ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Department of Computer Science
Institute of Theoretical Computer Science
Bernd Gärtner, David Steurer

Optimization for Data Science **Special Assignment 1** **FS19**

- The solution is due on **Sunday, April 28, 2019** by **11:59 pm**. Please send your solution as PDF to `hung.hoang@inf.ethz.ch`. After receiving your file, we will send you a confirmation on the following work day, and at the latest on Monday, April 29th. Make sure you receive this confirmation, otherwise complain timely.
- Please solve the exercises carefully and then write a nice and complete exposition of your solution using a computer, where we strongly recommend to use LaTeX. A tutorial can be found at `http://www.cadmo.ethz.ch/education/thesis/latex`. Handwritten solutions will not be graded!
- For geometric drawings that can easily be integrated into LaTeX documents, we recommend the drawing editor IPE, retrievable at `http://ipe7.sourceforge.net/` in source code and as an executable for Windows.
- Keep in mind the following premises:
  - When writing in English, write short and simple sentences.
  - When writing a proof, write precise statements.

  The conclusion is, of course, that your solution should consist of sentences that are short, simple, and precise!
- This is a theory course, which means: if an exercise does not explicitly say "you do not need to prove your answer" or "justify intuitively", then a formal proof is **always** required. You can of course refer in your solutions to the lecture notes and to the exercises, if a result you need has already been proved there.
- We would like to stress that the ETH Disciplinary Code applies to this special assignment as it constitutes part of your final grade. The only exception we make to the Code is that we encourage you to verbally discuss the tasks with your colleagues. It is strictly prohibited to share any (hand)written or electronic (partial) solutions with any of your colleagues. We are obligated to inform the Rector of any violations of the Code.
- There will be two special assignments. Both of them will be graded and the average grade will contribute 20% to your final grade. That is, if S_1 and S_2 are the (unrounded) grades from your respective special assignments and E is the (unrounded) grade from your exam, then your final grade will be $0.1 \cdot S_1 + 0.1 \cdot S_2 + 0.8 \cdot E$, rounded to the nearest quarter (rounding is only applied in this last step). If you do not hand in one of the special assignments, it will be counted with a grade of 1.0.
- As with all exercises, the material of the special assignments is relevant for the exam.

Approximate testing of intersections of balls

We are interested in testing whether a set of balls in $\mathbb{R}^d$ have a common intersection point. The idea is to use the machinery developed in class to find an algorithm that approximately answers the above question. Let $B_1, B_2, \ldots, B_n \subset \mathbb{R}^d$ be a collection of n d-dimensional balls, where each B_i is represented by a center $\mathbf{c}_i$ and a radius r_i such that $B_i = \{x : \|x - \mathbf{c}_i\| \leq r_i\}$, where $\|\cdot\|$ denotes the Euclidean norm. Intuitively, our problem is to decide if there exists a point that is "very close" to each B_i, or if every point in the $\mathbb{R}^d$ is sufficiently far away from at least one of the balls. In the former case, we may not be able to differentiate if the point is a common intersection point, or just very close to all the given balls. In the latter case however, we can guarantee that there is no common intersection point.

We proceed now to formalize this problem. Given two compact subsets $X, Y \subset \mathbb{R}^d$, their *distance* is denoted by $d(X, Y) = \min\{\|x - y\| : x \in X, y \in Y\}$. If $X = \{x\}$ consists of a single point, we simplify the notation and assume that $d(x, Y) = d(\{x\}, Y)$.

For $i \in [n]$, let $h_i(x) = d(x, B_i)$, and let $h : \mathbb{R}^d \to \mathbb{R}$ be the function such that

$$h(x) = \max_{i \in [n]} h_i(x).$$

The ε-*intersection problem* is defined as follows: Given $\varepsilon > 0$ and a collection of d-dimensional balls $B_1, B_2, \ldots, B_n \subset \mathbb{R}^d$ represented by the centers and radii, compute either a point $\bar{x} \in \mathbb{R}^d$ such that $h(\bar{x}) < \varepsilon$, or decide that $\cap_{i=1}^n B_i = \emptyset$. That is, show that for every $x \in \mathbb{R}^d$, it holds that $h(x) > 0$.

Note that in some cases one may be able to compute both a point $\bar{x}$ such that $h(\bar{x}) < \varepsilon$, and also have a certificate of no intersection. In this case both are valid solutions to the ε-intersection problem.

In this assignment you are asked to design algorithms to solve the ε-intersection problem using the techniques discussed in class.

Assignment 1. *Given* $i \in [n]$ *and* $x \in \mathbb{R}^d$*, let*

$$g_i(x) = \begin{cases} \mathbf{0} & \text{if } x \in B_i \\ \frac{x - \mathbf{c}_i}{\|x - \mathbf{c}_i\|} & \text{otherwise.} \end{cases}$$

Show that $g_i(x)$ *is a subgradient of* $h_i(x)$*, i.e.,* $g_i(x) \in \partial h_i(x)$*. Is* h_i *differentiable?*

Assignment 2. *For* $x \in \mathbb{R}^d$*, show that there is* $j \in [n]$ *such that* $g_j(x)$ *is a subgradient of* $h(x)$*, i.e.,* $g_j(x) \in \partial h(x)$*. Describe how to compute such subgradient and provide the running time of this procedure.*

Assignment 3. *Show that each* h_i *is convex and 1-Lipschitz. Then use this to show that* h *is convex and 1-Lipschitz.*

Assignment 4. *Assume that you are given* $x_0 \in \mathbb{R}^d$ *such that* $\|x_0 - x^*\| \le R$ *for some constant* $R \in \mathbb{R}$*, where* x^* *is a global minimum of* h.[1] *Show that there exists* $\gamma \in \mathbb{R}$ *such that after* T *iterations of subgradient descent on* $h(x)$ *we get:*

$$\min_{t \in \{0,\dots,T-1\}} (h(x_t) - h(x^*)) \le \frac{R}{\sqrt{T}}.$$

Assignment 5. *Given* $\varepsilon > 0$*, show that* $O(n/\varepsilon^2)$ *time suffices to solve the* ε*-intersection problem for the set of* B_i*'s.*

While the above algorithm is independent of the dimension, the dependency on ε is really bad for practical applications. In the next section we look at improving this dependency, at the expense of having a slightly worse dependency on n.

For $i \in [n]$, let $f_i(x) = \frac{1}{2} d(x, B_i)^2$, and let $f : \mathbb{R}^d \to \mathbb{R}$ be the function such that

$$f(x) = \sum_{i=1}^{n} f_i(x).$$

You may assume that f_i is convex and differentiable for each $i \in [n]$. Because the sum of convex and differentiable functions is also convex and differentiable, one can also prove that f is convex and differentiable. For this assignment you may assume these facts without a proof. We are now interested in computing the gradient of f to be able to apply gradient descent.

Assignment 6. *Using elementary calculus (e.g. partial derivatives) show that*

$$\nabla f_i(x) = \begin{cases} 0 & \text{if } x \in B_i \\ \left(1 - \frac{r_i}{\|x - c_i\|}\right)(x - c_i) & \text{otherwise.} \end{cases}$$

Show how to compute the gradient $\nabla f(x)$*. How much time is needed to compute this gradient?*

The following is a helpful result that will allow us to prove the smoothness of the function f.

Assignment 7. *For* $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$ *such that* $\|\mathbf{u}\| = \|\mathbf{v}\|$ *and for real numbers* $\alpha, \beta \ge 1$,

$$\|\mathbf{u} - \mathbf{v}\| \le \|\alpha \mathbf{u} - \beta \mathbf{v}\|.$$

Assignment 8. *Using Assignment 7, show that each for each* f_i*, it holds that*

$$\|\nabla f_i(x) - \nabla f_i(y)\| \le 2\|x - y\|.$$

That is, f_i *is smooth with parameter 2 (Lemma 2.4). Using this fact and Lemma 2.5 we also get that* f *is smooth with parameter* $2n$*. Is* f *always strongly convex?*

Assignment 9. *Given* $\varepsilon > 0$*, show that* $O(\frac{n^{3/2}}{\varepsilon})$ *time suffices to solve the* ε*-intersection problem for the set of* B_i*'s. Use accelerated gradient descent to show this bound.*

[1] For this specific problem, one can provide explicit bounds on R in terms of the radii and positions of the centers of the balls. For example, one can compute the bounding box of the set of centers, and choose x_0 as the center of this box.