Labs
Optimization for Machine Learning
Spring 2025

EPFL
School of Computer and Communication Sciences
Nicolas Flammarion
github.com/epfml/OptML_course

Problem Set 4, March 21, 2025 (Subgradient Descent)

Subgradient Descent

Solve Exercises 27, 28, 29, 30 and 32 from the lecture notes.

Random Walks

Gradient descent turns up in a surprising number of situations which apriori have nothing to do with optimization. In this exercise, we will see how performing a random walk on a graph can be seen as a special case of gradient descent.

We are given an *undirected* graph $G(V, E)$ with vertices $V = [n]$ labelled 1 through n, and edges $E \subseteq [n]^2$ such that if $(i, j) \in E$, then $(j, i) \in E$. Further, we assume that the graph is *regular* in the sense that every edge has the same degree. Let d be the degree of each node such that if we denote $\mathcal{N}(i) = \{j : (i, j) \in E\}$ to be the neighbors of i, then $|\mathcal{N}(i)| = d$. We assume that every node is connected to itself and so $(i, i) \in \mathcal{N}(i)$.

Now we start our random walk from node 1, jumping randomly from a node to its neighbor. More precisely, suppose at time step t we are at node i_t. Then i_{t+1} is picked uniformly at random from $\mathcal{N}(i)$. If we run this random walk for a large enough T steps, we expect that $\Pr(i_T = j) = 1/n$ for any $j \in [n]$. This is called the stationary distribution.

Problem A. Let us represent the position at time step t in the graph with $\mathbf{e}_{i_t} \in \mathbb{R}^n$ where the i_tth coordinate is 1 and all others are 0. Then, the vector $\mathbf{x}_t = \mathbb{E}[\mathbf{e}_{i_t}]$ denotes the probability distribtion over the n nodes of the graph. Further, let us denote $\mathbf{G} \in \mathbb{R}^{n \times n}$ be the transition probability matrix such that

$$\mathbf{G}_{i,j} = \begin{cases} \frac{1}{d} & \text{if } (i,j) \in E \\ 0 & \text{otherwise}\,. \end{cases}$$

Show that

$$\mathbf{x}_{t+1} = \mathbf{G}\mathbf{x}_t \tag{1}$$

Problem B. Simulate the random walk above over a torus and confirm that we indeed converge to a uniform distribution over the nodes. What is the *rate* at which this convergence occurs?

Follow the Python notebook provided here:

colab.research.google.com/github/epfml/OptML_course/blob/master/labs/ex04/template/notebook_lab04.ipynb

Problem C. Define $\boldsymbol{\mu} := \frac{1}{n}\mathbf{1}_n$ be a vector of all $1/n$, and a objective function $f : \mathcal{S} \to \mathbb{R}$ as

$$f(\mathbf{x}) = (\mathbf{x} - \boldsymbol{\mu})^\top (\mathbf{I} - \mathbf{G})(\mathbf{x} - \boldsymbol{\mu})\,,$$

defined over the subspace $\mathcal{S} \subseteq \mathbb{R}^n$ where $\mathcal{S} = \{\mathbf{v} : \mathbf{1}_n^\top \mathbf{v} = 1\}$.

1. Show that f defined above is convex and compute its smoothness constant.

2. Show that running gradient descent on f with the correct step-size is equivalent to the random walk step (1).

3. Prove that $\mathbf{x}_t$ converges to the distribution $\boldsymbol{\mu}$ at a linear rate i.e. for the random walk on a torus with n nodes,
$$\|\mathbf{x}_t - \boldsymbol{\mu}\|_2^2 \leq \left(1 - \frac{1}{n}\right)^t \|\mathbf{x}_0 - \boldsymbol{\mu}\|_2^2 \leq \left(1 - \frac{1}{n}\right)^t .$$
Hint: Use that the two largest eigenvalues of $\mathbf{G}$ are 1 and $1 - \frac{1}{n}$. Also $\mathbf{G}\boldsymbol{\mu} = \boldsymbol{\mu}$ and so $\boldsymbol{\mu}$ is the eigenvector corresponding to eigenvalue 1.