EECS 16B Designing Information Devices and Systems II
Spring 2021 UC Berkeley **Homework 10**

This homework is due on Friday, April 2, 2021, at 11:00PM. Self-grades and HW Resubmission are due on Tuesday, April 6, 2021, at 11:00PM.

1. Reading Lecture Notes

Staying up to date with lectures is an important part of the learning process in this course. Here are links to the notes that you need to read for this week: Note 10A

(a) Consider $A \in \mathbb{R}^{n \times n}$ where the columns of A (denoted by $\vec{a_k}, 1 \leq k \leq n$) are orthonormal. What does the least squares solution $(A^T A)^{-1} A^T \vec{y}$ simplify to?

(b) Suppose we have two vectors $\vec{v_1} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\vec{v_2} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \in \mathbb{R}^2$. Then, we use Gram-Schmidt Orthonormalization to construct $\vec{q_1}$ and $\vec{q_2}$. Are $\vec{q_1}$ and $\vec{q_2}$ the only vectors that form an orthogonal basis for the span$\{\vec{v_1}, \vec{v_2}\}$?

2. Gram-Schmidt Basic

(a) **Use Gram-Schmidt to find a matrix U whose columns form an orthonormal basis for the column space of V.**

$$V = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

(b) Show that you get the same resulting vector when you project $\vec{w} = \begin{bmatrix} 1 \\ -1 \\ 0 \\ -1 \\ 0 \end{bmatrix}$ onto the columns of V as you do when you project onto the columns of U, i.e. **show that**

$$V(V^T V)^{-1} V^T \vec{w} = U(U^T U)^{-1} U^T \vec{w}.$$

Feel free to use numpy. No need to grind this out by hand.

3. Symmetric Matrices

We want to show that every real symmetric matrix can be diagonalized by a matrix of its orthonormal eigenvectors. In other words, a symmetric matrix has an orthonormal eigenbasis. This is called the spectral theorem for real symmetric matrices.

In discussion section, you have seen a recursive derivation of a related fact. Formally however, such recursive derivations are usually turned into proofs by using induction. This problem serves to both freshen your mind regarding induction as well as to give you a chance to prove for yourself this very important theorem. (This is the same essential proof as that of Schur upper-triangularization. So understanding this problem will help solidify your understanding of that proof as well.)

(a) You will start by proving a basic lemma about real symmetric matrices under an orthonormal change of basis. **Prove that if S is an $m \times m$ symmetric matrix ($S = S^\top$) and U is any $m \times n$ matrix, then $U^T SU$ is also symmetric**.

(b) Another useful lemma is one about finding orthonormal bases. **Show that given a single nonzero vector $\vec{u}_0$ of dimension n, it is possible to find an orthonormal set of n vectors, $\vec{v}_0, \ldots, \vec{v}_{n-1}$ such that $\vec{v}_0 = \alpha \vec{u}_0$ for some scalar α.**

(Hint: Use the Gram-Schmidt process on the list of $n+1$ vectors obtained by starting with the given vector and appending the standard basis — i.e. the columns of the identity matrix. Follow the procedure as in lecture.)

(c) For the main proof that every real symmetric matrix is diagonalized by a matrix of its orthonormal eigenvectors, we will proceed by formal induction. Recall that for a proof by induction, we have to start with a base case - this is also the base case in a recursive derivation.

Consider the trivial case of S having dimensions $[1 \times 1]$ ($n = 1$). **Does S have an eigenvector? Does S have an eigenbasis? Can this eigenbasis be made orthonormal? Is the matrix diagonal in this basis? Are the entries real?**

(d) After the base case, we do an inductive stage of the main proof. The first step in the inductive stage is to write down the induction hypothesis. Assume that the property that every real symmetric matrix is diagonalized by a matrix of its orthonormal eigenvectors holds for all symmetric matrices with size $[(n-1) \times (n-1)]$. **Complete the proof template below by filling in the blanks**. *(Hint: In general for proofs by induction, you want to start with the strongest version of what you want to prove. This gives you the most powerful inductive hypothesis.)*

> **Goal**: we want to show that any $[n \times n]$ real symmetric matrix S can be diagonalized by a matrix of its orthonormal real eigenvectors.
> **Base case**: when $n = 1$, this holds for $[1 \times 1]$ matrix $S = [s]$ as you showed in Part (d).
> **Inductive hypothesis**: Let Q be an $[(n-1) \times (n-1)]$ real symmetric matrix with eigenvectors $\vec{u}_0, \ldots, \vec{u}_{n-2}$, then we can express Q as $U \Lambda_Q U^T$, where Λ_Q is a _____ matrix with _____ eigenvalues of Q along its _____, and U is an $[(n-1) \times (n-1)]$ matrix, where the columns are _____ of Q.

(e) Now think about a symmetric matrix S with size $[n \times n]$. Consider a real eigenvalue λ_0 of S and the corresponding eigenvector $\vec{u}_0$ (a column vector with size n). **Use an appropriate orthonormal change of basis V to show that $S = VXV^T$, where X is of the form**

$$X = \begin{bmatrix} \lambda_0 & x_{1,2} & \cdots & x_{1,n} \\ 0 & x_{2,2} & \cdots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & x_{n,2} & \cdots & x_{n,n} \end{bmatrix}.$$

That is, the first column of X is $\begin{bmatrix} \lambda_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$.

(Hint: Follow the proof strategy from lecture. What should the first column of V be? How can you fill out the rest?)

(f) Continue the previous part **to show that in fact S can be written as**

$$S = V \begin{bmatrix} \lambda_0 & \vec{0}^T \\ \vec{0} & Q \end{bmatrix} V^T$$

where Q is an [$(n-1) \times (n-1)$] symmetric matrix and V is the matrix that you found above. *Hint: Define $\vec{v}_0 = \frac{\vec{u}_0}{\|\vec{u}_0\|}$, think of V as $V = \begin{bmatrix} \vec{v}_0 & R \end{bmatrix}$, where the dimension of R is $[n \times (n-1)]$. Also recall that S is a symmetric matrix.*

(g) According to our induction hypothesis, we can write Q as $U\Lambda U^T$ where U is an orthonormal [$(n-1) \times (n-1)$] square matrix and Λ is a diagonal matrix with real entries along the diagonal and 0s everywhere else. **Substitute this in the previous part to show that indeed there must exist an orthonormal [$n \times n$] square matrix W such that**

$$S = W \begin{bmatrix} \lambda_0 & \vec{0}^T \\ \vec{0} & \Lambda \end{bmatrix} W^T$$

(Hint: If R and U are both orthonormal matrices, is RU also orthonormal? What is $(RU)^T(RU)$?)

(h) **Finally, prove that the eigenvalues λ of real, symmetric matrix S are real.** *Hint: Suppose that S had a complex eigenvalue λ with eigenvector $\vec{v}$. Because S is a real matrix, what do you know about $S\bar{\vec{v}}$, where $S\bar{\vec{v}}$ is the complex conjugate of $\vec{v}$? What happens when you take a potentially complex number and multiply it by its own complex conjugate? Consequently, what do you know happens if you multiply a complex vector $\vec{v}$ by the conjugate of its transpose: i.e. consider $\bar{\vec{v}}^T\vec{v}$? Since S is symmetric, what do you know about $\vec{v}^T S$?*

By induction, we are now done since we have proved that having the desired property for $n-1$ implies that we have the property for n and we also have a valid base case at $n = 1$.

According to the base case and inductive steps we just proved, the statement, "every real symmetric matrix is diagonalized by a matrix of its real orthonormal eigenvectors" is proved by induction.

4. Classification of Sinusoids

This HW problem can be viewed as a warm-up for the next topic in the course: which is going to be motivated by figuring out how to process signals recorded from the brain to decipher what a person wants to do in terms of a specific command to their robot arm. These kinds of problems are called "classification" problems. In this exercise, you will be using jupyter to classify sinusoids.

The iPython notebook Sinusoidal_Projection_fa19_prob.ipynb will guide you through the process of performing sinusoidal projections.

Suppose you already know the true potential frequencies f_i and potential phases ϕ_i of a set of sinusoidal signals

$$S := \left\{\sin(2\pi f_i k + \phi_i), i = 1, 2, \ldots, n\right\}, \tag{1}$$

and you have some noisy samples of these true sinusoidal signals. You want to determine the true sinusoidal signal for each of these noisy samples—How would you approach the problem?

We will show in this problem that we can project noisy sinusoidal signals onto noiseless sinusoids to achieve good classification.

In the realistic world, one often doesn't have the complete waveform of a continuous function, instead oftentimes one works with *samples* of the continous function.

In our case, we generate noisy samples of the true sinusoidal signals in the following way. For each of the $num\_sinusoids$ true frequencies, each noisy sample y_i consists of N sample points sampled with a sampling rate of F_s sample rate, and corrupted by noise scaled by σ.

$$y_i(k) = \sin\big(2\pi f \cdot k/F_s\big) + \sigma \cdot Noise. \quad k = 1, 2, \ldots, N.$$

In our example, we will work with $num\_sinusoids = 3$.

A higher σ corresponds to more noise in our measurements.

Please complete the notebook by following the instructions given.

(a) Run the first part of the jupyter notebook to generate our noisy data points. **Use $\sigma = 0.1, 1.0, 10.0, 100.0$ and comment on what you observe in the plots.**

(b) **Complete part (b) of the notebook** to project noisy sinusoids onto potential true sinusoids.
Sketch the resulting 3D plot of projections qualitatively. Comment on what happens when you try the noise scalings $\sigma = 0.1, 1.0, 10.0, 100.0$.

(c) **Complete part (c) of the notebook** to classify the data points and calculate the number of misclassified points.
Report the number of misclassifications for $\sigma = 0.1, 1.0, 10.0, 100.0$. Explain what happens when there is a high level of noise. Recall that our noisy process is random so that there can be cases where there are misclassifications even in low noise.

(d) **For what qualitative regions of the noise level is it very beneficial for us to use projections?** For very low values of noise, do you have to do projections to successfully classify? What else could you have done? This question is asking you to reflect on what you have observed.

5. Using upper-triangularization to solve differential equations

You know that for any square matrix A with real eigenvalues, there exists a real matrix V with orthonormal columns and a real upper triangular matrix R so that $A = VRV^T$. In particular, to set notation explicitly:

$$V = \begin{bmatrix} \vec{v}_1, \vec{v}_2, \cdots, \vec{v}_n \end{bmatrix}$$

$$R = \begin{bmatrix} \vec{r}_1^T \\ \vec{r}_2^T \\ \vdots \\ \vec{r}_n^T \end{bmatrix}$$

where the rows of the upper-triangular R look like

$$\begin{aligned} \vec{r}_1^T &= [\lambda_1, r_{1,2}, r_{1,3}, \ldots, r_{1,n}] \\ \vec{r}_2^T &= [0, \lambda_2, r_{2,3}, r_{2,4}, \ldots, r_{2,n}] \\ \vec{r}_i^T &= [\underbrace{0, \ldots, 0}_{i-1 \text{ times}}, \lambda_i, r_{i,i+1}, r_{i,i+2}, \ldots, r_{i,n}] \\ \vec{r}_n^T &= [\underbrace{0, \ldots, 0}_{n-1 \text{ times}}, \lambda_n] \end{aligned}$$

where the λ_i are the eigenvalues of A.

Suppose our goal is to solve the n-dimensional system of differential equations written out in vector/matrix form as:

$$\begin{aligned} \frac{d}{dt}\vec{x}(t) &= A\vec{x}(t) + \vec{u}(t), \\ \vec{x}(0) &= \vec{x}_0, \end{aligned}$$

where $\vec{x}_0$ is a specified initial condition and $\vec{u}(t)$ is a given vector of functions of time.

Assume that the V and R have already been computed and are accessible to you using the notation above.

Assume that you have access to a function $ScalarSolve(\lambda, y_0, \breve{u})$ that takes a real number λ, a real number y_0, and a real-valued function of time $\breve{u}$ as inputs and returns a real-valued function of time that is the solution to the scalar differential equation

$$\frac{d}{dt}y(t) = \lambda y(t) + \breve{u}(t) \tag{2}$$

with initial condition $y(0) = y_0$.

Also assume that you can do regular arithmetic using real-valued functions and it will do the right thing. So if u is a real-valued function of time, and g is also a real-valued funcion of time, then $5u + 6g$ will be a real valued function of time that evaluates to $5u(t) + 6g(t)$ at time t.

Use V, R to construct a procedure for solving this differential equation

$$\begin{aligned} \frac{d}{dt}\vec{x}(t) &= A\vec{x}(t) + \vec{u}(t), \\ \vec{x}(0) &= \vec{x}_0, \end{aligned}$$

for $\vec{x}(t)$ by filling in the following template in the spots marked ♣, ♢, ♡, ♠.

(Note: It will be useful to upper triangularize A by change of basis to get a differential equation in terms of R instead of A.)
(Note: We use the notation $\vec{v}[i]$ to be the ith component of the vector $\vec{v}$)
(HINT: The process here should be simillar to diagonalization with some modifications. Start from the last row of the system and work your way up to understand the algorithm.)

1: $\vec{\widetilde{x}}_0 = V^T \vec{x}_0$ ▷ Change the initial condition to be in V-coordinates
2: $\vec{\widetilde{u}} = V^T \vec{u}$ ▷ Change the external input functions to be in V-coordinates
3: **for** $i = n$ down to 1 **do** ▷ Iterate up from the bottom row
4: $\quad \breve{u}_i = \clubsuit + \sum_{j=i+1}^{n} \spadesuit$ ▷ Make the effective input for this level
5: $\quad \widetilde{x}_i = ScalarSolve(\diamondsuit, \widetilde{x}_0[i], \breve{u}_i)$ ▷ Solve this level's scalar differential equation
6: **end for**
7: $\vec{x}(t) = \heartsuit \begin{bmatrix} \widetilde{x}_1 \\ \widetilde{x}_2 \\ \vdots \\ \widetilde{x}_n \end{bmatrix} (t)$ ▷ Change back into original coordinates

(a) Give the expression for $\heartsuit$ on line 7 of the algorithm above. (i.e. How do you get from $\vec{\widetilde{x}}(t)$ to $\vec{x}(t)$?)

(b) Give the expression for $\diamondsuit$ on line 5 of the algorithm above. (i.e. What are the λ arguments to $ScalarSolve$, equation (2), for the i^{th} iteration of the for-loop?)
(HINT: Convert the differential equation to be in terms of R instead of A. It may be helpful to start with i = n and develop a general form for the ith row.)

(c) Give the expression for $\clubsuit$ on line 4 of the algorithm above.

(d) Give the expression for $\spadesuit$ on line 4 of the algorithm above.

(e) (**OPTIONAL**) Let us complete the algorithm by investigating how $ScalarSolve(\lambda, y_0, \breve{u})$ works. Consider an input that is a weighted polynomial times and exponential.

$$\breve{u}(t) = \alpha t^{\beta} e^{\gamma t}$$

Here, α is a real constant, β is a non-negative integer, and γ is a real exponent. In addition, we will assume that $\gamma = \lambda$ for simplicity. We encourage you to attempt solving this system if $\gamma \neq \lambda$ if you are curious.

What function should $ScalarSolve(\lambda, y_0, \breve{u})$ return for the above $\breve{u}$? Remember, we are only considering the case where $\gamma = \lambda$. Express the answer in terms of α, β, γ.

6. Homework Process and Study Group

Citing sources and collaborators are an important part of life, including being a student!
We also want to understand what resources you find helpful and how much time homework is taking, so we can change things in the future if possible.

(a) **What sources (if any) did you use as you worked through the homework?**

(b) **If you worked with someone on this homework, who did you work with?**
List names and student ID's. (In case of homework party, you can also just describe the group.)

Contributors:

- Siddharth Iyer.
- Yu-Yun Dai.
- Sanjit Batra.
- Anant Sahai.
- Sidney Buchbinder.
- Gaoyue Zhou.
- Kuan-Yun Lee.
- Daniel Abraham.