Matrix Algebra and Optimization for Statistics and Machine Learning

Yiyuan She

Department of Statistics, Florida State University

- Matrix differentiation

Kronecker product and the vec operator

- Given $A \in \mathbb{R}^{n \times p}$ and $B \in \mathbb{R}^{m \times q}$, $A \otimes B$ is an $nm \times pq$ matrix defined by

$$\begin{bmatrix} a_{11}B & \dots & a_{1p}B \\ \vdots & & \vdots \\ a_{n1}B & \dots & a_{np}B \end{bmatrix}$$

- $\mathrm{vec}(A)$ is obtained by stacking the columns of A:

$$[a_{11}, \dots, a_{n1}, \dots, a_{1p}, \dots, a_{np}]^T$$

Properties

- Some basics: $(A \otimes B) \otimes C = A \otimes (B \otimes C)$, $(A + B) \otimes (C + D) = A \otimes C + A \otimes D + B \otimes C + B \otimes D$, $a \otimes A = A \otimes a = aA$
- Vector outer-product can be viewed as a special case:

$$\alpha \otimes \beta^T = \alpha\beta^T = \beta^T \otimes \alpha$$

- Although $A \otimes B$ and $B \otimes A$ do not equal in general, there exist permutations so that $P(A \otimes B)Q = B \otimes A$

- Mixed-product property:

$$(A \otimes B)(C \otimes D) = AC \otimes BD$$

  where conformability is assumed
    - The RHS is often much more efficient to compute
    - Assume all are square: $2n^4 + n^6$ vs. $2n^3 + n^4$
- Let $A \in \mathbb{R}^{m \times m}$, $B \in \mathbb{R}^{n \times n}$. From the above property, $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$.
    - It actually holds generally for the MP inverse

- In addition, $(A \otimes B)^T = A^T \otimes B^T$
    - **No change in order**
    - A special case helps to remember: $A = I$
    - [Check the 2nd argument first when seeing $\otimes$]
- $|A \otimes B| = |A|^n |B|^m$ (a nice result)
- $tr(A \otimes B) = tr(A)tr(B)$

- $\operatorname{vec}(\alpha\beta^T) = \beta \otimes \alpha$ for any vectors α, β
- More generally,

$$\operatorname{vec}(AXB^T) = (B \otimes A) \operatorname{vec}(X)$$

- Let $A \in \mathbb{R}^{m \times n}$. Then

$$K_{m,n} \operatorname{vec}(A) = \operatorname{vec}(A^T)$$

where $K_{m,n}$ is the **commutation** matrix.

- Let $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{p \times q}$. Then

$$K_{p,m}(A \otimes B)K_{n,q} = B \otimes A$$

- In fact, for any $X \in \mathbb{R}^{q \times n}$, $K_{p,m}(A \otimes B)\,\mathrm{vec}X = K_{p,m}\,\mathrm{vec}(BXA^T) = \mathrm{vec}(AX^TB^T) = (B \otimes A)\,\mathrm{vec}(X^T) = (B \otimes A)K_{q,n}\,\mathrm{vec}X$, and $K_{n,m}^{-1} = K_{n,m}^T = K_{m,n}$.

Partial derivatives and the Jacobian matrix

- Given a (smooth) vector function $f : S \to \mathbb{R}^m$ with $S \subset \mathbb{R}^n$, the Jacobian matrix of f at x is

$$\mathrm{D}f(x) = [d_{i,j}] = [\,\mathrm{D}_j f_i(x)] \in \mathbb{R}^{m\times n},$$

  where $\mathrm{D}_j f_i(x)$ is the partial derivative $\frac{\partial f_i(x)}{\partial x_j}$
- The gradient $\nabla f(x) = (\,\mathrm{D}f(x))^T$ (but we will often use a *non-vectorized* version in <u>matrix</u> optimization)
- When f is **real**-valued, $\nabla f(x)$ has the **same** size as x; here we also denote $\nabla f(x)$ by $\frac{\partial f}{\partial x}$ (derivative: $\frac{\partial f(x)}{\partial x^T}$)

Differential of a vector function

- Again assume $f : S \to \mathbb{R}^m$ with $S \subset \mathbb{R}^n$ and let x be an interior point of S. f is differentiable at x if

  $$f(x + \mathrm{d}x) - f(x) = \mathrm{d}f(x; \mathrm{d}x) + o(\mathrm{d}x), \forall \mathrm{d}x \in S \cap B(0; r)$$

  with $\mathrm{d}f(x; \mathrm{d}x) = A(x)\,\mathrm{d}x$, $\lim_{\mathrm{d}x \to 0} o(\mathrm{d}x)/\|\mathrm{d}x\| = 0$
- $A(x)$: **derivative** of f at x
- If $\mathrm{D}_j f_i(\cdot x)$ exist in some neighborhood of c and are continuous at $x = c$, then f is differentiable at c and

  $$A(c) = \mathrm{D}f(c) =: \left.\frac{\partial f(x)}{\partial x^T}\right|_{x=c}$$

Matrix functions

- For a matrix function $F : S \to \mathbb{R}^{m\times p}$ with $S \subset \mathbb{R}^{n\times q}$, we define its Jacobian matrix at X as

$$\mathrm{D}F(X) = \frac{\partial\, \mathrm{vec} F(X)}{\partial(\,\mathrm{vec}(X))^T}$$

- That is, if we introduce $f(\,\mathrm{vec} X) = \,\mathrm{vec} F(X)$, $\mathrm{D}F(X) = \,\mathrm{D}f(\,\mathrm{vec} X)$

- Similarly, we apply vec to define differentiability:

$$F(X + \mathrm{d}X) - F(X) = \mathrm{d}F(X; \mathrm{d}X) + o(\mathrm{d}X), \text{or}$$
$$\mathrm{vec}F(X + \mathrm{d}X) - \mathrm{vec}F(X) = \mathrm{vec}\,\mathrm{d}F(X; \mathrm{d}X) + o(\mathrm{d}X),$$

  where $\mathrm{vec}\,\mathrm{d}F(X; \mathrm{d}X) = A(X)\,\mathrm{vec}(\mathrm{d}X)$
- $\mathrm{d}F(X; \mathrm{d}X) \in \mathbb{R}^{m\times p}$ is called the **differential** of F at X with increment $\mathrm{d}X$
- $A(X) \in \mathbb{R}^{mp\times nq}$ is called the derivative of F at X

The chain rule and Cauchy invariance

- Consider a smooth (vector) function $h(t) = g \circ f(t) = g(f(t))$ at t and $x = f(t)$
- Then $\mathrm{D}h(t) = \mathrm{D}g(x)\, \mathrm{D}f(t)$ (matrix product)
- It means that $\mathrm{d}h(t;\, \mathrm{d}t) = \mathrm{d}g(x;\, \mathrm{d}f(t;\, \mathrm{d}t))$ for any $\mathrm{d}t$ (LHS $= \mathrm{D}g(x)\, \mathrm{D}f(t)\, \mathrm{d}t = \mathrm{D}g(x)\, \mathrm{d}f(t;\, \mathrm{d}t) =$ RHS)
- We will use the shorthand notation $\mathrm{d}h$ from now on

A practical differentiation scheme

- Let $F(X)$ be a matrix function of X that is smooth
  1. Get the differential $\mathrm{d}F$ as a linear function of $\mathrm{d}X$
  2. Vectorize $\mathrm{d}F$ as a linear transformation of $\mathrm{d}\,\mathrm{vec}X$
- The transformation matrix gives all partial derivatives or the Jacobian matrix $\mathrm{D}F(X)$

An illustration

- Let $F(X) = AXB$.
- Then $\mathrm{d}F(X) = A(\mathrm{d}X)B$
- Vectorization: $\mathrm{d}\,\mathrm{vec}F(X) = (B^T \otimes A)\,\mathrm{d}\,\mathrm{vec}X$
- Hence $\mathrm{D}F(X) = B^T \otimes A$

Scalar functions

- In optimization, we typically deal with a real-valued objective function ϕ.
- Knowing its gradient often suffices (cf. the 1st step)
- Here, it is more convenient to use $\frac{\partial \phi(X)}{\partial X} := [\frac{\partial \phi(X)}{\partial X_{i,j}}]$
- We still call it the gradient $\nabla \phi(X)$ (abuse of notation!)
- [If you want to use a Newton-type algorithm, applying the vectorization step could be helpful]

Inner-product form of differential

- Concretely, when ϕ is a scalar function defined on a matrix X, we often try to express the differential as

$$\mathrm{d}\phi(X) = \langle A, \mathrm{d}X \rangle$$

- Then $\nabla\phi(X)(= \frac{\partial\phi(X)}{\partial X}) = A$, while $\mathrm{D}\phi(X) = (\mathrm{vec}A)^T$ (as $\mathrm{d}\phi(X) = (\mathrm{vec}A)^T \mathrm{vec}X$) gives $\nabla\phi(X) = \mathrm{vec}A$!
- Note the advantage of representing a problem in terms of a matrix in algorithm design and in implementation

Differential calculus

- Some properties:
  - $\mathrm{d}(U+V) = \mathrm{d}U + \mathrm{d}V$
  - $\mathrm{d}(UV) = (\mathrm{d}U)V + U(\mathrm{d}V)$
  - $\mathrm{d}(U^T) = (\mathrm{d}U)^T$, $\mathrm{d}\,\mathrm{vec}U = \mathrm{vec}\,\mathrm{d}U$, $\mathrm{d}tr(U) = tr\,\mathrm{d}U$
  - $\mathrm{d}(U \otimes V) = (\mathrm{d}U) \otimes V + U \otimes (\mathrm{d}V)$
- A few useful facts:
  - $\mathrm{d}Ax = A\,\mathrm{d}x$, $\nabla(Ax) = A^T$
  - $\mathrm{d}|X| = \langle |X|(X^T)^{-1}, \mathrm{d}X\rangle$ (use the adjoint matrix)
  - $\mathrm{d}X^{-1} = -X^{-1}(\mathrm{d}X)X^{-1}$

- We show the last result assuming X^{-1} is differentiable
- First we show the conclusion by definition

$$
\begin{aligned}
&(X+\mathrm{d}X)^{-1}-X^{-1}\\
=&(UDV^T+\mathrm{d}X)^{-1}-VD^{-1}U^T\\
=&VD^{-1/2}\{(I+D^{-1/2}U^T(\mathrm{d}X)VD^{-1/2})^{-1}-I\}D^{-1/2}U^T\\
=&VD^{-1/2}\{I-D^{-1/2}U^T(\mathrm{d}X)VD^{-1/2}-I\}D^{-1/2}U^T+o(\mathrm{d}X)\\
=&-X^{-1}(\mathrm{d}X)X^{-1}+o(\mathrm{d}X)
\end{aligned}
$$

- We can also use $XX^{-1}=I \Rightarrow \mathrm{d}(XX^{-1})=0 \Rightarrow$ $(\mathrm{d}X)X^{-1}+X(\mathrm{d}X^{-1})=0 \Rightarrow \mathrm{d}X^{-1}=-X^{-1}(\mathrm{d}X)X^{-1}$

Examples

- $\phi(x) = x^T A x$:
$$\begin{aligned} \mathrm{d}\phi(x) &= \mathrm{d}(x^T A x) = (\mathrm{d}x)^T A x + x^T A\,\mathrm{d}x \\ &= x^T (A^T + A)\,\mathrm{d}x \Rightarrow \mathrm{D}\phi(x) = x^T (A + A^T) \end{aligned}$$
- $\phi(x) = \|y - Ax\|_2^2/2$: Let $e = y - Ax$, $\phi(x) = e^T e/2$. $\mathrm{d}\phi(x) = (y - Ax)^T \mathrm{d}(y - Ax) = (Ax - y)^T A\,\mathrm{d}x$. Therefore, $\nabla\phi(x) = x^T (Ax - y)$

- $\phi(x) = \exp(x^T x)$:

$$\begin{aligned} \mathrm{d}\phi(x) &= \exp(x^T x)\,\mathrm{d}(x^T x) = \exp(x^T x)((\,\mathrm{d}x)^T x + x^T\,\mathrm{d}x) \\ &= 2\exp(x^T x)x^T\,\mathrm{d}x \end{aligned}$$

- GLM: $\min_\beta l(\beta) = -\langle y, X\beta\rangle + \langle 1, b(X\beta)\rangle$

$$\begin{aligned} \mathrm{d}l(\beta) &= -\langle y, X\,\mathrm{d}\beta\rangle + \langle 1, b'(X\beta)\circ(X\,\mathrm{d}\beta)\rangle \\ &= -\langle X^T y,\,\mathrm{d}\beta\rangle + \langle b'(X\beta), X\,\mathrm{d}\beta\rangle \\ &= \langle X^T(b'(X\beta) - y),\,\mathrm{d}\beta\rangle \end{aligned}$$

- $\phi(X) = tr(X)$: $\nabla\phi(X) = I$
- $\phi(X) = \log|X|$:

$$\mathrm{d}\phi(X) = |X|^{-1}\langle|X|(X^T)^{-1}, \mathrm{d}X\rangle = \langle(X^T)^{-1}, \mathrm{d}X\rangle$$
$$\Rightarrow \nabla(\log|X|) = (X^T)^{-1}$$

- $F(x) = xx^T$ (a matrix function defined on a vector)

$$\begin{aligned}\mathrm{d}(xx^T) =& (\mathrm{d}x)x^T + x(\mathrm{d}x)^T \text{(combine? vec!)}\\ =& (x \otimes I)\,\mathrm{d}\,\mathrm{vec}x + (I \otimes x)\,\mathrm{d}\,\mathrm{vec}(x^T)\\ =& (x \otimes I + I \otimes x)\,\mathrm{d}\,\mathrm{vec}x\end{aligned}$$

(What if x is replaced by X?)

- $F(X) = X$ $(X \in \mathbb{R}^{n\times q})$: $\mathrm{d}F(X) = \mathrm{d}X \Rightarrow$ $\mathrm{d}\,\mathrm{vec}F(X) = I_{nq}\,\mathrm{d}\,\mathrm{vec}X \Rightarrow \mathrm{D}F(X) = I_{nq}$
- $F(X) = X^T$ $(X \in \mathbb{R}^{n\times q})$: $\mathrm{d}F(X) = \mathrm{d}X^T \Rightarrow$ $\mathrm{d}\,\mathrm{vec}F(X) = \mathrm{d}\,\mathrm{vec}(X^T) = K_{n,q}\,\mathrm{d}\,\mathrm{vec}X \Rightarrow$ $\mathrm{D}F(X) = K_{n,q}$
- $F(X) = X^{-1}$: $\mathrm{d}F(X) = -X^{-1}(\mathrm{d}X)X^{-1} \Rightarrow$ $\mathrm{d}\,\mathrm{vec}F(X) = -(X^T)^{-1} \otimes X^{-1}\,\mathrm{d}\,\mathrm{vec}X$

Multivariate meta-analysis (regression)

- Assume a random effects meta-regression model with multiple outcomes: $y_i = X_i\beta + \epsilon_i + \delta_i$, $i = 1, \ldots, m$
- $y_i \in \mathbb{R}^n$: n outcomes from m **studies**. $X_i \in \mathbb{R}^{n \times p}$: design matrices containing study-level predictors
- $\epsilon_i \sim \mathcal{N}(0, \Sigma_i)$, $\delta_i \sim \mathcal{N}(0, \Sigma)$, $\epsilon_i \perp \delta_j$
- Σ_i: within-study covariances (known)
- Goal: Estimate β (fixed) as well as the between-study covariance Σ from the m studies

- MLE: $\min_{\beta,\Sigma\succeq 0} \sum_{i=1}^m \log\det(\Sigma_i + \Sigma) + \sum_{i=1}^m (y_i - X_i\beta)^T(\Sigma_i + \Sigma)^{-1}(y_i - X_i\beta)$ (Q: regularization?)
- We will assume β is **known** for simplicity (o/w: BCD)
- Let $R_i = (y_i - X_i\beta)(y_i - X_i\beta)^T$.

$$\min_{\Sigma\succeq 0} f(\Sigma) = \sum_{i=1}^m \log\det(\Sigma_i + \Sigma) + \sum_{i=1}^m tr((\Sigma_i + \Sigma)^{-1}R_i)$$

- A special case: $\Sigma_i = 0$ and $\Theta = \Sigma^{-1}$
- The problem is not convex and has a psd constraint

- We do not consider the constrained optimization for now. Let's use it to practice our differential skills.
- The gradient equation is

$$\sum(\Sigma+\Sigma_i)^{-1}-\sum(\Sigma+\Sigma_i)^{-1}R_i(\Sigma+\Sigma_i)^{-1}=0, \text{ or}$$
$$\sum(\Sigma+\Sigma_i)^{-1}(\Sigma+\Sigma_i-R_i)(\Sigma+\Sigma_i)^{-1}=0.$$

- <u>Assuming</u> Σ_i are not very different, $\sum \Sigma+\Sigma_i-R_i\approx 0$
- **GLS** (Berkey et al 96): $\Sigma=\sum(R_i-\Sigma_i)/m$ (given β)
- [But is the assumption reasonable in general? Does the estimate satisfy the psd constraint?]

- We could use Newton's method, which requires the gradient information of

$$g(\Sigma) = \sum (\Sigma + \Sigma_i)^{-1} - \sum (\Sigma + \Sigma_i)^{-1} R_i (\Sigma + \Sigma_i)^{-1}$$

- What is $H^{-1}g$ (no PSD constraint, no stepsize)? We did not introduce vech and Hessian..
- Using the differentiation technique, $\mathrm{d}g = -\sum(\Sigma + \Sigma_i)^{-1}\,\mathrm{d}\Sigma(\Sigma + \Sigma_i)^{-1} + \sum(\Sigma + \Sigma_i)^{-1}\,\mathrm{d}\Sigma(\Sigma + \Sigma_i)^{-1} R_i (\Sigma + \Sigma_i)^{-1} + \sum(\Sigma + \Sigma_i)^{-1} R_i (\Sigma + \Sigma_i)^{-1}\,\mathrm{d}\Sigma(\Sigma + \Sigma_i)^{-1}$

- So the second-order approximation of $f(\Sigma + \Delta)$ is

$$\sum tr\{(\Sigma + \Sigma_i)^{-1}\Delta\} - tr\{(\Sigma + \Sigma_i)^{-1}R_i(\Sigma + \Sigma_i)^{-1}\Delta\}$$
$$- \frac{1}{2}tr\{(\Sigma + \Sigma_i)^{-1}\Delta\Sigma(\Sigma + \Sigma_i)^{-1}\Delta\}$$
$$+ tr\{(\Sigma + \Sigma_i)^{-1}\Delta(\Sigma + \Sigma_i)^{-1}R_i(\Sigma + \Sigma_i)^{-1}\Delta\}$$

- The solution for the update Δ satisfies

$$\sum(\Sigma + \Sigma_i)^{-1}[\Delta - R_i(\Sigma + \Sigma_i)^{-1}\Delta - \Delta(\Sigma + \Sigma_i)^{-1}R_i](\Sigma + \Sigma_i)^{-1}$$
$$= \sum(\Sigma + \Sigma_i)^{-1}[\Sigma + \Sigma_i - R_i](\Sigma + \Sigma_i)^{-1},$$

- Now we can vectorize it to solve a linear system

- When Σ_i are not very different, we have

$$\sum(\Delta - R_i(\Sigma+\Sigma_i)^{-1}\Delta - \Delta(\Sigma+\Sigma_i)^{-1}R_i) \approx \sum(\Sigma+\Sigma_i -$$

- It is an example of the **Lyapunov** equation

$$A\Delta + \Delta A^T + Q = 0$$

with $A = \sum\left(R_i(\Sigma+\Sigma_i)^{-1} - \frac{1}{2}I\right)$, $Q = \sum(\Sigma+\Sigma_i - R_i)$

- There are standard algorithms (often treating it as a special Sylvester equation). Matlab: Δ `= lyap(A,Q)`.

Subgradients and subdifferential

- Another very useful concept is sudifferential
- Let f be a real-valued convex function. $g(x)$ is a subgradient of f at x if

$$f(y) \geq f(x) + \langle g(x), y - x \rangle, \forall y$$

- Subdifferential $\partial f(x)$: the set of all subgradients at x
- Why studying subgradients?
  - Nicely, if f is convex, $\partial f(x)$ is never null
  - If x is a minimizer of a convex f, then $0 \in \partial f(x)$
  - Finding just one subgradient is often easy

Two examples

- $f(x) = |x|$:

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

- $f(X) = \|X\|_*$. Let $X = UDV^T$ (compact form)

$$\begin{aligned} \partial \|X\|_* &= \{UV^T + W : U^T W = 0, WV = 0, \|W\|_2 \leq 1\} \\ &= \{UV^T + U_\perp Z V_\perp^T : \|Z\|_2 \leq 1\} \end{aligned}$$

- Quite useful in computation and in theory

Soft-thresholding

- For example, let's consider a simple lasso problem $\min_\beta \frac{1}{2}\|y-\beta\|_F^2 + \lambda\|\beta\|_1$ ($\lambda \geq 0$, $X = I$)
- Let β^o be the optimal solution. Then it satisfies

$$\beta - y + \lambda\partial\|\beta\|_1 \ni 0, \text{ or}$$
$$\beta - y + \lambda s(\beta) = 0 \text{ for some } s(\beta) \in \partial\|\beta\|_1$$

- If $\beta_j^o > 0$, then $y_j - \lambda = \beta_j^o$ (and so $y_j > \lambda$); if $\beta_j^o < 0$, $y_j + \lambda = \beta_j^o$ (and $y_j < \lambda$); if $\beta_j^o = 0$, we know $|y_j| \leq \lambda$
- Hence $\beta^o = \Theta(y;\lambda)$ where $\Theta(t;\lambda) = 1_{|t|>\lambda}(t - \text{sgn}(t)\lambda)$.

Singular-value (soft)thresholding

- Now consider the problem $\min_B \frac{1}{2}\|Y - B\|_F^2 + \lambda\|B\|_*$
- Let B_o be the optimal solution with $B_o = UD_oV$. Then $B_o - Y + \lambda UV^T + \lambda U_\perp Z V_\perp^T = 0$ for some $\|Z\|_2 \leq 1$, or

$$
\begin{aligned}
Y =& U(D_o + \lambda I)V^T + U_\perp \lambda Z V_\perp^T \\
=& [U \;\; U_\perp U_z] \begin{bmatrix} D_o + \lambda I & \\ & \lambda D_z \end{bmatrix} [V \;\; V_\perp V_z]^T
\end{aligned}
$$

- It implies that $B_o = \Theta^\sigma(Y; \lambda) = U_y \Theta(D_y; \lambda) V_y^T$ (well defined!), by (soft-)thresholding Y's singular values
- These are examples of **proximity** operators

Subgradient update

- If $f(x^+) < f(x)$ (convex), then $0 \le f(x^+) - f(x) - \langle g(x), x^+ - x \rangle < -\langle g(x), x^+ - x \rangle$ or $\langle g(x), x^+ - x \rangle < 0$, $\forall g(x) \in \partial f(x)$
- How about a subgradient update $x^+ = x - \alpha g(x)$?
  - $f(x^+) < f(x)$ may not hold; even $\lim_{\alpha \downarrow 0} \frac{1}{\alpha}[f(x^+(\alpha)) - f(x)] \le 0$ may not be true (f may not be in $\mathcal{C}^{(1)}$)
- We often apply subgradient algorithms on the dual.
- Subgradient methods can be slow, but when $\partial f(x)$ is a singleton, we get faster convergence (why?)

- In handling the dual problem, we often need to know the first-order information of $d(\lambda) = \sup_{x \in A} L(x; \lambda)$
- Then $\partial f(\lambda)$ is the closure of

$$\textbf{conv} \cup \{\partial L(x; \cdot) : L(x; \lambda) = d(\lambda), x \in A\}$$

- When $\sup_{x \in A} L(x; \lambda)$ has a unique solution, d is differentiable at λ!
    - ADMM uses the fact (actually, a variant) to update the dual variable efficiently

Example

- We know $f(X) = \lambda_{\max}(X)$ ($X \in \mathbf{S}^n$) is convex
- What is its subdifferential?
  1. $f(X) = \sup_{\|\alpha\|_2^2=1} \alpha^T X \alpha$
  2. $\mathrm{d}(\alpha^T X \alpha) = \mathrm{d}tr(\alpha^T X \alpha) = tr\, \mathrm{d}(\alpha\alpha^T X) = \langle \alpha\alpha^T, \mathrm{d}X \rangle$
  3. $\partial f(X) = \mathbf{conv}\{\alpha\alpha^T : \alpha^T X \alpha = \|X\|_2, \|\alpha\|_2^2 = 1\}$
- So for any eigenvector α (with norm 1) associated with the largest eigenvalue of X, $\alpha\alpha^T$ gives a subgradient
- What if the geometric multiplicity of $\lambda_{\max}(X)$ is 1?
- What about $f(X) = \|X\|_2$ for any X ($X \neq 0$, $X = 0$)?