EECS 127/227AT Optimization Models in Engineering UC Berkeley Spring 2023

Homework 13

This homework is due at 11 PM on April 27, 2023.

Submission Format: Your homework submission should consist of a single PDF file that contains all of your answers (any handwritten answers should be scanned), as well as a printout of your completed Jupyter notebook(s).

1. **Project Logistics**

   Fill out this form to let us know whether:

   - you plan on doing the project in a group;
   - you plan on doing the project by yourself; or
   - you are not planning to do the project;

   as well as some auxiliary information. Even if you do not plan to do the project, you **must** fill out the form.

   To get credit for the problem, attach a screenshot of the filled-out form to the PDF you submit to Gradescope.

2. Wasserstein distance between distributions

The Wasserstein distance is a measure of distance between probability distributions. The Wasserstein distance can roughly be thought of as the cost of turning one distribution to another distribution by moving probability mass around from one location to another. It is also sometimes called the earth-mover distance, because it may be visualized as the cost of moving a pile of dirt from one configuration to another.

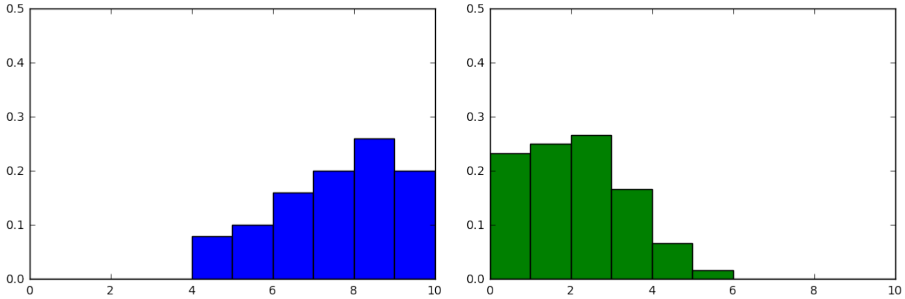

Figure 1: Visualization of μ histogram on left and ν histogram on right.

Let $n \in \mathbb{N}$. We define two discrete probability distributions $\vec{\mu} = (\mu_1, \cdots, \mu_n)$ and $\vec{\nu} = (\nu_1, \cdots, \nu_n)$; that is, $\mu_i, \nu_i \geq 0$ and $\sum_i \mu_i = \sum_i \nu_i = 1$.

We define $C \in \mathbb{R}^{n \times n}$ to be a cost matrix where $c_{ij} \geq 0$ is the cost of transporting one unit of probability mass from location $i \in \{1, \cdots, n\}$ to location $j \in \{1, \cdots, n\}$. We define a matrix $M \in \mathbb{R}^{n \times n}$ where $m_{ij} \geq 0$ denotes the quantity of probability mass to be moved from location i to location j. In summary, if we move m_{ij} units of probability mass from location i to location j, we incur cost $c_{ij} m_{ij}$.

In addition, the M matrix satisfies the following conditions. Row i of M indicates where all the probability mass in location i in the $\vec{\mu}$ distribution ends up. Hence, the sum of all the entries in row i must equal μ_i. Similarly, column j indicates where all the probability mass in location j in the $\vec{\nu}$ distribution came from. Hence, the sum of all the entries in column j must equal ν_j. We can summarize these conditions in math:

$$M\vec{1} = \vec{\mu} \tag{1}$$

$$M^\top \vec{1} = \vec{\nu}, \tag{2}$$

where $\vec{1}$ is a vector of 1s.

(a) What is the total cost of transporting the mass $\vec{\mu}$ into $\vec{\nu}$ by following the transportation plan dictated by the matrix M?

(b) Given the cost matrix C, write the optimization problem of finding the transportation plan $M^\star$ with minimal total cost. What type of optimization problem is it? (LP, QP, $\cdots$?).

Now, we apply the idea of Wasserstein distance to document similarity as illustrated in Fig. 2. Here, our application is that we want to identify words in two different documents that are most similar. This is mostly just a fun application, but may be of interest if you are trying to compare documents that are identical but in different languages. Here we consider a contrived example.

Natural Language Processing techniques have standard tools for converting words into vectors and embedding them in vector spaces, so that we can use machine learning and optimization tools on them. One such embedding is called *word2vec*. Assume we are provided with a *word2vec* embedding for the words in two documents. The word travel cost c_{ij} between word i and word j is the Euclidean distance $\|x_i - x_j\|_2$ in the word embedding space. We can compute the similarity between two documents as the minimum cumulative cost required to move all non-stop words from one document to the other.

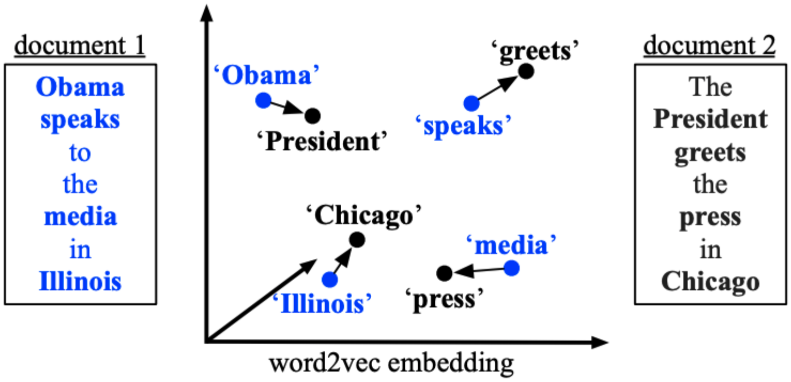

Figure 2: An illustration of the Wasserstein distance. All non-stop words (**bold**) of both documents are embedded into a *word embedding* space. The similarity between the two documents is the minimum cumulative distance that all words in document 1 need to travel to exactly match document 2.

(c) Using the `text_kantorovich.ipynb` Juypter notebook, implement the calculation of the Wasserstein distance in the notebook and use the provided code to visualize the resulting matrix M. Comment on the results.

3. **Linear Quadratic Regulator**

In this question, we will derive the Riccati equation for the LQR model studied in class. We first recall the statement of the LQR problem:

$$\begin{aligned} \min_{\vec{x}_t, \vec{u}_t} \quad & \sum_{t=0}^{T-1} \frac{1}{2}\left(\vec{x}_t^\top Q \vec{x}_t + \vec{u}_t^\top R \vec{u}_t\right) + \frac{1}{2}\vec{x}_T^\top Q \vec{x}_T && (3) \\ \text{s.t.} \quad & \vec{x}_{t+1} = A\vec{x}_t + B\vec{u}_t && (4) \\ & \vec{x}_0 = \vec{x}_{\text{init}} && (5) \end{aligned}$$

where $\vec{x}_t$ is thought of as the state of the system and $\vec{u}_t$ is the control input at time t and the matrices A and B define the dynamics of the system. Here $Q, R \succ 0$ are symmetric positive definite matrices determining how the state and control affect the cost. While the problem can be solved as a quadratic program, we will now take a slightly different approach. We start by defining the functions, J_k for $0 \leq k \leq T$, as follows:

$$\begin{aligned} J_k(\vec{x}) = \min_{\{\vec{u}_t\}_{t=k}^{T-1}} \quad & \sum_{t=k}^{T-1} \frac{1}{2}\left(\vec{x}_t^\top Q \vec{x}_t + \vec{u}_t^\top R \vec{u}_t\right) + \frac{1}{2}\vec{x}_T^\top Q \vec{x}_T && (6) \\ \text{s.t.} \quad & \vec{x}_{t+1} = A\vec{x}_t + B\vec{u}_t && (7) \\ & \vec{x}_k = \vec{x}. && (8) \end{aligned}$$

J_k can be thought of as the minimum cost that we would incur from time k assuming that we start at state $\vec{x}_k = \vec{x}$. We can now decompose J_k for $0 \leq k \leq T-1$ further as follows:

$$\begin{aligned} J_k(\vec{x}) = \min_{\vec{u}_k} \quad & \frac{1}{2}\left(\vec{x}_k^\top Q \vec{x}_k + \vec{u}_k^\top R \vec{u}_k\right) + \min_{\{\vec{u}_t\}_{t=k+1}^{T-1}} \sum_{t=k+1}^{T-1} \frac{1}{2}\left(\vec{x}_t^\top Q \vec{x}_t + \vec{u}_t^\top R \vec{u}_t\right) + \frac{1}{2}\vec{x}_T^\top Q \vec{x}_T && (9) \\ \text{s.t.} \quad & \vec{x}_{t+1} = A\vec{x}_t + B\vec{u}_t && (10) \\ & \vec{x}_k = \vec{x}. && (11) \end{aligned}$$

Note that in particular, the first constraint implies that $\vec{x}_{k+1} = A\vec{x}_k + B\vec{u}_k$. Therefore, the above characterization gives the following decomposition:

$$J_k(\vec{x}) = \min_{\vec{u}} \frac{1}{2}\left(\vec{x}^\top Q \vec{x} + \vec{u}^\top R \vec{u}\right) + J_{k+1}(A\vec{x} + B\vec{u}). \tag{12}$$

This equation is called the *Bellman equation* and is solved by dynamic programming, as we will show in the problem.

We will see that the functions, J_k, are all in fact quadratic functions in $\vec{x}$ and this will give us convenient ways to derive the optimal control inputs at each time.

(a) First, we will show by reverse induction that each of the functions J_k for $0 \leq k \leq T$ are convex quadratics. In particular, prove that $J_k(\vec{x}) = \frac{1}{2}\vec{x}^\top Q_k \vec{x}$ for some $Q_k \succ 0$ and determine the value of Q_k in terms of Q_{k+1}.

HINT: $J_T(\vec{x}) = \frac{1}{2}\vec{x}^\top Q \vec{x}$. Therefore, $Q_T = Q$. Also can use (12) above, and substitute $J_{k+1}(A\vec{x} + B\vec{u}) = \frac{1}{2}(A\vec{x} + B\vec{u})^\top Q_{k+1}(A\vec{x} + B\vec{u})$. Then solve the resulting QP to find the optimal $\vec{u}$.

HINT: You should get the following recursion for Q_k:

$$Q_k = Q + A^\top Q_{k+1} A - A^\top Q_{k+1} B (R + B^\top Q_{k+1} B)^{-1} B^\top Q_{k+1} A. \tag{13}$$

and you may assume that $A^\top Q_{k+1} A - A^\top Q_{k+1} B (R + B^\top Q_{k+1} B)^{-1} B^\top Q_{k+1} A$ is positive semi-definite when you try to establish that Q_k is positive definite.

(b) Now, show that the expression for Q_l is equivalent for the expression obtained by using the Lagrangian. That is, show that Q_l from the previous part is the same as:

$$Q_l = Q + A^\top (Q_{l+1}^{-1} + BR^{-1}B^\top)^{-1}A. \tag{14}$$

HINT: You may find useful the Sherman-Morrison-Woodbury matrix identity:

$$(M + UWV)^{-1} = M^{-1} - M^{-1}U(W^{-1} + VM^{-1}U)^{-1}VM^{-1}. \tag{15}$$

4. Soft-Margin SVM

Consider the soft-margin SVM problem,

$$p^\star(C) = \min_{\vec{w}\in\mathbb{R}^m, b\in\mathbb{R}, \vec{\xi}\in\mathbb{R}^n} \quad \frac{1}{2}\|\vec{w}\|_2^2 + C\sum_{i=1}^{n}\xi_i \tag{16}$$

$$\text{s.t.} \quad 1 - \xi_i - y_i(\vec{x}_i^\top\vec{w} - b) \leq 0, \quad i = 1, 2, \ldots, n \tag{17}$$

$$-\xi_i \leq 0, \quad i = 1, 2, \ldots, n, \tag{18}$$

where $\vec{x}_i \in \mathbb{R}^m$ refers to the i^{th} training data point, $y_i \in \{-1, 1\}$ is its label, and $C \in \mathbb{R}_+$ (i.e. $C > 0$) is a hyperparameter. Let α_i denote the dual variable corresponding to the inequality $1 - \xi_i - y_i(\vec{x}_i^\top\vec{w} - b) \leq 0$ and let β_i denote the dual variable corresponding to the inequality $-\xi_i \leq 0$. The Lagrangian is then given by

$$\mathcal{L}(\vec{w}, b, \vec{\xi}, \vec{\alpha}, \beta) = \frac{1}{2}\|\vec{w}\|_2^2 + C\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i(1 - \xi_i - y_i(\vec{x}_i^\top\vec{w} - b)) - \sum_{i=1}^{n}\beta_i\xi_i. \tag{19}$$

Suppose $\vec{w}^\star, b^\star, \vec{\xi}^\star, \vec{\alpha}^\star, \beta^\star$ satisfy the KKT conditions. Classify the following statements as true or false and justify your answers mathematically.

(a) Suppose the optimal solution $\vec{w}^\star, b^\star$ changes when the training point $\vec{x}_i$ is removed. Then originally, we necessarily have $y_i(\vec{x}_i^\top\vec{w}^\star - b^\star) = 1 - \xi_i^\star$.

(b) Suppose the optimal solution $\vec{w}^\star, b^\star$ changes when the training point $\vec{x}_i$ is removed. Then originally, we necessarily have $\alpha_i^\star > 0$.

(c) Suppose the data points are strictly linearly separable, i.e. there exist $\widetilde{\vec{w}}$ and $\widetilde{b}$ such that for all i,

$$y_i(\vec{x}_i^\top\widetilde{\vec{w}} - \widetilde{b}) > 0. \tag{20}$$

Then $p^\star(C) \to \infty$ as $C \to \infty$.

5. **Ridge Regression Classifier Vs. SVM**

In this problem, we explore Ridge Regression as a classifier, and compare it to SVM. Recall Ridge Regression solves the problem

$$\min_{\vec{w}} \|X\vec{w} - \vec{y}\|_2^2 + \lambda \|\vec{w}\|_2^2 , \tag{21}$$

where $X \in \mathbb{R}^{m \times n}$, and $\vec{y} \in \mathbb{R}^n$

(a) Ridge Regression as is solves a regression problem. Given data $X \in \mathbb{R}^{m \times n}$ and labels $\vec{y} \in \{-1, 1\}^m$, explain how we might be able to train a Ridge Regression model and use it to classify a test point.

(b) Complete the accompanying Jupyter notebook to compare Ridge Regression and SVM.

6. Modified SVM

Let $C > 0$. Suppose we have labeled data $(\vec{x}_i, y_i) \in \mathbb{R}^d \times \{-1, 1\}$ for $i = 1, \ldots, n$. For each i, define $\vec{z}_i \doteq y_i\vec{x}_i$. Finally, define $Z \doteq \begin{bmatrix}\vec{z}_1, \ldots, \vec{z}_n\end{bmatrix}^\top \in \mathbb{R}^{n\times d}$.

Recall that the soft-margin support vector machine problem can be expressed using slack variables as

$$\begin{aligned} p_1^\star = \min_{\vec{w},\vec{s}} \quad & \frac{1}{2}\|\vec{w}\|_2^2 + C\sum_{i=1}^{n} s_i \\ \text{s.t.} \quad & s_i = \max\{0, 1 - \vec{z}_i^\top\vec{w}\}, \qquad \forall i \in \{1, \ldots, n\}. \end{aligned} \tag{22}$$

In this problem we consider a modified SVM program with a squared penalty:

$$\begin{aligned} p_2^\star = \min_{\vec{w},\vec{s}} \quad & \frac{1}{2}\|\vec{w}\|_2^2 + \frac{C}{2}\sum_{i=1}^{n} s_i^2 \\ \text{s.t.} \quad & s_i = \max\{0, 1 - \vec{z}_i^\top\vec{w}\}, \qquad \forall i \in \{1, \ldots, n\}. \end{aligned} \tag{23}$$

We will use another representation of this program, namely one with affine constraints:

$$\begin{aligned} p^\star = \min_{\vec{w},\vec{s}} \quad & \frac{1}{2}\|\vec{w}\|_2^2 + \frac{C}{2}\|\vec{s}\|_2^2 \\ \text{s.t.} \quad & \vec{s} \geq \vec{0} \\ & \vec{s} \geq \vec{1} - Z\vec{w}, \end{aligned} \tag{24}$$

where the inequality constraints are componentwise (as usual).

(a) Choose the smallest class that problem (24) belongs to (LP/QP/SOCP/etc).

(b) Prove that strong duality holds for (24).

(c) Are the KKT conditions for problem (24) necessary, sufficient or both necessary and sufficient for global optimality?

(d) Let $\vec{\alpha}$ be the dual variable corresponding to the constraint $\vec{s} \geq \vec{0}$. What is the dimension (i.e., number of entries) of $\vec{\alpha}$?

(e) Show that the Lagrangian $L(\vec{w}, \vec{s}, \vec{\alpha}, \vec{\beta})$ of problem (24), where $\vec{\alpha}$ is the dual variable corresponding to the constraint $\vec{s} \geq \vec{0}$, and $\vec{\beta}$ is the dual variable corresponding to the constraint $\vec{s} \geq \vec{1} - Z\vec{w}$, is equal to

$$L(\vec{w}, \vec{s}, \vec{\alpha}, \vec{\beta}) = \frac{1}{2}\|\vec{w}\|_2^2 + \frac{C}{2}\|\vec{s}\|_2^2 - \vec{s}^\top(\vec{\alpha} + \vec{\beta}) - \vec{w}^\top Z^\top\vec{\beta} + \vec{1}^\top\vec{\beta}. \tag{25}$$

(f) Write the KKT conditions for problem (24). Show that if $(\vec{w}^\star, \vec{s}^\star, \vec{\alpha}^\star, \vec{\beta}^\star)$ obey the KKT conditions for problem (24), then

$$\vec{w}^\star = Z^\top\vec{\beta}^\star \qquad \text{and} \qquad \vec{s}^\star = \frac{\vec{\alpha}^\star + \vec{\beta}^\star}{C}. \tag{26}$$

HINT: For the first order/stationarity condition on the Lagrangian you will need to consider partial derivatives with respect to both $\vec{w}$ and $\vec{s}$.

(g) Compute the dual function of problem (24) as

$$g(\vec{\alpha}, \vec{\beta}) \doteq L(\vec{w}^\star(\vec{\alpha}, \vec{\beta}), \vec{s}^\star(\vec{\alpha}, \vec{\beta}), \vec{\alpha}, \vec{\beta}) \tag{27}$$

where from the previous part we have that

$$\vec{w}^{\star}(\vec{\alpha}, \vec{\beta}) = Z^{\top}\vec{\beta} \qquad \text{and} \qquad \vec{s}^{\star}(\vec{\alpha}, \vec{\beta}) = \frac{\vec{\alpha} + \vec{\beta}}{C}. \tag{28}$$

Your final expression for $g(\vec{\alpha}, \vec{\beta})$ should not contain any maximizations, minimizations or terms including $\vec{w}$, $\vec{s}$, $\vec{w}^{\star}$, or $\vec{s}^{\star}$. It should only contain $\vec{\alpha}$, $\vec{\beta}$, C, Z, and numerical constants.

(h) Let $\vec{\alpha}^{\star}$ and $\vec{\beta}^{\star}$ be optimal dual variables that solve the problem

$$d^{\star} \doteq \max_{\vec{\alpha}, \vec{\beta} \geq \vec{0}} g(\vec{\alpha}, \vec{\beta}). \tag{29}$$

It turns out that $\vec{\alpha}^{\star}$ can also be obtained by solving the quadratic program:

$$\begin{aligned} \min_{\vec{\alpha}} \quad & \left\|\vec{\alpha} + \vec{\beta}^{\star}\right\|_2^2 \\ \text{s.t.} \quad & \vec{\alpha} \geq \vec{0}. \end{aligned} \tag{30}$$

Solve this quadratic program (30) directly and find $\vec{\alpha}^{\star}$.

HINT: The duality or KKT approaches are not recommended. Consider $\vec{\alpha} = \begin{bmatrix} \alpha_1 & \cdots & \alpha_n \end{bmatrix}^{\top}$*, and use the components of* $\vec{\alpha}$ *to decompose the problem into* n *separate scalar problems. Solve each one by checking critical points; that is, points where the gradient is* 0*, the boundary of the feasible set, and* $\pm\infty$*.*

(i) Let $\beta^{\star}$ be a solution to the dual problem. Characterize the pairs $(\vec{x}_i, y_i)$ which are "support vectors", i.e., contribute to the optimal weight vector $\vec{w}^{\star}$, in terms of $\beta^{\star}$.

7. Homework Process

With whom did you work on this homework? List the names and SIDs of your group members.

NOTE: If you didn't work with anyone, you can put "none" as your answer.