EECS 127/227AT Optimization Models in Engineering UC Berkeley Spring 2024

Discussion 12

We might not cover all the problems on the worksheet, as discussion worksheets are not always designed to be finished within an hour. However, this is totally fine since they are deliberately made slightly longer so they can serve as resources you can use to practice, reinforce, and build upon concepts covered in lectures and homework.

1. Squaring SOCP Constraints

When considering a second-order cone (SOC) constraint, you might be tempted to square it to obtain a classical convex quadratic constraint. This problem explores why that might not always work, and how to introduce additional constraints to maintain equivalence and convexity.

(a) For $\vec{x} \in \mathbb{R}^2$, consider the constraint

$$x_1 - 2x_2 \geq \|\vec{x}\|_2\,, \tag{1}$$

and its squared counterpart

$$(x_1 - 2x_2)^2 \geq \|\vec{x}\|_2^2\,. \tag{2}$$

Are the two sets equivalent? Are they both convex?

Solution: The set defined by the first constraint is an SOC constraint and is thus convex. The second, squared constraint

$$(x_1 - 2x_2)^2 \geq \|\vec{x}\|_2^2 \tag{3}$$

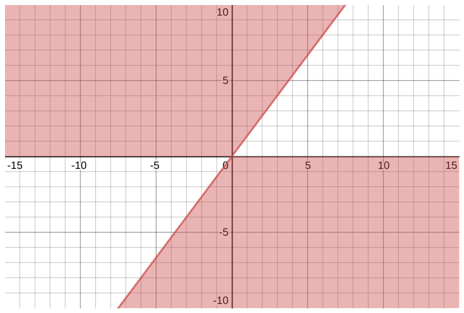
can be expressed as

$$x_1^2 - 4x_1x_2 + 4x_2^2 \geq x_1^2 + x_2^2, \tag{4}$$

and thus

$$x_2(4x_1 - 3x_2) \leq 0. \tag{5}$$

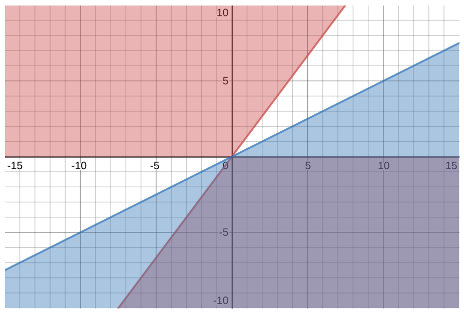
Plotting this region (*red*), we can immediately see that it is not convex:



and thus cannot possibly be equivalent to the first, unsquared constraint.

(b) What additional constraint must be imposed alongside the squared constraint to enforce the same feasible set as the unsquared SOC constraint?

Solution: The initial SOC constraint $x_1 - 2x_2 \geq \|\vec{x}\|_2$ implicitly imposes that $x_1 - 2x_2 \geq 0$, a condition that is lost when both sides of the inequality are squared. We can recover the same feasible set (*purple*) by imposing this constraint (*blue*) alongside $(x_1 - 2x_2)^2 \geq \|\vec{x}\|_2^2$ (*red*):



This principle holds in general; it is indeed possible to square an SOC constraint

$$\left\|A\vec{x}+\vec{b}\right\|_2 \leq \vec{c}^\top\vec{x}+d, \tag{6}$$

provided one takes care to include the implicit constraint that $\vec{c}^\top\vec{x}+d \geq 0$. The general SOC constraint is thus equivalent to

$$\left\|A\vec{x}+\vec{b}\right\|_2^2 \leq (\vec{c}^\top\vec{x}+d)^2 \tag{7}$$

$$0 \leq \vec{c}^\top\vec{x}+d. \tag{8}$$

Note, however, that this is often of limited use, as certain convex optimization solvers (e.g., CVX) may not accept constraints of the form $\left\|A\vec{x}+\vec{b}\right\|_2^2 - (\vec{c}^\top\vec{x}+d)^2 \leq 0$ in general, since, as shown above, the difference of two convex quadratic functions may well be nonconvex. Although you may know that your problem is convex — i.e., that you've provided the requisite additional constraints — your solver might not be that smart!

2. Newton's Method for Quadratic Functions

Give a symmetric positive defininte matrix $Q \in \mathbb{S}^n_{++}$ and $b \in \mathbb{R}^n$, consider minimizing

$$f(x) = \frac{1}{2}\vec{x}^\top Q\vec{x} - \vec{b}^\top\vec{x} \tag{9}$$

Let $\vec{x}^\star$ denote the point at which $f(\vec{x})$ is minimized, and define $\mathcal{B}(\vec{x}^\star)$ as the ball centered at $\vec{x}^\star$ with unit ℓ_2 norm:

$$\mathcal{B}(\vec{x}^\star) = \{\vec{x} \in \mathbb{R}^n : \|\vec{x}-\vec{x}^\star\|_2 \leq 1\} \tag{10}$$

Assume we use Newton's method to minimize f:

$$\vec{x}_{k+1} = \vec{x}_k - (\nabla^2 f(\vec{x}_k))^{-1}\nabla f(\vec{x}_k) \tag{11}$$

where the initial point is $\vec{x}_0 \in \mathcal{B}(\vec{x}^\star)$.

For any $k \in \mathbb{N}$, find

$$\max_{\vec{x}_0 \in \mathcal{B}(\vec{x}^\star)} \|\vec{x}_k - \vec{x}^\star\|_2. \tag{12}$$

Solution: For all $\vec{x}$, the Hessian is $\nabla^2 f(\vec{x}) = Q$. Therefore, the update rule is

$$\vec{x}_{k+1} = \vec{x}_k - Q^{-1}(Q\vec{x}_k - \vec{b}) = Q^{-1}\vec{b} = \vec{x}^\star \tag{13}$$

So the optimal value of $\max_{\vec{x}_0 \in \mathcal{B}(\vec{x}^\star)} \|\vec{x}_k - \vec{x}^\star\|_2$ is 1 for $k=0$ and 0 for $k \geq 1$

3. Generalized Linear Models

A wide class of machine learning models (e.g. classification and regression) can be modelled in a common framework called generalised linear models (GLMs). In this problem, we'll talk about exponential families, generalized linear models and use Newton's method to perform maximum likelihood estimation (MLEs). Consider a special class of probability distributions known as exponential families whose density is of the form

$$f(\vec{y};\vec{\theta}) = e^{\vec{y}^\top\vec{\theta}-b(\vec{\theta})}f_0(\vec{y}) \tag{14}$$

where $\vec{y},\vec{\theta}\in\mathbb{R}^n$ and $b(\vec{\theta}) = \log\left(\int_{\mathbb{R}^n} e^{\vec{y}^\top\vec{\theta}}f_0(\vec{y})d\vec{y}\right)$ is the normalizing constant which ensures f is a probability distribution over $\vec{y}$.

(a) **(OPTIONAL)** Show that $b(\vec{\theta})$ is a convex function.

Solution: In a previous homework, you proved that the following function $g(x_1,\ldots,x_n) = \log\left(\sum_{i=1}^{n} e^{x_i}\right)$ is convex. Certainly this implies that $g(A\vec{x}+\vec{c})$ is also convex for any $A,\vec{c}$. This can be seen as taking an A and $\vec{c}$ which are infinite dimensional $(n\to\infty)$ matrix and vector respectively as then the sum can be treated as an integral (recall Riemann sum). These are indexed by different values of $\vec{y}$ taking $A(\vec{y}) = \vec{y}^\top$ and $c(\vec{y}) = \log(f_0(\vec{y}))$. **We won't expect you to do infinite summations like this for the final but provide it here to give an example of such proofs.**

For more details on the above proof, recall the Riemann sum interpretation of integrals: $\int_a^b f(x)dx = \lim_{n\to\infty}\sum_{i=1}^n f(x_i)(x_i - x_{i-1})$ by picking n points $a = x_0 < x_1 < \ldots < x_n = b$ and making the distance between the points tending to zero by taking infinitely many points in this interval. This idea also works when the integral limits a and b are $-\infty$ and ∞. We do the proof for dimension $n = 1$ and the proof for higher dimensions is exactly similar. Now we see,

$$\log\left(\int_{-\infty}^{\infty} e^{y\theta}dy\right) = \log\left(\lim_{n\to\infty}\sum_{i=1}^{n} e^{y_i\theta}(y_i - y_{i-1})\right) \tag{15}$$

$$= \log\left(\lim_{n\to\infty}\sum_{i=1}^{n} e^{y_i\theta+\log(y_i-y_{i-1})}\right) \tag{16}$$

$$= \lim_{n\to\infty}\log\left(\sum_{i=1}^{n} e^{y_i\theta+\log(y_i-y_{i-1})}\right) \tag{17}$$

Now if $g(x_1,\ldots,x_n) = \log\left(\sum_{i=1}^{n} e^{x_i}\right)$ is convex, so is any affine transformation and so the above is convex. Convexity is preserved when taking limits so we get that $b(\vec{\theta})$ is convex.

(b) We model $\vec{\theta} = X\vec{\beta}$ where $X\in\mathbb{R}^{n\times d}$ is the data matrix. Under this parameterization of $\vec{\theta}$, the exponential family is called a generalized linear model. Prove that $b(X\vec{\beta})$ is convex in $\vec{\beta}$.

Solution: If $b(\vec{\theta})$ is convex, so is any linear transformation so $b(X\vec{\beta})$ is convex in $\vec{\beta}$.

(c) For a given exponential family/GLM model, MLE estimation for a data matrix X and corresponding output variables $\vec{y}\in\mathbb{R}^n$ corresponds to solving the following maximization problem:

$$\max_{\vec{\beta}} f(\vec{y};X\vec{\beta}) \tag{18}$$

Prove that this maximization problem is equivalent to

$$\min_{\vec{\beta}} g(\vec{\beta}) := -\vec{y}^\top X\vec{\beta} + b(X\vec{\beta}) \tag{19}$$

Show that this is a convex optimization problem. Which choice of $b(\cdot)$ recovers linear regression?

Solution: Consider the negative logarithm of $f(\vec{y}; X\vec{\beta})$

$$-\log\left(e^{\vec{y}^\top X\vec{\beta}-b(X\vec{\beta})}f_0(\vec{y})\right) = -\vec{y}^\top X\vec{\beta} + b(X\vec{\beta}) - \log(f_0(\vec{y})) \tag{20}$$

Since the last term doesn't depend on $\vec{\beta}$ we can ignore it. Since $\log(\cdot)$ is an increasing function, maximizing a function is same as maximizing logarithm of the function and hence equivalent to minimizing negative log of the function and so we are done. We recover linear regression by taking $b(\vec{x}) = \frac{1}{2}\|\vec{x}\|_2^2$. An algorithm for classification known as logistic regression can be recovered by taking $b(\vec{x}) = \sum_{i=1}^{n} \log(1 + e^{x_i})$.

(d) For the above convex minimization problem, find the undamped Newton's method (with step size 1) update. This update also goes by the name *iteratively reweighted least squares* (IRLS). Can you tell why? (For any iterate $\vec{\beta}$ and the Newton update on $\vec{\beta}$ denoted by $\vec{\beta}_+$, what optimization problem is $\vec{\beta}_+$ the optimum of?)

Solution:

$$\nabla_{\vec{\beta}} g(\vec{\beta}) = -X^\top \vec{y} + X^\top \nabla b(X\vec{\beta}) \tag{21}$$

$$\nabla^2_{\vec{\beta}} g(\vec{\beta}) = X^\top \nabla^2 b(X\vec{\beta}) X \tag{22}$$

So the Newton update looks like

$$\vec{\beta}_+ = \vec{\beta} - (X^\top H_{\vec{\beta}} X)^{-1}(X^\top \nabla_\beta b(\vec{\beta}) - X^\top \vec{y}) \tag{23}$$

$$= (X^\top H_{\vec{\beta}} X)^{-1} X^\top H_{\vec{\beta}} \vec{\alpha}_{\vec{\beta}} \tag{24}$$

where $H_{\vec{\beta}} = \nabla^2_{\vec{\beta}} b(X\vec{\beta})$ and $\vec{\alpha}_{\vec{\beta}} = X\vec{\beta} + H_{\vec{\beta}}^{-1}(\vec{y} - \nabla_{\vec{\beta}} b(X\vec{\beta}))$ You can verify that $\vec{\beta}_+$ is the solution to the following *weighted* least squares problem:

$$\min_{\vec{\theta}} \frac{1}{2}\|H_{\vec{\beta}}^{1/2}(X\vec{\theta} - \vec{\alpha}_{\vec{\beta}})\|_2^2 \tag{25}$$

so at each step we are solving a least squares like problem where we *reweigh* the observations by a PSD matrix $H_{\vec{\beta}}^{1/2}$ and change the output vector to $\vec{\alpha}_{\vec{\beta}}$. Note that both $H_{\vec{\beta}}$ and $\vec{\alpha}_{\vec{\beta}}$ depend on the current iterate $\vec{\beta}$. This is why this problem sometimes is a special case of a procedure known as *iteratively reweighted least squares*(IRLS)