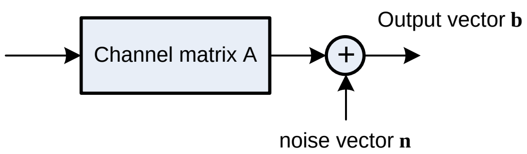
2021 Spring VLSI DSP Homework Assignment #1

Due date: 2021/3/30

Q1. Least Square Problem

Create an arbitrary matrix $\mathbf{A}_{6\times4}$ and an arbitrary vector $\mathbf{x}_{4\times1}$. The matrix $\mathbf{A}$ should have full column rank, which is 4, and with entries $|a_{i,j}| \le 10$. The vector $\mathbf{x}$ has it all its entry value $|x_i| \le 4$. Let $\mathbf{n}_{6\times1}$ be an arbitrary noise vector with all its entry value of ±1. Suppose $\mathbf{A}_{6\times4}$ is the channel matrix, $\mathbf{x}_{4\times1}$ is the original signal, and $\mathbf{b}_{6\times1}$ is the observed output with additive noise $\mathbf{n}_{6\times1}$



We therefore encounter a LS problem to solve $\mathbf{x}_{4\times1}$ subject to $\mathbf{A}_{6\times4} \cdot \mathbf{x}_{4\times1} = \mathbf{b}_{6\times1}$.

a) Calculate vector $\mathbf{b}_{6\times1}$ first use the $\mathbf{A}$, $\mathbf{x}$, $\mathbf{n}$ you create. Then calculate the least square estimate $\hat{\mathbf{x}}_{4\times1}$ using the pseudo inverse matrix scheme $\hat{\mathbf{x}} = \left(\mathbf{A}^t \cdot \mathbf{A}\right)^{-1} \cdot \mathbf{A}^t \cdot \mathbf{b}$ and determine its norm 2 value of error vector ($\mathbf{x}_{4\times1} - \hat{\mathbf{x}}_{4\times1}$).

b) Repeat problem a) by using the MatLab QR function, i.e. [Q,R]=qr[A]. Note that the returned Q is a 6×6 unitary matrix and R is a 6×4 upper triangular matrix with the last two rows nullified. Calculate $\tilde{\mathbf{b}}_{6\times1} = \mathbf{Q}^t \cdot \mathbf{b}$ first and set $\widehat{\mathbf{b}}_{4\times1}$ as the upper part of $\tilde{\mathbf{b}}_{6\times1}$.

   Then obtain the LS estimate as $\hat{\mathbf{x}}_{4\times1} = \mathbf{R}_{4\times4}^{-1} \cdot \widehat{\mathbf{b}}_{4\times1}$

c) Repeat problem b) except now using the Givens rotations to perform the QR factorization

d) Compare the LS solutions obtained from the three approaches. Are they identical? Are the norm 2 values of the three error vectors the same?

Q2. LMS filter design

For a least mean square (LMS) adaptive filter, assume the filter is of the form finite impulse response (FIR) and 15-tap long (i.e., with 15 coefficients b_0~b_{14} for x(n)~x(n-14)). Given an input signal consisting of 2 frequency components

$s(n) = \sin(2\pi*n/12)+\cos(2\pi*n/4)$

develop an adaptive low pass filter design

Set the target as a low pass filter to remove the high frequency component $\cos(2\pi*n/4)$ and use $\sin(2\pi*n/12)$ as the desired (or training) signal for LMS adaptation. Assume the step size μ is 2^{-2}.

- write a Matlab code to simulate the LMS based adaptive filtering. Calculate the RMS (root mean square) value of the latest 16 prediction errors (i.e., r = sqrt(($e^2(n)$ + $e^2(n-1)$ + ….+ $e^2(n-15)$)/16) and the adaptation is considered being converged if this value is less than 10% of RMS (root mean square) value of the desired signal, which equals 0.1/sqrt(2).
- Show the plot of "r" versus "n" and indicate when the filter converges, i.e. how many training samples are required
- Show the plot of filter coefficients $b_i(n)$, for i = 0~14, versus "n" and see if the values of filter coefficients remain mostly unchanged after convergence
- Apply a 64-point FFT to the impulse response of the converged filter and verify the filter is indeed a low pass one. Note that the input vector to the 64-point FFT is (b_0, b_1, ….., b_{14}, 0,0,…..,0) with 49 trailing zeros.
- Change the step size μ to 2^{-4} and see how the behavior of the adaptive filter changes.
- Conduct simulation with a sufficiently large number of samples to see how small the value of "r" can be (the convergence bias)