Convex Optimization Note

Zelin Yao

2024 年 9 月 23 日

前言

if people do not believe that mathematics is simple, it is only because they do not realize how complicated life is. ——John von Neumann

2024 年 9 月 23 日

与其焦虑不如先行动起来！

目录

[1] A. Nedic and A. Ozdaglar, "Distributed Subgradient Methods for Multi-Agent Optimization," in IEEE Transactions on Automatic Control, vol. 54, no. 1, pp. 48-61, Jan. 2009, doi: 10.1109/TAC.2008.2009515

[2] A. D. Domínguez-García and C. N. Hadjicostis, "Distributed strategies for average consensus in directed graphs," 2011 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, USA, 2011, pp. 2124-2129, doi: 10.1109/CDC.2011.6160462.

第一章　Optimization Basis

1.1　KKT Conditions

1.2　Derivatives

This note is summarized by [1].

Consider a function $f : \mathbb{R}^n \to \mathbb{R}$, *the partial derivatives, directional derivatives* have the following definitions,

Def. 1.2.1: Partial Derivatives, Directional Derivatives

Assume e_i is the unit vector in the x_i direction, the *partial derivative* of function $f(\vec{x})$ in the x_i direction is,

$$\frac{\partial f(\vec{x})}{\partial x_i} = \lim_{\alpha \to 0} \frac{f(\vec{x} + \alpha e_i) - f(\vec{x})}{\alpha}$$

The *directional derivative* of $f(\vec{x})$ in direction d is,

$$f^{'}(\vec{x}; d) = \lim_{\alpha \to 0} \frac{f(\vec{x} + \alpha d) - f(\vec{x})}{\alpha}$$

If the function $f(\vec{x})$ is differentiable of each directions, and the direction d is a **unit vector**, the following relation is satisfied,

$$f^{'}(\vec{x}; d) = \sum_{i=1}^{n} \frac{\partial f(\vec{x})}{\partial x_i} d_i = [\frac{\partial f(\vec{x})}{\partial x_i}]^T_{i \in [n]} d = \nabla f(\vec{x})^T d$$

If we check the the directional derivative of function $f(x, y, z) = y^2 d^{xyz}$ in direction $d = (0, 1, 2)$ and the gradient, the unit vector of the direction is necessary.

1.3 Matrix Calculus

If the variables about domains are considered as inputs of a system, and the variables about ranges are called outputs of a system, these inputs and outputs can be divided into three types, i.e. scalars, vectors, and matrices. **This article is based on *Numerator Layout*, whose means will demonstrate as following!** This table can clearly describe all situations. In

表 1.1: The Types of All Derivatives

Output \ Input	Scalar	Vector	Matrix
Scalar	$f^{'}(x)$	$f(\vec{x}):\mathbb{R}^{m\times 1}\to\mathbb{R}$	$f(X):\mathbb{R}^{m\times n}\to\mathbb{R}$
Vector	$\vec{f}(x):\mathbb{R}\to\mathbb{R}^{m\times 1}$	$\vec{f}(\vec{x}):\mathbb{R}^{m\times 1}\to\mathbb{R}^{n\times 1}$	$\vec{f}(X):\mathbb{R}^{m\times n}\to\mathbb{R}^{n\times 1}$
Matrix	$F(x):\mathbb{R}\to\mathbb{R}^{m\times n}$	$F(\vec{x}):\mathbb{R}^{m\times 1}\to\mathbb{R}^{p\times q}$	$F(X):\mathbb{R}^{m\times n}\to\mathbb{R}^{p\times q}$

optimization problem, the most common situations are scalar output, and learning it is critical!

Notes 1.3.1: The Meaning of *Numerator Layout* and *Denominator Layout*

Firstly, considering a function $\vec{f}(\vec{x}):\mathbb{R}^{n\times 1}\to\mathbb{R}^{m\times 1}$, which means that the input and output are both vector and not matrices. If its derivative is arranged as following and the dimension of each elements is equal to the numerator of derivative, we call this layout to be *Numerator Layout*. The results of *Denominator Layout* is just the transpose of the results of *Numerator Layout*.

We first construct the **outer matrix**, which decomposes the vector $\vec{x}$ to $x_1, x_2, ..., x_n$ and forms derivatives by $\partial\vec{f}(\vec{x})/\partial x_i, \forall i\in\{1,2,...,n\}$. And more important, the dimension of **outer matrix** is equal to the dimension of transposed denominator.

$$\frac{\partial\vec{f}(\vec{x})}{\partial\vec{x}}=\begin{pmatrix}\partial\vec{f}(\vec{x})/\partial x_1 & \partial\vec{f}(\vec{x})/\partial x_2 & ... & \partial\vec{f}(\vec{x})/\partial x_n\end{pmatrix}$$

Each elements is called **inner matrix**, which has the following form,

$$\frac{\partial\vec{f}(\vec{x})}{\partial x_i}=\begin{pmatrix}\partial f_1(\vec{x})/\partial x_i\\ \partial f_2(\vec{x})/\partial x_i\\ ...\\ \partial f_m(\vec{x})/\partial x_i\end{pmatrix}\in\mathbb{R}^{m\times 1}$$

1.3.1 Notation

1.3.2 Basic Methods

Method Based on Elements

This method uses the definition of Note 1.3. The key of this method is that based on single input and output problem decomposing other problems into single variable problem. Next we give some examples in subsection 1.3.4. Actually, this method can be used to prove some theorems or formulation, but it is costly.

Method Based on Limitation

Firstly, recall the differential of $f(x)$ that is **scalar input and scalar output**. The difference (not differential) has this form,

$$\Delta f = f(x + \Delta x) - f(x) = f^{'}(x)\Delta x + o(||\Delta x||)$$

Let $\Delta x \to 0$. The Δx is transformed to dx (maybe mathematical abuse), and high-order term is omitted. The *differential* can be reformulated as,

$$df = f(x + dx) - f(x) = f^{'}(x)dx$$

In this time, the simplest situation is solved, but how we know the definition of gradient and etc.

Secondly, some other situations about scalar and vector (the situations about matrices are a little bit different) are considered. According to the definition of scalar input and scalar output, the situation of **vector input and scalar output** is similar.
Consider $f(\vec{x}) : \mathbb{R}^n \to \mathbb{R}$. Similar with the situation of scalar input and scalar output, we have,

$$df = f(\vec{x} + d\vec{x}) - f(\vec{x}) = f^{'}(\vec{x})d\vec{x}$$

In fact, the $f^{'}(\vec{x})$ is called *derivative* and it is a row vector (or covector/dual vector). The reason why it is a row vector is that the dimension match that df belongs to $\mathbb{R}$ and $d\vec{x}$ is a column vector. Obvious, the result is in line with the definition by elements. The ***gradient*** is the transposed derivative.
The situation of **scalar input and vector output** is similar with the situation of vector input

and scalar output. The difference between them is the dimension of derivative (actually the relation of transposing).

The most critical situation is **vector input and vector output**. Using the definition by elements, the dimension of derivative is not obvious and the calculation is complex and trivial. But the method based on limitation can deal with it. Consider $\vec{f}(\vec{x}) : \mathbb{R}^{n\times 1} \to \mathbb{R}^{m\times 1}$. The similar formulation is,

$$d\vec{f} = \vec{f}(\vec{x} + d\vec{x}) - \vec{f}(\vec{x}) = Jd\vec{x}$$

Where $J \in \mathbb{R}^{m\times n}$ to match dimension of $d\vec{f}$ and $d\vec{x}$.

Thirdly, the special and critical situations are the input or output associated matrices. **The differential can be represented as the dot product of derivative and dX, but if the input or output is matrix, the dot product of matrix must be defined again to make the situation satisfy the definition of scalar and vector situation.** A new dot product definition will be introduced, and trace and vec are used to simplify expressions. New definition about dot product is defined as following.

Def. 1.3.1: The Definition of Matrices Dot Product

The meaning of dot product is the sum of multiplying all corresponding elements. The *dot product* of matrix A and B is that

$$\begin{aligned} A \bullet B &= \sum_i \sum_j A_{ij} B_{ij} \\ &= \langle A, B \rangle \\ &= vec(A)^T vec(B) \\ &= tr(A^T B) \\ &= tr(BA^T) \end{aligned}$$

The detailed examples about matrices are shown in subsection 1.3.4

1.3.3 Derivative Properties and Other Prerequisites

These properties is useful to the all situation whatever scalars, vectors and matrices. And *Numerator Layout* is adapted. I will list all of them, and proof some important theorems in subsection 1.3.4.

Properties 1.3.1: Derivative Properties

1. **(Sum Rule)** if $f(x) = g(x) + h(x); f, g, h$ are scalar, vector, or matrix function, then $df = dg + dh$ and $df/dx = dg/dx + dh/dx$
2. **(Product Rule)**if $f(x) = g(x)h(x); f, g, h$ are scalar, vector, or matrix function, then $df = dgh + gdh$. Notice that the order cannot be changed!
3. **(Chain Rule)**if $f(x) = g(h(x)); f, g, h$ are scalar, vector, or matrix function, then $df = dg(h(x))dh(x)$. Another equivalent formulation is that $\frac{df}{dx} = \frac{dg(h(x))}{dh(x)}\frac{dh(x)}{dx}$. Using method of elements to prove it easily.
   Notice that the order cannot be changed!
4. **(Linearity)** $d(au(x) + bv(x))/dx = d(u(x))/dx + d(v(x))/dx$. We have proofed this property is satisfied when scalar/vector/matrix input and scalar/vector/matrix output.

Their proofs are convenient when using the method of elements.

Properties 1.3.2: Trace

1. (Linearity) Assume a, b are constant, A, B are same dimensional matrices. $tr(aA \pm bB) = atr(A) + btr(B)$
2. (Transpose) $tr(A) = tr(A^T)$
3. (Matrix Dot Product) $tr(A^T B) = tr(B^T A) = tr(AB^T) = tr(BA^T) = \langle A, B\rangle = \sum_i \sum_j A_{ij}B_{ij}$
4. (Cyclic Property) Assume three asymmetric matrices $A, B, C \Rightarrow tr(ABC) = tr(BCA) = tr(CAB)$. Assume three symmetric matrices A, B, C, the order in trace can be changed arbitrarily.

Properties 1.3.3: Kronecker Product

1. (Trace) Assume A, B are symmetric, and then $tr(A \otimes B) = tr(A)tr(B)$.
2. (Transpose and Inverse) $(A \otimes B)^T = (A^T) \otimes (B^T)$ and $(A \otimes B)^{-1} = (A^{-1}) \otimes (B^{-1})$. Under Kronecker Product, the inverse and transpose cannot change the multiple order.
3. (Kronecker Sum/Tensor Sum) Assume $A \in \mathbb{R}^{n \times n}, B \in \mathbb{R}^{m \times m}$, $A \oplus B = (I_m \otimes A) + (B \otimes I_n) \in \mathbb{R}^{mn \times mn}$
4. (vec/span a matrix column by column) $vec(ABC) = (C^T \otimes A)vec(B)$ for any matrix A, B, C.

1.3.4 Some Examples, Proofs and Applications

$\mathbf{f}(\mathbf{x}) : \mathbb{R} \to \mathbb{R}$

Easy!

$\mathbf{f(x)} : \mathbb{R}^{\mathbf{m\times 1}} \to \mathbb{R}$

Examples 1.3.1

1. $f(\vec{x}) = \vec{x}^T A\vec{x}, x \in \mathbb{R}^{m\times 1}, A \in \mathbb{R}^{m\times m}$.

Sol1: (Method of elements)Firstly, the outer matrix can be written as,

$$\frac{\partial f(\vec{x})}{\partial x} = \left(\partial f(\vec{x})/\partial x_1, ..., \partial f(\vec{x})/\partial x_m\right)$$

where the dimension of outer matrix is the dimension of transposed denominator. Then each inner matrices can be expressed as,

$$\frac{\partial f(\vec{x})}{\partial x_i} = 2A_{ii}x_i + \sum_{k\neq i}(A_{ki} + A_{ik})x_k = \sum_{k=1}^{m}(A_{ki} + A_{ik})x_k$$

So, substitute the elements of outer matrix with inner matrix, then the derivative is

$$\frac{\partial \vec{x}^T A\vec{x}}{\partial \vec{x}} = x^T(A + A^T)$$

Sol2:(Method based on limitation) The method of elements is trivial. But if we use the properties of derivative fully, the situation will be easy to tackle.

$$\begin{aligned} df &= f(\vec{x} + d\vec{x}) - f(\vec{x}) \\ &= (\vec{x} + d\vec{x})^T A(\vec{x} + d\vec{x}) - \vec{x}^T A\vec{x} \\ &= d\vec{x}^T A\vec{x} + \vec{x}^T Ad\vec{x} + d\vec{x}^T Ad\vec{x} \end{aligned}$$

Because this method is based on limitation and omits the notation of limit, the limitation of $d\vec{x}$can be expressed as $d\vec{x} \to 0$. So the above equation can be,

$$\begin{aligned} df &= f(\vec{x} + d\vec{x}) - f(\vec{x}) \\ &= d\vec{x}^T A\vec{x} + \vec{x}^T Ad\vec{x} \\ &= (d\vec{x}^T A\vec{x})^T + \vec{x}^T Ad\vec{x} \qquad \textbf{the transpose of scalar is still this scalar} \\ &= \vec{x}^T(A + A^T)d\vec{x} \end{aligned}$$

So the derivative of the given function is $\vec{x}^T(A + A^T)$, and the gradient is $(A + A^T)\vec{x}$

Sol3:(Derivative properties) $df = d\vec{x}^T A\vec{x} + \vec{x}^T dA\vec{x} + \vec{x}^T Ad\vec{x}$. Transform it, and the result is obvious.

Examples 1.3.2: $l2-norm$

$f(\vec{x}) = ||\vec{x}||_2, \vec{x} \in \mathbb{R}^{m\times 1}$

Squaring both sides,

$$f(\vec{x})f(\vec{x}) = ||\vec{x}||_2^2 = \vec{x}^T\vec{x}$$

Using derivative properties,

$$2fdf = d\vec{x}^T\vec{x} + \vec{x}^T d\vec{x} = 2\vec{x}^T d\vec{x}$$

So the derivative of f is,

$$df(\vec{x}) = \frac{\vec{x}^T}{f(\vec{x})}d\vec{x} = \frac{\vec{x}^T}{||\vec{x}||_2}d\vec{x}$$

All in all, the derivative of $l2-norm$ is $\frac{\vec{x}^T}{||\vec{x}||_2}$, and the gradient is $\frac{\vec{x}}{||\vec{x}||_2}$

$\mathbf{f(x)} : \mathbb{R}^{\mathbf{m\times n}} \to \mathbb{R}$

Examples 1.3.3: Inverse

$f(X) = X^{-1}, X \in \mathbb{R}^{m\times m} \to \mathbb{R}$

In fact, the method based on limitation and derivative properties is effective in calculating derivative, but not useful in proof. We have,

$$X^{-1}X = I$$

By product rule of derivative, we have,

$$dX^{-1}X + X^{-1}dX = dI = 0$$

So, the derivative of the inverse of X is,

$$dX^{-1} = -X^{-1}dXX^{-1}$$

Examples 1.3.4: $F-norm$

$f(X) = ||X||_F = \sqrt{tr(X^T X)}, X \in \mathbb{R}^{m \times n} \to \mathbb{R}$

Firstly, using chain rule of derivative,

$$df(X) = \frac{d(tr(X^T X))}{2\sqrt{tr(X^T X)}}$$

According to the property of trace ($d(tr(A)) = tr(d(A))$),

$$\begin{aligned} df(X) &= \frac{tr(d(X^T X))}{2\sqrt{tr(X^T X)}} \\ &= \frac{tr(d(X^T)X))}{2\sqrt{tr(X^T X)}} + \frac{tr(X^T dX))}{2\sqrt{tr(X^T X)}} \end{aligned}$$

Use the property of trace ($d(A^T) = d(A)$),

$$\begin{aligned} df(X) &= \frac{tr(X^T dX))}{\sqrt{tr(X^T X)}} \\ &= \frac{X \bullet dX}{||X||_F} \qquad \textbf{(The definition of matrix dot product)} \end{aligned}$$

So, the derivative of $||X||_F$ is $X/||X||_F$, and the gradient is $X^T/||X||_F$

Examples 1.3.5: Determinant

$f(X) = det(X), X \in \mathbb{R}^{m\times m} \to \mathbb{R}$

Without doubt, the method based on limitation is used.

$$\begin{aligned} df(X) &= det(X + dX) - det(X) \\ &= det(X + XX^{-1}dX) - det(X) \\ &= det(X(I + X^{-1}dX)) - det(X) \\ &= det(X)det(I + X^{-1}dX) - det(X) \end{aligned}$$

The determinant can be treated as the multiplication of eigenvalues. Assume the eigenvalues of $X^{-1}dX$ are $\lambda_1, \lambda_2, ..., \lambda_m$, the eigenvalues of $I+X^{-1}dX$ are $1+\lambda_1, 1+\lambda_2, ..., 1+\lambda_m$. So the deternimant of $I + X^{-1}dX$ is,

$$\begin{aligned} det(I + X^{-1}dX) &= (1 + \lambda_1)(1 + \lambda_2)...(1 + \lambda_m) \\ &= 1 + (\lambda_1 + \lambda_2 + ... + \lambda_m) + ... \\ &\approx 1 + (\lambda_1 + \lambda_2 + ... + \lambda_m) \end{aligned}$$

So, $det(X) = det(X)(\lambda_1 + \lambda_2 + ... + \lambda_m) = tr(det(X)X^{-1}dX)$

$\mathbf{f(x)} : \mathbb{R}^{\mathbf{m\times 1}} \to \mathbb{R}^{\mathbf{n\times 1}}$

Examples 1.3.6

$\vec{f}(\vec{x}) = A\vec{x}, A \in \mathbb{R}^{m\times n} \to \mathbb{R}^{v\times 1}$

Similarly, using the method of elements is complex. We use properties and limitation here to solve it.

$$d\vec{f} = dA\vec{x} + Ad\vec{x} = Ad\vec{x}$$

The derivative is A, and the gradient is A^T.

$\mathbf{f(x)} : \mathbb{R}^{\mathbf{m}\times\mathbf{n}} \to \mathbb{R}^{\mathbf{p}\times\mathbf{q}}$

Examples 1.3.7

$F(X) = X^3, X \in \mathbb{R}^{m\times m} \to \mathbb{R}^{m\times m}$

Briefly writing the process,

$$dF = dXX^2 + XdXX + X^2dX$$

Meet some problems that the dF cannot be represented the explicit form of $(\cdot)\bullet dX$. But if the new notation of *vec* (span a matrix column by column and link with them)is used, the problem is solved. According to the property of $vec(ABC) = (C^T \otimes A)vec(B)$,

$$\begin{aligned}
&vec(dF) = vec(dXX^2) + vec(XdXX) + vec(X^2dX)\\
\Leftrightarrow &vec(dF) = vec(IdXX^2) + vec(XdXX) + vec(X^2dXI)\\
\Leftrightarrow &vec(dF) = (X^2 \otimes I)vec(dX) + (X \otimes X)vec(dX) + (I \otimes X^2)vec(dX)\\
\Leftrightarrow &vec(dF) = ((X^2 \otimes I) + (X \otimes X) + (I \otimes X^2))vec(dX)
\end{aligned}$$

1.4 Random Vectors, Random Matrices, and Their Expected Values

This note comes from the summary of James H. Steiger.

Some intuitive senses,

1. *random vectors* or *random matrices* are these vectors or matrices that each of their elements is a single random variable.
2. The expectation of random vector is vector that expects its all elements.

1.5 Convexity and Smoothness

The definition and some explanations are stemmed form [2], [3], [4].

Def. 1.5.1: Convexity

1. A *set* $\mathcal{K}$ is convex if $\forall x, y \in \mathcal{K}$, $\alpha \in [0,1]$, we have $\alpha x + (1-\alpha)y \in \mathcal{K}$
2. A *function* $f : \mathcal{K} \to \mathbb{R}$ is convex if $\forall x, y \in \mathcal{K}$, $\alpha \in [0,1]$, we have $f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y)$
3. A *differentiable function* $f : \mathcal{K} \to \mathbb{R}$ is convex if and only if $\forall x, y \in \mathcal{K}$, we have $f(y) \geq f(x) + \nabla f(x)^T (y - x)$

Def. 1.5.2: α-strongly Convex

1. A *function* f is α-strongly convex if $\forall x, y \in \mathcal{K}$, we have $f(y) \geq f(x) + \nabla f(x)^T (y-x) + \frac{\alpha}{2}||y-x||^2$ or $||\nabla f(x) - \nabla f(y)|| \geq \alpha ||x-y||$ or $f(x) - \frac{\alpha}{2}||x||^2$ is convex.
2. A *twice differentiable function* f is α-strongly convex if $\nabla^2 f(x) \succeq \alpha I$

Def. 1.5.3: β-smooth

1. A *function* f is β-smooth if $\forall x, y \in \mathcal{K}$, we have $f(y) \leq f(x) + \nabla f(x)^T (y-x) + \frac{\beta}{2}||y-x||^2$ or $||\nabla f(x) - \nabla f(y)|| \leq \beta ||x-y||$.
2. A *twice differentiable function* f is β-smooth if $\nabla^2 f(x) \preceq \beta I$
3. If f is β-smooth then $\frac{\beta}{2}||x||^2 - f(x)$ is convex.

The *condition number* of a γ-well-conditioned function is defined to be the ratio between strong convexity and smoothness.

$$\gamma = \frac{\alpha}{\beta} \leq 1$$

$f(y) \geq f(x) + \nabla f(x)^T (y-x) + \frac{\alpha}{2}||y-x||^2$ can be interpreted as fixing a point x. The part of $f(x) + \nabla f(x)^T (y-x)$ is the linear part or tangent of the function $f(x)$ at point x. This inequality shows that the curve of $f(y)$ is *above* the tangent plus a parabola. The analysis of smooth is similar.

Properties 1.5.1: The Properties of Convexity and Smoothness

1. If f_1 is a α_1-strongly convex, f_2 is a α_2-strongly convex. Then $f_1 + f_2$ is $(\alpha_1 + \alpha_2)$-strongly convex in $dom(f_1) \cap dom(f_2)$.
2. If f_1 is a β_1-smooth, f_2 is a β_2-smooth. Then $f_1 + f_2$ is $(\beta_1 + \beta_2)$-smooth in $dom(f_1) \cap dom(f_2)$.
3. **The function $\frac{\alpha}{2}||x - x_1||_2^2$ is α-strongly convex and α-smooth.**

1.6 Cones and Optimality Conditions

This note comes from Lecture 6 of [3].

There are two types of cones (*normal cone* and *tangent cone*) that are important to deduce optimality condition. And have to mention that sometimes the definition uses x to represent vector and sometimes point. x is a point in this note.

Def. 1.6.1: Normal Cone

Given any set $\mathcal{K}$ (**convex or not convex**), and a point $x \in \mathcal{K}$, the *normal cone* of set $\mathcal{K}$ at point x is defined as,

$$\mathcal{N}_{\mathcal{K}}(x) = \{g \mid g^T(y - x) \leq 0, \forall y \in \mathcal{K}\}$$

Def. 1.6.2: Tangent Cone (i.e. polar cone)

Based on the definition of normal cone, the *tangent cone* represents the set of feasible direction that walking along this direction is still possibly landing in set $\mathcal{K}$. Because the definition of normal cone is general including convex or non-convex, the definition of tangent cone is also do.

When $\mathcal{K}$ is a convex set, the *tangent cone* is the polar of the normal cone and is a convex cone, and is defined as,

$$T_{\mathcal{K}}(x) = \mathcal{N}_{\mathcal{K}}(x)^\circ = \{g \mid g^T(z - x) \leq 0, \forall z \in \mathcal{N}_{\mathcal{K}}(X)\}$$

The normal cones at different points of set $\mathcal{K}$ is different, and can be classified into three cases.

1. x inside the set $\mathcal{K}$. The normal cone of it is empty, just as the figure 1.6 shown.

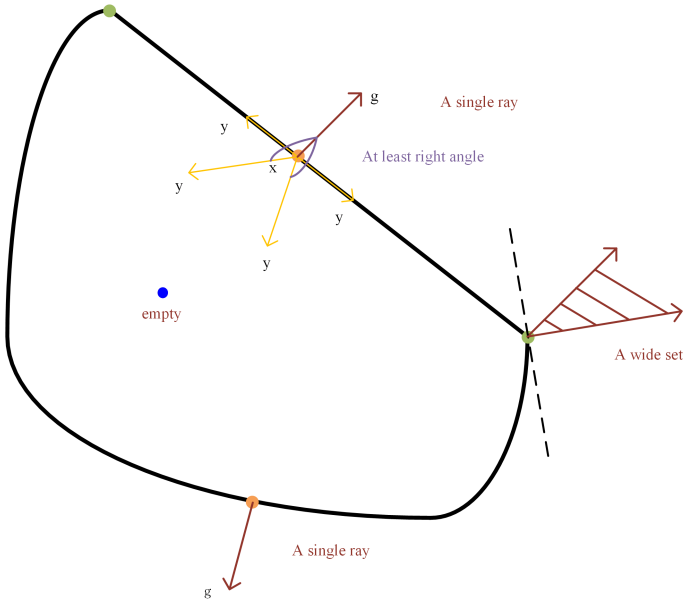

图 1.1: The Explanation of Normal Cone

2. x at the boundary where it is smooth. The normal cone is a single ray, otherwise it does not satisfy at least right angle between g and $(z - x)$.

3. x at the boundary where it is nonsmooth. This situation is a little bit complex. The all one-side tangents will be found, and the normal cone is the cone combined by these one-side tangents.

1.7 Optimality Conditions

This section will use the intuitive sense in section1.6.

Unconstrained Case

1. x^* is optimal $\Leftrightarrow 0 \in \partial f(x^*)$.

Constrained Case

1. x^* is optimal $\Leftrightarrow \nabla f(x^*)^T(y - x^*) \geq 0, \forall y \in \mathcal{K}$

2. x^* is optimal $\Leftrightarrow -\nabla f(x^*) \in \mathcal{N}_{\mathcal{K}}(x^*)$ **(by the definition of normal cone)**

Under the constrained case, the gradient has three possible cases corresponding to the three cases of normal cone at different points.

1. x^* inside the set $\mathcal{K}$. The minus gradient does not belong to $\mathcal{N}_{\mathcal{K}}(x^*)$, because the normal cone is empty. And this problem degenerates to a unconstrained problem, using $\nabla f(x) =$

0 to solve the minimum.

2. x^* at the boundary where it is smooth. The minus gradient is unique.
3. x^* at the boundary where it is nonsmooth. The minus gradient of optimal solution has a wide range.

1.8 Bregman Divergence

This is a kind of method to measure distance, like metric. Bregman divergence is useful when analyzes the convergence rate about norm, because Bregman divergence can use some special properties to simplify proof.

Def. 1.8.1: Bregman Divergence

Assume (a)Ω is a closed convex set, (b)function $f : \Omega \to \mathbb{R}$ is strictly convex, and continuously differentiable. Then the *Bregman Divergence* is defined as,

$$B_f(y, x) = f(y) - \underbrace{(f(x) + \nabla f(x)^T(y - x))}_{\text{the first Taylor expansion of } f \text{ around point } x}$$

Examples 1.8.1: The Bregman Divergence of Some Special Functions

1. **2-Norm**. If $f(x)$ is define $\frac{1}{2}||x||_2^2$, then the Bregman divergence of $f(x)$ is $B_f(y,x) = \frac{1}{2}||y-x||^2$.

2. **General Norm**. If $f(x)$ is define $\frac{1}{2}||x||_q^2$, then the Bregman divergence of $f(x)$ is $B_f(y,x) = \frac{1}{2}||x||_q^2 - \langle \nabla\left(\frac{1}{2}||y||_q^2\right), x\rangle + \frac{1}{2}||y||_q^2$.

proof: We just verify the correctness of the gradient of the general norm.

$$\begin{aligned}\nabla(\frac{1}{2}||y||_q^2) &= \nabla\left(\frac{1}{2}\left(\sum_{i=1}^{d}|y_i|^q\right)^{\frac{2}{q}}\right)\\ &= \begin{pmatrix} \frac{1}{q}\left(\sum_{i=1}^{d}|y_i|^q\right)^{\frac{2}{q}-1}\nabla(|y_1|^q)\\ \frac{1}{q}\left(\sum_{i=1}^{d}|y_i|^q\right)^{\frac{2}{q}-1}\nabla(|y_2|^q)\\ \vdots\\ \frac{1}{q}\left(\sum_{i=1}^{d}|y_i|^q\right)^{\frac{2}{q}-1}\nabla(|y_d|^q)\end{pmatrix}\end{aligned}$$

If all elements y_i are positive, then $\nabla(\frac{1}{2}||y||_q^2)^T y = ||y||_q^2$. **And notice that** $\nabla(\frac{1}{2}||y||_q^2) \neq ||y||_q^2$ **all the time** .

第二章　Gradient-besed Methods

2.1 Projected Gradient, Proximal Gradient and Coordinate Descent Method

Projected Gradient Method

About *projected gradient method*,

1. It is a special case of *proximal gradient method*
2. It solves the constrained problem under gradient method, which cannot solve constrained problem without some techniques. And the update rules are similar with gradient method, only adding the procedure of projecting.

$$x_{k+1} = [x_k - \eta_k \nabla f(x_k)]^+_{\mathcal{K}}$$

Some keypoints in *projected gradient method*,

1. The projection has two situations. If the point $x_k - \eta_k \nabla f(x_k)$ is within the feasible region, then the projection to set $\mathcal{K}$ is itself. If the point $x_k - \eta_k \nabla f(x_k)$ is out of the feasible region, then the projection to set $\mathcal{K}$ is a sub-problem, which is,

$$\begin{aligned} &\min_{y \in \mathcal{K}} ||y - \Big(x_k - \eta_k \nabla f(x_k)\Big)||_2 \\ \Leftrightarrow &\min_{y \in \mathcal{K}} \frac{1}{2} ||y - \Big(x_k - \eta_k \nabla f(x_k)\Big)||_2^2 \end{aligned}$$

2. The difficulties of projected gradient method are the calculation of projection. Sometimes the projection is not unique if the set $\mathcal{K}$ is nonconvex. The projection is easy to calculate if sub-problem has closed-form expression or the set is convex and the problem is easy to deal with.
3. The geometric explanation of projected gradient method. **[TODO]**

4. The ergodic convergence rate of projected gradient method. **[TODO]**

Remark 2.1.1: What Is Ergodic Convergence Rate

The critical role in "Ergodic" convergence rate is ergodic, which means some kinds of average value gradually get closer to the limit point. Or say "The centroid of a point cloud moving towards the limit point"[5].

If an sequence $\{x_1, x_2, ...\}$ is ergodic convergent, then $\lim_{K\to\infty} \frac{1}{K}\sum_{k=1}^{K} x_k - x^* = 0$. It is not equal to all x_k will get closer to x^* gradually, some of them can be moving away from x^*, but the centroid of them gets closer to x^*.

Why Projected Gradient Method is ergodic convergent? projection do not guarantee the decreases of function value.

Proximal Gradient Method

2.2 Mirror Descent

2.3 Subgradient Method

Like the steepest descent method, the subgradient method also has a similar updating formulations.

$$x_{k+1} = [x_k + \alpha_k g_k]_X^+$$

where g_k represents the subgradient of $f(x)$ at x_k. $[\cdot]_M^+$ means the Euclidean projection on set M, which is equal to this form $\arg\min_{x\in M} ||x - \cdot||_2^2$.

Notes 2.3.1: Proper and Properness

Def. 2.3.1

"a proper convex function is an extended real-valued convex function with a non-empty domain, that never takes on the value $-\infty$ and also is not identically equal to $+\infty$" [a].

[a] https://en.wikipedia.org/wiki/Proper_convex_function

This definition is for the situation of **minimizing a function**, and maximization is similar, which only need add minus to become a maximization problem to a minimization problem. If the minimum can be $-\infty$, then the optimal point is definitely the point corresponding to the $-\infty$, and this problem is meaningless. And similarly, if identically equal to $+\infty$, then the minimum point does not exist.

Properties 2.3.1

1. The sum of two proper convex function is convex but maybe proper. **why?***

Def. 2.3.2: Subgradient and Subdifferential

Given a proper convex function $f : \mathbb{R}^n \to (-\infty, +\infty]$, we say that a vector $g \in \mathbb{R}^n$is a subgradient of f at a point $x \in dom(f)$ if,

$$f(z) \geq f(x) + g^T(z - x) \quad , \forall z \in \mathbb{R}^n$$

And all subgradients of a point x is called subdifferetial, i.e. $\partial f(x) = \{g(x) | f(z) \geq f(x) + g^T(z - x) \quad , \forall z \in \mathbb{R}^n\}$.

Explanation: "a proper convex function $f : \mathbb{R}^n \to (-\infty, +\infty]$". Because a proper function cannot be $-\infty$, otherwise the minimum points are obvious and meaningless. And cannot always be $+\infty$,otherwise without minimum points.

Properties 2.3.2

1. Subdifferential is a closed set. The subdifferential is the union of all subgradients, and subgradient is a closed halfspace. The geometric meaning of a subgradient is that

$$\begin{pmatrix} -g \\ 1 \end{pmatrix}^T \begin{pmatrix} z \\ f(z) \end{pmatrix} \geq \begin{pmatrix} -g \\ 1 \end{pmatrix}^T \begin{pmatrix} x \\ f(x) \end{pmatrix}$$

is a closed halfspace $\{z | z$ satisfies the above inequality$\}$. (If normal and any point are known, the halfspace can be expressed.)

2. If $f(x)$ is real-valued, the $\partial f(x)$ is convex, nonempty and compact (closed and bounded).

Notes 2.3.2: The Geometric Explanation of Subgradient Using Definition

The definition of subgradient can be transformed into the following form,

$$f(z) - g^T z \geq f(x) - g^T x \quad , \forall z \in \mathbb{R}^n$$

If there exists a point x_0, then the right side of above formulation is a line with $(g(x_0), -1)$ normal and passing through the point x_0. The left side is equal to $h(z) = f(z) - g^T z, \forall z \in \mathbb{R}^n$, and it is a epigraph (the above part of any function) of f. So this definition can be explained as: if subgradient of one points is up to find, then we just draw some lines through it and the lines that are totally lower than epigraph of $f(z)$ are selected. Then this kind of normal is called a subgradient at this point.

Math is like this, learning something intuitively and deduce them mathematically!

Properties 2.3.3: Subdifferetial and Directional Derivative

Directional Derivative along the direction d at point x is defined as,

$$f^{'}(x;d) = \lim_{\alpha \downarrow 0} \frac{f(x+\alpha d) - f(x)}{\alpha}$$

This is monotonically nonincreasing as $\alpha \downarrow 0$, and proof is given to 2.3. It is worth to notice that this nonincreasing property is related to this part of $\frac{f(x+\alpha d)-f(x)}{\alpha}$ and it is monotonically nonincreasing as α getting closer to 0.

1. **(the relation of subdifferetial and directional derivative)** $f^{'}(x;d) = \max_{g \in \partial f(x)} g^{'} d$ when $\partial f(x)$ is compact, nonempty and convex. So if $f(x)$ is differentiable at x, then the subgradient is unique at x and equal to $\nabla f(x)$.

Properties 2.3.4

1. **(Subdifferential Boundness)**

Properties 2.3.5: Some Critical Convergence Properties

Notes 2.3.3: Some Proofs

Directional erivative is monotonically nonincreasing as $\alpha \downarrow 0$

2.4 Distributed Subgradient Methods [1]

2.4.1 The Origin of Distributed Strategies [2]

To study distributed computation and control/decision tasks, some methods have been proposed. In previous works, each node in undirected network repeatedly updates its value as the linear combination of its previous value and the previous values of its neighbors. That

[1] A. Nedic and A. Ozdaglar, "Distributed Subgradient Methods for Multi-Agent Optimization," in IEEE Transactions on Automatic Control, vol. 54, no. 1, pp. 48-61, Jan. 2009, doi: 10.1109/TAC.2008.2009515

[2] A. D. Domínguez-García and C. N. Hadjicostis, "Distributed strategies for average consensus in directed graphs," 2011 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, FL, USA, 2011, pp. 2124-2129, doi: 10.1109/CDC.2011.6160462.

is why in distributed subgradient method and etc. the update variable x is instituted by the weighted sum of its neighbors values.

Def. 2.4.1: The Links of Graph, Neighbors and In- and Out- Degrees

A distributed system can be conveniently described as a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. The ***link*** $(j, i) \in \mathcal{E}$ means that node j can receive information from node i, which is equal to say that node i is the in-degree node of node j.

The ***neighbors*** of node j is denoted by $\mathcal{N}_j$, which is the set of all nodes that can transmit information to node j.

The number of neighbors of node j is called the ***in-degree*** of node j and is denoted by $D_j^- = |\mathcal{N}_j|$. The ***out-degree*** is similar.

The intuitive meaning of in-degree of node j is that the number of the source of the information, and the out-degree of node j is that the number of transmitting by node j.

The critical procedures are that collecting information from its neighbors and update optimization variables in first step and adjusting the transition matrix (or say *mixing matrix*) by some strategies in second step. So we have the following result,

$$\begin{pmatrix} \pi_1(k+1) \\ \dots \\ \pi_n k+1 \end{pmatrix} = \begin{pmatrix} p_{11} & \dots & p_{1n} \\ \vdots & & \vdots \\ p_{n1} & \dots & p_{nn} \end{pmatrix} \begin{pmatrix} \pi_1(k) \\ \dots \\ \pi_n(k) \end{pmatrix}$$

Define $\pi(k) = \begin{pmatrix} \pi_1(k) & \dots & \pi_n(k) \end{pmatrix}^T$, $P = [p_{ij}]$, where the matrix P has definitely actual meaning. The index of each row and column can be treated as the index of node. The index from column to row represents the direction of transmitting information.

Assumption Node j can only change its self-weight and the weight of its neighbors (i.e. the weight of out-degree links or say the column of matrix P).

The Strategy of Weight Adjustment In order to produce a *column-stochastic* matrix P, the self-weight of node j can be set. And naturally, the other column weight can be set as $p_{ij} = c_{ij}(1 - p_{jj}), \forall j \in \mathcal{N}_j$, where $\sum_{i \in \mathcal{N}_j} c_{ij} = 1$ and $c_{ij} = 0, \forall i \notin \mathcal{N}_j \cup \{j\}, c_{ij} > 0, \forall i \in \mathcal{N}_j$.

By **The Strategy of Weight Adjustment**, the nonnegative column-stochastic matrix property can be easily verified and the matrix is also primitive (i.e. regular, $\exists k \in \mathbb{N}^*, s.t.$ all elements in A^k is bigger than 0). It can be shown that the matrix P will converge to a doubly stochastic matrix P_{ss} as $k \to \infty$. A kind of special selection of P will be shown

and the selection of c_{ij} is also critical.

The Selection of Transition Matrix P

1. At the beginning, set the self-weight of node j to be $1/(1+D_j^+)$, and the weight from node j to other neighbors is $c_{ij}/(1+D_j^+), \forall i \in \mathcal{N}_j$. So the first iteration of matrix P is that,

$$\pi_j(1) = \frac{1}{1+D_j^+}\pi_j(0) + \sum_{i\in\mathcal{N}_j} \frac{c_{ij}}{1+D_j^+}\pi_i(0)$$

Write update of all nodes,

$$\begin{aligned}\pi(1) &= P(0)\pi(0)\\ &= (\overline{P}\Delta(0) + (I-\Delta(0)))\pi(0)\end{aligned}$$

where the matrix $\overline{P} = [c_{ij}]$ is the column-stochastic matrix *corresponding to the algorithm where each node distributes its values among its neighbors*, and $\Delta(0) = diag(\delta_1(0), ..., \delta_n(0))$ is the diagonal matrix with $\delta_j(0) = D_j^+/(1+D_j^+) = 1 - p_{jj}(0)$ entry. Detailedly,

$$\overline{P} = \begin{pmatrix} 0 & c_{12} & \dots & c_{1n} \\ c_{21} & 0 & \dots & c_{2n} \\ \vdots & \vdots & \dots & \vdots \\ c_{n1} & c_{n2} & \dots & 0 \end{pmatrix}$$

Remark 2.4.1: The Meaning of Decomposing Matrix P

$$P(k+1) = \underbrace{\overline{P}\Delta(k)}_{\text{the info. to nbd.}} + \underbrace{(I-\Delta(k))}_{\text{the info. to itself}}$$

So, each optimization variable of node j is,

$$\pi_j(k+1) = \underbrace{\sum_{i\in\mathcal{N}_j}(c_{ji}\delta_i(k))\pi_i(k)}_{\text{the info. to itself}} + \underbrace{(1-\delta_j(k))\pi_j(k)}_{\text{the info. to nbd.}}$$

So, this kind of decomposition divides the effect of node j into two parts, including its self-effect and the effect from neighbors.

In order to make matrix P be a doubly stochastic matrix. **The core is that iterate the rows' update of the matrix P to get closer to 1**. 2. After that, define a critical index $\rho_j(k) = \sum_{i\in\mathcal{N}_j} p_{ji}(k)$, which measures the extend of the row sum of matrix P closing to 1. So

the update rule of $\delta_j(k)$ is,

$$\delta_j(k+1) = \begin{cases} \delta_j(k)\rho_j(k) & \rho_j(k) \leq 1 \\ 1 - \frac{1}{\rho_j(k)}(1 - \delta_j(k)) & \rho_j(k) > 1 \end{cases}$$

Remark 2.4.2: The Idea of Choosing Update Rule $\delta_j(k)$

When $\rho_j(k) \leq 1$, the $\delta_j(k+1)$ will decrease, which means the information to node j is not enough. Then the proportion of information to the neighbors of node j will also decrease, and the information will flow largely into itself. By some lemmas, it can be proofed that this update rule will make ρ larger when $\rho \leq 1$ and make ρ smaller when $\rho > 1$.

Finally, **by this kind of update rules, the $P(k)$ will converge to a doubly stochastic matrix as $k \to \infty$, and $\pi_j, \forall j \in \{1, 2, ..., n\}$ will converge to the average sum of initial points, i.e. $\sum_i x_i/n$.**

第三章　Traditional Second-Order Methods

3.1　Newton Method

3.2　Quasi-Newton Method

This note is concluded by the lecture in [6, 7]. This course is a state-of-art course and introduces online optimization, mirror descent, and etc.

The key idea is to find a appropriate estimation of Hessian matrix, and this estimation is good estimator and cheap to form. This approximate Hessian is only associated with function values and gradient. **The approximate Hessian or its inverse is kept symmetric as well as positive definite[7].**

There are two critical elements in the research Quasi-Newton Methods. Firstly, it is the update rule of approximate Hessian, and how to find a good enough method to approximate. Secondly, secant equation is important to measure the wellness of approximation.

3.2.1　Secant Equation

$$B_{k+1}s_k = y_k$$

where $s_k = \alpha_k p_k, y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$.

3.2.2　Hessian Updating

Rank-One Update

Update formula is $B_{k+1} = B_k + \beta_k u_k u_k^T$, where the parameters β_k, u_k are determined by *secant equation*. By substitute B_{k+1} of $B_{k+1}s_k = y_k$ with $B_{k+1} = B_k + \beta_k u_k u_k^T$, we have,

$$\begin{aligned} &(B_k + \beta_k u_k u_k^T)s_k = y_k \\ \Leftrightarrow &\beta_k u_k \langle u_k, s_k \rangle = y_k - B_k s_k \\ \Leftrightarrow &u_k = \frac{y_k - B_k s_k}{\beta_k \langle u_k, s_k \rangle} \quad (\langle u_k, s_k \rangle \text{ is a constant.}) \end{aligned} \tag{3.1}$$

Multiply a s_k^T on the left, the equation $\beta_k u_k \langle u_k, s_k \rangle = y_k - B_k s_k$ becomes,

$$\beta_k (\langle u_k, s_k \rangle)^2 = \langle s_k, y_k - B_k s_k \rangle \tag{3.2}$$

Substitute the u_k, β_k in $B_{k+1} = B_k + \beta_k u_k u_k^T$ with Formula.3.1, Formula.3.2.

Eventually, the update formula for B_{k+1} is,

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{\langle s_k, y_k - B_k s_k \rangle}$$

But this method is unstable, the size and sign of $\langle s_k, y_k - B_k s_k \rangle$ will effect this algorithm. The positive sign is to guarantee to generate a positive definite matrix B_{k+1}. The small size will lead to numerical difficulties.

Rank-Two Update

3.2.3 DFP(Davidon-Fletcher-Powell)

第四章　Online Optimization

This note is the conclusion of [2].

4.1 Unconstrained and Constrained Gradient Method for Ordinary Problems

The candidate points can be founded in whole space at unconstrained situation. And whatever the points is, it always lands in set. Different from this, the selected points maybe not in set at constrained situation. This causes tiny difference of update formulations.

$$x_{k+1} = x_k - \eta_k \nabla f(x_k) \qquad \text{Unconstrained Situation}$$
$$x_{k+1} = [x_k - \eta_k \nabla f(x_k)]^+_{\mathcal{K}} \qquad \text{Constrained Situation}$$

Properties 4.1.1: The Convergence Rate of Constrained Gradient Descent

For constrained minimization of γ-*well-conditioned functions* and step size $\eta_k = 1/\beta$, the algorithm of projected gradient method converges as,

$$h_{k+1} = f(x_{k+1}) - f(x^*) \leq h_1 e^{-\frac{\gamma k}{4}}$$

The previous proof methods are complex and abundant and have to analyses from scratch, so a new reduction analysis is explored by [**?**]. This **reduction method** will used to derive near-optimal convergence rate for non-strongly convex, or non-smooth and so on.

4.2 First-Order Algorithms for Online Optimization

The problem setting is like that,

$$\min_{x} \sum_{t=1}^{T} f_t(x) \text{ or } \min_{x_i, \forall i \in [T]} \sum_{t=1}^{T} f_t(x_t)$$
$$\text{s.t. } x \in \mathbb{R}^d \text{ or } x_i \in \mathbb{R}^{d_i}, \forall i \in 1, 2, 3, ..., T$$

4.2.1 Online Gradient Descent

This method is proposed by Zinkevich. In this note, this method only considers the situation that there only has one optimization variable.

Theorem 4.2.1: Regret Bound of Online Gradient Descent with Step Size $\eta_t = \frac{D}{G\sqrt{t}}$

Online gradient descent with step size $\eta_t = \frac{D}{G\sqrt{t}}$ guarantees that,

$$Regret_T = \sum_{t=1}^{T} f_t(x_t) - \min_{z \in \mathcal{K}} \sum_{t=1}^{T} f_t(z) \leq \frac{3}{2} GD\sqrt{t}$$

Theorem 4.2.2: General Lower Bound of Regret in the Worst Case

Any online convex optimization algorithm incurs $\Omega(DG\sqrt{t})$ regret in the worst case.

Theorem 4.2.3: The Regret Bound for Online Gradient Descent for Strongly Convex Functions

Online gradient descent with step size $\eta_t = \frac{1}{\alpha t}$ achieves the following guarantee for all T.

$$Regret_T \leq \frac{G^2}{2\alpha}(1 + \log T)$$

The selection of step size will affect the regret bound considerably.

A special case of online convex optimization is the stochastic optimization, which is to minimize a function $f(x)$ with constrain $x \in \mathcal{K}$. In Alg.4.2.1, the notation $\mathcal{O}(x_t)$ means that sample a gradient at point x_t, which is equal to $\tilde{\nabla}_t$, and $\mathbb{E}\{\tilde{\nabla}_t\} = \nabla f(x_t), \mathbb{E}\{||\tilde{\nabla}_t||^2\} \leq G^2$.

Algorithm 1 Stochastic Gradient Descent

Require: $\mathcal{O}, \mathcal{K}, \{\eta_t\}, x_1 \in \mathcal{K}$

Ensure: $\bar{x}_T = \frac{1}{T}\sum_{t=1}^{T} x_t$

1: **for** $t = 1, 2, ..., T$ **do**

2: Let $\tilde{\nabla}_t = \mathcal{O}(x_t)$;

3: Update and Project

$$\begin{aligned} y_{t+1} &= x_t - \eta_t \tilde{\nabla}_t \\ x_{t+1} &= \prod_{\mathcal{K}} y_{t+1} \end{aligned} \tag{4.1}$$

4: **end for**

Theorem 4.2.4: The Convergence of Stochastic Gradient Descent

Alg.4.2.1 with step size $\eta_t = \frac{D}{G\sqrt{t}}$ guarantees,

$$\mathbb{E}\{f(\bar{x}_T)\} \leq \min_{z\in\mathcal{K}} f(z) + \frac{3GD}{2\sqrt{T}}$$

proof:

$$\begin{aligned} &\mathbb{E}\{f(\bar{x}_T)\} - f(x^*) \\ =&\mathbb{E}\{f(\frac{1}{T}\sum_{t=1}^{T} x_t)\} - f(x^*) \\ \leq&\mathbb{E}\{\frac{1}{T}\sum_{t=1}^{T} f(x_t)\} - f(x^*) \text{ (the convexity of function } f) \\ =&\frac{1}{T}\mathbb{E}\{\sum_{t=1}^{T}(f(x_t)) - Tf(x^*)\} \\ =&\frac{1}{T}\mathbb{E}\{\sum_{t=1}^{T}(f(x_t) - f(x^*))\} \\ \leq&\frac{1}{T}\mathbb{E}\{\sum_{t=1}^{T}(\nabla f(x_t)^T(x_t - x^*))\} \quad (f(x^*) \leq f(x_t) + \nabla f(x_t)^T(x^* - x_t)) \\ =&\frac{1}{T}\mathbb{E}\{\sum_{t=1}^{T}(\tilde{\nabla}_t^T(x_t - x^*))\} \text{Frame.4.2.1} \\ =&\frac{1}{T}\mathbb{E}\{\sum_{t=1}^{T}(f_t(x_t) - f_t(x^*))\} \text{Frame.4.2.1} \\ =&\frac{\mathbb{E}\{Regret_T(f_t(x_t))\}}{T} \\ \leq&\frac{3}{2}GD\sqrt{T} \quad (\text{the } G, D \text{ is belong to the function } f_t(z) = \tilde{\nabla}_t^T z) \end{aligned}$$

"That is the magic of SGD; we have matched the nearly optimal convergence rate of first order methods using extreme cheap iterations." [2] using extremely cheap iterations.

$$
\begin{aligned}
&\frac{1}{T}\mathbb{E}\{\sum_{t=1}^{T}(\nabla f(x_t)^T(x_t-x^*))\}x_t \in \mathbb{R}^d \\
=&\frac{1}{T}\mathbb{E}\{\sum_{t=1}^{T}\sum_{j=1}^{d}[\nabla f(x_t)]_j[(x_t-x^*)]_j\} \\
=&\frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{d}\mathbb{E}\{[\nabla f(x_t)]_j[(x_t-x^*)]_j\} \\
=&\frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{d}\mathbb{E}\{[\tilde{\nabla}_t]_j[(x_t-x^*)]_j\} \\
=&\frac{1}{T}\mathbb{E}\{\sum_{t=1}^{T}(\tilde{\nabla}_t^T(x_t-x^*))\}
\end{aligned}
\tag{4.2}
$$

Firstly define $f_t(z) = \tilde{\nabla}_t^T z$. The regret of it is,

$$Regret_T(f_t(z)) = \sum_{t=1}^{T} f_t(z) - \min_{x^*\in\mathcal{K}}\sum_{t=1}^{T} f_t(x^*)$$

So, $Regret_T(f_t(x_t)) = \sum_{t=1}^{T}(f_t(x_t) - f_t(x^*))$. And according to Thm.4.2.1,

$$Regret_T(f_t(x_t)) \le \frac{3}{2}GD\sqrt{T}$$

Consider its expectation,

$$\mathbb{E}\{Regret_T(f_t(x_t)) = \sum_{t=1}^{T}\mathbb{E}\{f_t(x_t)\} - \sum_{t=1}^{T}\mathbb{E}\{f_t(x^*)\} = \sum_{t=1}^{T}\nabla f(x_t)^T(x_t-x^*)$$

4.3 Second-Order Methods for Online Optimization

4.4 Regularization

4.5 Bandit Convex Optimization

The only difference between *Bandit Convex Optimization(BCO)* and *Online Convex Optimization(OCO)* is that *BCO* is value oracle but *OCO* is gradient and value oracle. **Bandit algorithm is a kind of policy to balance exploration and exploitation**[8]. For example, a citizen bought a Lottery and won 10 dollars (the reward is 10 dollars), and next time if he sticks to buy this, the exploration is not enough. Because a better lottery may appears.

Remark 4.5.1: What is "oracle"?

Assume a network with some nodes, these nodes maintains some local functions. Different with OCO, these local functions in BCO setting likes a machine, which will return value rather than gradient. Such situations are normal, for trying selling items, we only know the prices each time, and do not know the gradient because the price is not a definite function. There are so many oracle models, such as,

1. Value oracle. Given x, and return $f(x)$.
2. Gradient oracle. Given x, and return $\nabla f(x)$.
3. First-order oracle. Given x, and return $f(x), \nabla f(x)$.
4. Stochastic oracle. Given x, and return $\nabla f(\tilde{x}), \{\nabla f(\tilde{x})\} = \mathbb{E}\{\nabla f(x)\}$.

第五章　Gossip-based Computation

第六章　Second-order Methods

6.1　Newton's Method and Damped Newton's Method

This classification may be not right.

第七章　Latex Notes

7.1　Instructions

\protect

Sometimes if some vulnerable instructions are used, errors will occur. So add this instruction to the front of them. Such as adding footnote in brackets of \section.

abc

第八章　Github Notes

The Difference Between Git and Github Git is a version control system, Github is a platform which provides service of git for users.

参考文献

[1] None. (None) Directional derivatives and the gradient vector. [Online]. Available: https://math.berkeley.edu/~arash/53/notes/14_6.pdf

[2] E. Hazan, "Introduction to online convex optimization," 2023. [Online]. Available: https://arxiv.org/abs/1909.05207

[3] M. Gormley. (2023) Introduction to convex optimization. [Online]. Available: https://www.cs.cmu.edu/~mgormley/courses/10425/

[4] A. Sidford. (2017) Msande213cs269o:mathematical optimization. [Online]. Available: https://www.studocu.com/en-us/document/missouri-university-of-science-and-technology/convex-optimization/sidford-mse213-2019-fa-chap-1-intro/59892783?origin=university-course-page

[5] A. Ang. (2024) Projected gradient algorithm. [Online]. Available: https://angms.science/doc/CVX/CVX_PGD.pdf

[6] Y. Chen. (2024) Cs/isye/math/stat 726 nonlinear optimization i (spring 2024). [Online]. Available: https://pages.cs.wisc.edu/~yudongchen/cs726_sp24/

[7] J. S. Arora. (2004) Computational methods for unconstrained minimization. [Online]. Available: https://user.engineering.uiowa.edu/~design1/53-235%20AOD/

[8] 安室透. (2021) bandit 知识分享与总结. [Online]. Available: https://zhuanlan.zhihu.com/p/442717115