/\|X_i\|_2^2}(\frac{X_i^T(y - X_{-i}\beta_{-i})}{X_i^T X_i})$$

Repeat this for $i = 1, 2, \ldots, p, 1, 2, \ldots$.

19.4 Coordinate gradient descent

For a smooth function f, the iterations

$$x_i^{(k)} = x_i^{(k-1)} - t_{ki} \cdot \nabla_i f(x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)}), i = 1, \ldots, n$$

for $k = 1, 2, 3, \ldots$ are called coordinate gradient descent, and when $f = g + h$, with g smooth and $h = \sum_{i=1}^n h_i$, the iterations

$$x_i^{(k)} = prox_{h_i,t_{ki}}\left(x_i^{(k-1)} - t_{ki} \cdot \nabla_i g(x_1^{(k)}, \ldots, x_{i-1}^{(k)}, x_i, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)})\right), i = 1, \ldots, n$$

for $k = 1, 2, 3, \ldots$ are called coordinate proximal gradient descent.

When g is quadratic, (proximal) coordinate gradient descent is the same as coordinate descent under proper step sizes.

19.5 Convergence analyses

Theory for coordinate descent moves quickly. Each combination of the following cases has (probably) been analyzed:

- Coordinate descent or (proximal) coordinate gradient descent?
- Cyclic rule, permuted cyclic, or greedy rule, randomized rule?

Roughly speaking, results are similar to those for proximal gradient descent: under standard conditions, get standard rates

But constants differ and this matters! Much recent work is focused on improving them

Some references are Beck and Tetruashvili (2013), Wright (2015), Sun and Hong (2015), Li et al. (2016).

20 Dual ascent

20.1 Conjugate functions

Given $f : \mathbb{R}^n \to \mathbb{R}$, the function

$$f^\star(y) = \max_x \; y^T x - f(x)x$$

is called its conjugate.

- Conjugates appear frequently in dual programs, since

$$-f^\star(y) = \min_x \; f(x) - y^T x$$

- If f is closed and convex, then $f^{\star\star} = f$. Also,

$$x \in \partial f^\star(y) \Longleftrightarrow y \in \partial f(x) \Longleftrightarrow x \in \operatorname*{argmin}_z \; f(z) - y^T z$$

Proof. We only prove the "$\Longrightarrow$" of the second "$\Longleftrightarrow$". Since $y \in \partial f(x)$, we get $0 \in f(x) - y$. By the subgradient optimality, this means that x is a minimizer of the problem $\min_z \; f(z) - y^T z$, i.e., $x \in \operatorname{argmin}_z \; f(z) - y^T z$. □

- If f is strictly convex, then $x = \nabla f^\star(y) \in \operatorname{argmin}_z \; f(z) - y^T z$.

  Proof. If f is strictly convex, then the minimizer of the problem $\min_z \; f(z) - y^T z$ is unique. Hence, $y = \nabla f(x)$. By the second item, we have $x = \nabla f^\star(y) = \operatorname{argmin}_z \; f(z) - y^T z$. □

20.2 Dual-based (sub)gradient methods

Even if we can't derive dual (conjugate) in closed form, we can still use dual-based gradient or subgradient methods.

Consider the problem

$$\min \; f(x) \text{ subject to } Ax = b$$

Its dual problem is

$$\max \; -f^\star(-A^T u) - b^T A u$$

where $f^\star$ is conjugate of f. Defining $g(u) = -f^\star(-A^T u) - b^T u$. Note that

$$\partial g(u) = A \partial f^\star(-A^T u) - b = Ax - b$$

. where $x \in \partial f^\star(-A^T u)$.

Therefore, using the second property of what we know about conjugates in the last subsection,

$$\partial g(u) = Ax - b \text{ where } x \in \operatorname*{argmin}_z \; f(z) + u^T A z$$

The dual subgradient method (for maximizing the dual objective) starts with an initial dual guess $u^{(0)}$, and repeats for $k = 1, 2, 3, \ldots$

$$x^{(k)} \in \operatorname*{argmin}_x \; f(x) + (u^{(k-1)})^T A x$$

$$u^{(k)} = u^{(k-1)} + t_k \cdot (Ax^{(k)} - b) \text{ an ascent update}$$

Step sizes $t_k, k = 1, 2, 3, \ldots$, are chosen in standard ways.

Recall that if f is strictly convex, then $f^\star$ is differentiable, and so this becomes dual gradient ascent, which repeats for $k = 1, 2, 3, \ldots$.

$$x^{(k)} = \operatorname*{argmin}_x \; f(x) + (u^{(k-1)})^T A x$$

$$u^{(k)} = u^{(k-1)} + t_k \cdot (Ax^{(k)} - b)$$

(Difference is that each $x^{(k)}$ is unique, here.) Again, step sizes $t_k, k = 1, 2, 3, \ldots$ are chosen in standard ways.

Lastly, proximal gradients and acceleration can be applied as they would usually.

20.3 Smoothness of f and $f^\star$

Assume that f is a closed and convex function. Then f is strongly convex with parameter $m \Longleftrightarrow \nabla f^\star$ Lipschitz with parameter $1/m$.

Proof. We first show "$\Longrightarrow$". If g is strongly convex with parameter m, we have

$$g(y) \geq g(x) + \nabla g(x)^T (y - x) + \frac{m}{2} \|y - x\|_2^2, \text{ for all } y$$

If x is a minimizer, then $\nabla g(x) = 0$. Thus,

$$g(y) \geq g(x) + \frac{m}{2} \|y - x\|_2^2, \text{ for all } y$$

We define $g(y) = f(y) - u^T y$. Hence, defining $x_u = \nabla f^\star(u)$ and $x_v = \nabla f^\star(v)$, we have

$$f(x_v) - u^T x_v \geq f(x_u) - u^T x_u + \frac{m}{2} \|x_v - x_u\|_2^2$$

$$f(x_u) - v^T x_u \geq f(x_v) - v^T x_v + \frac{m}{2} \|x_v - x_u\|_2^2$$

Adding these together,

$$\begin{aligned}
u^T x_u - u^T x_v - v^T x_u + v^T x_v &\geq m\|x_v - x_u\|_2^2 \\
u^T(x_u - x_v) - v^T(x_u - x_v) &\geq m\|x_v - x_u\|_2^2 \\
(u-v)^T(x_u - x_v) &\geq m\|x_v - x_u\|_2^2 \\
\|x_v - x_u\|_2\|v-u\|_2 &\geq m\|x_v - x_u\|_2^2 \text{ (Cauchy-Schwartz inequality)} \\
\|v-u\|_2 &\geq m\|x_v - x_u\|_2 \\
\|x_v - x_u\|_2 &\leq \frac{1}{m}\|v-u\|_2
\end{aligned}$$

Now we show the "$\Longleftarrow$". For simplicity, call $g = f^\star$ and $L = 1/m$. As ∇g is Lipschitz with constant L, so is $g_x(z) = g(z) - \nabla g(x)^T z$, hence

$$g_x(z) \leq g_x(y) + \nabla_y g_x(y)^T(z-y) + \frac{L}{2}\|z-y\|_2^2$$

Minimizing each side over z,

$$\nabla g_x(z) = \nabla g(z) - \nabla g(x) = 0 \Longrightarrow z = x$$

$$\nabla g(y) - \nabla g(x) + L(z-y) = 0 \Longrightarrow z = y - \frac{1}{L}(\nabla g(y) - \nabla g(x))$$

Thus,

$$g(x) - \nabla g(x)^T x \leq g(y) - \nabla g(x)^T y - \frac{1}{L}(\nabla g(y) - \nabla g(x))^T(\nabla g(y) - \nabla g(x)) + \frac{L}{2}\cdot\frac{1}{L^2}\|\nabla g(y) - \nabla g(x)\|^2$$

$$\frac{1}{2L}\|\nabla g(y) - \nabla g(x)\|^2 \leq g(y) - g(x) + \nabla g(x)^T(x-y)$$

Exchanging roles of x, y,

$$\frac{1}{2L}\|\nabla g(x) - \nabla g(y)\|^2 \leq g(x) - g(y) + \nabla g(y)^T(y-x)$$

Adding together, gives

$$\frac{1}{L}\|\nabla g(y) - \nabla g(x)\|^2 \leq \nabla(g(y) - g(x))^T(y-x)$$

Let $u = \nabla g(y), v = \nabla g(x)$, then $x = \nabla f(u), y = \nabla f(v)$. Hence, and the above reads $(x-y)^T(u-v) \geq \|u-v\|^2/L$. Then, by the mean-value theorem,

$$(\nabla f(u) - \nabla f(v))^T(u-v) = (u-v)^T\nabla^2 f(w)(u-v) \geq \|u-v\|^2/L = m\|u-v\|^2$$

which implies

$$\nabla^2 f(w) \succeq mI$$

This means f is strongly convex with m. □

20.4 Convergence guarantees

The following results hold from combining the last fact with what we already know about gradient descent:

- If f is strongly convex with parameter m, then dual gradient ascent with constant step sizes $t_k = m$ converges at sublinear rate $O(1/\epsilon)$
- If f is strongly convex with parameter m and ∇f is Lipschitz with parameter L, then dual gradient ascent with step sizes $t_k = 2/(1/m + 1/L)$ converges at linear rate $O(\log(1/\epsilon))$

Note that these results describe convergence of the dual objective to its optimal value

20.5 Dual decomposition

Consider

$$\min_x \sum_{i=1}^{B} f_i(x_i) \text{ subject to } Ax = b$$

Here $x = (x_1, \dots, x_B) \in R^n$ divides into B blocks of variables, with each $x_i \in \mathbb{R}^{n_i}$. We can also partition A accordingly

$$A = [A_1 \dots A_B] \text{ where } A_i \in \mathbb{R}^{m \times n_i}$$

Simple but powerful observation, in calculation of (sub)gradient, is that the minimization decomposes into B separate problems:

$$\begin{aligned} x^+ &\in \underset{x}{\text{argmin}} \sum_{i=1}^{B} f_i(x_i) + u^T Ax \\ \Longrightarrow x_i^+ &\in \underset{x_i}{\text{argmin}}\ f_i(x_i) + u^T A_i x_i \end{aligned}$$

Dual decomposition algorithm: repeat for $k = 1, 2, 3, \dots$

$$x_i^{(k)} = \underset{x_i}{\text{argmin}}\ f(x_i) + (u^{(k-1)})^T A x_i,\ i = 1, 2, \dots, B$$

$$u^{(k)} = u^{(k-1)} + t_k \cdot (\sum_{i=1}^{B} A_i x_i^{(k)} - b)$$

As shown in Fig. 10, we can think of these steps as:

- Broadcast: send u to each of the B processors, each optimizes in parallel to find x_i
- Gather: collect $A_i x_i$ from each processor, update the global dual variable u

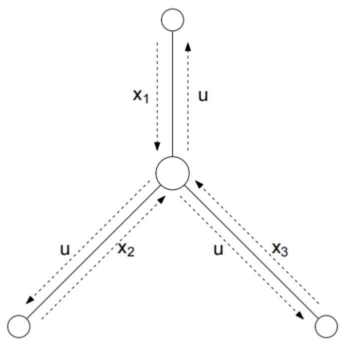


Figure 10: Dual decomposition update.

20.6 Dual decomposition with inequality constraints

Consider

$$\min_x \sum_{i=1}^{B} f_i(x_i) \text{ subject to } \sum_{i=1}^{B} A_i x_i \le b$$

Dual decomposition, i.e., projected subgradient method:

$$x_i^{(k)} \in \underset{x_i}{\text{argmin}}\ f(x_i) + (u^{(k-1)})^T A x_i,\ i = 1, 2, \dots, B$$

$$u^{(k)} = \left(u^{(k-1)} + t_k \cdot (\sum_{i=1}^{B} A_i x_i^{(k)} - b) \right)_+$$

where u^+ denotes the positive part of u, i.e., $(u_+)_i = \max\{0, u_i\}, i = 1, \dots, m$.

Price coordination interpretation (Vandenberghe):

- Have B units in a system, each unit chooses its own decision variable x_i (how to allocate its goods)
- Constraints are limits on shared resources (rows of A), each component of dual variable u_j is price of resource j
- Dual update:

$$u_j^+ = (u_j - ts_j)_+, \ j = 1, \dots, m$$

where $s = b - \sum_{i=1}^{B} A_i x_i$ are slacks

- Increase price u_j if resource j is over-utilized, $s_j < 0$
- Decrease price u_j if resource j is under-utilized, $s_j > 0$
- Never let prices get negative

21 Alternating direction method of multipliers(ADMM)

21.1 Augmented Lagrangian method(also known as: method of multipliers)

Disadvantage of dual ascent: require strong conditions to ensure convergence. Improved by augmented Lagrangian method, also called method of multipliers. We transform the primal problem:

$$\begin{aligned} &\min_x \quad f(x) + \frac{\rho}{2}\|Ax - b\|_2^2 \\ &\text{subject to } Ax = b \end{aligned}$$

where $\rho > 0$ is a parameter. Clearly equivalent to original problem, and objective is strongly convex when A has full column rank. Use dual gradient ascent:

$$x^{(k)} = \underset{x}{\operatorname{argmin}} \ f(x) + (u^{(k-1)})^T Ax + \frac{\rho}{2}\|Ax - b\|_2^2$$

$$u^{(k)} = u^{(k-1)} + \rho \cdot (Ax^{(k)} - b)$$

Notice step size choice $t_k = \rho$ in dual algorithm. Why? Since $x^{(k)}$ minimizes $f(x) + (u^{(k-1)})^T Ax + \frac{\rho}{2}\|Ax - b\|_2^2$ over x, we have

$$\begin{aligned} 0 &\in \partial f(x^{(k)}) + A^T(u^{(k-1)} + \rho(Ax^{(k)} - b)) \\ &\in \partial f(x^{(k)}) + A^T u^{(k)} \end{aligned}$$

This is the stationarity condition for original primal problem; **under mild conditions as $k \to \infty$ (primal iterates become feasible, i.e., $Ax \to b$)**, so KKT conditions are satisfied in the limit and $x^{(k)}, u^{(k)}$ converge to solutions.

- Advantage: much better convergence properties
- Disadvantage: lose decomposability! (Separability is ruined by augmented Lagrangian ...)

21.2 Alternating direction method of multipliers

Alternating direction method of multipliers or ADMM: try for best of both worlds. Consider the problem

$$\min_{x,z} \ f(x) + g(z), \ \text{subject to} \ Ax + Bz = c$$

As before, we augment the objective

$$\begin{aligned} &\min_{x,z} \ f(x) + g(z) + \frac{\rho}{2}\|Ax + Bz - c\|^2 \\ &\text{subject to} \ Ax + Bz = c \end{aligned}$$

for a parameter $\rho > 0$. We define augmented Lagrangian

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

ADMM repeats the steps, for $k = 1, 2, 3, \dots$

$$\begin{aligned} x^{(k)} &= \underset{x}{\operatorname{argmin}}\ L_\rho(x, z^{(k-1)}, u^{(k-1)}) \\ z^{(k)} &= \underset{z}{\operatorname{argmin}}\ L_\rho(x^{(k)}, z, u^{(k-1)}) \\ u^{(k)} &= u^{(k-1)} + \rho(Ax^{(k)} + Bz^{(k)} - c) \end{aligned}$$

where the update for u follows from the stationarity condition as follows

$$\begin{aligned} 0 &\in \partial g(z^{(k)}) + B^T(u^{(k-1)} + \rho(Ax^{(k)} + Bz^{(k)} - c)) \\ &\in \partial g(z^{(k)}) + B^T u^{(k)} \end{aligned}$$

Note that the usual method of multipliers would have replaced the first two steps by a joint minimization

$$(x^{(k)}, z^{(k)}) = \underset{x,z}{\operatorname{argmin}}\ L_\rho(x, z, u^{(k-1)})$$

21.3 Convergence guarantees

Under modest assumptions on f, g (these do not require A, B to be full rank), the ADMM iterates satisfy, for any $\rho > 0$:

- Residual convergence: $r^{(k)} = Ax^{(k)} + Bz^k - c \to 0$ as $k \to \infty$, i.e., primal iterates approach feasibility.
- Objective convergence: $f(x^{(k)}) + g(z^{(k)}) \to f^\star + g^\star$, where $f^\star + g^\star$ is the optimal primal objective value.
- Dual convergence: $u(k) \to u^\star$, where $u^\star$ is a dual solution.

For details, see Boyd et al. (2010). Note that we do not generically get primal convergence, but this is true under more assumptions.

Convergence rate: roughly, ADMM behaves like first-order method. Theory still being developed, see, e.g., in Hong and Luo (2012), Deng and Yin (2012), Iutzeler et al. (2014), Nishihara et al. (2015).

21.4 Scaled form ADMM

Scaled form: denote $w = u/\rho$, so augmented Lagrangian becomes

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

and ADMM updates become

$$\begin{aligned} x^{(k)} &= \underset{x}{\operatorname{argmin}}\ f(x) + \frac{\rho}{2}\|Ax + Bz^{(k-1)} - c + w^{(k-1)}\|_2^2 \\ z^{(k)} &= \underset{z}{\operatorname{argmin}}\ g(z) + \frac{\rho}{2}\|Ax^{(k)} + Bz - c + w^{(k-1)}\|_2^2 \\ w^{(k)} &= w^{(k-1)} + (Ax^{(k)} + Bz^{(k)} - c) \end{aligned}$$

where the update for w follows from the stationarity condition as follows

$$\begin{aligned} 0 &\in \partial g(z^{(k)}) + \rho B^T(w^{(k-1)} + (Ax^{(k)} + Bz^{(k)} - c)) \\ &\in \partial g(z^{(k)}) + \rho B^T w^{(k)} \end{aligned}$$

Note that here kth iterate $w^{(k)}$ is just a running sum of residuals:

$$w^{(k)} = w^0 + \sum_{i=1}^{k}(Ax^{(i)} + Bz^{(i)} - c)$$

21.5 Example: alternating projections

Consider finding a point in intersection of convex sets $C, D \subset \mathbb{R}^n$

$$\min_x \; I_C(x) + I_D(x)$$

To get this into ADMM form, we express it as

$$\min_x \; I_C(x) + I_D(x) \, \text{subject to} \, x - z = 0$$

Each ADMM cycle involves two projections:

$$\begin{aligned} x^{(k)} &= \operatorname*{argmin}_x P_C(z^{(k-1)} - w^{(k-1)}) \\ z^{(k)} &= \operatorname*{argmin}_z P_D(x^{(k)} + w^{(k-1)}) \\ w^{(k)} &= w^{(k-1)} + (x^{(k)} - z^{(k)}) \end{aligned}$$

Compare classic alternating projections algorithm (von Neumann):

$$\begin{aligned} x^{(k)} &= \operatorname*{argmin}_x P_C(z^{(k-1)}) \\ z^{(k)} &= \operatorname*{argmin}_z P_D(x^{(k)}) \end{aligned}$$

Difference is ADMM utilizes a dual variable w to offset projections. When (say) C is a linear subspace, ADMM algorithm becomes

$$\begin{aligned} x^{(k)} &= \operatorname*{argmin}_x P_C(z^{(k-1)}) \\ z^{(k)} &= \operatorname*{argmin}_z P_D(x^{(k)} + w^{(k-1)}) \\ w^{(k)} &= w^{(k-1)} + (x^{(k)} - z^{(k)}) \end{aligned}$$

Due to linearity, w does not matter here. Initialized at $z^{(0)} = y$, this is equivalent to Dykstra's algorithm for finding the closest point in $C \cap D$ to y.

21.6 Connection to proximal operators

Consider

$$\min_x \; f(x) + g(x) \Longleftrightarrow \min_{x,z} \; f(x) + g(z), \text{ subject to } \; x - z = 0$$

$$\begin{aligned} x^{(k)} &= \operatorname*{argmin}_x L_\rho(x, z^{(k-1)}, w^{(k-1)}) \\ &= \operatorname*{argmin}_x f(x) + \frac{\rho}{2}\|x - z^{(k-1)} + w^{(k-1)}\|^2 \\ &= \operatorname*{argmin}_x \frac{1}{2 \cdot \frac{1}{\rho}}\|z^{(k-1)} - w^{(k-1)} - x\|^2 + f(x) \\ &= \operatorname{prox}_{f,1/\rho}(z^{(k-1)} - w^{(k-1)}) \end{aligned}$$

and similarly for updating $z^{(k)}$. ADMM steps (equivalent to Douglas-Rachford, here):

$$\begin{aligned} x^{(k)} &= \operatorname{prox}_{f,1/\rho}(z^{(k-1)} - w^{(k-1)}) \\ z^{(k)} &= \operatorname{prox}_{g,1/\rho}(x^{(k)} + w^{(k-1)}) \\ w^{(k)} &= w^{(k-1)} + (x^{(k)} - z^{(k)}) \end{aligned}$$

where $\operatorname{prox}_{f,1/\rho}$ is the proximal operator for f at parameter $1/\rho$, and similarly for $\operatorname{prox}_{g,1/\rho}$.

In general, the update for block of variables reduces to prox update whenever the corresponding linear transformation is the identity.

21.7 Example: lasso regression

Given $y \in \mathbb{R}^n$, $X \in \mathbb{R}^{n\times p}$, recall the lasso problem:

$$\min_{\beta} = \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1.$$

We can rewrite this as:

$$\min_{\beta,\alpha} = \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\alpha\|_1, \text{ subject to } \beta - \alpha = 0$$

To update β:

$$\begin{aligned}
&\nabla_\beta\left(\frac{1}{2}\|y - X\beta\|_2^2 + \frac{\rho}{2}\|\beta - \alpha - w\|_2^2\right) = 0\\
\Longrightarrow & X^T(X\beta - y) + \rho(\beta - \alpha - w) = 0\\
\Longrightarrow & (X^TX + \rho I)\beta - (X^Ty + \rho(\alpha + w)) = 0\\
\Longrightarrow & \beta = (X^TX + \rho I)^{-1}(X^Ty + \rho(\alpha + w))
\end{aligned}$$

To update α:

$$\begin{aligned}
\alpha &= \operatorname*{argmin}_{\alpha} \frac{\rho}{2}\|\beta - \alpha + w\|_2^2 + \lambda\|\alpha\|_1\\
&= \operatorname*{argmin}_{\alpha} \frac{1}{2\cdot\frac{\lambda}{\rho}}\|\beta + w - \alpha\|_2^2 + \|\alpha\|_1\\
&= S_{\frac{\lambda}{\rho}}(\beta + w)
\end{aligned}$$

where the soft-thresholding operator $S_t(x)$ is defined as

$$[S_t(x)]_j = \begin{cases} x_j - t, & \text{if } x_j > 0\\ 0, & \text{if } |x_j| \le t\\ x_j + t, & \text{if } x_j < 0.\end{cases}$$

ADMM steps:

$$\begin{aligned}
\beta^{(k)} &= (X^TX + \rho I)^{-1}(X^Ty + \rho(\alpha^{(k-1)} + w^{(k-1)}))\\
\alpha^{(k)} &= S_{\lambda/\rho}(\beta^{(k)} + w^{(k-1)})\\
w^{(k)} &= w^{(k-1)} + (\beta^{(k)} - \alpha^{(k)})
\end{aligned}$$

Notes:

- The matrix $X^TX + \rho I$ is always invertible, regardless of X
- If we compute a factorization (say Cholesky) in $O(p^3)$ flops, then each β update takes $O(p^2)$ flops
- ADMM steps are "almost" like repeated soft-thresholding of ridge regression coefficients

21.8 Practicalities

In practice, ADMM usually obtains a relatively accurate solution in a handful of iterations, but it requires a large number of iterations for a highly accurate solution (like a first-order method).

Choice of ρ can greatly influence practical convergence of ADMM:

- ρ too large $\to$ not enough emphasis on minimizing $f + g$
- ρ too small $\to$ not enough emphasis on feasibility

Boyd et al. (2010) give a strategy for varying ρ; it can work well in practice, but does not have convergence guarantees.

Like deriving duals, transforming a problem into one that ADMM can handle is sometimes a bit subtle, since different forms can lead to different algorithms.

21.9 Example: group lasso regression

Given $y \in \mathbb{R}^n, X \in \mathbb{R}^{n\times p}$, recall the group lasso problem:

$$\min_\beta \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \sum_{g=1}^G c_g\|\beta_g\|_2.$$

Rewrite as:

$$\min_{\beta,\alpha} = \frac{1}{2}\|y - X\beta\|_2^2 + \lambda \sum_{g=1}^G c_g\|\alpha_g\|_2, \text{ subject to } \beta - \alpha = 0$$

ADMM steps:

$$\begin{aligned}
\beta^{(k)} &= (X^TX + \rho I)^{-1}(X^Ty + \rho(\alpha^{(k-1)} + w^{(k-1)}))\\
\alpha_g^{(k)} &= R_{c_g\lambda/\rho}(\beta_g^{(k)} + w_g^{(k-1)}) \quad g = 1, \ldots, G\\
w^{(k)} &= w^{(k-1)} + (\beta^{(k)} - \alpha^{(k)})
\end{aligned}$$

Notes:

- The matrix $X^TX + \rho I$ is always invertible, regardless of X
- If we compute a factorization (say Cholesky) in $O(p^3)$ flops, then each β update takes $O(p^2)$ flops
- The α update applies the group soft-thresolding operator R_t, which is defined as

$$R_t(x) = (1 - \frac{t}{\|x\|_2})_+ x$$

- Similar ADMM steps follow for a sum of arbitrary norms of as regularizer, provided we know prox operator of each norm
- ADMM algorithm can be rederived when groups have overlap (hard problem to optimize in general!). See Boyd et al. (2010)

21.10 Consensus ADMM

Consider a problem of the form: $\min_x \sum_{i=1}^B f_i(x)$.

The consensus ADMM approach begins by reparametrizing:

$$\min_{x_1,x_2,\ldots,x_B,x} \sum_{i=1}^B f_i(x_i) \text{ subject to } x_i = x, i = 1, \ldots, B$$

This yields the decomposable ADMM steps:

$$x_i^k = \operatorname*{argmin}_{x_i} f_i(x_i) + \frac{\rho}{2\|x_i - x^{(k-1)} + w_i^{(k-1)}\|_2^2}, \; i = 1, \ldots, B$$

$$x^k = \frac{1}{B}\sum_{i=1}^B (x_i^{(k)} + w_i^{(k-1)})$$

$$w_i^{(k)} = w_i^{(k-1)} + x_i^{(k)} - x^{(k)}, \; i = 1, \ldots, B$$

Write $\bar{x} = \frac{1}{B}\sum_{i=1}^B x_i$ and similarly for other variables. Not hard to see that $\bar{w}^k = 0$ for all iterations $k \geq 1$

Hence ADMM steps can be simplified, by taking $x^{(k)} = \bar{x}^{(k)}$

$$x_i^k = \operatorname*{argmin}_{x_i} f_i(x_i) + \frac{\rho}{2\|x_i - \bar{x}^{(k-1)} + w_i^{(k-1)}\|_2^2}, \; i = 1, \ldots, B$$

$$w_i^{(k)} = w_i^{(k-1)} + x_i^{(k)} - \bar{x}^{(k)}, \; i = 1, \ldots, B$$

To reiterate, the xi, i = 1, . . . B updates here are done in parallel.

Intuition:

- Try to minimize each $f_i(x_i)$, use (squared) ℓ_2 regularization to pull each x_i towards the average $\bar{x}$.
- If a variable x_i is bigger than the average, then w_i is increased.
- So the regularization in the next step pulls x_i even closer.

21.11 General consensus ADMM

Consider a problem of the form: $\min_x \sum_{i=1}^B f_i(a_i^T x + b_i) + g(x)$.

For consensus ADMM, we again reparametrize:

$$\min_{x_1,x_2,\ldots,x_B,x} \sum_{i=1}^B f_i(a_i^T x + b_i) + g(x) \text{ subject to } x_i = x, i = 1, \ldots, B$$

This yields the decomposable ADMM updates:

$$x_i^k = \underset{x_i}{\text{argmin}}\ f_i(a_i^T x + b_i) + \frac{\rho}{2\|x_i - x^{(k-1)} + w_i^{(k-1)}\|_2^2},\ i = 1, \ldots, B$$
$$x^k = \frac{B\rho}{2} \underset{x_i}{\text{argmin}} \|x_i - x^{(k-1)} + w_i^{(k-1)}\|_2^2 + g(x)$$

$w_i^{(k)} = w_i^{(k-1)} + x_i^{(k)} - x^{(k)},\ i = 1, \ldots, B$

Notes:

- It is no longer true that $\bar{w}^{(k)} = 0$ at a general iteration k, so ADMM steps do not simplify as before.
- To reiterate, the $x_i, i = 1, \ldots, B$ updates are done in parallel
- Each x_i update can be thought of as a loss minimization on part of the data, with ℓ_2 regularization
- The x update is a proximal operation in regularizer g
- The w update drives the individual variables into consensus
- A different initial reparametrization will give rise to a different ADMM algorithm

See Boyd et al. (2010), Parikh and Boyd (2013) for more details on consensus ADMM, strategies for splitting up into subproblems, and implementation tips.

21.12 Special decompositions

ADMM can exhibit much faster convergence than usual, when we parametrize subproblems in a "special way".

- ADMM updates relate closely to block coordinate descent, in which we optimize a criterion in an alternating fashion across blocks of variables
- With this in mind, get fastest convergence when minimizing over blocks of variables leads to updates in nearly orthogonal directions
- Suggests we should design ADMM form (auxiliary constraints) so that primal updates de-correlate as best as possible
- This is done in, e.g., Ramdas and Tibshirani (2014), Wytock et al. (2014), Barbero and Sra (2014)

22 Frank Wolfe method

A large part of this section is taken from the corresponding lecture notes of Ryan Tibshirani's Convex Optimization course[17].

22.1 Projected gradient descent

The motivation for Frank-Wolfe is projected gradient descent. Projected gradient descent is a special case of proximal gradient descent. Consider a constrained optimization problem, where the solution set is constrained C,

$$\min_x\ f(x),\ \text{subject to}\ x \in C$$

where f is convex and smooth, and C is convex. Recall projected gradient descent uses an initial $x^{(0)}$, and then updates for $k = 1, 2, 3, \cdot$ by first performing gradient descent on the then current solution and then projecting it back onto the constraint set. This can be expressed as

$$x^{(k)} = P_C(x^{(k-1)} - t_k \nabla f(x^{(k-1)}))$$

[17]https://www.stat.cmu.edu/~ryantibs/convexopt-F18/scribes/Lecture_23.pdf

where P_C is the projection operator onto the set C. This is a special case of proximal gradient which is motivated by local quadratic expansion of f. Specifically, recall the proximal mapping

$$\begin{aligned}
x^+ &= \operatorname{prox}_{t,h}(x - t\nabla g(x)) \\
&= \underset{u}{\operatorname{argmin}} \left\{ h(u) + \frac{1}{2t}\|u - (x - t\nabla g(x))\|_2^2 \right\} \\
&= \underset{u}{\operatorname{argmin}} \left\{ h(u) + g(x) + \frac{1}{2t}\|u - x + t\nabla g(x)\|_2^2 \right\} \\
&= \underset{u}{\operatorname{argmin}} \left\{ h(u) + g(x) + \nabla g(x)^T(u - x) + \frac{1}{2t}\|u - x\|_2^2 \right\}
\end{aligned}$$

Note that x^+ minimizes $h(u)$ plus a simple quadratic local model of $g(u)$ around x. When $h(u) = I_C(u)$, we get

$$\begin{aligned}
x^+ &= \underset{u}{\operatorname{argmin}} \left\{ I_C(u) + g(x) + \nabla g(x)^T(u - x) + \frac{1}{2t}\|u - x\|_2^2 \right\} \\
&= \underset{u \in C}{\operatorname{argmin}} \left\{ \nabla g(x)^T(u - x) + \frac{1}{2t}\|u - x\|_2^2 \right\} \\
&= P_C(x - t\nabla g(x))
\end{aligned}$$

So, projected gradient method is a special case of proximal gradient method when h is an indicator function. But we need to know the projection operator for a given constraint beforehand which could be computationally expensive. For example, if your constraint is a polyhedron $C = \{x : Ax \le b\}$, then projection onto it is typically very hard, while there are some special cases, such as if the polyhedra is a simplex $C = \{x : 1^T x = 1,\ x \in [0,1]\}$, then the projection can be computed in linear time.

22.2 Frank-Wolfe Method

The Frank-Wolfe method is an alternative to Projected Gradient Descent which doesn't involve projections. The Frank-Wolfe method is also known as conditional gradient method. Instead of using a quadratic expansion as shown above for the projected GD method, it uses a local linear expansion of f.

$$\begin{aligned}
x^+ &= \underset{u}{\operatorname{argmin}} \left\{ I_C(u) + f(x) + \nabla f(x)^T(u - x) \right\} \\
&= \underset{u \in C}{\operatorname{argmin}}\, \nabla f(x)^T u
\end{aligned}$$

In the Frank-Wolfe method they minimize the linear approximation over the constraint set, instead of projecting afterwards separately. Let s represent u, then

$$s^{(k-1)} \in \underset{s \in C}{\operatorname{argmin}}\ \nabla f(x^{(k-1)})^T s$$

$$x^{(k)} = (1 - \gamma_k)x^{(k-1)} + \gamma_k s^{(k-1)}$$

We take a convex combination of the new point $s^{(k-1)}$ and $x^{(k-1)}$. There is no projection involved, this method is solved directly over the constraint set C. The default choice of step sizes is $\gamma_k = 2/(k+1), k = 1, 2, 3, \cdots$. This choice has been made to facilitate the convergence proof and to derive the convergence rate. Note for any $0 \le \gamma_k \le 1$, we have $x^{(k)} \in C$ by convexity. So we can rewrite the update as

$$x^{(k)} = (1 - \gamma_k)x^{(k-1)} + \gamma_k s^{(k-1)}$$

Here, $(s^{(k-1)} - x^{(k-1)}$ represents the direction we are traveling in. This update states that we are moving less and less in the direction of the linearization minimizer as the algorithm proceeds.

22.3 Solving a problem with a norm constraint

We have the following problem

$$s \in \underset{\|x\| \le t}{\operatorname{argmin}}\, \nabla f(x^{(k-1)})^T s$$

We see that,

$$\begin{aligned}
\min_{\|s\| \le t} \nabla f(x)^T s &= -\max_{\|s\| \le t} -\nabla f(x)^T s \\
&= -t \cdot \max_{\|z\| \le 1} -\nabla f(x)^T z \quad (s = tz) \\
&= -t \cdot \max_{\|z\| \le 1} \nabla f(x)^T z \quad (s = -tz)
\end{aligned}$$

which gives $\text{argmin}_{\|s\|\leq t} \nabla f(x)^T s = -t \cdot \text{argmax}_{\|z\|\leq 1} \nabla f(x)^T z$. $\text{argmax}_{\|z\|\leq 1} \nabla f(x)^T z$ is equal to the subdifferential of $\|\cdot\|$'s dual norm which is defined as

$$\|z\|_\star = \max_{\|x\|\leq 1} z^T x$$

where $\|\cdot\|_\star$ denotes the dual norm of $\|\cdot\|$. That is, if we know how to compute the subgradients of the dual norm, then we can easily perform the Frank-Wolfe steps. Since we have a closed form update now, this can often be much cheaper or simpler than projection onto $C = \{x : \|x\|\leq t\}$. We look at the updates for some special cases of norm constraints and compare it to the projected gradient descent method for these cases.

22.3.1 ℓ_1 Regularization

If we were to solve the constrained form of lasso, we'd have an ℓ_1 ball as the constraint set C. To use the Frank-Wolfe method, we'd need to know what the dual norm is and what its subgradient looks like for ℓ_1 ball.

$$\min_x \ f(x), \ \text{subject to } \|x\|_1 \leq t$$

The dual norm of the ℓ_1 norm is the ℓ_∞ norm. So, we have $s^{(k-1)} \in -t\partial\|\nabla f(x^{(k-1)})\|_\infty$. If $|\nabla f(x^{(k-1)})|$ has its component-wise maximum at i, then the subgradient is the standard basis vector e_i. This can be written as

$$i^{(k-1)} \in \underset{i=1,\cdots,p}{\text{argmax}} |\nabla_i f(x^{(k-1)})|$$

$$x^{(k)} = (1-\gamma_k)x^{(k-1)} - \gamma_k t \cdot \text{sgn}(\nabla_{i^{(k-1)}} f(x^{(k-1)})) \cdot e_{i^{(k-1)}}$$

This looks like greedy coordinate descent since coordinate descent goes through the vector cyclically whereas Frank-Wolfe here picks the largest component at each iteration. Note that this update is simpler than projecting onto a ℓ_1 ball, although they both have the same computational complexity, i.e., $O(n)$.

22.3.2 ℓ_p Regularization

More generally, for the ℓ_p regularized problem

$$\min_x \ f(x), \ \text{subject to } \|x\|_p \leq t$$

for $1 \leq p \leq \infty$, we have $s^{(k-1)} \in -t\partial\|\nabla f(x^{(k-1)})\|_q$, where $1/p + 1/q = 1$. It is interesting to note that the subgradient of a given dual norm can be computed efficiently using the following function, where α is a scaling factor and the rest is the scaled form of the subgradient of max over ℓ_q norm.

$$s_i^{(k-1)} \in -\alpha \cdot \text{sgn}(\nabla_i f(x^{(k-1)})) \cdot |\nabla_i f(x^{(k-1)})|^{p/q}, \ i = 1, \cdots, n$$

where α is used to satisfy the constraint $\|s^{(k-1)}\|_q = t$.

This is followed by the main update, i.e., a convex combination (γ_k) of $(s^{(k-1)}, x^{(k-1)})$. Note that these update rules are a lot simpler than projection onto the ℓ_p ball for any general p since there exists no general projection rule. Aside from special cases ($p = 1, 2, \infty$), these projections cannot be computed directly, the projection step must be treated as an optimization.

22.3.3 Trace norm regularization

The trace-regularized problem takes the following form,

$$\min_X \ f(X), \ \text{subject to } \|X\|_{\text{tr}} \leq t$$

Recall that the trace norm is the sum of the singular values of X. Now, to derive our update S we need to (1) compute the dual norm, (2) find the subgradients of the resultant dual norm. We first note that the dual of the trace norm is the operator norm (largest singular value of X), and obtain the following general update,

$$S^{(k-1)} \in -t\partial\|\nabla f(X^{(k-1)}\|_{\text{op}}$$

We now claim that the update $S^{(k-1)}$ can be explicitly written as, $S^{(k-1)} = -tuv^T$, where u, v are the leading left and right singular vectors of $\nabla f(X^{(k-1)})$. Hence, this means that $\partial\|\nabla f(X^{(k-1)}\|_{\text{op}} = uv^T$. Further, note that we can compute u, v using the power method on $\nabla f(X^{(k-1)})$, which is very cheap if the matrix is sparse.

However, taking projection onto the norm ball would require computing the full SVD, which is more complicated and expensive than Frank-Wolfe.

22.3.4 Method Comparisons over Various norms

- ℓ_1 norm: Frank-Wolfe update scans for maximum of gradient; proximal operator soft-thresholds the gradient step; both use $O(n)$ flops
- ℓ_p norm: Frank-Wolfe update raises each entry of gradient to power and sums, in $O(n)$ flops; proximal operator not generally directly computable
- Trace norm: Frank-Wolfe update computes top left and right singular vectors of gradient; proximal operator soft-thresholds the gradient step, requiring a singular value decomposition

Various other constraints yield efficient Frank-Wolfe updates, e.g., special polyhedra or cone constraints, **sum-of-norms (group-based) regularization**, atomic norms, See Martin Jaggi's doctoral dissertation[18].

22.4 Example: lasso comparison

Comparing projected and conditional gradient for constrained lasso problem, with $n = 100, p = 500$: Note that

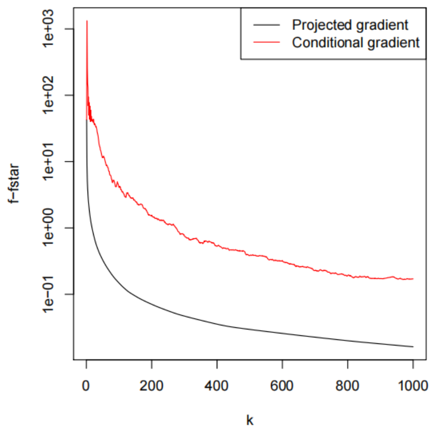


Figure 11: Frank-Wolfe vs projected GD.

both projected gradient and Frank-Wolfe are $O(n)$. From Fig. 11, we notice that Frank-Wolfe is not a descent estimator as the objective is not monotonically decreasing over each k. Lastly, Frank-Wolfe in this problem uses standard step sizes, and a different step size method such as line search would probably help in terms of convergence.

22.5 Duality gap

Frank-Wolfe iterations admit a very natural duality gap:

$$\nabla f(x^{(k)})^T(x^{(k)} - s^{(k)})$$

Claim: this upper bounds the duality gap.

Proof. By the first-order convexity condition,

$$f(s) \geq f(x^{(k)}) + \nabla f(x^{(k)})^T(s - x^{(k)})$$

Minimizing both sides over all $s \in C$ yields,

$$\begin{aligned} f^\star &\geq f(x^{(k)}) + \min_{s\in C} \nabla f(x^{(k)})^T(s - x^{(k)}) \\ &= f(x^{(k)}) + \min_{s\in C} \nabla f(x^{(k)})^T(s^{(k)} - x^{(k)}) \end{aligned}$$

which can be re-written as,

$$\begin{aligned} f^\star &\geq f(x^{(k)}) + \nabla f(x^{(k)})^T(s^{(k)} - x^{(k)}) \\ f^\star - f(x^{(k)}) &\geq \nabla f(x^{(k)})^T(s^{(k)} - x^{(k)}) \\ f(x^{(k)}) - f^\star &\leq \nabla f(x^{(k)})^T(x^{(k)} - s^{(k)}) \end{aligned}$$

Hence, it is the upper bound. $\square$

[18]Martion Jaggi. Sparse Convex Optimization Methods for Machine Learning, 2011. https://www.research-collection.ethz.ch/bitstream/handle/20.500.11850/72811/eth-5262-02.pdf?sequence=2&isAllowed=y

Why do we call this a "duality gap"? Re-write original problem as

$$\min_x \; f(x) + I_C(x)$$

where I_C is the indicator function of C. The dual problem is then (by conjugate and dual of indicator function),

$$\max_u \; -f^\star(u) - I^\star(-u)$$

where I_C^* is the support function of C. If we choose x, u that are feasible in the primal and dual, respectively, then $I_C(x) = 0$ and we obtain the following duality gap,

$$f(x) - (-f^*(u) - I_C^*(-u)) = f(x) + f^*(u) + I_C^*(-u)$$

Then, by Fenchel's inequality,

$$f(x) + f^*(u) + I_C^*(-u) \geq x^T u + I_C^*(-u)$$

Evaluating this at $x = x^{(k)}, u = \nabla f(x^{(k)})$, we obtain:

$$\nabla f(x^{(k)})^T x^{(k)} + \max_{s \in C} -\nabla f(x^{(k)})^T s = \nabla f(x^{(k)})^T (x^{(k)} - s^{(k)})$$

Here, $\nabla f(x^{(k)})^T (x^{(k)} - s^{(k)})$ lower bounds the difference between the primal value and the dual value. However, in the previous claim, $\nabla f(x^{(k)})^T (x^{(k)} - s^{(k)})$ upper bounds the difference between $f(x^{(k)})$ and the optimal objective value $f(x^*)$.

22.6 Convergence analysis

Following Jaggi's dissertation[19], define the **curvature constant** of f over C:

$$M = \max_{\substack{\gamma \in [0,1] \\ x,s,y \in C \\ y = (1-\gamma)x + \gamma s}} \frac{2}{\gamma^2} (f(y) - f(x) - \nabla f(x)^T (y - x))$$

Note that $M = 0$ for linear f, big if f is highly curved over the set C, and $f(y) - f(x) - \nabla f(x)^T (y - x)$ is called the **Bregman divergence** from x to y, defined by f.

Theorem 16. The Frank-Wolfe method using standard step sizes $\frac{2}{k+2}, k = 1, 2, 3, \cdots$ satisfies,

$$f(x^{(k)}) - f^\star \leq \frac{2M}{k+2}$$

Then, the number of iterations needed for $f(x^{(k)}) - f^\star \leq \epsilon$ is $O(\epsilon)$. This matches the sublinear rate for projected gradient descent for Lipschitz ∇f, but how do the assumptions compare? For Lipschitz ∇f with constant L, recall,

$$f(y) - f(x) - \nabla f(x)^T (y - x) \leq \frac{L}{2} \|y - x\|_2^2$$

Maximizing over all $y = (1 - \gamma)x + \gamma s$, and multiplying by $\frac{2}{\gamma^2}$,

$$\begin{aligned} M &\leq \max_{\substack{\gamma \in [0,1] \\ x,s,y \in C \\ y = (1-\gamma)x + \gamma s}} \frac{2}{\gamma^2} \cdot \frac{L}{2} \|y - x\|_2^2 \\ &\leq \max_{x,s \in C} \frac{2}{\gamma^2} \cdot \frac{L}{2} \|(1 - \gamma)x + \gamma s - x\|_2^2 \\ &= \max_{x,s \in C} \frac{L}{\gamma^2} \|\gamma(s - x)\|_2^2 = L \cdot \mathrm{diam}^2(C) \end{aligned}$$

where $\mathrm{diam}^2(C)$ is the squared diameter of the set C. So, if f has a gradient that is Lipschitz, and C is compact, then it immediately has a curvature that is finite and that is at most $L \cdot \mathrm{diam}^2(C)$.

Hence assuming a bounded curvature is basically **no stronger** than what we assumed for projected GD method.

[19]Martion Jaggi. Sparse Convex Optimization Methods for Machine Learning, 2011. https://www.research-collection.ethz.ch/bitstream/handle/20.500.11850/72811/eth-5262-02.pdf?sequence=2&isAllowed=y

Proof. We denote the duality gap $\max_{s \in C} \nabla f(x)^T(x-s)$ by $g(x)$. By the definition of M,

$$f\left(x+\gamma(s-x)\right)-f(x)-\gamma\nabla f(x)^T(s-x) \leq \frac{\gamma^2}{2}M$$

Let $x^+ = x+\gamma(s-x)$. Then,

$$\begin{aligned} f(x^+) &\leq f(x)+\gamma\nabla f(x)^T(s-x)+\frac{\gamma^2}{2}M \\ &= f(x)-\gamma g(x)+\frac{\gamma^2}{2}M \end{aligned}$$

Let $x^+ = x^{(k)}, x = x^{(k-1)}, \gamma = \gamma_{k-1}, s = s^{(k-1)}$. Denote by $h(x) = f(x) - f^\star$ the suboptimality gap at x. Then

$$\begin{aligned} h(x^{(k)}) &\leq h(x^{(k-1)})-\gamma_k\nabla g(x^{(k-1)})+\frac{\gamma^2}{2}M \\ &\leq f(x)-\gamma_{k-1}h(x^{(k-1)})+\frac{\gamma_{k-1}^2}{2}M \\ &= (1-\gamma_{k-1})h(x^{(k-1)})+\frac{\gamma_{k-1}^2}{2}M \end{aligned} \tag{46}$$

where the second line follows from $g(x^{(k-1)}) \geq h(x^{(k-1)})$ since $g(x^{(k-1)})$ is the upper bound of the difference between $h(x^{(k-1)}) = f(x^{(k-1)}) - f^\star$. We will now use induction over k to prove our claimed bound, i.e.,

$$h(x^{(k)}) \leq \frac{2M}{k+2}$$

The base-case $k = 0$ follows from (46) applied for the first step of the algorithm, using $\gamma_0 = \frac{2}{0+2} = 1$. Now considering $k \geq 1$, the bound (46) reads as

$$\begin{aligned} h(x^{(k)}) &\leq (1-\gamma_{k-1})h(x^{(k-1)})+\frac{\gamma_{k-1}^2}{2}M \\ &= (1-\frac{2}{k+1})h(x^{(k-1)})+(\frac{2}{k+1})^2\frac{M}{2} \\ &\leq \frac{k-1}{k+1}\cdot\frac{2M}{k+1}+(\frac{2}{k+1})^2\frac{M}{2} \\ &= \frac{2M(k-1)}{(k+1)^2}+\frac{2M}{(k+1)^2} \\ &= \frac{2kM}{(k+1)^2} \\ &\leq \frac{2kM}{k(k+2)} = \frac{2M}{k+2} \end{aligned}$$

where in the third line we use the induction hypothesis for $h(x^{(k-1)})$ and in the last line we use $(k+1)^2 - k(k+2) = 1 > 0$. This completes the proof for the claimed bound for $k \geq 1$.

□

22.7 Affine invariance

One of important properties that are not shared with the projected GD method is affine invariance. For nonsingular matrix A, define $x = Ax'$ and $F(x') = f(x) = f(Ax')$. Consider Frank-Wolfe on F and $x \in C, x = Ax' \Leftrightarrow x' \in A^{-1}C$, we have

$$s' = \underset{z \in A^{-1}C}{\operatorname{argmin}} \ \nabla F(x')^T z$$

$$(x')^+ = (1-\gamma)x' + \gamma s'$$

By multiplying A on both sides,

$$A(x')^+ = (1-\gamma)Ax' + \gamma As'$$

and then by applying $\nabla F(x') = A^T f(Ax')$,

$$\begin{aligned} As' &= A \cdot \underset{z \in A^{-1}C}{\operatorname{argmin}} \nabla F(x')^T z \\ &= A \cdot \underset{Az \in C}{\operatorname{argmin}} \nabla f(Ax')^T Az \\ &= A \cdot A^{-1} \underset{w \in C}{\operatorname{argmin}} \nabla f(Ax')^T w \\ &= \underset{w \in C}{\operatorname{argmin}} \nabla f(x)^T w \end{aligned}$$

which produces the same Frank-Wolfe update as that from f. Convergence analysis is also affine invariant. The curvature constant M of F:

$$\begin{aligned} M' &= \max_{\substack{\gamma \in [0,1] \\ x', s', y' \in A^{-1}C \\ y' = (1-\gamma)x' + \gamma s'}} \frac{2}{\gamma^2}(F(y') - F'(x') - \nabla F(x')^T(y' - x')) \\ &= \max_{\substack{\gamma \in [0,1] \\ Ax', As', Ay' \in C \\ Ay' = (1-\gamma)Ax' + \gamma As'}} \frac{2}{\gamma^2}(F(y') - F'(x') - \nabla f(Ax')^T A(y' - x')) \\ &= \max_{\substack{\gamma \in [0,1] \\ x, s, y \in C \\ y = (1-\gamma)x + \gamma As}} \frac{2}{\gamma^2}(f(y) - f(x) - \nabla f(x)^T(y - x)) \end{aligned}$$

matches that of f.

References

[Beck, 2014] Beck, A. (2014). Introduction to nonlinear optimization - theory, algorithms, and applications with matlab. In MOS-SIAM Series on Optimization.

[Boyd and Vandenberghe, 2004] Boyd, S. and Vandenberghe, L. (2004). Convex Optimization.

[Garber and Hazan, 2015] Garber, D. and Hazan, E. (2015). Faster rates for the frank-wolfe method over strongly-convex sets. In Proceedings of the 32th International Conference on Machine Learning, pages 541–549.

[Lee et al., 2014] Lee, J. D., Sun, Y., and Saunders, M. A. (2014). Proximal newton-type methods for minimizing composite functions. SIAM Journal on Optimization, 24(3):1420–1443.