Convex Optimization 10-725/36-725
Homework 2, due Oct 3

Instructions:

- You must complete Problems 1–3 and **either Problem 4 or Problem 5** (your choice between the two).
- When you submit the homework, upload a single PDF (e.g., produced by LaTeX, or scanned handwritten exercises) for the solution of each problem separately, to blackboard. You should your name at the top of each file, except for the first problem. **Your solution to Problem 1 (mastery set) should appear completely anonymous to a reader.**

1 Mastery set [25 points] (Aaditya)

Be very concise. If written up and scanned, should be less than two sides in total for the whole mastery set. If Latex-ed up, should be less than one side.

A [2 + 2] Let S be the set of all minimizers of a convex function f. Prove that S is convex. Now suppose that f is strictly convex. Prove that S contains only one element, i.e., f has a unique minimizer.

B [2 + 2] What are the singular values of an $n \times n$ square orthogonal matrix? (justify in one line) Show with a short example that the set of all orthogonal matrices is non-convex.

C [2 + 2 + 2] Show that every convex combination of orthogonal matrices has spectral norm at most 1. In the second recitation, we showed that the spectral norm $\|A\|_{\text{op}}$ is a convex function of A. Here, prove that the unit spectral norm ball $\{A : \|A\|_{\text{op}} \leq 1\}$ is a convex set. Is the spectral norm a Lipschitz function with respect to the spectral norm, i.e. is $|\|A\|_{\text{op}} - \|B\|_{\text{op}}| \leq L\|A - B\|_{\text{op}}$ for some L? (Hint: use properties of norms from first recitation)

D [5] Use the above part to show that the convex hull of orthogonal matrices is the unit spectral norm ball (hint: the reverse direction still needs to be shown).

E [4+2] For a differentiable λ-strongly convex function, starting from the Taylor-like first order definition, prove that $\|\nabla f(y) - \nabla f(x)\|_2 \geq \lambda \|y - x\|_2$. (hint: proof is four lines long, plenty of hints in the second recitation). If additionally, its gradient is L-Lipschitz, what can we say about λ, L?

2 Subgradients of matrix norms [30 points, 8+9+9+4] (Yifei)

In this problem, you'll consider two types of matrix norms: the trace norm (also called the nuclear norm) and the operator norm (also called the spectral norm). Recall that for a matrix A, its trace norm is $\|A\|_* = \sum_{i=1}^r \sigma_i(A)$, the sum of the singular values of A, and its operator norm is $\|A\|_{\text{op}} = \sigma_i(A)$, the largest singular value of A.

An important fact to know is that, in the matrix world, the inner product between matrices A, B is given by $\text{tr}(B^T A)$, where $\text{tr}(\cdot)$ is the trace operator (sum of diagonal elements). [To convince yourself of this, think about unraveling A, B as vectors and performing a usual inner product—check that this matches $\text{tr}(B^T A)$.]

Now assume that $A \in \mathbb{R}^{m \times n}$ has rank r and singular value decomposition $A = U\Sigma V^T$, where $U \in \mathbb{R}^{m \times r}, \Sigma \in \mathbb{R}^{r \times r}, V \in \mathbb{R}^{n \times r}$. Parts (a) and (b) concern the subgradients of the trace norm evaluated at A (we used these subgradients in lecture to derive a proximal gradient algorithm for matrix completion). Parts (c) and (d) concern the subgradients of the operator norm at A.

(a) [8 points] Prove that for any $W \in \mathbb{R}^{m \times n}$ with $\|W\|_{\text{op}} \leq 1$, $U^T W = 0$, $WV = 0$, we have $\|UV^T + W\|_{\text{op}} \leq 1$.

(Hint: look at the singular value decomposition of $UV^T + W$.)

(b) [4 points] Prove that for any such W as in (a), we have $\text{tr}((UV^T + W)^T A) = \text{tr}(\Sigma)$.

(c) [4 points] Prove that the subdifferential of the trace norm evaluated at A satisfies

$$\partial \|A\|_* \supseteq \{UV^T + W : W \in \mathbb{R}^{m \times n}, \|W\|_{\text{op}} \leq 1, U^T W = 0, WV = 0\}.$$

[Hint: it will be helpful to use the fact that the dual of the trace norm is the operator norm, i.e.,

$$\|A\|_* = \max_{\|B\|_{\text{op}} \leq 1} \text{tr}(B^T A).$$

Now recall the rule for subgradients of functions defined via a max operation.]

Note: the above is actually an equality, not a containment, for $\partial\|A\|_*$. Proving the reverse direction is only a little bit more tricky.

(d) [10 points] Using the same strategy that was used in parts (a)–(c), prove that

$$\partial\|A\|_{\text{op}} \supseteq \text{conv}\big(\{u_j v_j^T : \Sigma_{jj} = \Sigma_{11}\}\big),$$

where again we use $A = U\Sigma V^T$ to denote the SVD of A, and u_j, v_j are the jth columns of U, V, respectively. (Note that Σ_{11} is the largest entry in Σ.)

[Hint: as before, use the dual relationship between the trace and operator norms,

$$\|A\|_{\text{op}} = \max_{\|B\|_* \le 1} \text{tr}(B^T A).$$

and the rule for subgradients of functions defined by a maximum.]

(e) [4 points] The result in (d) is actually an equality for $\partial\|A\|_{\text{op}}$, though again, we will not prove the reverse containment since it is a little more tricky. Assuming this fact, when does the subdifferential $\partial\|A\|_{\text{op}}$ contain a single element? Hence what can you conclude about the differentiability of the operator norm at a matrix A?

3 Algorithms for matrix completion [30 points] (Sashank)

In this problem we will compare generalized gradient descent and sub-gradient descent on the task of matrix completion. Given a partially observed matrix Y, we can formulate matrix completion as optimizing the following objective:

$$B_\lambda = \underset{B\in\mathbb{R}^{m\times n}}{\text{argmin}} \ \frac{1}{2}\sum_{(i,j)\in\Omega}(Y_{ij} - B_{ij})^2 + \lambda\|B\|_* \tag{1}$$

$$= \underset{B\in\mathbb{R}^{m\times n}}{\text{argmin}} \ \frac{1}{2}\|P_\Omega(Y - B)\|_F^2 + \lambda\|B\|_*, \tag{2}$$

where $Y \in \mathbb{R}^{m\times n}$ is the partially observed matrix, $\Omega \subset \{1,\dots m\} \times \{1,\dots n\}$ is the observed entry set, $\|\cdot\|_*$ is the trace norm, and $P_\Omega(\cdot) : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ is the projection operator onto the observed set Ω:

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

We are going to compare different algorithms for optimizing the above objective on a subset of the MovieLens dataset. The MovieLens dataset consists of a large, sparse matrix Y, where the rows and columns of Y represent users and movies respectively. Specifically, the entry Y_{ij} represents the rating of user i for movie j. Since not all users rated all movies, many entries of Y are missing. Your task in this question is to complete the movie ratings matrix Y.

For detailed analysis, we actually provide two matrices Y: a 'training' matrix Y^{tr}, and 'test' matrix Y^{te}. You can think of the two matrices as two different subsamples from some ground-truth full matrix. We will use Y^{tr} to learn a completed matrix B_λ for each value of λ, and then use the error between Y^{te} and B_λ to choose the best λ. In other words, we will choose a λ which generalizes well to new incomplete information of the same kind.

Let P_Ω^{te} be the projection onto the observed entry set for the test matrix, and k the number of observed entries in Y^{te}. The test error (RMSE) used to choose λ is defined as follows:

$$RMSE = \frac{\|P_\Omega^{te}(Y^{te} - B_\lambda)\|_F}{\sqrt{k}}.$$

Download the data from `http://www.stat.cmu.edu/~ryantibs/convexopt/homeworks/movie_data.zip`.

(a) [**10**] Recall the soft-impute algorithm discussed in lecture, with updates:

$$B^+ = S_\lambda\big(P_\Omega(Y^{tr}) + P_\Omega^\perp(B)\big),$$

where $S_\lambda(\cdot)$ is the matrix soft-thresholding operator, and $P_\Omega^\perp = I - P_\Omega$, the projector onto the unobserved set. Recall that this is just proximal gradient descent with a fixed step size $t = 1$. Implement the soft impute function in MATLAB or R with $\Lambda = \text{logspace}(0, 3, 30)$. You can just implement a "naive" function for matrix soft-thresholding, which just computes the entire SVD and uses it for thresholding. For each of the 30 values of λ, run soft-impute on training data until convergence, starting from $B = 0$. The stopping criteria is $\frac{|f_{k+1} - f_k|}{f_k} < 10^{-4}$ where f_k is the objective function value at k^{th} iteration or maximum of 500 iterations (whichever occurs first) .

(b) [**5**] Record and plot the number of iterations it took to converge at each value of $\log(\lambda)$. Plot the RMSE error on training and test data versus $\log(\lambda)$. What value of λ would you choose, and what is the rank of the corresponding solution?

(c) [**5**] Now across the 30 values of λ, again run soft-impute until convergence, but this time using warm starts. i.e., at each successive value of λ (in sorted order, from largest to smallest) start the algorithm at the previously computed solution. Again record the number of iterations required to converge at each λ value. Did the number of iterations change in comparison to without warm start? If so, then why, roughly speaking, do warm starts work?

(d) [**10**] Derive the sugradient method steps for the optimization problem 1, and implement it. Run soft impute and subgradient method for 500 iterations and compare their objective function values over $\Lambda = \{1, 5, 10\}$. For the subgradient method, use the step size $t_k = 1/k$ for the k^{th} iteration.

Your comparison should involve three figures, one for each λ value. In each, plot $f_k - f_{\min}$ versus the iteration number k for both the subgradient and soft impute methods. Here $f_{\min}$ is the minimum value of the objective function found overall (by either the subgradient or soft impute method), and f_k is the current value of the criterion at the kh iteration (for the subgradient method, since it is not a descent method, this is the best value seen up until the kth iteration). It will be helpful to put the y-axis in these figures on a log scale. Which algorithm performed better?

4 Convergence rate of generalized gradient descent [15 points] (Adona)

Recall that the generalized gradient descent method for minimizing a composite function $f(x) = g(x) + h(x)$, for convex, differentiable $g : \mathbb{R}^n \to \mathbb{R}$ and convex $h : \mathbb{R}^n \to \mathbb{R}$, begins with an initial point $x^{(0)} \in \mathbb{R}^n$, and repeats

$$x^{(k)} = \text{prox}_{t_k}\big(x^{(k-1)} - t_k \nabla g(x^{(k-1)})\big). \tag{3}$$

Here we will assume that ∇g is Lipschitz with constant $L > 0$,

$$\|\nabla g(x) - \nabla g(y)\|_2 \le L\|x - y\|_2 \quad \text{all } x, y \in \mathbb{R}^n,$$

and we will prove that by taking a fixed step size $t_k = t \le 1/L$ for all $k = 1, 2, 3, \ldots$, the generalized gradient algorithm has the exact same convergence guarantees as does gradient descent.

It will be helpful to define the "generalized gradient" G_t of f so that the updates (3) look like gradient descent updates. This is

$$\text{prox}_t\big(x - t\nabla g(x)\big) = x - tG_t(x), \tag{4}$$

i.e.,

$$G_t(x) = \frac{x - \text{prox}_t\big(x - t\nabla g(x)\big)}{t}.$$

(a) [**2**] Show that ∇g being Lipschitz with constant L implies

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

for all x, y.

(b) [**2**] By plugging in $y = x^+ = x - tG_t(x)$ into the result from (a), and letting $t \le 1/L$, show that

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

(c) [**4**] Arguing directly from (4), use the definition of the proximal operator,

$$\text{prox}_t(x) = \underset{z \in \mathbb{R}^n}{\text{argmin}} \ \frac{1}{2t}\|x - z\|_2^2 + h(z),$$

and the zero subgradient characterization of optimality, to show that

$$G_t(x) - \nabla g(x) \in \partial h\big(x - tG_t(x)\big).$$

(d) [**3**] Starting from the bound you had in (b), use the result of part (c), and the convexity of g, h around an arbitrary z, to show that

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

for all z.

(e) [**2**] Take $z = x$ in (d) to verify that the generalized gradient descent algorithm decreases the criterion f at each iteration. Take $z = x^\star$, a minimizer of f, to yield

$$f(x^+) \le f(x^\star) + \frac{1}{2t}\big(\|x - x^\star\|_2^2 - \|x^+ - x^\star\|_2^2\big).$$

(f) [**2**] Complete the proof, following the same strategy as in gradient descent, to conclude

$$f(x^{(k)}) - f(x^\star) \le \frac{\|x^{(0)} - x^\star\|_2^2}{2tk}.$$

5 Accelerating matrix completion [15 points] (Adona)

In this problem we will continue exploring the matrix completion algorithms of Q3, in particular how they combine with acceleration and backtracking, and how they behave on a realistic dataset.

(a) [**7**] Implement the generalized gradient descent (soft-impute) algorithm of Q3(a) with acceleration. Run both the non-accelerated and accelerated versions for different λs in $\Lambda = \text{logspace}(0, 3, 30)$, and plot the number of iterations the two algorithms took to converge for each value of λ, starting from $B = 0$. As before, run the algorithms until either $\frac{|f_{k+1} - f_k|}{f_k} < 10^{-4}$ (where f_k is the objective function value at k^{th} iteration), or after a maximum of 500 iterations (whichever occurs first). Further, for fixed $\lambda = 10$, plot and compare the value of the objective at each iteration # with and without acceleration. What do you observe? Does acceleration help for this problem?

(b) [**2**] Do the same as in part (a), but now using warm starts. Is there any difference in the comparison between no acceleration and acceleration?

(c) [**6**] Next we will explore how matrix completion performs as an algorithm for image reconstruction. Download the image of Mona Lisa from `http://www.stat.cmu.edu/~ryantibs/convexopt/homeworks/mona_bw.jpg`. Construct a test image by randomly subsampling 50% of the pixels, and setting the remaining pixels to zero. Run matrix completion with accelerated generalized gradient descent on the subsampled image for a few (10-15) different λs in the range $10^{-2} - 10^{2}$. What do you observe? Show the original image, the subsampled image, and 3-4 reconstructions across the range of λs. Which λ returns the best results? Repeat this experiment at 20% subsampling level, and show the best reconstructed image. What λ did you choose in this case, and was it different from the best λ at 50% subsampling level?