EECS 16B Designing Information Devices and Systems II
Spring 2016 Anant Sahai and Michel Maharbiz

Homework 13

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.)

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

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

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:

$$\dot{\vec{x}}(t) = A\vec{x}(t) + B\vec{u}(t)$$
$$\vec{y}(t) = C\vec{x}(t)$$

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

$$\vec{x}_d(k+1) = A_d\vec{x}_d(k) + B_d\vec{u}_d(k) \tag{1}$$
$$\vec{y}_d(k) = C_d\vec{x}_d(k) \tag{2}$$

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.

(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 this one step at a time. Everytime a new control is applied, this is a simple differential equation with a new constant input. How does $\dot{x}(t) = \lambda x(t) + u$ 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.)*

(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?

(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.

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

(f) Now consider a 2×2 diagonal matrix A that has the same eigenvalue repeated twice and a vector $B = \vec{b}$. Is it ever possible for the pair $(A, \vec{b})$ to be controllable? Show such a case or prove that it cannot exist.

(g) Now consider the case of complex eigenvalues for a diagonal matrix A (with all the eigenvalues distinct) with a vector $B = \vec{b}$ that consists of all 1s. Can you find a case in which $(A, \vec{b})$ is controllable but (A_d, B_d) is not controllable? What has to be true about the sampling period T in relation to the eigenvalues for this to happen?

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.

$$\vec{x} = \begin{bmatrix} x[0] & \dots & x[n-1] \end{bmatrix}^T \tag{3}$$

As we learned from the DFT module, this signal can be represented in the DFT basis. Let $\vec{X} = \begin{bmatrix} X[0] & \dots & X[n-1] \end{bmatrix}^T$ be the coordinates of $\vec{x}$ in the DFT basis.

$$\vec{X} = U^{-1}\vec{x} = U^*\vec{x} \tag{4}$$

where U is a matrix of the DFT basis vectors.

$$U = \begin{bmatrix} | & & | \\ \vec{u}_0 & \cdots & \vec{u}_{n-1} \\ | & & | \end{bmatrix} = \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & e^{\frac{2\pi i}{n}} & e^{\frac{2\pi i(2)}{n}} & \cdots & e^{\frac{2\pi i(n-1)}{n}} \\ 1 & e^{\frac{2\pi i(2)}{n}} & e^{\frac{2\pi i(4)}{n}} & \cdots & e^{\frac{2\pi i2(n-1)}{n}} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & e^{\frac{2\pi i(n-1)}{n}} & e^{\frac{2\pi i2(n-1)}{n}} & \cdots & e^{\frac{2\pi i(n-1)(n-1)}{n}} \end{bmatrix} \tag{5}$$

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 $X[m]$.

$$\vec{x} = X[0]\vec{u}_0 + \cdots + X[n-1]\vec{u}_{n-1} \tag{6}$$

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: $x = e^{i\square}$

(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.

(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 $k = 0, \ldots, n-1$.

(d) An alternative perspective on the DFT treats the $s_0, \ldots, s_{n-1}$ as places where we can evaluate 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 $k = 0, \ldots, n-1$.

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,

$$\vec{x} = \begin{bmatrix} 2 \\ 1 + \frac{\sqrt{3}i}{2} + \frac{1}{2} \\ x[2] \\ 0 \\ x[4] \\ x[5] \end{bmatrix}, \tag{7}$$

along with its corresponding DFT coefficients, $\vec{X}$. 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 $\vec{x}$ based on the above information. Do so and write down all entries of $\vec{x}$ and $\vec{X}$ explicitly.

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

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 $\vec{X}$ is the DFT coefficient of a real vector, $\vec{x}$, the k-th component of $\vec{X}$ 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 $\frac{f_0}{f_s}$ (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.

(h) Assume n (f_s) is big enough. Let the resulting signal to be $\vec{z}$. Consider its DFT coefficients, $\vec{Z}$, we can find $Z[m] = 0$ for all $|m| > \pm k$. How is k related to f, f_s and 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 $0 \leq t < 1$.

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?

(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 nth sample taken — this is the sample taken at real time nT_s.) Are they the same or different?

(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?

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 $\vec{x}$ 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 $\vec{x}$ at the coordinates $0, 2, 4$, to find $\vec{x}[0] = 1, \vec{x}[2] = -1, \vec{x}[4] = 0$. Use Lagrange interpolation to find a polynomial f of degree at most 2, such that the i-th coordinate of $\vec{x}$ is the evaluation of f at i, i.e. $f(i) = \vec{x}[i]$ for $i = 0, 2, 4$.

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

(c) Suppose that we sample $\vec{x}$ again and find that $\vec{x}[1] = 5$. Find an interpolating polynomial g, this time of degree at most 3, that satisfies this additional condition, and compute the remaining coordinates of $\vec{x}$ in this case. Compare this with the vector $\vec{x}$ from the previous part.

(d) The difference between our two vectors $\vec{x}$ is that, when we sample $\vec{x}$ 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 $\vec{x}$, 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 $\vec{x}$.

Find a 5×5 matrix V that projects $\vec{x}$ 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.

(e) Suppose that the actual vector $\vec{x}$ is $\vec{x} = \begin{pmatrix} 1 & 5 & -1 & 2 & 0 \end{pmatrix}^T$. 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 $0 \leq i \leq 4$.

(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 5×5 matrix V such that $V\vec{x}$ is the projection of $\vec{x}$ onto S. We call V an anti-aliasing filter.

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

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

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.

(b) Implement the sampling step.

(c) Implement the upsampling/interpolation function.

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

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.