Solutions to hw3 homework on Convex Optimization https://web.stanford.edu/class/ee364b/homework.html

Andrei Keino

March 9, 2021

3.1 (4 points)

Consider the optimization problem

$$minimize_{\{x_j\}_{j=1}^J} f(x_1, \ldots, x_j) \coloneqq \frac{1}{2}\|b - \sum_{j=1}^{J} A_j x_j\|_2^2 + \lambda \cdot \sum_{j=1}^{J} \|x_j\|_2,$$
$$s.t.\ A_j x_j \geq 0,\ \forall j \in \{1, 2, \ldots, J\}$$

with variable $x_1, \ldots, x_J \in R^n$, and problem data $A_1, \ldots, A_J \in R^{m \times n}$, $b \in R^m$, and $\lambda > 0$. for constrained optimization given on page 11 (really p. 12) of the lecture slides for subgradient methods for constrained problems.

Let $J = 3$, $n = 100$, $m = 10$ and $\lambda = 0.5$ Generate random matrices $A_1, \ldots, A_J \in R^{m \times n}$ with independent uniformly distributed entries in the interval $[0, \frac{1}{\sqrt{m}})$, and, random vectors $x_1, \ldots, x_J \in 2R^n$uniformly distributed entries in the interval $[0, \frac{1}{\sqrt{n}})$, then set $b = \sum_{j=1}^{J} A_j x_j$. Plot convergence in terms of the objective $f(x_1^{(k)}, \ldots x_1^{(J)})$. Try different step length schedules. Also, plot the maximal violation for the linear constraints at each step.

Solution:
The code is in the file `solution_3_1_b.m`. The code is nearly the same as one for the task 2.4, the only difference that on every step of the gradient descent we are calculating the constraint violation vector and if there are at least one violation, we replace the gradient of the minimized function with the gradient of any constraint violation found.

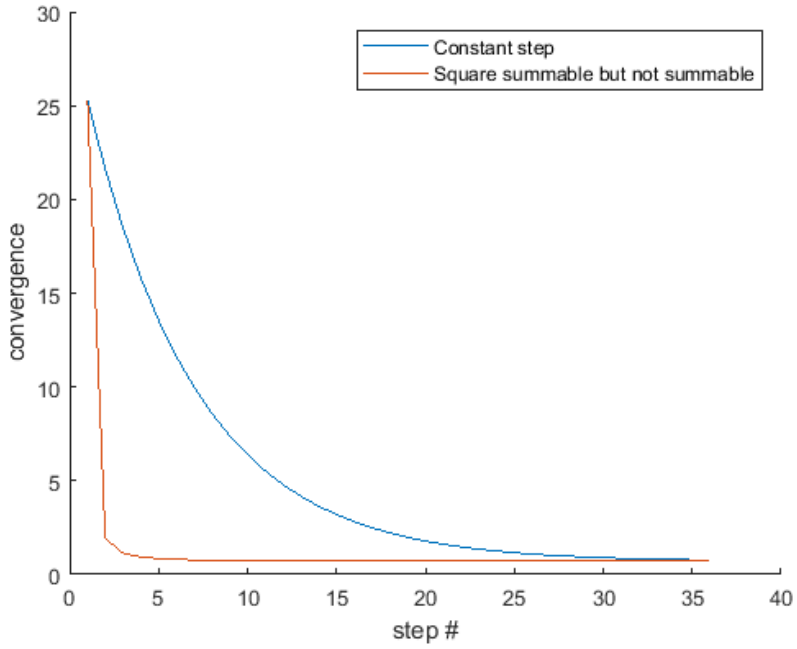

Figure 1: Convergence with different step length.

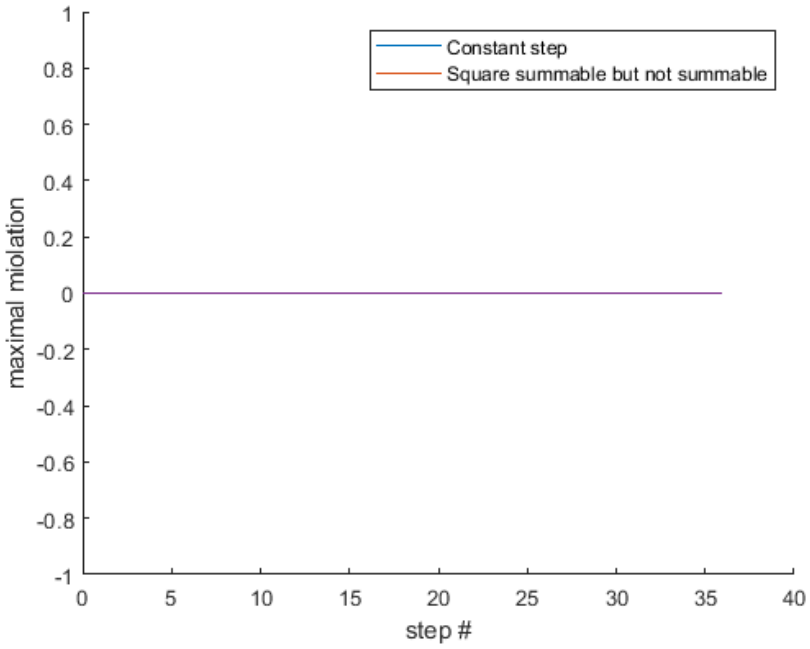

Figure 2: Maximal violation for the linear constraints.

3.2 (4 points)

A randomized least squares solver. Consider the Least Squares minimization problem

$$\begin{aligned} &\text{minimize } \frac{1}{2m} \underbrace{\sum_{i=1}^{m} (b_i - a_i^T x)^2}_{f(x)} \\ &\text{subject to } x \in R^n \end{aligned}$$

where $a_1, \ldots, a_m$ are rows of matrix A. We will consider the stochastic subgradient descent iterates

$$x^{t+1} = x^t - \alpha_t g_t \tag{1}$$

where g_t is a noisy unbiased gradient of the objective function, i.e., $E[g^T | x^T] \in \partial f(x^t)$.

a) (1 point) Let j will be a random index chosen from $\{1, \ldots, m\}$ such that for every index $i \in \{1, \ldots, m\}$ the probability that $j = i$ is p_i, i.e.,

$$P(i = j) = p_i,$$

for a given discrete probability distribution $p_1, \ldots, p_m \geq 0$, $\sum_{i=1}^{m} = 1$. Show that

$$E(\frac{(a_j^T x - b_j)}{mp_j} a_j) \in \partial f(x)$$

where the expectation is taken over the random variable j.

Solution:

$$f(x) = \frac{1}{2m}(b - Ax)^T(b - Ax)$$

The gradient of $f(x)$ is

$$\frac{(Ax - b)^T}{m} A$$

or, component-wize

$$e_i \frac{(a_i^T x - b_i)}{m} a_i$$

there e_i is the i-th component of the unit vector.
The expectation is

$$E\frac{(a_j^T x - b_j)}{mp_j} a_j =$$
$$\frac{1}{m}(E(a_j^T x - b_j)a_j/p_j) =$$
$$\frac{1}{m}(E(a_j^T x a_j/p_j) - E(b_j a_j/p_j))$$

For the second member of equation:

$$E(b_j a_j/p_j) = \sum_{j=1}^{m} e_j p_j b_j a_j/p_j = \sum_{j=1}^{m} e_j b_j a_j$$
$$= e_j b_j a_j.$$

there e_j is the j-th component of the unit vector.
So, $E(b_j a_j/p_j) = e_j b_j a_j$.
The equality for the first member of the equation:

$$E(a_j^T x a_j/p_j) = e_j a_j^T x a_j$$

can be proved in the same way. So, we're done.

(b)
Assume that $b = Ax^*$ for some vector $x*$, i.e., $x \in arg\ min f(x)$. Define the error vector $e_t = x_t - x^*$, where x_t is the subgradient descent iterate in (1). Consider the the constant step size $\alpha_t = \frac{m}{||A||_F^2}$, the unbiased the unbiased subgradient from part (a) sampled i.i.d. at every iteration, and the probability distribution

$$p_i = \frac{||a_i||_2^2}{\sum_k ||a_k||_2^2} = \frac{||a_i||_2^2}{||A||_F^2}$$

Show that the error vector e_t obeys the time-varying linear dynamical system

$$e_{t+1} = P_t e_t$$

where P_t is a (random) symmetric projection matrix, i.e., $P_t^T P_t = P_t^2 = P_t$ obeying $\mathbf{E}P_t = I - \frac{1}{||A||_F^2} A^T A$.

Solution:

$e_{t+1,j} =$
(from (1))
$x_{t,j} - \alpha_t g_{t,j} - x_j^* =$
$e_{t,j} - \alpha_t g_{t,j} =$
(using the equation for subgradient)
$e_{t,j} - y_j \alpha_t (a_j^T x_{t,j} - b_j) a_j / m p_j =$
(here y_j is a random variable which is equal either to 0 or 1 with probability p_j)
(using the equation $e_t = x_t - x^*$)
$e_{t,j} - y_j \alpha_t (a_j^T (e_{t,j} + x_j^*) - b_j) a_j / m p_j =$
(as $a_j^T x_j^* - b_j = 0$)
$e_{t,j} - y_j \alpha_t a_t^T e_{t,j} a_j / m p_j =$
$e_{t,j}(1 - y_j \alpha_t a_j^T a_j / m p_j) =$
$e_{t,j}(1 - y_j a_j^T a_j / (m * p_j ||A||_F^2) =$
(b1)
using the given expression for p_j :
$e_{t,j}(1 - y_j a_j^T a_j / ||a_j||_2^2 =$
$e_{t,j}(1 - y_j)$

If we rewrite the last equation in the matrix form we will get:

$$e_{t+1} = Pe_t$$

where $P = I - Y_j$, where Y_j is the matrix those elements are equal $\delta_{ij} y_i$, i.e., all non-diagonal elements are zero, and diagonal of matrix consists of the y_i. This is evident that this matrix is a symmetric projection matrix, as

$$P = P^T,\ PP^T = P^2 = P.$$

Using equation (b1)

$$e_{t+1,j} = e_{t,i}(1 - y_j a_j^T a_j/(mp_j||A||_F^2))$$

we get:

$$P_{j,j} = (1 - y_j a_j^T a_j/(mp_j||A||_F^2))$$

where $P_{j,j}$ is the $j, j-$ th element of matrix P. then

$$\mathbf{E}P_{j,j} = 1 - \sum_{i=1}^{m} p_j a_j^T a_j/(mp_j||A||_F^2)$$

i.e,

$$\mathbf{E}P = I - A^T A/||A||_F^2$$

(c) (1 point) Show that

$$\mathbf{E}||e_{t+1}||_2^2 \leq \big(1 - \frac{\sigma_{min}(A)^2}{||A||_F^2}\big) E(||e_t||_2^2)$$

where $\sigma_{min}(A)$ is the smallest singular value of A. Hint: note that $\mathbf{E}[||e_{t+1}||_2^2|e_t] = \mathbf{E}[e_t^T P_t^T P_t e_t|e_t] = \mathbf{E}[e_t^T P_t e_t|e_t] = e_t^T \mathbf{E}[P_t] e_t$, and that

$$e^T A^T A e \geq \sigma_{min}(A)^2 e^T e.$$

for every vector $e \in R^n$. Apply this bound recursively to obtain a bound on $\mathbf{E}||e_t||_2^2$, involving only $\mathbf{E}||e_0||_2^2$, $||A||_F$, $\sigma_{min}(A)$.

Solution:

$$\begin{aligned}
&\mathbf{E}[||e_{t+1}||_2^2|e_t] = \\
&\mathbf{E}e_t^T \mathbf{E}[P_t] e_t = \\
&\mathbf{E}e_t^T (I - A^T A/||A||_F^2) e = \\
&\mathbf{E}(e^T e) - \mathbf{E}(e^T A^T A e)/||A||_F^2 \leq \\
&\mathbf{E}(e^T e) - \sigma_{min}(A)^2 \mathbf{E}(e^T e)/||A||_F^2 = \\
&\mathbf{E}(e^T e)(1 - \sigma_{min}(A)^2/||A||_F^2)
\end{aligned}$$

Applying this inequality recursively we will get

$$\mathbf{E}||e_{t+1}||_2^2 \leq \big(1 - \frac{\sigma_{min}(A)^2}{||A||_F^2}\big)^t E(||e_1||_2^2)$$

(d) (1 point) Assuming zero initialization, $x_0 = 0$, how many iterations are needed to obtain $\mathbf{E}||x_t - x^*||_2^2 \leq 10^{-5}$? Your answer should depend on $||x^*||_2$, $||A||_F$, $\sigma_{min}(A)$. What is the computational complexity (number of real number multiplications) in $Big\ O$ notation?
Solution:

Let $a = 1 - \frac{\sigma_{min}(A)^2}{||A||_F^2}$.
Then we have $\mathbf{E}||x_t - x^*||_2^2 \leq 10^{-5} \leq a^{t-1}||x^{*2}||_2^2$, or $a^{t-1} \geq 10^{-5}/||x^*||_2^2$. Then $t \geq 1 + \log_a(10^{-5}/||x^*||_2^2)$ or $t \geq 1 + \frac{\log(10^{-5}/||x^*||_2^2)}{\log(a)}$

The computational complexity is:

$$C((a_j - b_j)a_j/m) = O(n(n+1))$$
$$C\big(\frac{m}{||A||_F^2}\big) = O(1 + mn)$$

So, the computational complexity is $O(\frac{\log(10^{-5}/||x^*||_2^2)}{\log(a)} n(n+1) + mn)$

3.3 (4 points)

(3 points) Log-optimal portfolio optimization using return oracle. We consider the portfolio optimization problem

$$\begin{aligned} &\text{maximize } \mathbf{E}_r log(r^T x) \\ &\text{subject to } \mathbf{1}^T x = 1,\ x \succeq 0 \end{aligned}$$

with variable (portfolio weights) $x \in R^n$. The expectation is over the distribution of the (total) return vector $r \in R^n_{++}$, which is a random variable. (Although not relevant in this problem, the log-optimal portfolio maximizes the long-term growth of an initial investment, assuming the investments are re-balanced to the log-optimal portfolio after each investment period, and ignoring transaction costs.) In this problem we assume that we do not know the distribution of r (other than that we have $r \succ 0$ almost surely). However, we have access to an oracle that will generate independent samples from the return distribution. (Although not relevant, these samples could come from historical data, or stochastic simulations, or a known or assumed distribution.)

(a) (1 point) Explain how to use the (projected) stochastic subgradient method, using one return sample for each iteration, to

find (in the limit) a log-optimal portfolio. Describe how to carry out the projection required, and how to update the portfolio in each iteration.

Hint. Note that projecting $x^{(k)}$ onto the constraints involves projecting onto a simplex, or solving the optimization problem

$$\begin{aligned} &\text{minimize } (1/2)||z - x^{(k)}||_2^2 \\ &\text{subject to } 1^T z = 1,\ z \succeq 0 \end{aligned} \tag{1}$$

with variable z. One way to solve the problem is to introduce a dual variable ν for the equality constraint and write the (partial) Lagrangian as

$$L(z, \nu) = (1/2)||z - x^{(k)}||_2^2 + \nu(1^T z - 1)$$

with $\mathbf{dom} L(z, \nu) \in R_+^n \times R$ (in other words, the inequality constraint $z \succeq 0$ is implicit). Consider the single variable function $g(\nu)$ obtained by minimizing $L(z, \nu)$ over z. The value ν^* that maximizes $g(\nu)$ can be found using bisection, and the solution z^* to problem 1 can be found given ν^*.

Solution:

As mentioned in the subgradient method slides, the projection method given by:

$$x^{(k+1)} = P(x^{(k)} - \alpha_k g^{(k)})$$

where P is the Euclidean projection on the constraints, and $g^{(k)} \in \partial f(x^{(k)})$. The Euclidean projection on the linear subspace $\mathbf{1}^T x = 1$ is given by the minimization problem, as mentioned in the corresponding hint:

$$\begin{aligned} &\text{minimize } (1/2)||z - x^{(k)}||_2^2 \\ &\text{subject to } 1^T z = 1,\ z \succeq 0 \end{aligned} \tag{2}$$

We can find the equation for the z^* using the dual problem, as described in the hint.

$$\nabla_z L(z, \nu) = z - x^{(k)} + \nu z$$

then value z which minimizes $L(z, \nu)$ is

$$z = x^{(k)}/(1 + \nu)$$

substituting this value in the Lagrangian we get:

$$L(\nu) = (1/2)\left(\frac{\nu}{\nu + 1}\right)^2 ||x^{(k)}||_2^2 + \nu\big(\frac{1}{\nu + 1}\mathbf{1}^T x^{(k)} - 1\big)$$

and the *argmin* of $L(\nu)$ can be found using bisection. After that we can calculate the z^*, which is the Euclidean projection of $x^{(k)} - \alpha_k g^{(k)}$ on the problem constraints.

b) (1 point) Implement the method and run it on the problem with $n = 10$ assets, with return sample `oracle log_opt_return_sample` in the

file `log_opt_return_sample.m`. This function called with argument m returns an $n \times n$ matrix whose columns are independent return samples. We have also provided a function to project onto the simplex in p `rojToSmplx.m`. You are welcome to look inside `log_opt_return_sample.m` to see how we are generating the sample. The distribution is a mixture of two log-normal distributions; you can think of one as the standard return model and the other as the return model in some abnormal regime. However, your stochastic subgradient algorithm can only call `log_opt_return_sample(1),` once per iteration; you cannot use any information found inside the

le in your implementation. To get a Monte Carlo approximation of the objective function value, you can generate a block of, say, 105 samples (using `R_emp=log_opt_return_sample(1e5),` which only needs to be done once), and then use `obj_hat = mean(log(R_emp'*x))` as your estimate of the objective function. Plot the (approximate) objective value versus iteration, as well as the best approximate objective value obtained up to that iteration. (Note that evaluating the objective will require far more compu- tation than each stochastic subgradient step.) You may need to play around with the step size selection in your method to get reasonable convergence. Remember that your objective value evaluation is only an approximation.

I can't solve problems (b) and (c) because I don't have the required files.