This homework is extra credit (but in scope) and is due May 4, 2016, at Noon.

1. Homework process and study group

(a) Who else did you work with on this homework? List names and student ID’s. (In case of hw party, you can also just describe the group.)

Solution: I worked on this homework with...

(b) How long did you spend working on this homework? How did you approach it?

Solution: I spent a total of X hours working on this assignment.
I first worked by myself for 2 hours, but got stuck on Problem 5 so I went to office hours on...
Then I went to homework party for a few hours, where I finished the homework.

For self-grading, be sure to enter the number of hours alone in the comment field for this problem. This will help us better automatically collect this information.

2. Lecture Attendance

Did you attend live lecture this week? (the week you were working on this homework) What was your favorite part? Was anything unclear? Answer for each of the subparts below. If you only watched on YouTube, write that for partial credit.

(a) Monday lecture
(b) Wednesday lecture
(c) Friday lecture

Solution: Full credit for attending live lecture and giving a comment (what you liked best, what was unclear) about that lecture. 8 points for attending live lecture but giving no comment. 5 points for watching on YouTube and giving a comment. 2 points for just watching on YouTube. 0 points for blank or not watching lecture at all.

3. Sampling a Continuous-Time Control System to Get a Discrete-Time Control System

The goal of this problem is to help us understand how given a linear continuous-time system:

\[
\begin{align*}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &=Cx(t)
\end{align*}
\]

we can sample it every T seconds and get a discrete-time form of the control system. The discretization of the state equations is a sampled discrete time-invariant system given by
\[
\begin{align*}
\vec{x}_d(k+1) &= A_d\vec{x}_d(k) + B_d\vec{u}_d(k) \\
\vec{y}_d(k) &= C_d\vec{x}_d(k)
\end{align*}
\] (1)

Here, the \( \vec{x}_d(k) \) denotes \( \vec{x}(kT) \). This is a snapshot of the state. Similarly, the output \( \vec{y}_d(k) \) is a snapshot of \( \vec{y}(kT) \).

The relationship between the discrete-time input \( \vec{u}_d(k) \) and the actual input applied to the physical continuous-time system is that \( \vec{u}(t) = \vec{u}_d(k) \) for all \( t \in [kT, (k+1)T) \).

While it is clear from the above that the discrete-time state and the continuous-time state have the same dimensions and similarly for the control inputs, what is not clear is what the relationship should be between the matrices \( A, B \) and the matrices \( A_d, B_d \). By contrast it is immediately clear that \( C_d = C \).

(a) Argue intuitively why if the continuous-time system is stable, the corresponding discrete-time system should be stable too. Similarly, argue intuitively why if the discrete-time system is unstable, then the continuous-time system should also be unstable.

**Solution:** If the continuous-time system is bounded with all bounded inputs, then a bounded discrete-time input is also bounded when realized in continuous time and hence results in a bounded continuous-time state. Sampling this state would stay bounded.

Similarly, if there is a bounded input that would send the discrete-time system spiraling out to infinity, then that particular input is also valid in continuous-time since it is realized in a piecewise constant way. It sends the sampled state out of bounds and so must not be bounded in continuous-time either.

(b) Consider the scalar case where \( A \) and \( B \) are just constants. What are the new constants \( A_d \) and \( B_d \)?

**HINT:** Think about solving one step at a time. Every time a new control is applied, this is a simple differential equation with a new constant input. How does \( x(t) = \lambda x(t) + u \) evolve with time if it starts at \( x(0) \)? Notice that \( x(0)e^{\lambda t} + \frac{u}{\lambda} (e^{\lambda t} - 1) \) seems to solve this differential equation.

**Solution:** When we discretize the system, the input \( u \) becomes constant between each sampling point. Therefore, we can use the solution to the differential equation provided in the hint. We can only use this solution between each time step but not across time steps because \( u \) is guaranteed to be constant only within each time interval \( T \).

Let’s suppose that at time \( t = 0 \), we set \( u = 0 \). We will look at what happens with \( x(t) \) and see how we can generalize what we see to find the constants \( A_d \) and \( B_d \).

At time \( t = 0 \), the system starts the initial condition \( x(0) \). When we step to time \( t = T \), the system had evolved according to the solution to the differential equation with a constant \( u = 0 \):

\[
x(T) = x(0)e^{\lambda T} + \frac{0B}{A} (e^{\lambda T} - 1) \\
= x(0)e^{\lambda T}
\] (3)

As we move to time \( t = 2T \), assume now that the input \( u = u \) — some generic input. We can again apply the equation from the hint, but we now use \( x(0)e^{\lambda T} \) as our initial condition and take a step of size \( T \) as follows:

\[
x(2T) = x(0)e^{\lambda T}e^{\lambda T} + \frac{uB}{A} (e^{\lambda T} - 1) \\
= x(0)e^{2\lambda T} + \frac{uB}{A} (e^{\lambda T} - 1)
\] (4)
Here, we can see a pattern forming. Specifically, at each time step, we the term $e^{AT}$ multiplies the previous state, which leads us to conclude that $A_d = e^{AT}$. Similarly, at each time step for the input $u$, we add in a new term $\frac{e^{AT} - 1}{A} Bu_d(k)$. Therefore, we can see that $B_d = \frac{e^{AT} - 1}{A} B$.

(c) Consider now the case where $A$ is a diagonal matrix and $B$ is some general matrix. What is the new matrix $A_d$ and $B_d$?

**Solution:** The diagonal case is just a bunch of parallel scalar differential equations. The only coupling between them comes from the inputs. The $j$-th equation is thus:

$$\dot{x}_j(t) = a_{j,j} x_j(t) + B_j \bar{u}(t)$$

where $B_j$ is the $j$-th row of the matrix $B$ and $a_{j,j}$ is the $j$-th element along $A$’s diagonal. Treating all of $B_j \bar{u}(t)$ as a single scalar input, we can see from the previous step that in discrete-time, we have:

$$x_j(k+1) = e^{a_{j,j}T} x_j(k) + \frac{e^{a_{j,j}T} - 1}{a_{j,j}} B_j \bar{u}_d(k)$$

From this it is clear that $A_d$ is a diagonal matrix with $e^{a_{j,j}T}$ in the $j$-th position in the diagonal. Similarly, the $j$-th row of $B_d$ must be $\frac{e^{a_{j,j}T} - 1}{a_{j,j}} B_j$ or in other words $B_d = \bar{A}_T B$ where $\bar{A}_T$ is a diagonal matrix with $\frac{e^{a_{j,j}T} - 1}{a_{j,j}}$ in the $j$-th diagonal position.

(d) Consider the case where $A$ is a diagonalizable matrix. Use a change of coordinates to figure out the new matrix $A_d$ and $B_d$.

**Solution:** If $A$ is diagonalizable, then there exists an invertible transformation $S$ so that if we change variables to $\bar{z} = S \bar{x}$, then the dynamics expressed in terms of $\bar{z}$ — namely

$$\dot{\bar{z}}(t) = SAS^{-1} \bar{z}(t) + SB \bar{u}(t)$$

If we choose $S$ so that $SAS^{-1} = \Lambda$ which is a diagonal matrix with $\lambda_j$ in the $j$-th position, then we have reduced to the previous case. This means that in discrete-time, we know that

$$\bar{z}_d(k+1) = e^{\Lambda T} \bar{z}_d(k) + \bar{\Lambda}_T SB \bar{u}_d(k)$$

where we use the shorthand $e^{\Lambda T}$ to refer to a diagonal matrix with $e^{\lambda_j T}$ in the $j$-th position and the shorthand $\bar{\Lambda}_T$ to refer to the diagonal matrix with $\frac{e^{\lambda_j T} - 1}{\lambda_j}$ in the $j$-th diagonal position.

Changing back to original coordinates by multiplying both sides by $S^{-1}$ gives us:

$$\bar{x}_d(k+1) = S^{-1} e^{\Lambda T} S \bar{x}_d(k) + S^{-1} \bar{\Lambda}_T S B \bar{u}_d(k)$$

So $A_d = S^{-1} e^{\Lambda T} S$ and $B_d = S^{-1} \bar{\Lambda}_T S B$.

(e) Consider a general diagonal matrix $A$ with distinct eigenvalues and a vector $B = \bar{b}$ that consists of all 1s. Is the pair $(A, \bar{b})$ necessarily controllable? Prove that it must be or show a case where it isn’t. (HINT: Polynomials)

**Solution:** Look at the controllability matrix $[\bar{b}, A \bar{b}, A^2 \bar{b}, \ldots, A^{n-1} \bar{b}]$. By the properties of diagonal matrix multiplication, this must be:

$$\begin{bmatrix}
1 & \lambda_0 & \lambda_0^2 & \ldots & \lambda_0^{n-1} \\
1 & \lambda_1 & \lambda_1^2 & \ldots & \lambda_1^{n-1} \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
1 & \lambda_{n-1} & \lambda_{n-1}^2 & \ldots & \lambda_{n-1}^{n-1}
\end{bmatrix}$$
4. Sampling and the DFT

Recap of DFT: Consider a continuous-time signal \( x(t) \). We can collect a vector of discrete samples of \( x(t) \) over time as \( \vec{x} \), which is a finite time signal of length \( n \).

\[
\vec{x} = [x[0] \ldots x[n-1]]^T
\]  

(5)

As we learned from the DFT module, this signal can be represented in the DFT basis. Let \( \vec{X} = [X[0] \ldots X[n-1]]^T \) be the coordinates of \( \vec{x} \) in the DFT basis.

\[
\vec{X} = U^{-1}\vec{x} = U^*\vec{x}
\]

(6)

where \( U \) is a matrix of the DFT basis vectors.

\[
U = \begin{bmatrix}
\vec{u}_0 & \cdots & \vec{u}_{n-1}
\end{bmatrix} = \frac{1}{\sqrt{n}} \begin{bmatrix}
1 & 1 & \cdots & 1 \\
e^{2\pi i/n} & e^{2\pi i(2)/n} & \cdots & e^{2\pi i(n-1)/n} \\
e^{2\pi i(2)/n} & e^{2\pi i(4)/n} & \cdots & e^{2\pi i(2(n-1))/n} \\
de^{2\pi i(n-1)/n} & e^{2\pi i(2(n-1))/n} & \cdots & e^{2\pi i[(n-1)(n-1)]/n}
\end{bmatrix}
\]

(7)
Recall that \( U \) is an orthonormal basis, and the above equation shows that \( \vec{x} = U \vec{X} \). That is, \( \vec{x} \) can be always represented as a linear combination of the DFT basis signals \( \vec{u}_m \) with coefficients \( X[m] \).

\[
\vec{x} = X[0] \vec{u}_0 + \cdots + X[n-1] \vec{u}_{n-1}
\]  

(8)

Applying \( U \) to \( \vec{X} \) is a process called “taking the inverse DFT,” which is the reconstruction of the original signal as a superposition of its sinusoidal components.

(a) The \( n \)-th roots of unity are roots of the cyclotomic equation

\[
x^n = 1,
\]

and are known as the de Moivre numbers. Write down all complex roots of the above cyclotomic equation. Make sure they satisfy this equation. The roots should be in this form: \( x = e^{i \pi / n} \) \textbf{Solution:} \( e^{2\pi i k / n} \) for \( k = 0, 1, \ldots, n - 1 \)

(b) Assume \( n = 4 \). Label all the 4-th roots of unity on the complex plane. Call them \( s_0, s_1, s_2, s_3 \).

\textbf{Solution:} \( s_0 = e^{2\pi i (0)/4} = 1 + 0i, s_1 = e^{2\pi i (1)/4} = 0 + 1i, s_2 = e^{2\pi i (2)/4} = -1 + 0i, s_3 = e^{2\pi i (3)/4} = 0 + (-i) \). They are uniformly distributed on the unit circle.

(c) One perspective on the DFT basis vectors is that these are obtained by looking at the powers of each of the roots-of-unity. Verify that the \( j \)-th DFT basis element is just \( \frac{1}{\sqrt{n}} (s_j)^k \) as \( k = 0, \ldots, n - 1 \).

\textbf{Solution:} The \( j \)-th DFT basis is \( \frac{1}{\sqrt{n}} \)

\[
\begin{bmatrix}
    e^{\frac{2\pi i (0)}{n}} \\
    e^{\frac{2\pi i (1)}{n}} \\
    \vdots \\
    e^{\frac{2\pi i (n-1)}{n}}
\end{bmatrix}
\]

\[
= \frac{1}{\sqrt{n}}
\begin{bmatrix}
    (s_j)^0 \\
    (s_j)^1 \\
    \vdots \\
    (s_j)^{n-1}
\end{bmatrix}
\]

(d) An alternative perspective on the DFT treats the \( s_0, \ldots, s_{n-1} \) as polynomials. Verify that the \( j \)-th DFT basis element is just \( \frac{1}{\sqrt{n}} r^j \) evaluated as \( r \) goes through the \( n \) \( s_k \) points, as \( k = 0, \ldots, n - 1 \).

\textbf{Solution:} The \( j \)-th DFT basis is \( \frac{1}{\sqrt{n}} \)

\[
\begin{bmatrix}
    e^{\frac{2\pi i (0)}{n}} \\
    e^{\frac{2\pi i (1)}{n}} \\
    \vdots \\
    e^{\frac{2\pi i (n-1)}{n}}
\end{bmatrix}
\]

\[
= \frac{1}{\sqrt{n}}
\begin{bmatrix}
    (s_0)^0 \\
    (s_1)^0 \\
    \vdots \\
    (s_{n-1})^0
\end{bmatrix}
\]

Now let’s think about interpolating a discrete-time signal \( \vec{x} \) by assuming it is in a natural subspace defined by a few consecutive DFT basis vectors.

(e) Consider the complex signal \( \vec{x} \) below,

\[
\vec{x} = \begin{bmatrix}
    2 \\
    1 + \frac{\sqrt{2}}{2} + \frac{1}{2} \\
    x[2] \\
    0 \\
    x[4] \\
    x[5]
\end{bmatrix},
\]

(9)
along with its corresponding DFT coefficients, \( \tilde{X} \). Assume that we know \( X[m] = 0 \), for all \( m \geq 3 \). Because we have 3 samples and exactly 3 unknowns, we are able to recover all entries of \( \tilde{x} \) based on the above information. Do so and write down all entries of \( \tilde{x} \) and \( \tilde{X} \) explicitly.

**Solution:** This discrete signal can be represented as a linear combination of columns of the DFT basis with \( n = 6 \), \( \tilde{x} = X[0]u_0 + X[1]u_1 + X[2]u_2 + X[3]i\bar{u}_1 + X[4]i\bar{u}_2 + X[5]i\bar{u}_3 \). Because \( X[m] = 0 \), for all \( m \geq 3 \), \( \tilde{x} = X[0]u_0 + X[1]u_1 + X[2]u_2 \). If we write out the DFT basis explicitly, we can get the following three equations, with three variables, \( X[0] \), \( X[1] \) and \( X[2] \):

\[
\begin{align*}
2 &= x[0] = \frac{1}{\sqrt{6}} (X[0]e^{0\pi} + X[1]e^{0\pi} + X[2]e^{0\pi}) \\
1 + \frac{\sqrt{3}i}{2} + \frac{1}{2} &= x[1] = \frac{1}{\sqrt{6}} (X[0]e^{1\pi} + X[1]e^{1\pi} + X[2]e^{1\pi}) \\
0 &= x[3] = \frac{1}{\sqrt{6}} (X[0]e^{3\pi} + X[1]e^{3\pi} + X[2]e^{3\pi})
\end{align*}
\]

Solve the three equations, we can get \( X[0] = \sqrt{6} \), \( X[1] = -\sqrt{6} \) and \( X[2] = 0 \). Use the DFT coefficients to find

\[
\tilde{x} = \begin{bmatrix}
2 \\
1 + \frac{\sqrt{3}i}{2} + \frac{1}{2} \\
1 + \frac{\sqrt{3}i}{2} - \frac{1}{2} \\
1 - \frac{\sqrt{3}i}{2} - \frac{1}{2} \\
1 - \frac{\sqrt{3}i}{2} + \frac{1}{2}
\end{bmatrix}
\]

(f) Consider a length \( n \) discrete-time signal \( \tilde{x} \). Suppose that we know that its DFT coefficients \( \tilde{X} \) satisfy \( X[m] = 0 \), for all \( m \geq k \). What is the minimum number of sampling points we need to interpolate a unique \( \tilde{x} \)?

**Solution:** If we know \( X[m] = 0 \), for all \( m \geq k \), we only have \( X[0], \ldots, X[k-1] \) unknown. Hence we only need \( k \) samples to write down \( k \) equations to solve them.

In many real world applications, the time-domain signals are real. As you have proved in Homework 1 Question 4, the DFT coefficients for each real-vector exhibit the conjugate symmetry property. In other words, if \( \tilde{X} \) is the DFT coefficient of a real vector, \( \tilde{x} \), the \( k \)-th component of \( \tilde{X} \) satisfies \( X[k] = (X[n-k])^* \), for \( 0 \leq k \leq \lfloor \frac{n}{2} \rfloor \).

You can think about this in another way: we know \( u_{n-k} = u_{-k} \), which is the conjugate of \( u_k \). To construct a cosine signal \( \cos(\frac{2\pi k}{n}t) \) or a sine signal \( \sin(\frac{2\pi k}{n}t) \), you need \( e^{i\frac{2\pi k}{n}t} \) and \( e^{-i\frac{2\pi k}{n}t} \), along with conjugate coefficients to cancel the complex terms.

From Discussion 13A, you know that for a length \( n \) signal, if its DFT coefficients are zero for indexes beyond \( \pm k \), the minimum number of samples needed to reconstruct it is \( 2k + 1 \). But how would we ever know something like that? Why can we predict the zeros in DFT coefficients without knowing the whole signal?

Remember that we can implement circuits as low-pass filters, which allow sinusoidal signals less than certain frequency to pass but reject all higher-frequency sinusoids. Assume the cut-off frequency is \( f \).

Consider a continuous time periodic \( z(t) \), which consists of the sum of several sinusoids with different frequencies.
Let \( \tilde{z}(t) \) be the output signal after applying the low-pass filter to \( z(t) \). \( \tilde{z}(t) \) also consists of several sinusoids with different frequencies, none of which are greater than \( f \).

Now let’s perform sampling within a fixed duration, \( 0 \leq t < 1 \), which means if the sampling rate is \( f_s \) Hz (the unit of Hz is 1/second), we get \( n = f_s \), while the sampling period is \( \frac{1}{f_s} \) (second). Notice that when we do this type of sampling, we will make a sinusoid signal of frequency \( f_0 \) into a discrete-time signal that is periodic with period \( \frac{L}{f_s} \) (Check discussion)

(g) Consider the case where the sampling rate is \( f_s \) Hz for the time period \( 0 \leq t < 1 \). The number of sample points is \( n \). What is the highest frequency of real continuous-time sinusoids which can be represented purely in the DFT basis with size \( n \)? Consider both the cases of \( n \) odd or even.

**Solution:** Notice that under this sampling condition, a sinusoidal signal with frequency \( f \) is interpreted as \( \frac{f}{f_s} \) in the sampled discrete time signal. For \( n \) being odd, we have \( u_{\frac{n+1}{2}} \) and \( u_{\frac{n+1}{2}} + 1 = u_{\frac{n-1}{2}} \) to represent sinusoidal signal with frequency equal to \( \frac{n-1}{2n} \) (after scaling by \( f_s \)). Hence the frequency of the corresponding real continuous-time sinusoidal is \( \frac{(n-1)f_s}{2n} \).

For \( n \) being even, we can use \( u_{\frac{n}{2}} \) to represent a continuous-time cosine wave with frequency equal to \( \frac{n}{2n} \).

(h) Assume \( n \) (\( f_s \)) is big enough. Let the resulting signal to be \( \tilde{z} \). Consider its DFT coefficients, \( \tilde{Z} \), we can find \( Z[m] = 0 \) for all \( |m| > \pm k \). How is \( k \) related to \( f \), \( f_s \) and \( n \)?

**Solution:** Under the sampling rate \( f_s \) Hz, the cut-off frequency is interpreted as \( \frac{f}{f_s} \) on the DFT basis of size \( n \). Also, since \( Z[m] = 0 \) for all \( |m| > \pm k \), \( \frac{f}{f_s} \) must be less than \( \frac{k}{n} \); \( \frac{f}{f_s} < \frac{k}{n} \).

(i) As we know, the number of sample points we need to reconstruct \( \tilde{z}(t) \) is \( 2k + 1 \). What is the sampling rate if we take \( 2k + 1 \) samples during \( 0 \leq t < 1 \).

**Solution:** The sampling rate is \( 2k + 1 \) Hz because we take \( 2k + 1 \) samples within one second. The sampling period is \( \frac{1}{2k+1} \).

5. **Aliasing intuition in continuous time**

The concept of “aliasing” is intuitively about having a signal of interest whose samples look identical to a different signal of interest — creating an ambiguity as to which signal is actually present.

While the concept of aliasing is quite general, it is easiest to understand in the context of sinusoidal signals.

(a) Consider two signals,

\[
x_1(t) = a \cos(2\pi f_0 t + \phi)
\]

and

\[
x_2(t) = a \cos(2\pi (-f_0 + mf_s) t - \phi)
\]

where \( f_s = 1/T_s \). Are these two signals the same or different when viewed as functions of continuous time \( t \)?

**Solution:** The are different in the continuous time. \( x_1 \) has a frequency of \( f_0 \), which is different from the frequency of \( x_2 \), which is \( f_0 - mf_s \). They would only be the same if \( m = 0 \), in which case,

\[
x_2(t) = a \cos(-2\pi f_0 t - \phi) = a \cos(2\pi f_0 t + \phi) = x_1(t)
\]
(b) Consider the two signals from the previous part. These will both be sampled with the sampling interval \( T_s \). What will be the corresponding discrete-time signals \( x_{d,1}[n] \) and \( x_{d,2}[n] \)? (The \([n]\) refers to the \(n\)th sample taken — this is the sample taken at real time \(nT_s\).) Are they the same or different?

**Solution:**
Using the fact that the \(n\)th sample is taken at \( t = nT_s \), we can write out:

\[
x_{d,1}[n] = x_1(nT_s) = a \cos(2\pi f_0 n T_s + \phi)
\]

and

\[
x_{d,2}[n] = x_2(nT_s)
\]

\[
= a \cos(2\pi(-f_0 + m f_s) n T_s - \phi)
\]

\[
= a \cos(2\pi(-f_0 + m \frac{1}{T_s}) n T_s - \phi)
\]

\[
= a \cos(2\pi(-f_0 n T_s + mn) - \phi)
\]

\[
= a \cos(2\pi(-f_0 n T_s - \phi)
\]

\[
= a \cos(2\pi f_0 n T_s + \phi) = x_{d,1}[n]
\]

Therefore, for any \( t = nT_s \), \( x_{d,2}[n] = x_2(nT_s) = x_1(nT_s) = x_{d,1}[n] \). As can be seen, the sampled signals are exactly the same.

(c) What is the sinusoid \( a \cos(\omega t + \phi) \) that has the smallest \( \omega \geq 0 \) but still agrees at all of its samples (taken every \( T_s \) seconds) with \( x_1(t) \) above?

**Solution:**
As we can see from (b), we can choose any values of \( m \) for \( x_2(t) \) and the corresponding sampled signal will be identical to the sampled version of \( x_1(t) \), under the same sampling frequency, \( f_s \). To find the smallest \( \omega \geq 0 \), we need to find \( m \) such that \( \omega = 2\pi|f_0 + mf_s| \) is the smallest. Notice that we assume \( f_s \) and \( f_0 \) follow the sampling theorem, which implies \( \frac{T_s}{2} > |f_0| \).

6. Anti-aliasing filters

In this problem, we explore the difference between (1) projecting onto a subspace \( S \) and (2) sampling a vector and interpolating those samples assuming that those samples came from something truly in \( S \).

(a) Let \( \bar{x} \) be a vector of length 5. Our first approach is to sample \( f \) at 3 points, and to find a polynomial interpolation of those 3 points. More concretely, we sample \( \bar{x} \) at the coordinates 0, 2, 4, to find \( \bar{x}[0] = 1, \bar{x}[2] = -1, \bar{x}[4] = 0 \). Use Lagrange interpolation to find a polynomial \( f \) of degree at most 2, such that the \(i\)-th coordinate of \( \bar{x} \) is the evaluation of \( f \) at \( i \), i.e. \( f(i) = \bar{x}[i] \) for \( i = 0, 2, 4 \).

**Solution:** Our three data points are \((0, 1), (2, -1), (4, 0)\). Then our Lagrange interpolating polynomial is

\[
f(x) = \frac{3}{8}x^2 - \frac{7}{4}x + 1.
\]

(b) Using the interpolating polynomial \( f \) that you found in the previous part, compute all the remaining coordinates of \( \bar{x} \).

**Solution:** Using the polynomial \( f \) from the previous part, we compute \( f(1) = -\frac{3}{8}, f(3) = -\frac{7}{8} \). So we have \( \bar{x} = (1 \ -\frac{3}{8} \ -1 \ -\frac{7}{8} \ 0)^T \).
(c) Suppose that we sample \( \bar{x} \) again and find that \( \bar{x}[1] = 5 \). Find an interpolating polynomial \( g \), this time of degree at most 3, that satisfies this additional condition, and compute the remaining coordinates of \( \bar{x} \) in this case. Compare this with the vector \( \bar{x} \) from the previous part.

**Solution:** First, we note that any polynomial of the form \( f + ax(x - 2)(x - 4) \), for \( a \in \mathbb{R} \), is also an interpolating polynomial for the sampled points in part (a). This holds because the second term vanishes at \( x = 0, 2, 4 \). Thus we can simply find an \( a \in \mathbb{R} \) such that this polynomial evaluates to 5 at \( x = 1 \). This amounts to solving

\[
-\frac{3}{8} + a(1-2)(1-4) = 5
\]

which gives us \( a = \frac{43}{24} \), so our desired polynomial is

\[
g(x) = \frac{3}{8}x^2 - \frac{7}{4}x + 1 + \frac{43}{24}x(x-2)(x-4) = \frac{43}{24}x^3 - \frac{83}{8}x^2 + \frac{151}{12}x + 1.
\]

We compute \( g(1) = 5, g(3) = -\frac{25}{4} \). So we have \( \bar{x} = (1 \ 5 \ -1 \ -\frac{25}{4} \ 0)^T \). Observe that this differs from the vector \( \bar{x} \) from the previous part, in the entries at positions 1 and 3.

(d) The difference between our two vectors \( \bar{x} \) is that, when we sample \( \bar{x} \) at 0, 2, 4, all the other positions are still in principle free and could be anything. This means that anything in the subspace that has 0s in these positions could be added to \( \bar{x} \), and we would not be able to tell the difference from looking at the samples, so just interpolating these samples need not give us a good estimate of the actual \( \bar{x} \).

Find a \( 5 \times 5 \) matrix \( V \) that projects \( \bar{x} \) onto the subspace of polynomials of degree at most 2 evaluated at 0, 1, \ldots, 4, by considering the monomial basis. Here we want to be closest in the usual sense of 5-dimensional real vectors.

**Solution:** The monomial basis of size 5 is \( \{1, x, x^2, x^3, x^4\} \). We evaluate these monomials at 0, 1, \ldots, 4, and arrange them into a matrix as follows.

\[
A = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 \\
1 & 2 & 4 & 8 & 16 \\
1 & 3 & 9 & 27 & 81 \\
1 & 4 & 16 & 64 & 256
\end{pmatrix}.
\]

For any degree 4 polynomial \( p \), we can write the evaluation of \( p \) at 0, 1, \ldots, 4 as a linear sum of the columns of \( A \). If \( p \) is an interpolating polynomial of \( \bar{x} \), then this evaluation of \( p \) at these points should be equal to \( \bar{x} \). So we can write \( \bar{x} \) as a linear combination of the columns of \( A \), i.e. for some \( \alpha_0, \ldots, \alpha_4 \), we have

\[
\bar{x} = \alpha_0 \bar{a}_0 + \cdots + \alpha_4 \bar{a}_4,
\]

where \( \bar{a}_0, \ldots, \bar{a}_4 \) are the columns of \( A \). We want to find a \( 5 \times 5 \) matrix \( V \) that projects \( \bar{x} \) onto the subspace of polynomials of degree at most 2 evaluated at 0, 1, \ldots, 4, which is the subspace spanned by \( \bar{a}_0, \bar{a}_1, \bar{a}_2 \). We can write this condition as follows.

\[
V(\alpha_0 \bar{a}_0 + \cdots + \alpha_4 \bar{a}_4) = \alpha_0 \bar{a}_0 + \alpha_1 \bar{a}_1 + \alpha_2 \bar{a}_2.
\]

Let \( B \) be the matrix

\[
B = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 \\
1 & 2 & 4 & 0 & 0 \\
1 & 3 & 9 & 0 & 0 \\
1 & 4 & 16 & 0 & 0
\end{pmatrix}.
\]
Then the condition above translates to finding a matrix $V$ such that, for all $\alpha_0, \ldots, \alpha_4 \in \mathbb{R}$, we have

$$VA = B \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_4 \end{pmatrix}.$$ 

So we can compute $V$ as

$$V = BA^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ \frac{3}{8} & -\frac{1}{4} & -\frac{7}{4} & -\frac{1}{4} & \frac{5}{4} \\ \frac{8}{27} & -\frac{28}{27} & \frac{13}{27} & -\frac{20}{27} & \frac{1}{27} \\ \frac{63}{8} & -\frac{27}{8} & \frac{135}{8} & -17 & \frac{7}{8} \\ 16 & -\frac{160}{3} & 64 & -32 & \frac{8}{3} \end{pmatrix}.$$ 

(e) Suppose that the actual vector $\vec{x}$ is $\vec{x} = (1 \ 5 \ -1 \ 2 \ 0)^T$. Compute $\vec{y} = V\vec{x}$. Now assume that we sample $\vec{y}$ in the same positions $0, 2, 4$. Find an interpolating polynomial $h$ such that $h(i) = y[i]$ for $0 \leq i \leq 4$.

**Solution:** We first compute $\vec{y}$.

$$\vec{y} = V\vec{x} = \begin{pmatrix} 1 & -\frac{121}{24} & -\frac{211}{3} & -\frac{1559}{8} & -\frac{1136}{3} \end{pmatrix}^T$$

Using Lagrange interpolation on the points $(0, 1), (2, -\frac{211}{3}), (4, -\frac{1136}{3})$, we have

$$h(x) = -\frac{237}{8}x^2 + \frac{283}{12}x + 1.$$ 

We can check that $h(i) = y[i]$ for $0 \leq i \leq 4$.

(f) Suppose instead that we are interested in projecting onto the subspace $S$ spanned by the three DFT basis vectors $\vec{u}_{-1}, \vec{u}_0, \vec{u}_1$. Compute the $5 \times 5$ matrix $V$ such that $V\vec{x}$ is the projection of $\vec{x}$ onto $S$. We call $V$ an anti-aliasing filter.

**Solution:** We repeat the argument in part (d) for this case, where our matrices $A, B$ are now

$$A = \begin{pmatrix} \vec{u}_{-2} & \vec{u}_{-1} & \vec{u}_0 & \vec{u}_1 & \vec{u}_2 \end{pmatrix}$$

$$B = \begin{pmatrix} \vec{0} & \vec{u}_{-1} & \vec{u}_0 & \vec{u}_1 & \vec{0} \end{pmatrix}$$

where $\vec{0}$ is the vector with all entries zero. We want to find $V$ such that $VA = B$, and we do this by computing

$$V = BA^{-1} = \begin{pmatrix} \vec{0} & \vec{u}_{-1} & \vec{u}_0 & \vec{u}_1 & \vec{0} \end{pmatrix} \begin{pmatrix} \vec{u}^*_{-2} \\ \vec{u}^*_{-1} \\ \vec{u}^*_0 \\ \vec{u}^*_1 \\ \vec{u}^*_2 \end{pmatrix}.$$ 

(Here $*$ means conjugate transpose.) We can explicitly compute the columns of $V$, where the $k$-th
column, for \(0 \leq k \leq 4\), is

\[
\vec{v}_k = \vec{u}_{-2}[k]\bar{\vec{0}} + \vec{u}_{-1}[k]\bar{\vec{u}}_{-1} + \vec{u}_0[k]\bar{\vec{u}}_0 + \vec{u}_1[k]\bar{\vec{u}}_1 + \vec{u}_2[k]\bar{\vec{u}}_2
\]

\[
= 0 + \frac{1}{\sqrt{5}}e^{\frac{2\pi}{5}(k)}\bar{\vec{u}}_{-1} + \frac{1}{\sqrt{5}}\bar{\vec{u}}_0 + \frac{1}{\sqrt{5}}e^{-\frac{2\pi}{5}(k)}\bar{\vec{u}}_1
\]

\[
= \frac{1}{5}
\begin{pmatrix}
2\cos\frac{2\pi}{5}(k) + 1 \\
2\cos\frac{2\pi}{5}(k-1) + 1 \\
2\cos\frac{2\pi}{5}(k-2) + 1 \\
2\cos\frac{2\pi}{5}(k-3) + 1 \\
2\cos\frac{2\pi}{5}(k-4) + 1
\end{pmatrix}.
\]

(g) Check that the nullspace of \(V\) contains all vectors orthogonal to \(S\) for both the previous two cases.

**Solution:** The image of \(V\) is contained in \(S\), hence any vector orthogonal to \(S\) is in the nullspace of \(V\).

(h) Is the anti-aliasing filter matrix in the DFT case circulant? Why?

**Solution:** Yes, it is circulant.

We can see this because in the DFT basis, it is simply a diagonal matrix that zeros out the basis vectors orthogonal to the desired \(S\) and leaves the others preserved with a multiple of 1. Because it is diagonal in the DFT basis, it must be circulant in the standard basis.

7. GPS

The Global Positioning System (GPS) is a satellite-based navigation system that allows a receiver to accurately determine its location to within a meter. From a very high level view, the receiver measures the time it takes a signal to arrive from a GPS satellite to the receiver. If at least four satellites have a line-of-sight to the receiver, it can use the time delays to solve for its location. You have seen a version of this in the Locationing lab in 16A.

In this problem we study the methods used to determine the time delay between the satellite and receiver. We have simplified some details, while keeping the spirit of the method intact.

For the purpose of this problem, a satellite is assumed to transmit at rate of 1.023 Gbps (billion bits per second). It transmits a special pseudo-random sequence that is known to the receiver in advance. This sequence has the property that its correlation with any shifted version of itself is very low, but its correlation with itself, without a shift, is high. (Again, this is just like the locationing lab in 16A)

This pseudo-random sequence is periodic. Each period is composed of 1023 chips. A chip is 1000 repeated bits. (So this could be +1 sent 1000 times or -1 sent 1000 times.) The sequence itself repeats itself every 1 millisecond (1023 chips, each composed of 1000 bits, sent at a rate of 1.023 Gbps).

For the purpose of the problem, we consider a single 1 millisecond window—a single period of the signal. The received signal is processed according the following steps:

- The signal is passed through a low-pass filter, keeping the first 1000 frequency components in both the positive and negative frequencies.
- The filtered signal is sampled according to the Nyquist-Shannon sampling theorem, at a rate strictly higher than twice the highest frequency. We choose to sample once every 500 bits, or twice per chip, resulting in sampling rate of 2046 samples per window. (Compare this to the 2001 frequency components that were kept.)
• The result is then cross-correlated with the reference signal. (Which is just a clean nondelayed version of the reference signal that has been processed as above.) The delay which results in the highest correlation is returned as the estimated delay of the transmitted signal.

This approach works very well, but its accuracy is low. At best, it is accurate to a single half-chip, about a half-microsecond which corresponds to about 150m (recall that the speed of light is 300,000,000 m/s)

We can improve the accuracy by upsampling the signal to the original 1023,000 samples per window and therefore achieving nanosecond accuracy, which corresponds to less than a meter, by using the following steps:

• The signal from the second step above, is interpolated to have 1023,000 samples per window.
• The result is cross-correlated with a low-pass filtered reference signal. The delay that corresponds to the highest correlation is returned as the delay of the signal.

You will see that this approach is quite robust to noise, as it averages the noise over each chip.

Complete the accompanying iPython notebook as described below.

(a) Implement the low-pass filter.

**Solution:** It is easier to implement a low-pass filter in the frequency domain, so the first step is to compute the DFT of the signal. In the frequency domain, removing a frequency component $k$ is just replacing $\tilde{X}[k]$ with zeros. We replace all frequency components above $P$ and below $-P$ with zeros and then compute the inverse DFT to return the filtered signal to the time domain.

See the solution iPython notebook for the implementation.

(b) Implement the sampling step.

**Solution:** To sample, we can just pick one of every period samples. See the solution iPython notebook for the implementation.

(c) Implement the upsampling/interpolation function.

**Solution:** Let $n$ be the size of the original vector and $N > n$ be the size of the vector after upsampling. We would like to resample the same signal, having $N$ samples instead of $n$, all while keeping the same frequency components.

This is done by taking the DFT of the original signal and adding zeros in the middle of the vector (or on both sides if you consider it centered around zero). Finally the inverse DFT is taken to return the signal to the time domain.

The only problem is that the DFT matrices has different normalization factors. The problem is most easily noticeable using all-ones vector. The DFT representation of the all-one vector is $\sqrt{n}\tilde{u}_0$, but the upsampled all-ones vector has the DFT representation $\sqrt{N}\tilde{u}_0$. We need to normalize by multiplying by

$$\sqrt{\frac{N}{n}}$$

See the solution iPython notebook for the implementation.

(d) (Bonus) Do you think that you could’ve used these ideas to improve the performance of your system in 16A locationing lab?

**Solution:**

In the locationing lab, we were using 44kHz samples of audio. The speed of sound is 340 m/s which means that in one sample at 44kHz, sound moves almost 8mm. This effect limited our resolution of position to about a cm. If we had done interpolation, we could have gotten sub-cm precision estimates.
8. Your Own Problem

Write your own problem related to this week’s material and solve it. You may still work in groups to brainstorm problems, but each student must submit a unique problem. What is the problem? How to formulate it? How to solve it? What is the solution?

Contributors:

• Ioannis Konstantakopoulos.
• Baruch Sterin.