UNIVERSITY OF CALIFORNIA, SANTA CRUZ
BOARD OF STUDIES IN COMPUTER ENGINEERING

CMPE 240: INTRODUCTION TO LINEAR DYNAMICAL SYSTEMS

PROBLEM SET 3

---

1. *Derivative with vector variable.* Let $y(x)$ be a real-valued (scalar) function that takes a vector argument x in $\mathbf{R}^n$. We define the derivative of the function y (taken at a given point x) as the vector $g \in \mathbf{R}^n$ such that

$$\frac{g^T \Delta x}{\|\Delta x\|} = \lim_{h \to 0} \frac{y(x + h\Delta x) - y(x)}{\|h\Delta x\|},$$

for any $\Delta x \in \mathbf{R}^n$. Equivalently you could constrain Δx to be a unit vector, and then

$$g^T \Delta x = \lim_{h \to 0} \frac{y(x + h\Delta x) - y(x)}{h}$$

Equivalently, and perhaps more clearly, the derivative is the vector g such that, for small Δx,

$$g^T \Delta x \approx \Delta y$$

where $\Delta y = y(x + \Delta x) - y(x)$ (with equality in the limit as $\|\Delta x\| \to 0$).

Use this definition to obtain the derivative g for:

   (a) $y(x) = b^T x$, where $b \in \mathbf{R}^n$.

   (b) $y(x) = x^T A x$, for a general matrix $A \in \mathbf{R}^{n \times n}$.

   (c) $y(x) = x^T A x$, where the matrix A is symmetric, *i.e.*, $A^T = A$.

2. *Bessel's inequality.* Suppose the columns of $U \in \mathbf{R}^{n \times k}$ are orthonormal. Show that $\|U^T x\| \leq \|x\|$. When do we have $\|U^T x\| = \|x\|$?

3. *Orthogonal matrices.*

   (a) Show that if U and V are orthogonal, then so is UV.

   (b) Show that if U is orthogonal, then so is U^{-1}.

   (c) Suppose that $U \in \mathbf{R}^{2 \times 2}$ is orthogonal. Show that U is either a rotation or a reflection. Make clear how you decide whether a given orthogonal U is a rotation or reflection.

Remark: this result is true in higher dimensions: an orthogonal matrix is either a rotation about some axis, or a reflection.

4. *Projection matrices.* A matrix $P \in \mathbf{R}^{n\times n}$ is called a *projection matrix* if $P = P^T$ and $P^2 = P$.

   (a) Show that if P is a projection matrix then so is $I - P$.

   (b) Suppose that the columns of $U \in \mathbf{R}^{n\times k}$ are orthonormal. Show that UU^T is a projection matrix. (Later we will show that the converse is true: every projection matrix can be expressed as UU^T for some U with orthonormal columns.)

   (c) Suppose $A \in \mathbf{R}^{n\times k}$ is full rank, with $k \leq n$. Show that $A(A^TA)^{-1}A^T$ is a projection matrix.

   (d) If $S \subseteq \mathbf{R}^n$ and $x \in \mathbf{R}^n$, the point y in S closest to x is called the *projection of x on S*. Show that if P is a projection matrix, then $y = Px$ is the projection of x on $\mathcal{R}(P)$. (Which is why such matrices are called projection matrices ...)

5. *Sensor integrity monitor.* A suite of m sensors yields measurement $y \in \mathbf{R}^m$ of some vector of parameters $x \in \mathbf{R}^n$. When the system is operating normally (which we hope is almost always the case) we have $y = Ax$, where $m > n$. If the system or sensors fail, or become faulty, then we no longer have the relation $y = Ax$.

   We can exploit the redundancy in our measurements to help us identify whether such a fault has occured. We'll call a measurement y *consistent* if it has the form Ax for some $x \in \mathbf{R}^n$. If the system is operating normally then our measurement will, of course, be consistent. If the system becomes faulty, we hope that the resulting measurement y will become inconsistent, *i.e.*, not consistent. (If we are *really* unlucky, the system will fail in such a way that y is still consistent. Then we're out of luck.)

   A matrix $B \in \mathbf{R}^{k\times m}$ is called an *integrity monitor* if the following holds:

   - $By = 0$ for any y which is consistent.
   - $By \neq 0$ for any y which is inconsistent.

   If we find such a matrix B, we can quickly check whether y is consistent; we can send an alarm if $By \neq 0$. Note that the first requirement says that every consistent y does not trip the alarm; the second requirement states that every inconsistent y does trip the alarm.

   Finally, the problem. Find an integrity monitor B for the matrix

$$A = \begin{bmatrix} 1 & 2 & 1 \\ 1 & -1 & -2 \\ -2 & 1 & 3 \\ 1 & -1 & -2 \\ 1 & 1 & 0 \end{bmatrix}.$$

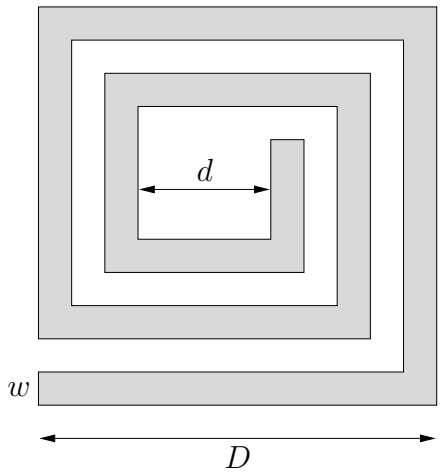
   Your B should have the smallest k (*i.e.*, number of rows) as possible.

As usual, you have to explain what you're doing, as well as giving us your explicit matrix B. You must also verify that the matrix you choose satisfies the requirements.

Hints:

- You might find one or more of the Matlab commands `orth`, `null`, or `qr` useful. Then again, you might not; there are many ways to find such a B.
- When checking that your B works, don't expect to have By exactly zero for a consistent y; because of roundoff errors in computer arithmetic, it will be really, really small. That's OK.
- Be very careful typing in the matrix A. It's not just a random matrix.

6. *Approximate inductance formula.* The figure below shows a *planar spiral inductor*, implemented in CMOS, for use in RF circuits.



The inductor is characterized by four key parameters:

- n, the number of turns (which is a multiple of 1/4, but that needn't concern us)
- w, the width of the wire
- d, the inner diameter
- D, the outer diameter

The inductance L of such an inductor is a complicated function of the parameters n, w, d, and D. The inductance L can be found by solving Maxwell's equations, which takes considerable computer time, or by fabricating the inductor and measuring the inductance.

In this problem you will develop a simple approximate inductance model of the form

$$\hat{L} = \alpha n^{\beta_1} w^{\beta_2} d^{\beta_3} D^{\beta_4},$$

where $\alpha, \beta_1, \beta_2, \beta_3, \beta_4 \in \mathbf{R}$ are constants that characterize the approximate model. (since L is positive, we have $\alpha > 0$, but the constants $\beta_2, \ldots, \beta_4$ can be negative.) This

simple approximate model, if accurate enough, can be used for design of planar spiral inductors.

The file `inductor_data.m` on the course web site contains data for 50 inductors. (The data is real, not that it would affect how you solve the problem ...) For inductor i, we give parameters n_i, w_i, d_i, and D_i (all in μm), and also, the inductance L_i (in nH) obtained from measurements. (The data are organized as vectors of length 50. Thus, for example, w_{13} gives the wire width of inductor 13.)

Your task, *i.e.*, the problem, is to find α, $\beta_1, \ldots, \beta_4$ so that

$$\hat{L}_i = \alpha n_i^{\beta_1} w_i^{\beta_2} d_i^{\beta_3} D_i^{\beta_4} \approx L_i \qquad \text{for } i = 1, \ldots, 50.$$

Your solution must include a clear description of how you found your parameters, as well as their actual numerical values. Note that we have not specified the criterion that you use to judge the approximate model (*i.e.*, the fit between $\hat{L}_i$ and L_i); we leave that to your engineering judgment. But be sure to tell us what criterion you use.

We define the *percentage error* between $\hat{L}_i$ and L_i as

$$e_i = 100|\hat{L}_i - L_i|/L_i.$$

Find the average percentage error for your model, *i.e.*, $(e_1 + \cdots + e_{50})/50$. (We are only asking you to find the average percentage error for your model; we do not require that your model minimize the average percentage error.)

Hint: you might find it easier to work with $\log L$.

7. *Least-squares deconvolution.* A communications channel is modeled by a finite-impulse-response (FIR) filter:

$$y(t) = \sum_{\tau=0}^{n-1} u(t-\tau)h(\tau),$$

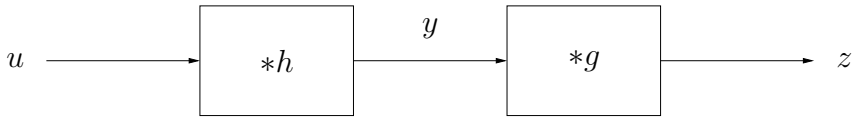
where $u : \mathbf{Z} \to \mathbf{R}$ is the channel input sequence, $y : \mathbf{Z} \to \mathbf{R}$ is the channel output, and $h(0), \ldots, h(n-1)$ is the impulse response of the channel. In terms of discrete-time convolution we write this as $y = h * u$.

You will design a *deconvolution filter* or *equalizer* which also has FIR form:

$$z(t) = \sum_{\tau=0}^{m-1} y(t-\tau)g(\tau),$$

where $z : \mathbf{Z} \to \mathbf{R}$ is the filter output, y is the channel output, and $g(0), \ldots, g(m-1)$ is the impulse response of the filter, which we are to design.

This is shown in the block diagram below.



The goal is to choose $g = (g(0), \ldots, g(m-1))$ so that the filter output is approximately the channel input delayed by D samples, *i.e.*, $z(t) \approx u(t-D)$. Since $z = g * h * u$ (discrete-time convolution), this means that we'd like

$$(g * h)(t) \approx \begin{cases} 0 & t \neq D, \\ 1 & t = D \end{cases}$$

We will refer to $g * h$ as the *equalized impulse response*; the goal is to make it as close as possible to a D-sample delay.

Specifically, we want the *least-squares* equalizer is g that minimizes the sum-of-squares error

$$\sum_{t \neq D} (g * h)(t)^2,$$

subject to the constraint

$$(g * h)(D) = 1.$$

To solve the problem below you'll need to get the file `deconv_data.m` from the class web page in the *Figures files* section. It will define the channel impulse response h as a Matlab vector `h`. (Indices in Matlab run from 1 to n, while the argument of the channel impulse response runs from $t = 0$ to $t = n - 1$, so `h(3)` in Matlab corresponds to $h(2)$.)

(a) Find the least-squares equalizer g, of length $m = 20$, with delay $D = 12$. Plot the impulse responses of the channel (h) and the equalizer (g). Plot the equalized impulse response ($g * h$).

(b) The vector y (also defined in `deconv_data.m`) contains the channel output corresponding to a signal u passed through the channel (*i.e.*, $y = h * u$). The signal u is binary, *i.e.*, $u(t) \in \{-1, 1\}$, and starts at $t = 0$ (*i.e.*, $u(t) = 0$ for $t < 0$). Pass y through the least-squares equalizer found in part a, to form the signal z. Give a histogram plot of the amplitude distribution of both y and z. (You can remove the first and last D samples of z before making the histogram plot.) Comment on what you find.

Figures hints: The command `conv` convolves two vectors; the command `hist` plots a histogram (of the amplitude distribution).