Convex Optimization
Homework 5

Winter 1401
Due date: 26th of Farvardin

---

1. $l_{1.5}$ *optimization* Optimization and approximation methods that use both an l_2-norm (or its square) and an l_1-norm are currently very popular in statistics, machine learning, and signal and image processing. Examples include Huber estimation, LASSO, basis pursuit, SVM, various l_1-regularized classification methods, total variation de-noising, etc. Very roughly, an l_2-norm corresponds to Euclidean distance (squared), or the negative log-likelihood function for a Gaussian; in contrast the l_1-norm gives 'robust' approximation, *i.e.*, reduced sensitivity to outliers, and also tends to yield sparse solutions (of whatever the argument of the norm is). (All of this is just background; you don't need to know any of this to solve the problem.)

   In this problem we study a natural method for blending the two norms, by using the $l_{1.5}$-norm, defined as

$$||z||_{1.5} = \left(\sum_{i=1}^{k} |z_i|^{3/2}\right)^{2/3}$$

   for $z \in \mathbf{R}^k$. We will consider the simplest approximation or regression problem:

$$minimize ||Ax - b||_{1.5},$$

   with variable $x \in \mathbf{R}^n$, and problem data $A \in \mathbf{R}^{m\times n}$ and $b \in \mathbf{R}^m$. We will assume that $m > n$ and the A is full rank (*i.e.,* rank n). The hope is that this $l_{1.5}$-optimal approximation problem should share some of the good features of l_2 and l_1 approximation.

   (a) Give optimality conditions for this problem. Try to make these as simple as possible.

   (b) Explain how to formulate the $l_{1.5}$-norm approximation problem as an SDP. (Your SDP can include linear equality and inequality constraints.)

   (c) Solve the specific numerical instance generated by the following code:

   $\mathbf{randn}('\mathbf{state}', \mathbf{0});$

   $\mathbf{A} = \mathbf{randn}(\mathbf{100}, \mathbf{30});$

   $\mathbf{b} = \mathbf{randn}(\mathbf{100}, \mathbf{1});$

   Numerically verify the optimality conditions. Give a histogram of the residuals, and repeat for the l_2-norm and l_1-norm approximations. You can use any method you like to solve the problem (but of course you must explain how you did it); in particular, you do not need to use the SDP formulation found in part (b).

2. *Piecewise-linear fitting.* In many applications some function in the model is not given by a formula, but instead as tabulated data. The tabulated data could come from empirical measurements, historical data, numerically evaluating some expression or solving some problem, for a set of values of the argument. For use in a convex optimization model, we then have to fit these data with a convex function that is compatible with the solver or other system that we use. In this problem we explore a very simple problem of this general type. Suppose we are given the data (x_i, y_i), $i = 1, ..., m$, with $(x_i, y_i) \in \mathbf{R}$. We will assume that x_i are sorted, *i.e.,* $x_1 < x_2 < ... < x_m$. Let $a_0 < a_1 < a_2 < ... < a_K$ be a set of fixed knot points, with $a_0 \leq x_1$ and $a_K \geq x_m$. Explain how to find the convex piecewise linear function f, defined over $[a_0, a_K]$, with knot points a_i, that minimizes the least-squares fitting criterion

$$\sum_{i=1}^{m} (f(x_i) - y_i)^2.$$

You must explain what the variables are and how they parametrize f, and how you ensure convexity of f. *Hints.* One method to solve this problem is based on the Lagrange basis, $f_0, ..., f_K$, which are the piecewise linear functions that satisfy

$$f_j(a_i) = \delta_{ij}\,,\ i, j = 0, ..., K.$$

Another method is based on defining $f(x) = \alpha_i x + \beta_i$, for $x \in (a_{i-1}, a_i]$. You then have to add conditions on the parameters α_i and β_i to ensure that f is continuous and convex. Apply your method to the data in the file **pwl_fit_data.m**, whichcontains data $x_j \in [0, 1]$. Find the best affine fit (which corresponds to $a = (0, 1)$), and the best piecewise-linear convex function fit for 1,2, and 3 internal knot points, evenly spaced in [0,1]. (for example, for 3 internal knot points we have $a_0 = 0, a_1 = 0.25, a_2 = 0.5, a_3 = 0.75, a_4 = 1$,.) Give the least-squares fitting cost for each one. Plot the data and the piecewise-linear fits found. Express each function in the form

$$f(x) = \max_{i=1,..,K}(\alpha_i x + \beta_i)$$

.

(In this form the function is easily incorporated into an optimization problem.)

3. *Multi-label support vector machine.* The basic SVM described in the book is used for classification of data with two labels. In this problem we explore an extension of SVM that canbe used to carry out classification of data with more than two labels. Our data consists of pairs $(x_i, y_i) \in R^n \times \{1, ..., K\}, i = 1, ..., m$, where x_i is the feature vector and y_i is the label of the ith data point. (So the labels can take the values $1, ..., K$.) Our classifier will use K affine functions, $f_k(x) = a_k^T x + b_k, k = 1, ..., K$, which we also collect into affine function from R^n into R^k as $f(x) = Ax+b$. (The rows of A are a_k^T.) Given feature vector x, we guess the label $\hat{y} = argmax_k f_k(x)$. We assume that exact ties never occur, or if they do, an arbitrary choice can be made. Note that if a multiple of $\mathbf{1}$ is added to b, the classifier does not change. Thus, without loss of generality, we can assume that $\mathbf{1}^T b = 0$.

   To correctly classify the data examples, we need $f_{y_i}(x_i) > max_{k \neq y_i} f_k(x_i)$ for all i. This is a set of homogeneous strict inequalities in a_k and b_k, which are feasible if and only if the set of nonstrict inequalities $f_{y_i}(x_i) \geq 1 + max_{k \neq y_i} f_k(x_i)$ are feasible. This motivates the loss function

$$L(A, b) = \sum_{i=1}^{m}(1 + \max_{k \neq y_i} f_k(x_i) - f_{y_i}(x_i))_+,$$

   where $(u)_+ = max\{u, 0\}$. The multi-label SVM chooses A and b to minimize

$$L(A, b) + \mu ||A||_F^2,$$

   subject to $\mathbf{1}^T b = 0$, where $\mu > 0$ is a regularization parameter. (Several variations on this are possible, such as regularizing b as well, or replacing the Frobenius norm squared with the sum of norms of the columns of A.)

   (a) Show how to find A and b using convex optimization. Be sure to justify any change of variables or reformulation (if needed), and convexity of the objective and constraints in your formulation.

   (b) Carry out multi-label SVM on the data given in **multi_label_svm_data.m**. Use the data given in $\mathbf{X}$ and $\mathbf{y}$ to fit the SVM model, for a range of values of μ. This data set includes an additional set of data **Xtest** and**ytest**, that you can use to test the SVM models. Plot the test set classification error rate (*i.e.*, the fraction of data examples in the test set fir which $\hat{y} \neq y$) versus μ. You don't need to try more than 10 or 20 values of μ, and we suggest choosing them uniformly on a log scale, from (say) $10^{-2}to10^2$.

4. *Maximum likelihood prediction of team ability.* A set of n teams compete in a tournament. We model each team's ability by a number $a_j \in [0,1], j = 1, ..., n$. When teams j and k play each other, the probability that team j wins is equal to $\mathbf{prob}(a_j - a_k + \nu > 0)$, where $\nu \sim \mathcal{N}(0, \sigma^2)$. You are given the outcome of m past games. These are organized as

$$(j^i, k^i, y^i), i = 1, ..., m,$$

meaning that game i was played between teams j^i and k^i; $y^i = 1$ means that team j^i won, while $y^i = -1$ means that team k^i won. (We assume there are no ties.)

(a) Formulate the problem of finding the maximum likelihood estimate of team abilities, $\hat{a} \in \mathbf{R}^n$, given the outcomes, as a convex optimization problem. You will find the *game incidence matrix* $A \in \mathbf{R}^{m \times n}$, defined as

$$A = \begin{cases} y^{(i)} & l = j^{(i)} \\ -y^{(i)} & l = k^{(i)} \\ 0 & otherwise, \end{cases}$$

useful. The prior constraints $\hat{a}_i \in [0,1]$ should be included in the problem formulation. Also, we note that if a constant is added to all team abilities, there is no change in the probabilities of game outcomes. This means that $\hat{a}$ is determined only up to a constant, like a potential. But this does'nt affect the ML estimation problem, or any subsequent predictions made using the estimated parameters.

(b) Find $\hat{a}$ for a tournament data given in **team_data.m**, in the matrix **train**. (This matrix gives the outcomes for a tournament in which each team plays each other team once.) You may find the CVX function **log_normcdf** helpful for this problem. You can form A using the commands

$$A = sparse(1 : m, train(:, 1), train(:, 3), m, n) + ...$$
$$sparse(1 : m, train(:, 2), -train(:, 3), m, n);$$

(c) Use the maximum likelihoood estimate $\hat{a}$ found in part (b) to predict the outcomes of next year's tournament games, given in the matrix **test**, using $\hat{y}^{(i)} = \mathbf{sign}(\hat{a}_{j^{(i)}} - \hat{a}_{k^{(i)}})$. Compare these predictions with the actual outcomes, given in the third column of **test**. Give the fraction of correctly predicted outcomes. The games played in **train** and **test** are the same, so another, simpler method for predicting the outcomes in **test** is to just assume the team that won last year's math will also win this year's match. Give the percentage of correctly predicted outcomes using this simple method.

5. *Worst-case probability of loss.* Two investments are made, with random returns R_1 and R_2. The total return for the two investments is $R_1 + R_2$, and the probability of a loss (including breaking even, *i.e.*, $R_1 + R_2 = 0$) is $p^{loss} = \mathbf{prob}(R_1 + R_2 \leq 0)$. The goal is to find the worst-case (*i.e.*, *maximumpossible*) value of p^{loss}, consistent with the following information. Both R_1 and R_2 have Gaussian marginal distributions, with known means μ_1 and μ_2 and knwon standard deviations σ_1 and σ_2. In addition, it is known that R_1 and R_2 are correlated with correlation coefficient ρ, *i.e.*,

$$\mathbf{E}(R_1 - \mu_1)(R_2 - \mu_2) = \rho\sigma_1\sigma_2.$$

Your job is to find the worst-case p^{loss} over any joint distribution of R_1 and R_2 consistent with the given marginals and correlation coefficient. We will consider the specific case with data

$$\mu_1 = 8, \mu_2 = 20, \sigma_1 = 6, \sigma_2 = 17.5, \rho = -0.25.$$

We can compare the results to the case when R_1 and R_2 are jointly Gaussian. In this case we have

$$R_1 + R_2 \sim \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2 + 2\rho\sigma_1\sigma_2),$$

which for the data given above gives $p^{loss} = 0.050$. Your job is to see how much larger p^{loss} can possibly be. This is an infinite-dimensional optimization problem, since you must maximize p^{loss} over an infinite-dimensional set of joint distributions. To (approximately) solve it, we discretize the values that R_1 and R_2 can take on, to $n = 100$ values $r_1, ..., r_n$, uniformly spaced from $r_1 = -30$ to $r_n = 70$. We use the discretized marginals $p^(1)$ and $p^(2)$ for R_1 and R_2, given by

$$p_i^{(k)} = \mathbf{prob}(R_k = r_i) = \frac{\exp\left(-\left(r_i - \mu_k\right)^2 / \left(2\sigma_k^2\right)\right)}{\sum_{j=1}^n \exp\left(-\left(r_j - \mu_k\right)^2 / \left(2\sigma_k^2\right)\right)},$$

for $k = 1, 2, i = 1, ..., n$.

Formulate the (discretized) problem as a convex optimization problem, and solve it. Report the maximum value of p^{loss} you find. Plot the joint distribution that yields the maximum value of p^{loss} using the MATLAB commands *mesh* and *contour*.

Remark. You might be surprised at both the maximum value of p^{loss}, and the joint distribution that achieves it.

6. *Rank one nonnegative matrix approximation.* We are given some entries of an $m \times n$ matrix A with positive entries, and wish to approximate it as the outer product of vectors x and y with positive entries, *i.e.*, xy^T. We will use the average relative deviation between the entries of A and xy^T as our approximation criterion,

$$\frac{1}{mn} \sum_{i=1}^m \sum_{j=1}^n R\left(A_{ij}, x_i y_j\right),$$

where R is the relative deviation of two positive numbers, defined as

$$R(u, \nu) = max\{u/\nu, \nu/u\} - 1,$$

If we scale x by the positive number α, and y by $1/\alpha$, the outer product $(\alpha x)(y/\alpha)^T$ is the same as xy^T, so we will normalize x as $\mathbf{1}^T x = 1$.

The data in the problem consists of some of the values of A. Specifically, we are given A_{ij} for $(i, j) \in \Omega \subseteq \{1, ..., m\} \times \{1, ..., n\}$. Thus, your goal is to find $x \in R_{++}^m$ (which satisfies $\mathbf{1}^T x = 1$), $y \in R_{++}^n$, and $A_{ij} > 0$ for $(i, j) \notin \Omega$, to minimize the average relative deviation between the entries of A and xy^T.

(a) Explain how to solve this problem using convex or quasiconvex optimization.

(b) Solve the problem for the data given in **rank_one_nmf_data**.∗. This includes a matrix A, and a set of indexes *Omega* for the given entries. (The other entries of A are filled in with zeros.) Report the optimal average relative deviation between A and xy^T. Give your values for x_1, y_1 and $A_{11} = x_1 y_1$.

Good Luck!