AM 221: Advanced Optimization	Spring 2018
Kojin Oshiba	*Problem Set 2 — Due Wed, Feb 7th at 23:59*

Instructions: All your solutions should be prepared in LaTeX and the PDF and .tex should be submitted to canvas. For each question, the best and correct answers will be selected as sample solutions for the entire class to enjoy. If you prefer that we do not use your solutions, please indicate this clearly on the first page of your assignment.

The programming parts can be written in the programming language of your choice and the code should be submitted alongside your solutions.

1. Convex Sets. Prove or give a counterexample:

a. The intersection of convex sets is a convex set.

b. A half-space is a convex set.

c. Every polyhedron is a convex set (remember that a polyhedron is the feasible set of a linear program, *i.e.* it is defined by a finite set of linear inequalities)

a. If the intersection is an empty set or has a single element, the intersection is convex by definition. If not, take any two points in the intersection. Since the two points must be contained in the same convex set (because we are taking the intersection), the intersection of convex sets must be convex.

b. WLOG, let the half space be $H = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{a}^T\mathbf{x} \geq \alpha\}$. Let $\mathbf{x}_1, \mathbf{x}_2 \in H$. Then, $\forall \lambda \in [0, 1]$

$$\begin{aligned} \mathbf{a}^T(\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) &= \lambda \mathbf{a}^T\mathbf{x}_1 + (1-\lambda)\mathbf{a}^T\mathbf{x}_2 \\ &\geq \lambda\alpha + (1-\lambda)\alpha = \alpha, \end{aligned}$$

Hence, $\lambda \mathbf{x}_1 + (1-\lambda)\mathbf{x}_2 \in H, \forall \lambda \in [0, 1]$. Therefore, it is a convex set.

c. Polyhedron can be defined as $P = \{\mathbf{x} \in \mathbb{R}^n : A\mathbf{x} \leq \mathbf{b}\}$ for some $A \in \mathbb{R}^{m \times n}$ and some $\mathbf{b} \in \mathbb{R}^m$. Note that $P = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{a}_1^T\mathbf{x} \leq \mathbf{b}, \mathbf{a}_2^T\mathbf{x} \leq \mathbf{b}, ..., \mathbf{a}_n^T\mathbf{x} \leq \mathbf{b}\}$. This means that polyhedron is an intersection of half spaces. Since in questions a, b, we showed that the intersection of convex sets are convex and that half spaces are convex, we have that polyhedron is convex.

2. Convex Hulls. Let us define the *convex hull* of a set X as the smallest (for the partial order defined by inclusion) convex set containing X and denote it by $C(X)$. In other words, there is no other convex set C' such that $X \subseteq C' \subset C(X)$.

a. Show that $C(X)$ is the intersection of all convex sets containing X that is:

$$C(X) = \bigcap \{C \mid X \subseteq C \text{ and } C \text{ is convex}\}$$

b. What is the convex hull of a convex set?

c. Show that when $X = \{\mathbf{x}_1, \ldots, \ldots \mathbf{x}_n\}$ is finite, then $C(X)$ can also be written as:

$$C(X) = \left\{ \sum_{i=1}^{n} \lambda_i \mathbf{x}_i \mid \lambda_i \geq 0,\ 1 \leq i \leq n \text{ and } \sum_{i=1}^{n} \lambda_i = 1 \right\}$$

d. What is the convex hull of two points? three points?

a. Let $Z = \bigcap\{C|X \subseteq C \text{ and } C \text{ is convex}\}$. Z is convex because it is an intersection of convex sets (using the result from 1.a). Suppose $Z \neq C(X)$. Then, there exists a convex set C' such that $X \subseteq C' \subset Z$. However, from how Z is defined, it must be the case that $Z \subseteq C'$. Hence, contradiction. Therefore, $Z = C(X)$.

b. Convex hull of a convex set is itself.

c. Let $Comb(X) = \{\sum_{i=1}^{n} \lambda_i \mathbf{x}_i | \lambda_i \geq 0, 1 \leq i \leq n, \sum_{i=1}^{n} \lambda_i = 1\}$. We would like to show that $C(X) = Comb(X)$. First we show that $Comb(X) \subseteq C(X)$. To show this, it suffices to show that if a set C is convex, then any convex combination of points in C is in C. We prove this by induction. This is because $Comb(X)$ is a set of all convex combination of points in X, which is a subset of a convex set $C(X)$.

$n = 1$ is trivial. $n = 2$ follows from the definition of convex sets. Suppose, for $n = k$, we have $\mathbf{x} = \sum_{i=1}^{k} \lambda_i \mathbf{x}_i \in C$, where $\mathbf{x}_1, ..., \mathbf{x}_k \in C, \sum_{i=1}^{k} \lambda_i = 1, \lambda_i \geq 0 \forall i$ Then, for $n = k + 1$,

$$\begin{aligned}
\mathbf{x} &= \sum_{i=1}^{k+1} \lambda_i \mathbf{x}_i \\
&= \sum_{i=1}^{k} \lambda_i \mathbf{x}_i + \lambda_{k+1} \mathbf{x}_{k+1} \\
&= (1 - \lambda_{k+1}) \sum_{i=1}^{k} \frac{\lambda_i}{1 - \lambda_{k+1}} \mathbf{x}_i + \lambda_{k+1} \mathbf{x}_{k+1}
\end{aligned}$$

Because $\sum_{i=1}^{k} \frac{\lambda_i}{1-\lambda_{k+1}} = 1, \frac{\lambda_i}{1-\lambda_{k+1}} \geq 0\ \forall i, \sum_{i=1}^{k} \frac{\lambda_i}{1-\lambda_{k+1}} \mathbf{x}_i \in C$ from the assumption we made for $n = k$. Hence, simply by the definition of convex sets, $\mathbf{x}$ is also in C. Therefore, we have $Comb(X) \subseteq C(X)$.

So, all we need to show now is that $Comb(X)$ is convex. Because if $Comb(X) \subseteq C(X)$ and $Comb(X)$ is convex, it must be the case that $Comb(X) = C(X)$ by the definiton of $C(X)$. Let $y, z \in Comb(X)$ and $y = \sum_i \mathbf{a}_i \mathbf{x}_i, z = \sum_i \mathbf{b}_i \mathbf{x}_i$. Then,

$cy + (1 - c)z = c \sum_i \mathbf{a}_i \mathbf{x}_i + (1 - c) \sum_i \mathbf{b}_i \mathbf{x}_i = \sum_i \mathbf{d}_i \mathbf{x}_i$ where $\mathbf{d}_i = c\mathbf{a}_i + (1 - c)\mathbf{b}_i$. Hence, $cy + (1 - c)z \in Comb(X)$. Therefore, we proved that $C(X) = Comb(X)$.

d. Convex hull of two points is the straight line connecting the two points. Convex hull of three points is inside of the triangle (edge inclusive) where the three points correspond to the vertices.

3. Strict Convexity of the ℓ_2 norm. Let y be an arbitrary point in $\mathbb{R}^n$ and let us define the function "distance to y" by:

$$d_{\mathbf{y}}(\mathbf{x}) = \|\mathbf{x} - \mathbf{y}\|_2^2, \quad \mathbf{x} \in \mathbb{R}^n$$

Show that the function $d_{\mathbf{y}}$ is strictly convex. Remember that a function f defined over $\mathbb{R}^n$ is strictly convex if and only if for any pair of points $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ with $\mathbf{x} \neq \mathbf{y}$ and any $\lambda \in (0,1)$:

$$f\big(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}\big) < \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{y})$$

$$\begin{aligned}
d_{\mathbf{y}}(\lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2) &= \|\lambda\mathbf{x}_1 + (1-\lambda)\mathbf{x}_2 - \mathbf{y}\|_2^2 \\
&= \|\lambda(\mathbf{x}_1 - \mathbf{y}) + (1-\lambda)(\mathbf{x}_2 - \mathbf{y})\|_2^2 \\
&\leq (\|\lambda(\mathbf{x}_1 - \mathbf{y})\|_2 + \|(1-\lambda)(\mathbf{x}_2 - \mathbf{y})\|_2)^2 \text{ } (\because \text{triangle inequality}) \\
&= (\lambda\|\mathbf{x}_1 - \mathbf{y}\|_2 + (1-\lambda)\|\mathbf{x}_2 - \mathbf{y}\|_2)^2 \\
&< \lambda\|\mathbf{x}_1 - \mathbf{y}\|_2^2 + (1-\lambda)\|\mathbf{x}_2 - \mathbf{y}\|_2^2 \text{ } (\because \text{strict convexity of the quadratic}) \\
&= \lambda d_{\mathbf{y}}(\mathbf{x}_1) + (1-\lambda) d_{\mathbf{y}}(\mathbf{x}_2)
\end{aligned}$$

Hence, we have that $d_{\mathbf{y}}$ is strictly convex.

4. Infeasibility and Unboundedness. Discuss the feasibility and boundedness of the following linear programs:

$$\begin{array}{ll}
\text{maximize} & 2x_2 + x_3 \\
\text{subject to} & x_1 - x_2 \leq 5 \\
& -2x_1 + x_2 \leq 3 \\
& x_1 - 2x_3 \leq 5 \\
& x_1, x_2, x_3 \geq 0
\end{array}
\qquad
\begin{array}{ll}
\text{minimize} & x + y + z + w \\
\text{subject to} & x + 3y + 2z + 4w \leq 5 \\
& 3x + y + 2z + w \leq 4 \\
& 5x + 3y + 3z + 3w = 9 \\
& x, y, z, w \geq 0
\end{array}$$

First LP

Let $A = \begin{pmatrix} 1 & -1 & 0 \\ -2 & 1 & 0 \\ 1 & 0 & -2 \end{pmatrix}$, $\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} 5 \\ 3 \\ 5 \end{pmatrix}$.

$A\mathbf{x} = \mathbf{b}$ has a single solution $\mathbf{x} = \begin{pmatrix} -8 \\ -13 \\ -6.5 \end{pmatrix}$. However, this doesn't satisfy $\mathbf{x} \geq 0$. Hence from Farka's lemma, this LP is infeasible. We can't really discuss boundedness here since LP is infeasible.

Second LP

Let $A = \begin{pmatrix} -1 & -3 & -2 & -4 \\ -3 & -1 & -2 & -1 \\ -5 & -3 & -3 & -3 \\ 5 & 3 & 3 & 3 \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} -5 \\ -4 \\ -9 \\ 9 \end{pmatrix}$, $\mathbf{c} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$, $\mathbf{x} = \begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix}$.

Then the primal LP can be written as:

$min\ \mathbf{c}^T\mathbf{x}$, subject to $A\mathbf{x} \geq \mathbf{b}, \mathbf{x} \geq 0$.

Then, the dual LP can be written as:

$max\ \mathbf{b}^T\mathbf{y}$ subject to $A^T\mathbf{y} = \mathbf{c}, \mathbf{y} \geq 0$

Since row-echelon form of A can be written as $\left(\begin{array}{cccc|c} 1 & 0 & 0 & -0.375 & 0 \\ 0 & 1 & 0 & 1.125 & 0.25 \\ 0 & 0 & 1 & 0.5 & -1 \\ 0 & 0 & 0 & 0 & 2 \end{array}\right)$,

$A^T\mathbf{y} = \mathbf{c}$ doesn't have a solution. Hence, this dual LP is infeasible.

From strong duality, if dual LP is infeasible, primal is either infeasible or unbounded. But since we have shown that the primal is feasible, it must be the case that the primal is unbounded.

5. Linear Classifiers and the Perceptron Algorithm In this problem, we will work on the Iris dataset available at `https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data`. The dataset is a single comma-separated value (CSV) file. The first 4 fields of each line contain the measurements of a sample of Iris flower, the last field is the name of this sample's species of Iris. More information about the dataset can be found at `https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.names`.

The goal of this problem is to construct a classifier to distinguish the *Iris setosa* from the other species of Iris. That is, we want to construct a function f which takes as input the vector $\mathbf{x} \in \mathbb{R}^4$ of measurements of a sample and return $f(\mathbf{x}) \in \{0, 1\}$ such that:

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if the sample is an Iris setosa} \\ 0 & \text{otherwise} \end{cases}$$

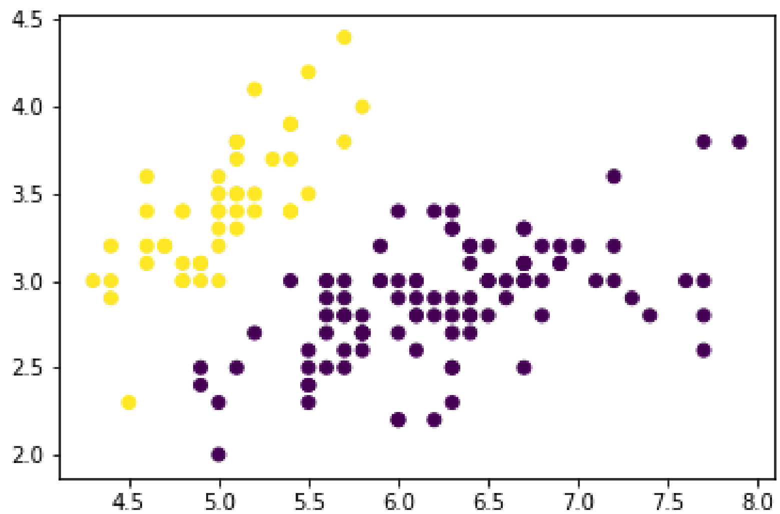
a. Download the dataset, choose a pair of coordinates (that is, any two of the first four fields) and draw a scatter plot of the dataset for this pair of coordinates. The color of the points should be determined by the Iris species.

b. Are the samples of Iris setosa linearly separable (that is, does there exist a separating hyperplane) from the samples from the other species? Is a linearly separable dataset always separable after having been projected on an arbitrary pair of coordinates? (prove of describe a counter-example)?

c. Explain why finding a separating hyperplane as described in the previous part is sufficient to construct a classifier f as described in the introduction of this problem. A classifier constructed in such a way is called a *linear classifier*.

d. Implement the perceptron algorithm in the programming language of your choice. Run your algorithm on the Iris dataset to find a hyperplane separating the Iris setosa samples from the other samples. Report the weights defining the hyperplane as well as the code you wrote.

a. I chose the first two coordinates.



b. To answer the first question, yes, it is linearly separable because it is linearly separable in a subset of all features. If it is separable in a lower dimension, it must be the case that in a higher dimension, it is also separable. To answer the second question, not necessarily, because the converse of what I just said is not always true. Even if the data is linearly separable given all features, it does not mean that it is liearly separable in the projected space. Consider a separating (hyper)plane in 3D space that is parallel to the x and y axes. Then the data is clearly separable in 3D, but when projected onto the x-y 2D plane, the data is no longer separable if, say one point from two clusters have the same x,y coordinates (but different z coordinate).

c. Because $f(\mathbf{x})$ can be defined as an indicator variable that returns 1 if a data point is above the hyperplane and 0 otherwise. If the hyperplane completely separates the two labels, then that means that f classifies the datapoints entirely.

d. The weights are [-0.03152507, -0.19633574, 0.29397583, 0.12135334, 0.04675983]. The code is in ‘hw2.py‘.