CS 189 Introduction to Machine Learning
Spring 2018

HW0

Your self-grade URL is http://eecs189.org/self_grade?question_ids=1_1,1_2,2,3_1,3_2,3_3,4_1,4_2,4_3,5_1,5_2,

This homework is due **Friday, January 19 at 10 p.m.**

2 Sample Submission

Please submit a plain text file to the Gradescope programming assignment "Homework 0 Test Set":

1. Containing 5 rows, where each row has only one value "1".
2. No spaces or miscellaneous characters.
3. Name it "submission.txt".

3 Linear Algebra Review

Consider the vectors $\mathbf{u} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ and $\mathbf{v} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$. Define the matrix $\mathbf{M} = \mathbf{u}\,\mathbf{v}^\top$.

Solution: There are two ways to approach the problem: (1) Perform numerical computations for parts (a) and (b), as one would do if the matrix $\mathbf{M}$ was a general matrix. This has the benefit of immediately making visible the patterns involved. (2) Exploit the special structure of the matrix $\mathbf{M}$ and make use of the general solution from part (c).

(a) Compute the eigenvalues and eigenvectors of the matrix $\mathbf{M}$.

Solution: The matrix $\mathbf{M}$ is given by $\begin{bmatrix} 2 & 3 \\ 4 & 6 \end{bmatrix}$. To find the eigenvalues, we have to find solutions of the equation $\det(\mathbf{M} - \lambda I) = 0$. We have

$$\begin{aligned} \det(\mathbf{M} - \lambda I) = 0 \Rightarrow \det \begin{bmatrix} 2-\lambda & 3 \\ 4 & 6-\lambda \end{bmatrix} &= 0 \\ \Rightarrow (2-\lambda)(6-\lambda) - 12 &= 0 \\ \Rightarrow \lambda^2 - 8\lambda &= 0 \\ \Rightarrow \lambda &= 0, 8. \end{aligned}$$

To compute the eigenvector corresponding to an eigenvalue λ, we have to determine a basis of the nullspace of the matrix $\mathbf{M} - \lambda I$. In other words, we have to find a maximal set of linearly independent solutions for the equation $(\mathbf{M}-\lambda I)\mathbf{x}=0$. In our case, these equations are given by

$$\lambda=0: \quad \begin{bmatrix}2 & 3\\ 4 & 6\end{bmatrix}\mathbf{x}=0$$
$$\lambda=8: \quad \begin{bmatrix}-6 & 3\\ 4 & -2\end{bmatrix}\mathbf{x}=0.$$

To solve the first linear equation, we have

$$\begin{bmatrix}2 & 3\\ 4 & 6\end{bmatrix} \sim \begin{bmatrix}2 & 3\\ 2 & 3\end{bmatrix} \Rightarrow 2x_1+3x_2=0 \Rightarrow \mathbf{x}=\begin{bmatrix}-3\\ 2\end{bmatrix}$$

We immediately notice that the resulting $\mathbf{x}$ is orthogonal to $\mathbf{v}$. And in hindsight, it makes complete sense. This makes $\mathbf{u}\mathbf{v}^\top\mathbf{x}=\mathbf{u}(\mathbf{v}^\top\mathbf{x})=\mathbf{0}$.

Similarly for the second linear eigenvector equation, we have

$$\begin{bmatrix}-6 & 3\\ 4 & -2\end{bmatrix} \sim \begin{bmatrix}-2 & 1\\ -2 & 1\end{bmatrix} \Rightarrow -2x_1+x_2=0 \Rightarrow \mathbf{x}=\begin{bmatrix}1\\ 2\end{bmatrix}$$

Hmm... This is clearly just $\mathbf{u}$ itself. And in hindsight, it makes complete sense. This makes $\mathbf{u}\mathbf{v}^\top\mathbf{x}=\mathbf{u}(\mathbf{v}^\top\mathbf{u})=(\mathbf{v}^\top\mathbf{u})\mathbf{u}$ and so $\mathbf{u}$ has to be an eigenvector.

(b) Compute the rank and the determinant of the matrix $\mathbf{M}$. What is the dimension of the nullspace of the matrix $\mathbf{M}$?

Solution: The rank of a matrix is equal to the number of linearly independed columns and hence the rank of the matrix $\mathbf{M}$ is 1. Alternatively, we know that the rank isn't zero since the matrix isn't the zero matrix. The rank isn't 2 since the matrix has an eigenvalue of 0 and hence has a nontrivial nullspace and hence is not invertible. So, it must have rank 1 by the process of elimination.

Since the determinant of any matrix is equal to the product of all its eigenvalues (repeated with multiplicity if any have multiplicity), the determinant of the matrix $\mathbf{M}$ is 0. Alternatively, since the matrix is not invertible, the determinant must be 0.

Furthermore, the dimension of the nullspace of any matrix is equal to the number of columns minus the rank and hence for $\mathbf{M}$, it is 1. Alternatively, it is the number of linearly indepenent eigenvectors corresponding to eigenvalue 0. Either way, it is 1.

(c) Now consider two non-zero vectors $\mathbf{p}$ and $\mathbf{q}$ in $\mathbb{R}^d$ and the matrix $\mathbf{N}=\mathbf{p}\ \mathbf{q}^\top$. Repeat the computations for parts (a) and (b) for the matrix $\mathbf{N}$.

Solution: To solve this part, we exploit the special structure of the matrix $\mathbf{N}$. We use the following three important facts:

- Any matrix of the form $\mathbf{p}\mathbf{q}^\top$ is a rank one matrix, because all its rows are just a multiple of the row vector $\mathbf{q}^\top$. Hence it has **at most one non-zero eigenvalue**.

- For two matrices $\mathbf{A}$ and $\mathbf{B}$ of compatible size, we have

$$\text{trace}(\mathbf{AB}) = \text{trace}(\mathbf{BA}).$$

Recalling the special case that we already say, we split the analysis in two parts: (1) $\mathbf{q}^T\mathbf{p} \neq 0$ and (2) $\mathbf{q}^T\mathbf{p} = 0$.

Case 1: To find the non-zero eigenvalue, follow what we observed earlier.

$$\mathbf{Np} = \mathbf{pq}^\top\mathbf{p} = (\mathbf{q}^\top\mathbf{p})\,\mathbf{p}.$$

That is for matrix $\mathbf{N}$, the vector $\mathbf{p}$ is an eigenvector with corresponding eigenvalue $(\mathbf{q}^\top\mathbf{p})$ which is not zero. (Notice that even if $\mathbf{q}^\top\mathbf{p} = 0$, that $\mathbf{p}$ would still be an eigenvector.)

Now, all the other $d-1$ eigenvalues are zero since the matrix is clearly rank 1 by contruction. In other words, the nullity of the matrix is $d-1$. Finding $d-1$ linearly independent (LI) eigenvectors corresponding to the zero eigenvalue is equivalent to finding a basis for the nullspace of the matrix $\mathbf{N}$, which in turn is equivalent to finding a set of $d-1$ LI vectors that are orthogonal to the vector q. That is we have to find the solutions to the equation

$$q_1x_1 + q_2x_2 + \ldots + q_dx_d = 0. \tag{1}$$

Since $\mathbf{q} \neq \mathbf{0}$, there exists a coordinate i such that $q_i \neq 0$. Without loss of generality, we can assume that $q_d \neq 0$. We now try to find a general solution of the equation above. We have

$$x_d = -(q_1x_1 + q_2x_2 + \ldots + q_{d-1}x_{d-1})/q_d.$$

As a result a general solution has the form

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{d-1} \\ -q_1/q_dx_1 - q_2/q_dx_2 - \ldots - q_{d-1}/q_dx_{d-1} \end{bmatrix}$$
$$= x_1 \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \\ -q_1/q_d \end{bmatrix} + x_2 \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \\ -q_2/q_d \end{bmatrix} + \ldots + x_{d-1} \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ -q_{d-1}/q_d \end{bmatrix}$$

which yields that we have the following set of LI solutions for the equation (1):

$$\mathscr{S} = \left\{ \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \\ -q_1/q_d \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \\ -q_2/q_d \end{bmatrix}, \ldots, \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ -q_{d-1}/q_d \end{bmatrix} \right\}. \tag{2}$$

To verify that the set $\mathscr{S}$ has linearly independent vectors, we observe that

$$\alpha_1 \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \\ -q_1/q_d \end{bmatrix} + \alpha_2 \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \\ 0 \\ -q_2/q_d \end{bmatrix} + \ldots + \alpha_{d-1} \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ -q_{d-1}/q_d \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix} \Leftrightarrow \alpha_1 = \alpha_2 = \cdots = \alpha_{d-1} = 0.$$

Remark: Students are not required to prove all the arguments in extensive detail. However an statement of the form" the eigenvectors are solutions of the equation (1)" without further explanation is not sufficient.

To summarize for the case $\mathbf{p}^T \mathbf{q} \neq 0$:

- The eigenvalues of the matrix $\mathbf{N}$ are given by $(\mathbf{q}^\top \mathbf{p}), 0$ and corresponding eigenvectors are $\mathbf{p}$ and the set $\mathscr{S}$ mentioned above.
- The rank of the matrix $\mathbf{N}$ is 1 and its determinant is 0. The nullity of the matrix is $d-1$.

Case 2: When $\mathbf{p}^T \mathbf{q} = 0$, then the situation becomes quite interesting. Note that as argued before the matrix $\mathbf{N}$ can have at most one non-zero eigenvalue, due to its rank being one. We also know that the sum of eigenvalues is equal to the trace of the matrix. Furthermore using the trace trick, $\text{trace}(\mathbf{N}) = \text{trace}(\mathbf{p}\mathbf{q}^T) = \text{trace}(\mathbf{p}^T\mathbf{q}) = \mathbf{p}^T\mathbf{q} = 0$. Thus the sum of the eigenvalues is zero. Since $d-1$ eigenvalues are known to be zero, adding the fact that the sum of d eigenvalues is zero, implies that all eigenvalues are zero.

Thus, we try to find d linearly independent eigenvectors corresponding to the 0 eigenvalue. A set of $d-1$ LI eigenvectors can be found as before. We need to determine if there exists a d-th LI eigenvector corresponding to the 0 eigenvalue. Note that $\mathbf{p}$ is orthogonal to q and is an eigenvector corresponding to 0. In fact it now lies in the linear span of the set $\mathscr{S}$, defined in equation (2).

To find the d-th eigenvector, we need to look outside the linear span of the set $\mathscr{S}$. In this case, this space is simply linear multiple of the vector $\mathbf{q}$. However, $\mathbf{q}$ is not an eigenvector because

$$\mathbf{N}\mathbf{q} = \mathbf{p}\mathbf{q}^T\mathbf{q} = \|\mathbf{q}\|^2 \mathbf{p},$$

which is not a scalar multiple of $\mathbf{q}$ because $\mathbf{q}$ is orthogonal to the vector $\mathbf{p}$. As a result, there is no d-th linearly independent eigenvector.

This example illustrates the phenomenon where the geometric multiplicity (dimension of the eigenspace) is less than the algebraic multiplicity of the eigenvalue. Notice that this is because the matrix $\mathbf{N}$ is clearly nilpotent and not zero. $\mathbf{N}\mathbf{N} = \mathbf{p}\mathbf{q}^\top\mathbf{p}\mathbf{q}^\top = \mathbf{p}(\mathbf{q}^\top\mathbf{p})\mathbf{q}^\top = \mathbf{p}0\mathbf{q}^\top = \mathbf{0}$. Clearly a nilpotent matrix can't be diagonalizable since all of its eigenvalues have to zero while the matrix itself is nonzero. So, it has to be lacking a full complement of eigenvectors.

To summarize for the case $\mathbf{p}^T \mathbf{q} = 0$:

- The eigenvalues of the matrix $\mathbf{N}$ are given by $0, \ldots, 0$ and corresponding eigenvectors are given by the set $\mathscr{S}$ mentioned above. The matrix has only $d-1$ linearly independent eigenvectors.
- The rank of the matrix $\mathbf{N}$ is still 1 and its determinant is 0. The nullity of the matrix is $d-1$.

Remark: A common mistake is to assume that the eigenvectors are usually orthogonal to each other. That is necessary only for symmetric matrices and need not hold for a general matrix. Also, note that the matrix $\mathbf{N}$ is not symmetric unless $\mathbf{p} = \mathbf{q}$.

Please explain your computations/arguments precisely.

4 Linear Regression and Adversarial Noise

In this question, we will investigate how the presence of noise in the data can adversely affect the model that we learn from it.

Suppose we obtain a training dataset consisting of n points (x_i, y_i) where $n \geq 2$. In case of no noise in the system, these set of points lie on a line given by $y = w_1 x + w_2$, i.e, for each i, $y_i = w_1 x_i + w_2$. The variable x is usually referred to as the features [1] and y's are referred to as the observation. Suppose that all x_i's are distinct and non-zero. Our task is to estimate the slope w_1 and the y-intercept w_2 from the training data. We call the pair (w_1, w_2) as the true model.

Suppose that an adversary modifies our data by corrupting the observations and we now have the training data $(x_i, \tilde{y}_i)$ where $\tilde{y}_i = y_i + \varepsilon_i$ and the noise ε_i is chosen by the adversary. Note that the adversary has access to the features x_i but *can not* modify them. Its goal is to trick us into learning a wrong model $(\hat{w}_1, \hat{w}_2)$ from the dataset $\{(x_i, \tilde{y}_i), i = 1, \ldots, n\}$. We denote by $(\hat{w}_1, \hat{w}_2)$ the model that we learn from this dataset $\{(x_i, \tilde{y}_i), i = 1, \ldots, n\}$ using the standard ordinary least-squares regression.

(a) Suppose that the adversary wants us to learn a particular wrong model (w_1^*, w_2^*). If we use standard ordinary least-squares regression, can the adversary *always* (for any choice of w_1^* and w_2^*) fool us by setting a particular value for exactly one ε_i (and leaving other observations as it is, i.e., $\tilde{y}_j = y_j, j \neq i$), so that we obtain $\hat{w}_1 = w_1^*$ and $\hat{w}_2 = w_2^*$? If yes, justify by providing a mathematical mechanism for the adversary to set the value of the noise term as a function of the dataset $\{(x_i, y_i), i = 1, \ldots, n\}$ and (w_1^*, w_2^*)? If no, provide a counter example.

Solution: The answer is no. Intuitively, the adversary is only in control of one degree of freedom. Here is a counterexample with just two data points. Let the two pairs of observations be $(x_1, y_1) = (1, 0), (x_2, y_2) = (-1, 1)$. Let $\mathbf{X}$ be the feature matrix:

$$\mathbf{X} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}.$$

[1] The earlier version of this homewoek calls this matrix covariate matrix, which is commonly used in statistics.

and let $\mathbf{y}$ denote the observation vector $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. The corrupted observation vector can take one of the forms $\tilde{\mathbf{y}} = \begin{bmatrix} 0+\varepsilon_1 \\ 1 \end{bmatrix}$ or $\tilde{\mathbf{y}} = \begin{bmatrix} 0 \\ 1+\varepsilon_2 \end{bmatrix}$. Standard OLS in this case simplifies to finding the solution to the equation $\mathbf{X}\mathbf{w} = \tilde{\mathbf{y}}$ and hence we have

$$\mathbf{X}\mathbf{w} = \tilde{\mathbf{y}} \Rightarrow \hat{\mathbf{w}} = \mathbf{X}^{-1}\tilde{\mathbf{y}} = \frac{1}{2}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}\tilde{\mathbf{y}} \Rightarrow \hat{\mathbf{w}} = \frac{1}{2}\begin{bmatrix} \tilde{y}_1 - \tilde{y}_2 \\ \tilde{y}_1 + \tilde{y}_2 \end{bmatrix}.$$

Note that we can take inverse because $\mathbf{X}$ is a square matrix. For a more general case, refer to part (b). Thus the OLS solutions are of the form

$$\hat{\mathbf{w}} = \frac{1}{2}\begin{bmatrix} \varepsilon_1 - 1 \\ \varepsilon_1 + 1 \end{bmatrix} \quad \text{or} \quad \hat{\mathbf{w}} = \begin{bmatrix} -1-\varepsilon_2 \\ 1+\varepsilon_2 \end{bmatrix}$$

and it is immediately clear that this adversary **CANNOT fool us in choosing $\hat{\mathbf{w}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ or say $\hat{\mathbf{w}} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$**. In fact, there are infinitely many $\mathbf{w}^*$ that the adversary cannot trick us into learning. The reason being that, in both cases, if the adversary uses a certain value of noise to trick us to hallucinate a particular w_1^*, it can no longer choose w_2^* of its choice, as the OLS solution determines it automatically.

(b) Repeat part (a) for the case when the adversary can corrupt two observations, i.e., for the case when the adversary can set up at most two of the ε_i's to any non-zero values of its choice.

Solution: Yes. Intuitively, the adversary now has control over two degrees of freedom.

The adversary needs control of two points that have different x coordinates. Consider n points $\{(x_i, y_i)\}_{i=1}^n$, where linear regression recovers $\hat{w}_1, \hat{w}_2$. We can start by examining the closed form solution for least-squares. Without loss of generality, let us assume that the first two points are the ones that the adversary is going to want to corrupt. Let us construct the standard feature matrix $\mathbf{X} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix}$ so that we are minimizing $\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2$. At this point, it is immediately obvious that since the first two rows of $\mathbf{X}$ are linearly independent since all x_i's are distinct, that $\mathbf{X}$ has full rank 2. The solution of linear least-squares is:

$$\hat{\mathbf{w}} = \begin{bmatrix} \hat{w}_1 \\ \hat{w}_2 \end{bmatrix} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\tilde{\mathbf{y}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y}+\varepsilon) = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\begin{bmatrix} y_1 \\ y_2 \\ \mathbf{y}_{rest} \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \mathbf{0} \end{bmatrix})$$

Note that the adversary chooses $(\varepsilon_1, \varepsilon_2)$ and wants least squares to yield exactly the line the adversary desires. Note that that the adversary needs to solve the following equation:

$$\begin{bmatrix} w_1^* \\ w_2^* \end{bmatrix} - (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \mathbf{0} \end{bmatrix} = (\mathbf{X}^T\mathbf{X})^{-1}\begin{bmatrix} x_1 & x_2 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix}.$$

But we know that since $x_1 \neq x_2$, the matrix $\begin{bmatrix} x_1 & x_2 \\ 1 & 1 \end{bmatrix}$ is invertible and so this has the solution:

$$\begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \end{bmatrix} = (\begin{bmatrix} x_1 & x_2 \\ 1 & 1 \end{bmatrix})^{-1}(\mathbf{X}^T\mathbf{X})(\begin{bmatrix} w_1^* \\ w_2^* \end{bmatrix} - (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}).$$

This explicit formula lets the adversary move the least-square solution to wherever it wants assuming it has control over the two "outliers" in the first two positions.

(a) Adversary solves for ε_i in terms of w_1^*, w_2^*.

(b) Adversary applies ε_i and gives data to you.

(c) You run least squares using the ε_i and retrieve $\hat{w}_1, \hat{w}_2$, which we have already determined to be equal to w_1^*, w_2^*.

Food for thought: If there are not two distinct points with different x coordinates, this means all the training points have the same x coordinate. At this point, it is clear that ordinary least squares will simply fail since it will not have enough rank in $\mathbf{X}$ itself. The adversary can't remedy this failure since, as stated, it can only perturb the y measurements, not the x ones.

(c) In the context of machine learning and applications, what lessons do you take-away after working through this problem?

Solution: As you can see, the adversary can fool us into "learning" an arbitrary model by (substantially) tweaking a few data points. Machine learning is learning from data and thus, our accuracy relies on the noise in the data and the mechanism used to learn the model. As this problem illustrates, when subject to unbounded noise or outliers, ordinary linear least-squares regression can yield a line *very* far from the true model. Here we saw that if an adversary corrupts a certain number of data points (equal to the degree of freedom in the data generation mechanism), we can be forced to hallucinate an arbitrary model.

In certain applications, it is desirable to learn a model that is not affected too much by a small fraction of the data points. In this course, we would learn about many techniques on making the learning algorithm robust to different types of noise in the data.

Some examples from other classes: For those of you who have taken EE16A at Berkeley, you would have learned about Orthogonal Matching Pursuit (OMP). It is actually possible to modify OMP to help hunt for outliers and remove them from the data in many cases for a more robust version of least-squares. Another example is from CS 70 where you would have learned about Reed-Solomon codes. These codes are robust to arbitrary amount of noise in a few data points. The idea generally is to exploit redundancy in the data to make the model robust to potentially big noise, instead of only trying to leverage the redundancy to best average away the small noises. (There is no big or small in the CS70 finite-field perspective, but there is in the reals and for complex numbers.)

5 Background Review

Please describe the coursework that you have undertaken on the following topics:

(a) Linear Algebra, e.g., EE 16A/B

Solution:

The standard expectation is that students will have learned linear algebra in EE16A and EE16B. This means that students are comfortable with the SVD as well as understanding how linear algebra is used in modeling.

Alternatively, some students might come having taken Math 54 and Math 110 (a lower-division and upper-division basic standard course in linear algebra). Sometimes such students haven't learned enough of the modeling side of linear algebra, but courses like DS8 and DS100 can help.

Alternatively, some students might just have a lower-division course in linear algebra like Math 54, and coupled with the solid review of linear algebra in a course like EECS 127 (optimization), this gives enough exposure to both the concepts and the modeling perspective on linear algebra.

There are other possibilities as well. Physics-oriented students might have gotten enough linear algebra through their basic lower-division and then courses like upper-division quantum mechanics and classical mechanics.

(b) Optimization, e.g., EECS 127

Solution:

There is a lot of optimization thinking in EE 16AB, and coupled with the treatment of Lagrange multipliers and the like from Math 53 and linear programming and duality from CS170, that can be enough for some students.

Of course, taking a course like EECS 127 (or equivalent) is the best way to get a solid grounding in optimization.

There are other possibilities as well. Physics-oriented students might have gotten many of the right ideas in an upper-division classical mechancs course. And mechanical engineers in an optimal control course. Etc.

A Data-science oriented student should probably have taken EECS 127 since that is a highly recommended core for that major as well.

(c) Probability and stochastic processes, e.g., EECS 126

Solution:

The probability in CS70 is barely enough if it is completely mastered. But most everyone requires a couple of courses before probability actually gets anywhere close to mastery. The best case is of course to have taken EECS 126.

However, Data Science students should be fine with DS100 plus Prob140 or equivalent.

A course like Stat 134 is barely enough, largely because of the same one-course issue. It usually takes everyone at least two courses before they really get probability.

(d) Vector Calculus, e.g., EECS 127, Math 53.

Solution:

In principle, everything that we need should have been taught in a lower-division course like Math 53. In practice, this is often not enough. Understanding the material from EECS 127 (or any continuous optimization course) should suffice. Alternatively, upper-division mathematics courses that involve vector calculus are good. Differential geometry courses are probably overkill.

(e) Also describe your experience with programming, in particular with python, e.g., like in your coursework in CS 61A/B and EE 16A/B.

Solution:

For this course, the programming from CS61A and EE16AB is the bare minimum. Having the maturity that comes from CS61B is very helpful as well. Any alternative programming background is also probably fine.

6 Your Own Question

Write your own question, and provide a thorough solution.

Writing your own problems is a very important way to really learn the material. The famous "Bloom's Taxonomy" that lists the levels of learning is: Remember, Understand, Apply, Analyze, Evaluate, and Create. Using what you know to create is the top-level. We rarely ask you any HW questions about the lowest level of straight-up remembering, expecting you to be able to do that yourself. (e.g. make yourself flashcards) But we don't want the same to be true about the highest level.

As a practical matter, having some practice at trying to create problems helps you study for exams much better than simply counting on solving existing practice problems. This is because thinking about how to create an interesting problem forces you to really look at the material from the perspective of those who are going to create the exams.

Besides, this is fun. If you want to make a boring problem, go ahead. That is your prerogative. But it is more fun to really engage with the material, discover something interesting, and then come up with a problem that walks others down a journey that lets them share your discovery. You don't have to achieve this every week. But unless you try every week, it probably won't happen ever.