MIT 6.7220/15.084 — Nonlinear Optimization Spring 2025

Problem Set 5

Due: Apr 29th 2025 11:59pm ET

Collaboration policy. We encourage working together whenever possible: in the recitations, problem sets, and general discussion of the material and assignments. Keep in mind, however, that for the problem sets the solutions you hand in should reflect your own understanding of the class material, and should be written solely by you. It is not acceptable to copy (in whole or in part) a solution that somebody else has written.

1. Distance-generating functions [20pts]

Given a distance-generating function $\varphi(x) : \Omega \to \mathbb{R}$, the Bregman divergence centered in $y \in \Omega$ is the function $\mathrm{D}_\varphi(x \parallel y)$ defined as

$$\mathrm{D}_\varphi(x \parallel y) := \varphi(x) - \varphi(y) - \langle \nabla\varphi(y), x - y\rangle.$$

The mirror descent algorithm is defined by the update

$$x_{t+1} := \mathrm{Prox}_\varphi(\eta\nabla f(x_t), x_t) = \arg\min_{x\in\Omega}\left\{\eta\langle\nabla f(x_t), x\rangle + \mathrm{D}_\varphi(x \parallel x_t)\right\}.$$

Given the specified distance-generating function $\varphi(x)$ and domain Ω for each question below, compute the closed-form expression of x_{t+1} in the mirror descent update. Your expression should be formulated in terms of $\nabla f(x_t)$, x_t and η.

(1.a) [5pts] $\varphi(x) = \frac{1}{2}\|Ax\|_2^2$ for an invertible matrix $A \in \mathbb{R}^{n\times n}$; $\Omega = \mathbb{R}^n$.

Solution.

$$\begin{aligned} x_{t+1} &= \arg\min_{x\in\Omega}\langle\eta\nabla f(x_t), x\rangle + D_\varphi(x\|x_t) \\ &= \arg\min_{x\in\Omega}\langle\eta\nabla f(x_t), x\rangle + \frac{1}{2}x^\top A^\top Ax - x^\top A^\top Ax_t \\ &= x_t - \eta(A^\top A)^{-1}\nabla f(x_t) \end{aligned}$$

◂

(1.b) [5pts] $\varphi(x) = \sum_{i=1}^n -\log x_i$; $\Omega = \{x \in \mathbb{R}^n : x_i \geq 1 \ \forall i \in [n]\}$.

Solution.

$$\begin{aligned} x_{t+1} &= \arg\min_{x\in\Omega}\langle\eta\nabla f(x_t), x\rangle + D_\varphi(x\|x_t) \\ &= \arg\min_{x\in\Omega}\langle\eta\nabla f(x_t), x\rangle - \sum_i \log x_i + \sum_i \frac{x_i}{(x_t)_i} \end{aligned}$$

This function is separable, minimizing for a component i implies minimizing

$$\arg\min_{x_i\geq 1}\eta\nabla_i f(x_t)x_i - \log(x_i) + \frac{x_i}{(x_t)_i}$$

Setting the gradient to zero yields $\eta\nabla_i f(x_t) - \frac{1}{x_i} + \frac{1}{(x_t)_i} = 0$

thus we get

$$(x_{t+1})_i = \max\left(1, \frac{(x_t)_i}{1 + \eta(x_t)_i\nabla_i f(x_t)}\right)$$

Because the function is strictly convex, if the minimizer of the unconstrained problem is less than 1, then the constrained problem's minimizer is 1.

◂

(1.c) [10pts] $\varphi(x) = \sum_{i=1}^{n} x_i \log x_i$; $\Omega = \{x \in \mathbb{R}^n : 1^\top x = 1, x \geq 0\}$.

Solution.

$$\begin{aligned} x_{t+1} &= \arg\min_{x \in \Omega} \langle \eta \nabla f(x_t), x \rangle + D_\varphi(x \| x_t) \\ &= \arg\min_{x \in \Omega} \langle \eta \nabla f(x_t), x \rangle + \sum_i x_i \log\left(\frac{x_i}{(x_t)_i}\right) - x_i \\ &= \arg\min_{x \in \Omega} \langle \eta \nabla f(x_t), x \rangle + \sum_i x_i \log\left(\frac{x_i}{(x_t)_i}\right) \end{aligned}$$

Since $\sum_i x_i = 1$, we can ignore it. Like homework 1, we know that the minimizer exists and is not on the boundary.

The gradient of the objective (wrt to x) is $\eta \nabla f(x_t) + \log\left(\frac{x}{x_t}\right) + 1$. The normal cone of the simplex (outside of its boundary) is $\mu \mathbb{1}$ for a real μ thus $\exp(\mu - \eta \nabla f(x_t) + \log(x_t) - 1) = x$. This implies that $x_{t+1} \propto x_t \exp \nabla f(x_t)$ and that μ is chosen such that x_{t+1} is in the unit simplex. ◂

2. Stochastic gradient descent and strong convexity [20pts]

Consider the stochastic gradient descent update

$$x_{t+1} = x_t - \eta_t \tilde{\nabla} f(x_t)$$

on a differentiable, μ-strongly convex function f, *i.e.*,

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2}\|x - y\|_2^2 \quad \forall x, y \in \mathbb{R}^n.$$

We assume there exists constant $\tilde{L}_f > 0$ such that for any $t \geq 0$,

$$\mathbb{E}_t\left[\tilde{\nabla} f(x_t)\right] = \nabla f(x_t) \quad \text{and} \quad \mathbb{E}_t\left[\left\|\tilde{\nabla} f(x_t)\right\|_2^2\right] \leq \tilde{L}_f^2. \tag{1}$$

(2.a) [10pts] Suppose $x^\star$ is the minimizer of f. Prove that

$$\mathbb{E}_t\left[\left\|x_{t+1} - x^\star\right\|_2^2\right] \leq (1 - \mu\eta_t)\|x_t - x^\star\|_2^2 - 2\eta_t(f(x_t) - f(x^\star)) + \left(\eta_t \tilde{L}_f\right)^2 \tag{2}$$

by expanding $\left\|x_{t+1} - x^\star\right\|_2^2$ and making use of the strong convexity of f.

Solution.

$$\begin{aligned}
&\mathbb{E}\left[\|x_{t+1} - x^*\|^2\right] \\
&= \mathbb{E}\left[\|x_t - \eta_t \tilde{\nabla} f(x_t) - x^*\|^2\right] \\
&= \|x_t - x\|^2 - 2\eta_t \left\langle \mathbb{E}\left[\tilde{\nabla} f(x_t)\right], x_t - x^* \right\rangle + \eta_t^2 \mathbb{E}\left[\|\tilde{\nabla} f(x_t)\|^2\right] \\
&\leq \|x_t - x\|^2 - 2\eta_t \langle \nabla f(x_t), x_t - x^* \rangle + \eta_t^2 \tilde{L}_f^2
\end{aligned} \tag{3}$$

Since f is strongly convex,

$$f(x^*) \geq f(x_t) + \langle \nabla f(x_t), x^* - x_t \rangle + \frac{\mu}{2}\,\|x\_x^*\|^2, \tag{4}$$

holds. Plugging it into the above inequality yields

$$\mathbb{E}\left[\|x_{t+1} - x^*\|^2\right] \leq (1 - \mu\eta_t)\|x_t - x\|^2 - 2\eta_t(f(x_t) - f(x^*)) + \eta_t^2 \tilde{L}_f^2 \tag{5}$$

◂

(2.b) [10pts] Set the stepsize $\eta_t := \frac{2}{\mu(t+1)}$. Prove that

$$\mathbb{E}\left[f\left(x_T^{\text{best}}\right)\right] \leq f(x^\star) + \frac{2\tilde{L}_f^2}{(T+1)\mu}, \tag{6}$$

where $f\left(x_T^{\text{best}}\right) := \min_{t=0,\ldots,T} f(x_t)$.

▸ *Hint:* Consider a weighted sum of (2).

Solution. Rearranging the above inequality and plugging $\eta_t = \frac{2}{\mu(t+1)}$ yields:

$$f(x^t) - f(x^*) \leq$$
$$\left(\frac{\mu(t-1)}{4}\right)\|x_t - x^*\| - \left(\mu\frac{t+1}{4}\right)\mathbb{E}[\|x_{t+1} - x^*\|^2] + \frac{\tilde{L}_f^2}{\mu(t+1)} \tag{7}$$

Multiplying both sides by t and taking the expectation wrt to x_t yields

$$t\mathbb{E}[f(x^t) - f(x^*)] \leq$$
$$\left(\mu\frac{t(t-1)}{4}\right)\mathbb{E}[\|x_t - x^*\|] - \left(\mu t\frac{t+1}{4}\right)\mathbb{E}[\|x_{t+1} - x^*\|^2] + t\frac{\tilde{L}_f^2}{\mu(t+1)} \tag{8}$$

Summing over $t = 1, ..., T$ gives

$$\sum_{t=0}^{T} t\mathbb{E}[f(x^t) - f(x^*)] \leq$$
$$-\left(\mu T\frac{T+1}{4}\right)\mathbb{E}[\|x_{T+1} - x^*\|^2] + \tilde{L}_f^2 \sum \frac{t}{t+1} \leq T\frac{\tilde{L}_f^2}{\mu} \tag{9}$$

Therefore, it follows that

$$\frac{T(T+1)}{2}\mathbb{E}[f(x_{\text{best}}) - f(x^*)] \leq T\frac{\tilde{L}_f^2}{\mu} \tag{10}$$

◂

3. Least-square problem on the simplex [35pts]

Consider a least square problem on a probability simplex:

$$\begin{aligned} \min_{x} \quad & f(x) := \tfrac{1}{2}\|Ax - b\|_2^2 \\ \text{s.t.} \quad & x \geq 0 \\ & 1^\top x = 1. \end{aligned} \tag{11}$$

where the data $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$ are given. Specifically, we set $m = 500$ and $n = 200$.

The next four questions are coding problems, and you can use any language you are familiar with. We recommend using Python, and we have provided a template on Canvas to help you answer these questions.

(3.a) [5pts] Solve the optimal value f^* of this problem via a solver (*e.g.*, by using CVXPY to call a solver).

> *Solution.* Please refer to the notebook ◂

For the following questions (3.b)-(3.d), you are NOT allowed to use any libraries that assist with modeling or solving (such as *CVXPY, PyTorch, sklearn*, etc.). Using Numpy is allowed. For each algorithm, please perform 30 iterations and then plot the relationship between the number of iterations and the difference in objective value $\frac{f(x^t)-f^*}{\max\{1,|f^*|\}}$.

(3.b) [10pts] Apply the projected gradient descent algorithm to solve (11), starting from the initial point $x_0 = \left(\frac{1}{n}, ..., \frac{1}{n}\right)$. How did you choose an appropriate stepsize?

> *Solution.* Note that $f(x)$ is $\lambda_{\max}(A^\top A)$-smooth, η can be chosen to be $\frac{1}{\lambda_{\max}(A^\top A)}$ ◂

(3.c) [10pts] Apply the projected gradient descent algorithm with Nesterov Acceleration, defined as,

$$\begin{aligned} x_{t+1} &:= \Pi_\Omega(y_t - \eta \nabla f(y_t)), \quad \text{where} \\ y_t &:= x_t + \frac{t-2}{t+1}(x_t - x_{t-1}), \\ x_0, x_{-1} &:= \left(\frac{1}{n}, ..., \frac{1}{n}\right), \end{aligned}$$

to solve (11).

> *Solution.* Please refer to the notebook ◂

(3.d) [10pts] Apply the mirror descent algorithm with distance-generating function $\varphi(x) = \sum_{i=1}^{n} x_i \log x_i$ to solve (11).

Solution. Please refer to the notebook ◂

4. MNIST [25pts]

In this assignment, you will implement logistic regression to classify handwritten digits from the MNIST dataset. The MNIST dataset consists of grayscale images of handwritten digits, where each image is 28x28 pixels in size.

Given a training dataset $D = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(n)}, y^{(n)})\}$, where $x^{(i)} \in \mathbb{R}^d$ represents the input features (pixel values) of the i-th image, and $y^{(i)} \in \{0, 1, ..., 9\}$ represents the corresponding digit label, the goal is to learn a logistic regression model with weights $W \in \mathbb{R}^{K \times (d+1)}$ that maps input features to class probabilities. Here $K = 10$ denotes the number of classes.

The logistic regression model computes the raw scores for each class using the equation $z = Wx$, where $x \in \mathbb{R}^{d+1}$ is the input feature vector (with an added bias term), and $z \in \mathbb{R}^K$ is the vector of raw scores.

The softmax function is applied to convert raw scores into class probabilities

$$\hat{y} = \text{softmax}(z),$$

where $\hat{y} \in \mathbb{R}^K$ is the vector of predicted probabilities, and

$$\text{softmax}(z)_i := \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \qquad i = 1, ..., K.$$

The objective is to minimize the negative log-likelihood

$$L(W) := -\frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} \Delta(i, k) \log \hat{y}_k^{(i)},$$

where $\Delta(i, k)$ is 1 if $y^{(i)} = k$ and 0 otherwise.

(4.a) [5pts] Derive the gradient of the loss function $L(W)$ with respect to the weights W.

> *Solution.* Let w_j be the $j-$th row of the matrix W,
>
> $$\nabla_{w_j} L = -\frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} \delta(i, k) \nabla_{w_j} \log\left(\hat{y}_k^{(i)}\right).$$
>
> It is clear that
>
> $$\nabla_{w_j} \log\left(\hat{y}_k^{(i)}\right) = \begin{cases} x^{(i)}(1 - \hat{y}^{(i)}) & \text{if } j = k \\ -x^{(i)}\hat{y}^{(i)} & \text{otherwise.} \end{cases}$$
>
> Thus we get

$$\nabla_{w_j} L = -\frac{1}{n}\sum_{i=1}^{n} -\frac{1}{n}\sum_{i=1}^{n} x^{(i)}\left(\Delta(i,j) - \hat{y}_j^{(i)}\right).$$

For the coding questions, refer to the notebooks.

◂

The next two questions are coding problems, and you can use any programming language you are familiar with. We recommend using Python, and we have provided a template on Canvas to help you answer these questions. You are required to write your own code for *training* the logistic regression model without using any pre-existing libraries or frameworks. However, you can use libraries to help you preprocess the data, evaluate accuracy, and plot the results.

(4.b) [10pts] Implement logistic regression for `MNIST` and use stochastic gradient descent (SGD) to minimize the loss function. Experiment with different batch sizes and learning rates. Plot the train accuracy as a function of the number of iterations for each combination of hyperparameters.

(4.c) [10pts] Extend your implementation to include momentum in the SGD updates. Experiment with different batch sizes, learning rates, and momentum values. Plot the train accuracy as a function of the number of iterations for each combination of hyperparameters.

6.7220 Nonlinear Optimization

Spring 2025