Some Notes from the Book: Priciples of Adaptive Filters and Self-learning Systems by Anthony Zaknick

John L. Weatherwax*

October 8, 2004

Introduction

Here are some notes that I wrote up as I worked through this excellent book. I've worked hard to make these notes as good as I can, but I have no illusions that they are perfect. If you feel that that there is a better way to accomplish or explain an exercise or derivation presented in these notes; or that one or more of the explanations is unclear, incomplete, or misleading, please tell me. If you find an error of any kind – technical, grammatical, typographical, whatever – please tell me that, too. I'll gladly add to the acknowledgments in later printings the name of the first person to bring each problem to my attention.

*wax@alum.mit.edu

Linear Systems and Stochastic Processes

Problem Solutions

Problem 3 (FIR filters)

Part (a): In the notation here the list given represents the first four values of $h[n]$. That is

$$\{0.2, 0.3, 0.3, 0.3\} = \{h[0], h[1], h[2], h[3]\} .$$

Thus we have $N = 4$. To have a linear phase filter when we are given the finite impulse response in this form requires

$$h^*[n] = h[N-1-n] , \tag{1}$$

or

$$h^*[n] = -h[N-1-n] . \tag{2}$$

For this problem none of these conditions hold true. Thus this is not a linear phase filter.

Part (b): Since $h[0] = 0.1 \neq h[N-1] = h[5] = 0.2$ (or its negative), neither of these two conditions hold and this is not a linear phase filter.

Part (c): For this part Equation 1 holds and this is a linear phase filter.

Part (d): This is not a linear phase filter.

Part (e): For this part Equation 2 holds and this is a linear phase filter.

Problem 4 (the $H(z)$ for some FIR filters)

To have a linear phase filter we need to write $H(e^{j\theta})$ as

$$H(e^{j\theta}) = A(e^{j\theta})e^{j(\beta-\alpha\theta)} .$$

Part (a): For a linear time-invariant system we will rely on the definition of $H(z)$ of

$$H(z) = \sum_{m=0}^{q} h[m]z^{-m} .$$

We have

$$H(z) = 0.5 + 0.5z^{-1} = 0.5(1 - z^{-1}) .$$

This has a zero of 1 and no pole. We now write $H(e^{j\theta})$ as

$$H(e^{j\theta}) = 0.5(1 + e^{-j\theta}) = 0.5(e^{j\theta/2} + e^{j\theta/2})e^{-j\theta/2} = \cos(\theta/2)e^{-j\theta/2} .$$

Part (b): We have

$$H(z) = 0.5 - 0.5z^{-2} = 0.5(1 - z^{-1})(1 + z^{-1}) .$$

This has two zeros of ± 1. We now write $H(e^{j\theta})$ as

$$\begin{aligned} H(e^{j\theta}) &= 0.5(1 - e^{+2j\theta}) = 0.5(e^{-j\theta} - e^{j\theta})e^{j\theta} = -j\sin(\theta)e^{j\theta} \\ &= \sin(\theta)e^{j(\theta - \frac{\pi}{2})} . \end{aligned}$$

Part (c): As in the previous part we have

$$H(z) = 0.5 + 0.5z^{-2} = 0.5z^{-2}(z^2 + 1) = 0.5z^{-2}(z - i)(z + i) .$$

This has two roots of $\pm i$. Now we write $H(e^{j\theta})$ as

$$H(e^{j\theta}) = 0.5 + 0.5e^{2j\theta} = 0.5(e^{-j\theta} + e^{j\theta})e^{j\theta} = \cos(\theta)e^{j\theta} .$$

Part (d): We have

$$H(z) = 0.25 - 0.5z^{-1} + 0.25z^{-2} = 0.25z^{-2}(z^2 - 2z + 1) = 0.25z^{-2}(z - 1)^2 .$$

This FIR has only a single root of 1 that is of multiplicity one. Now we write $H(e^{j\theta})$ as

$$\begin{aligned} H(e^{j\theta}) &= 0.25 - 0.5e^{j\theta} + 0.25e^{2j\theta} = 0.25(e^{-j\theta} - 2 + e^{j\theta})e^{j\theta} \\ &= 0.25(e^{-j\theta/2} - e^{j\theta/2})^2 e^{j\theta} = 0.25(2j\sin(\theta/2))^2 e^{j\theta} = \sin(\theta/2)^2 e^{j(\theta+\pi)} . \end{aligned}$$

Problem 5

To be orthogonal means that $a^T b = 0$. To be orthonormal means that they are orthogonal and also that $||a|| = 1$ and $||b|| = 1$.

Problem 6 (Toeplitz matrices)

The matrices a, c, e, and f are Toeplitz. The matrices c, d, e, and f are symmetric which means that they are *centrosymmetric* matrices.

Problem 7 (the inverse of an orthogonal matrix)

For a matrix A to be orthogonal means that $A^T A = I$. Its inverse is then $A^{-1} = A^T$.

Problem 8 (the importance of ergodicity)

An ergodic process are important because we can then use them to generate meaningful ensemble statistics from a single time series.

Problem 11 (computing the autocorrelation function)

Consider the discrete sequence $x[n] = A\sin(n\omega_0 + \phi)$ where ϕ is a uniform random variable between $(-\pi, \pi)$. Then the mean process, $\bar{x}$, is given by

$$\begin{aligned}\bar{x} = E\{x[n]\} &= \frac{1}{2\pi}\int_{-\pi}^{\pi} A\sin(n\omega_0+\phi)d\phi \\ &= -\frac{1}{2\pi}\, A\cos(n\omega_0+\phi)|_{-\pi}^{\pi} \\ &= -\frac{A}{2\pi}(\cos(n\omega_0+\pi) - \cos(n\omega_0-\pi)) \\ &= -\frac{A}{2\pi}(-\cos(n\omega_0) + \cos(n\omega_0)) = 0\,.\end{aligned}$$

The autocorrelation function for this series is computed as

$$\begin{aligned}r_x(m) &= E\{x[k]x[k+m]^*\} \\ &= E\{A\sin(k\omega_0+\phi)A\sin((k+m)\omega_0+\phi)\} \\ &= \frac{A^2}{2\pi}\int_{-\pi}^{\pi}\sin(k\omega_0+\phi)\sin((k+m)\omega_0+\phi)d\phi \\ &= \frac{A^2}{4\pi}\int_{-\pi}^{\pi}(\cos(m\omega_0) + \cos(2k\omega_0+m\omega_0+2\phi))d\phi \\ &= \frac{A^2}{2}\cos(m\omega_0) + \frac{A^2}{4\pi}\int_{-\pi}^{\pi}\cos(2k\omega_0+m\omega_0+2\phi)d\phi\,.\end{aligned}$$

This last integral evaluates to 0 and we end with

$$r_x(m) = \frac{A^2}{2}\cos(m\omega_0)\,.$$

If we now consider the harmonic process $x[n] = Ae^{j(n\omega_0+\phi)}$ we find its mean to be

$$\begin{aligned}\bar{x} = E\{x[n]\} &= \frac{A}{2\pi}\int_{-\pi}^{\pi} e^{j(n\omega_0+\phi)}d\phi = \frac{A}{2\pi}e^{jn\omega_0}\left(\frac{e^{j\phi}}{j}\right|_{-\pi}^{\pi} \\ &= \frac{A}{2\pi}\frac{e^{jn\omega_0}}{j}(e^{\pi j} - e^{-\pi j}) = 0\,.\end{aligned}$$

The autocorrelation function for this process is given by using the definition or

$$\begin{aligned}r_x(m) &= E\{x[k]x[k+m]^*\} \\ &= \frac{|A|^2}{2\pi}\int_{-\pi}^{\pi} e^{j(k\omega_0+\phi)}e^{-j((k+m)\omega_0+\phi)}d\phi \\ &= \frac{|A|^2}{2\pi}e^{-jm\omega_0}\int_{-\pi}^{\pi} d\phi = |A|^2 e^{-jm\omega_0}\,.\end{aligned}$$

Problem 12 (the autocorrelation matrix)

The 2×2 requested autocorrelation matrix is

$$R_x = \begin{bmatrix} r_x(0) & r_x^*(1) \\ r_x(1) & r_x(0) \end{bmatrix} = \begin{bmatrix} \frac{A^2}{2} & \frac{A^2}{2}\cos(\omega_0) \\ \frac{A^2}{2}\cos(\omega_0) & \frac{A^2}{2} \end{bmatrix}$$
$$= \frac{A^2}{2} \begin{bmatrix} 1 & \cos(\omega_0) \\ \cos(\omega_0) & 1 \end{bmatrix}.$$

Problem 13 (computing the power spectrum)

A first-order discrete autoregressive process has an autocorrelation function given by $r_x(m) = \alpha^{|m|}$, with $|\alpha| < 1$. We can compute the power spectrum $P_x(e^{j\theta})$ by its definition. We find

$$P_x(e^{j\theta}) = \sum_{k=-\infty}^{\infty} r_x(k)e^{-jk\theta} = \sum_{k=-\infty}^{\infty} \alpha^{|k|}e^{-jk\theta} = \sum_{k=-\infty}^{-1} \alpha^{-k}e^{-jk\theta} + \sum_{k=0}^{\infty} \alpha^k e^{-jk\theta}$$
$$= \sum_{k=1}^{\infty} \alpha^k e^{jk\theta} + \sum_{k=0}^{\infty} (\alpha e^{-j\theta})^k = \sum_{k=0}^{\infty} \alpha^k e^{jk\theta} - 1 + \frac{1}{1-\alpha e^{-j\theta}}$$
$$= \frac{1}{1-\alpha e^{j\theta}} + \frac{1}{1-\alpha e^{-j\theta}} - 1 = \frac{1-\alpha^2}{(1-\alpha e^{j\theta})(1-\alpha e^{-j\theta})}, \tag{3}$$

when we simplify.

Problem 14 (the power spectrum of filtered white noise)

Since the variance of white noise is $\sigma_w^2 = 1$ the power spectral density of w is $P_w(e^{j\theta}) = \sigma_w^2$ or a constant. After filtering with the suggested first order LSI filter $H(z) = \frac{1}{1+\frac{1}{4}z^{-1}}$ the output signal (denoted $x[n]$) has a power spectral density $P_x(e^{j\theta}) = P_x(e^{j\theta})|H(e^{j\theta})|^2$ so we get

$$P_x(e^{j\theta}) = \frac{\sigma_w^2}{(1+\frac{1}{4}e^{j\theta})(1+\frac{1}{4}e^{-j\theta})}.$$

The autocorrelation of $x[n]$ is computed via the inverse Fourier transform of $P_x(e^{j\theta})$ as

$$r_x(k) = \frac{1}{2\pi}\int_{-\pi}^{\pi} P_x(e^{j\theta})e^{jk\theta}d\theta.$$

Since we have seen a power spectral density in this form in Equation 3 we don't have to do the inverse Fourier transform directly. Instead we can make Equation 3 look like the above expression by replacing α in with $-\frac{1}{4}$. There is a constant factor of $1-\alpha^2$ that needs to be "removed" from Equation 3. Thus from the above power spectral density we see that the autocorrelation function that it corresponds to is given by

$$\left(\frac{\sigma_w^2}{1-\left(\frac{1}{4}\right)^2}\right)\left(-\frac{1}{4}\right)^{|m|}.$$

Problem 15

Yes this statement is true, since for any wide sense stationary process x with an autocorrelation function $r_x(m)$ that is passed through a LSI system with impulse response $h[m]$ we have

$$\sigma_y^2 = \sum_{l=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} h[l] r_x(m-l) h^*[m] \,.$$

This equation is discussed in the book. If x is the white noise process then it has a delta function for its autocorrelation function $r_x(\cdot)$. That is $r_x(m-l) = 0$ if $m \neq l$ and $r_x(0) = \sigma_x^2$ in that case. Thus in this case σ_y^2 above simplifies and we have

$$\sigma_y^2 = \sum_{m=-\infty}^{\infty} h[m] \sigma_x^2 h^*[m] = \sigma_x^2 \sum_{m=-\infty}^{\infty} |h[m]|^2 \,.$$

Optimization and Least Squares Estimation

Notes on the text

Notes on linear regression–fitting data to a line

We begin with our χ^2 objective function we seek to minimize with respect to a and b

$$\chi^2(a,b) = \sum_{i=1}^{N} \left(\frac{y_i - a - bx_i}{\sigma_i} \right)^2 .$$

To perform this minimization we take the a and the b derivatives of $\chi^2(a,b)$ and set the result equal to zero. Doing this for a and then simplifying we find

$$\frac{\partial \chi^2}{\partial a} = 2 \sum_{i=1}^{N} \left(\frac{y_i - a - bx_i}{\sigma_i} \right) (-1) = 0 .$$

Distributing the sums to each term gives

$$\sum_{i=1}^{N} \frac{y_i}{\sigma_i} - a \sum_{i=1}^{N} \frac{1}{\sigma_i} - b \sum_{i=1}^{N} \frac{x_i}{\sigma_i} = 0 .$$

or

$$S_y - aS - bS_x = 0 .$$

Taking the derivative of $\chi^2(a,b)$ with respect to b now gives

$$\frac{\partial \chi^2}{\partial b} = 2 \sum_{i=1}^{N} \left(\frac{y_i - a - bx_i}{\sigma_i} \right) (-x_i) = 0 .$$

Again distributing the sums to each term gives

$$\sum_{i=1}^{N} \frac{x_i y_i}{\sigma_i} - a \sum_{i=1}^{N} \frac{x_i}{\sigma_i} - b \sum_{i=1}^{N} \frac{{x_i}^2}{\sigma_i} = 0 .$$

or

$$S_{xy} - aS_x - bS_{xx} = 0 .$$

These two equations as a system for a and b we have

$$\begin{bmatrix} S & S_x \\ S_x & S_{xy} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} S_y \\ S_{xy} \end{bmatrix} .$$

The solution for a and b can be obtained with Crammer's rule

$$\begin{bmatrix} a \\ b \end{bmatrix} = \frac{1}{SS_{xx} - S_x^2} \begin{bmatrix} S_{xx} & -S_x \\ -S_x & S \end{bmatrix} \begin{bmatrix} S_y \\ S_{xy} \end{bmatrix} .$$

Thus we have shown that

$$a = \frac{S_{xx}S_y - S_x S_{xy}}{\Delta} \quad \text{and}$$
$$b = \frac{SS_{xy} - S_x S_y}{\Delta} \quad \text{with}$$
$$\Delta = SS_{xx} - S_x^2 .$$

Notes on diagonalization using Jacobi's algorithm

To make $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ symmetric using the Jacobi algorithm we need to find $J = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$ such that $J^T A$ is symmetric. Considering the elements in the product of $J^T A$ we see that this requires

$$\begin{aligned} ca_{12} - sa_{22} &= sa_{11} + ca_{21} \quad \text{or} \\ c(a_{12} - a_{21}) &= s(a_{11} + a_{22}) \quad \text{or} \\ \frac{c}{s} &= \frac{a_{11} + a_{22}}{a_{12} - a_{21}} . \end{aligned}$$

We define the ratio of c to s as ρ. Now given the matrix A, the value of ρ is fixed via the above expression. Since $c^2 + s^2 = 1$ we can put $c = \pm\sqrt{1 - s^2}$ into the expression for ρ to get

$$\rho = \pm\frac{\sqrt{1 - s^2}}{s} ,$$

or

$$1 - s^2 = s^2\rho^2 .$$

or

$$\frac{1}{1 + \rho^2} = s^2 \quad \text{so} \quad s = \pm\frac{1}{\sqrt{1 + \rho^2}} .$$

Given this value for s we find that c is then given by

$$c = \pm\sqrt{1 - s^2} = \pm\sqrt{1 - \frac{1}{1 + \rho^2}} = \pm\sqrt{\frac{\rho^2}{1 + \rho^2}} = \pm\frac{\rho}{\sqrt{1 + \rho^2}} = \rho s .$$

We can take the plus sign above since we just need to know that s and c exist i.e. we only need one solution. Let the symmetric matrix this J produces be denoted as B. That is $B = J^T A = \begin{bmatrix} b_{11} & b_{12} \\ b_{12} & b_{22} \end{bmatrix}$. Note that via its components we can see that B is symmetric (as it must be by the way we picked s and c above). We next want to find a rotation, J_2, such that $J_2^T B J_2$ is *diagonal*. In the same way as we did the first part of this section, this means that we want to pick c and s such that

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} b_{11} & b_{12} \\ b_{12} & b_{22} \end{bmatrix} \begin{bmatrix} c & s \\ -s & c \end{bmatrix} = \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix} .$$

When we multiply the left-hand-side matrices to get

$$\begin{aligned} J_2^T B J_2 &= \begin{bmatrix} c & -s \\ s & c \end{bmatrix} \begin{bmatrix} b_{11}c - sb_{12} & sb_{11} + cb_{12} \\ b_{12}c - sb_{22} & sb_{12} + cb_{22} \end{bmatrix} \\ &= \begin{bmatrix} c(b_{11}c - sb_{12}) - s(b_{12}c - sb_{22}) & c(sb_{11} + cb_{12}) - s(sb_{12} + cb_{22}) \\ s(cb_{11} - sb_{12}) + c(cb_{12} - sb_{22}) & s(sb_{11} + cb_{12}) + c(sb_{12} + cb_{22}) \end{bmatrix} \\ &= \begin{bmatrix} c^2b_{11} + s^2b_{22} - 2scb_{12} & c^2b_{12} - s^2b_{12} + sc(b_{11} - b_{22}) \\ c^2b_{12} - s^2b_{12} + sc(b_{11} - b_{22}) & c^2b_{22} + s^2b_{11} + 2scb_{12} \end{bmatrix} . \end{aligned}$$

To make this product matrix diagonal requires

$$c^2 b_{12} - s^2 b_{12} + sc(b_{11} - b_{22}) = 0\,.$$

Lets define the variable ξ as $\xi = \frac{b_{22}-b_{11}}{2b_{12}}$, so that the above becomes

$$c^2 - s^2 - 2\xi sc = 0\,.$$

Introduced another variable t such that $t = \frac{s}{c}$ by dividing the above equation by c^2 to get

$$1 - t^2 - 2\xi t = 0\,.$$

This is the quadratic equation $t^2 + 2\xi t - 1 = 0$ in t. When we solve for t we get

$$t = \frac{-2\xi \pm \sqrt{4\xi^2 + 4}}{2} = -\xi \pm \sqrt{\xi^2 + 1}\,.$$

Since B is a fixed matrix (once we specify the first rotation J) the value of ξ is fixed and from the above the value of t is fixed. As earlier, $s = \pm\sqrt{1 - c^2}$ so that

$$t = \frac{s}{c} = \frac{\pm\sqrt{1 - c^2}}{c}\,.$$

by squaring we get $c^2 t^2 = 1 - c^2$. Solving for c^2 we get $c^2 = \frac{1}{1+t^2}$ so

$$c = \frac{1}{\sqrt{1 + t^2}}\,,$$

when we take the positive root. Then

$$s = \pm\sqrt{1 - \frac{1}{1 + t^2}} = \pm\frac{t}{\sqrt{1 + t^2}} = tc\,,$$

as claimed. The total transformation on A, first by J and then J_2, is then $J_2^T(J^T A)J_2 = D$ is a diagonal matrix. These two combined transformations can be written as $(JJ_2)^T A J_2$ and provide the two matrices JJ_2 and J_2 that diagonalize the matrix A.

Notes on the QR algorithm with Householder transformations

The book claims that if we form the Householder matrix $H(v)$ as

$$H(v) = I - \frac{2vv^T}{||v||^2}\,, \tag{4}$$

and given x if we take $v = x - ||x||e_1$ we get

$$H(v)x = ||x||e_1\,.$$

To show that first note that $H(v)x = x - \frac{2vv^T x}{||v||^2}$. Now consider $v^T x$. We have

$$v^T x = (x - ||x||e_1)^T x = x^T x - ||x||e_1^T x = ||x||^2 - ||x||e_1^T x = ||x||(||x|| - e_1^T x)\,.$$

Now consider $||v||^2$ or

$$\begin{aligned} ||v||^2 &= (x - ||x||e_1)^T(x - ||x||e_1) = ||x||^2 - 2||x||e_1^T x + ||x||^2 \\ &= 2||x||^2 - 2||x||e_1^T x = 2||x||(||x|| - e_1^T x)\,. \end{aligned}$$

Thus

$$\frac{v^T x}{||v||^2} = \frac{||x||(||x|| - e_1^T x)}{2||x||(||x|| - e_1^T x)} = \frac{1}{2}\,.$$

Therefore we have shown

$$H(v)x = x - \frac{2}{2}v = x - v = x - (x - ||x||e_1) = ||x||e_1\,.$$

as claimed.

Problem Solutions

Problem 1 (maximizing volume)

After we cut out the corners the dimension of the "base" is $(6-2x)\times(6-2x)$ with a height of x giving a volume of

$$V(x) = (6-2x)^2 x = 4x^3 - 24x^2 + 36x\,. \tag{5}$$

To optimize $V(x)$ with respect to x we take the derivative with respect to x, set the result equal to zero and then solve for x. We find

$$V'(x) = 12x^2 - 48x + 36 = 0\,.$$

When we divide by 12 and factor we get $(x-1)(x-3) = 0$, thus $x = 1$ or $x = 3$. We find the second derivative of $V(x)$ given by $V''(x) = 2x - 4$. We find $V''(1) = -2$ and $V''(3) = 2$. For V to be a maximum at x we want $V''(x) < 0$ thus $x = 1$ gives the maximum volume of $V(1) = 4^2 = 16$.

Problem 4 (formulating of the acoustic position system example)

We could use the values $T_{C_1}, T_{C_2}, T_{C_3}, \cdots$ directly, but that would not be an optimal choice due to the fact that V is typically very large all of these expressions are very small. This means that the set of five equations could be ill-conditioned. We can obtain a more robust solution by using differences between the original measurements $T_{C_1}, T_{C_2}, T_{C_3}, \cdots$. The differences are chosen to be from points that will give the largest measurements. That is we would want to differ T_{C_1} and T_{C_3} (for G_0) and T_{C_2} and T_{C_0} (for H_0). We also scale these differences by V (the velocity of sound in sea water).

Parametric Signal and System Modeling

Problem Solutions

Problem 1 (AR, MA, and ARMA models)

Recall that our linear time invariant filter can be represented in a rational transform form

$$\frac{B_q(z)}{A_p(z)} = \frac{\sum_{k=0}^{q} b_k z^{-k}}{1 + \sum_{k=1}^{p} a_k z^{-k}} ,$$

which when input the discrete signal $x[n]$ produces the discrete output signal $y[n]$ given by

$$y[n] = \sum_{k=0}^{q} b_k x[n-k] + \sum_{k=1}^{p} a_k y[n-k] .$$

We have a moving average model if all $a_k = 0$. We have an autoregressive model if all $b_k = 0$. We have a mixed ARMA model if neither a_k or b_k are all zero.

Problem 2 (fitting ARMA models)

Finding the coefficients of an AR(p) model is simplified in that the relationship between the desired AR parameters a_k, and the autocorrelation function of the time series $x[n]$ leads to the set of simultaneous *linear* equations. Models with moving average terms requires the solution of nonlinear equations to estimate their parameters b_k. Thus estimating a_k is easier.

Problem 3 (deriving $\frac{\partial \varepsilon}{\partial a_k^*}$)

Warning: I think I might have something wrong with this derivation since it does not match exactly the result from the book. If anyone sees that anything is wrong (or correct and the book is wrong) please contact me.

Equation 4.5 from the book is given by

$$\begin{aligned}
\varepsilon &= \frac{1}{2\pi} \int_{-\pi}^{\pi} \left| E(e^{j\theta}) \right|^2 d\theta \\
&= \frac{1}{2\pi} \int_{-\pi}^{\pi} \left| X(e^{-j\theta}) - \frac{B(e^{-j\theta})}{A(e^{-j\theta})} \right|^2 d\theta \\
&= \frac{1}{2\pi} \int_{-\pi}^{\pi} \left[X(e^{-j\theta}) - \frac{\sum_{k=0}^{q} b_k e^{-j\theta k}}{1 + \sum_{k=1}^{p} a_k e^{-j\theta k}} \right] \left[X(e^{-j\theta})^* - \frac{\sum_{k=0}^{q} b_k^* e^{j\theta k}}{1 + \sum_{k=1}^{p} a_k^* e^{j\theta k}} \right] d\theta .
\end{aligned}$$

When we take the derivative of this expression with respect to the parameter a_k^* we get

$$\frac{\partial \varepsilon}{\partial a_k^*} = \frac{1}{2\pi}\int_{-\pi}^{\pi}\left[X(e^{-j\theta}) - \frac{\sum_{k=0}^{q} b_k e^{-j\theta k}}{1+\sum_{k=1}^{p} a_k e^{-j\theta k}}\right]\frac{\sum_{k=1}^{q} b_k^* e^{j\theta k}}{[1+\sum_{k=1}^{p} a_k^* e^{j\theta k}]^2} e^{j\theta k} d\theta$$
$$= \frac{1}{2\pi}\int_{-\pi}^{\pi}\left[X(e^{-j\theta}) - \frac{\sum_{k=0}^{q} b_k e^{-j\theta k}}{1+\sum_{k=1}^{p} a_k e^{-j\theta k}}\right]\frac{B(e^{-j\theta})^*}{A(e^{-j\theta})^{2^*}} e^{j\theta k} d\theta$$

Warning: In the book the external exponent is $e^{-j\theta k}$ while in the above we have $e^{j\theta k}$. There maybe a typo in the book. Again if anyone sees anything wrong with what I have done please let me know.

Problem 5 (problems with the Pade approximation method)

In the Pade' approximation method requires that the signal $x[n]$ be the output from the LSI model with transfer function $H(z)$ given by

$$H(z) = \frac{B(z)}{A(z)} = \frac{\sum_{k=0}^{q} b_k z^{-k}}{1+\sum_{k=1}^{p} a_k z^{-k}} = \sum_{n=0}^{\infty} h[n] z^{-n}\,.$$

Since we desire the output from this linear filter to equal $x[n]$ and we have $p+q+1$ unknowns given by the values of $\{a_k\}_{k=1}^{p}$ and $\{b_k\}_{k=0}^{q}$ to determine these we equate the output from the above LSI system to $x[n]$ for this many points (enough values so that we can uniquely determine a_k and b_k). One problem with this method is that in practice $x[n]$ may have noise mixed in with the signal and as such filtering to be specific to the of points $x[n]$ might be. One problem with this method is that only the first $p+q+1$ points from $x[n]$ are used to estimate the coefficients $\{a_k\}_{k=1}^{p}$ and $\{b_k\}_{k=0}^{q}$ thus a large sample history will be ignored unless we take p and q very large.

Problem 6 (using the Pade' approximation method)

Pade's method is based on the following. Recall that we desire to have $H(z) \approx \frac{B(z)}{A(z)}$. If we assume that $B(z) = A(z)H(z)$ then the impulse response form of this equation means that

$$b_n = \sum_{k=0}^{\infty} a_k h[n-k] = h[n] + \sum_{k=1}^{p} a_k h[n-k]\,. \tag{6}$$

for $0 \le n \le q$ and is zero for all other value for n. Take $n = 0, 1, \cdots q-1, q, q+1, q+2, \cdots, p+q$ sequentially in the above equation we get

$$h[n] + \sum_{k=1}^{p} a_k h[n-k] = \begin{cases} b_n & n = 0, 1, 2, \cdots, q \\ 0 & n = q+1, q+2, \cdots q+p \end{cases}\,. \tag{7}$$

If we assume h is causal ($h[n] = 0$ when $n < 0$) when we write out the above, we get the following system of equations

$$\begin{aligned}
b_0 &= h[0] \\
b_1 &= h[1] + a_1 h[0] \\
b_2 &= h[2] + a_1 h[1] + a_2 h[0] \\
b_3 &= h[3] + a_1 h[2] + a_2 h[1] + a_3 h[0] \\
&\vdots \\
b_{q-1} &= h[q-1] + a_1 h[q-2] + a_2 h[q-3] + \cdots + a_{p-1} h[q-p] + a_p h[q-p-1] \\
b_q &= h[q] + a_1 h[q-1] + a_2 h[q-2] + \cdots + a_{p-1} h[q-p+1] + a_p h[q-p] \\
0 &= h[q+1] + a_1 h[q] + a_2 h[q-1] + \cdots + a_{p-1} h[q-p+2] + a_p h[q-p+1] \\
&\vdots \\
0 &= h[q+p-1] + a_1 h[q+p-2] + a_2 h[q+p-3] + \cdots + a_{p-1} h[q] + a_p h[q-1] \\
0 &= h[q+p] + a_1 h[q+p-1] + a_2 h[q+p-2] + \cdots + a_{p-1} h[q+1] + a_p h[q] \,.
\end{aligned}$$

We then enforce that our systems impulse response matches the given signal or $h[n] = x[n]$ and then use the above to determine the coefficients a_k and b_k. For this problem we take the very short signal $x = \begin{bmatrix} 1 & 1.5 & 0.75 \end{bmatrix}$ and use Pade's method to determine the coefficients a_k and b_k.

Part (a): When $q = 0$ and $p = 2$ using Equation 7 with $h[n] = x[n]$ we get the following $p + q + 1 = 3$ equations

$$\begin{aligned}
x[0] + a_1 x[-1] + a_2 x[-2] &= b_0 \\
x[1] + a_1 x[0] + a_2 x[-1] &= 0 \\
x[2] + a_1 x[1] + a_2 x[0] &= 0 \,.
\end{aligned}$$

Since $x[n]$ is zero for negative arguments, as a matrix equation this becomes

$$\begin{bmatrix} x[0] & 0 & 0 \\ x[1] & x[0] & 0 \\ x[2] & x[1] & x[0] \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} b_0 \\ 0 \\ 0 \end{bmatrix} .$$

Thus $b_0 = 1$ and the system for the vector $\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$ becomes

$$\begin{bmatrix} 1 & 0 \\ 1.5 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = - \begin{bmatrix} 1.5 \\ 0.75 \end{bmatrix} .$$

We thus find that $a_1 = -1.5$ and $a_2 = 1.5$ and our model for $x[n]$ is

$$\hat{x}[n] = -\sum_{k=1}^{2} a_k x[n-k] = 1.5x[n-1] - 1.5x[n-2] \,,$$

for $n \geq 2$.

Part (b): When $p = 0$ and $q = 2$ using Equation 7 with $h[n] = x[n]$ we get the following $p + q + 1 = 3$ equations

$$\begin{bmatrix} x[0] \\ x[1] \\ x[2] \end{bmatrix} [1] = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix},$$

Thus $b_0 = 1$, $b_1 = 1.5$ and $b_2 = 0.75$.

Part (c): For $p = 1$ and $q = 1$ using Equation 7 with $h[n] = x[n]$ gives

$$\begin{aligned} x[0] + a_1 x[-1] &= b_0 \\ x[1] + a_1 x[0] &= b_1 \\ x[2] + a_1 x[1] &= 0\,. \end{aligned}$$

As a matrix system this is

$$\begin{bmatrix} x[0] & 0 \\ x[1] & x[0] \\ x[2] & x[1] \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \\ 0 \end{bmatrix}.$$

Solving the last equation or $x[2] + x[1]a_1 = 0$ we have $0.75 + 1.5a_1 = 0$ so $a_1 = -\frac{1}{2}$. With this the coefficients b_0 and b_1 are given by the first two equations above or

$$\begin{bmatrix} b_0 \\ b_1 \end{bmatrix} = \begin{bmatrix} x[0] \\ x[1] + a_1 x[0] \end{bmatrix} = \begin{bmatrix} 1 \\ 1.5 - 0.5(1.5) \end{bmatrix} = \begin{bmatrix} 1.5 \\ 0.75 \end{bmatrix}.$$

Problem 4.7 (using Prony's method)

Prony's method is also used to to model our signal $x[n]$ as the output of an LSI system. As discussed in the book we first solve for the autoregressive coefficients a and then use these to compute the moving average coefficients b. As discussed in the text to solve for the AR(1) part of the system we need to consider the system

$$\tilde{x} + \tilde{X}_2 \tilde{a} = 0\,,$$

or $\tilde{X}_2 \tilde{a} = -\tilde{x}$. In component form this matrix equation is

$$\begin{bmatrix} x[q] \\ x[q+1] \\ x[q+2] \\ \vdots \end{bmatrix} \tilde{a}_1 = - \begin{bmatrix} x[q+1] \\ x[q+2] \\ x[q+3] \\ \vdots \end{bmatrix}.$$

The dots indicate that we would consider this system over the entire length of $x[n]$. The least squares solution for $\tilde{a}_1$ from this system is obtained by multiplying by the transpose of the leading coefficient matrix to get

$$\left(\sum_{k=q}^{\infty} x[k]^2 \right) \tilde{a}_1 = - \sum_{k=q}^{\infty} x[k] x[k+1]\,,$$

so solving for $\tilde{a}_1$ we get

$$\tilde{a}_1 = -\frac{\sum_{k=q}^{\infty} x[k]x[k+1]}{\sum_{k=q}^{\infty} x[k]^2}\,.$$

If we take $q = 1$ and for this system we get

$$\tilde{a}_1 = -\frac{\sum_{k=1}^{\infty} x[k]x[k+1]}{\sum_{k=1}^{\infty} x[k]^2} = -\frac{x[1]x[2] + x[2]x[3] + \cdots + x[19]x[20]}{20} = -\frac{19}{20}\,.$$

Then with this value for a_1 we can find b_0 and b_1. They are given by

$$\begin{bmatrix} b_0 \\ b_1 \end{bmatrix} = X_1 a = \begin{bmatrix} x[0] & 0 \\ x[1] & x[0] \end{bmatrix} \begin{bmatrix} 1 \\ \tilde{a}_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 + \tilde{a}_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{1}{20} \end{bmatrix}\,.$$

Problem 8 (finding all-pole models)

Lets use the all pole Prony modeling method with when our signal is $x[n] = \delta[n] - \delta[n-1]$. To do this this we first need to evaluate the needed autocorrelation functions $r_x(k,m)$. Since we know the signal $x[n]$ exactly we can compute these explicitly. We find

$$\begin{aligned}
r_x(k,m) &= \sum_{n=0}^{\infty} x[n-m]x^*[n-k] \\
&= \sum_{n=0}^{\infty} (\delta_[n-m] - \delta[n-m-1])(\delta[n-k] - \delta[n-k-1]) \\
&= \sum_{n=0}^{\infty} (\delta[n-m]\delta[n-k] - \delta[n-m]\delta[n-k-1]) \\
&\quad - \sum_{n=0}^{\infty} (\delta[n-m-1]\delta[n-k] - \delta[n-m-1]\delta[n-k-1]) \\
&= \delta[m-k] - \delta[m-k-1] - \delta[k-m-1] + \delta[m-k] \\
&= 2\delta[m-k] - \delta[m-k-1] - \delta[m-k+1]\,.
\end{aligned}$$

Note that $r_x(k,m)$ is really a function of the difference $m-k$. Using the expression for $r_x(k,m)$ to compute a we need to solve

$$\sum_{m=1}^{p} a_m r_x(k,m) = -r_x(k,0) \quad \text{for} \quad k = 1, 2, \ldots, p-1, p\,. \tag{8}$$

This gives the system

$$\begin{bmatrix}
r_x(0) & r_x^*(1) & r_x^*(2) & \cdots & r_x^*(p-2) & r_x^*(p-1) \\
r_x(1) & r_x(0) & r_x^*(1) & \cdots & r_x^*(p-3) & r_x^*(p-2) \\
r_x(2) & r_x(1) & r_x(0) & \cdots & r_x^*(p-4) & r_x^*(p-3) \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
r_x(p-2) & r_x(p-3) & r_x(p-4) & \cdots & r_x(0) & r_x^*(1) \\
r_x(p-1) & r_x(p-2) & r_x(p-3) & \cdots & r_x^*(1) & r_x(0)
\end{bmatrix}
\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{p-1} \\ a_p \end{bmatrix}
= -\begin{bmatrix} r_x(1) \\ r_x(2) \\ r_x(3) \\ \vdots \\ r_x(p-1) \\ r_x(p) \end{bmatrix}$$

Part (a): In the case where $p = 1$ we only have one coefficient a_1 so using the first row in the above matrix system we get

$$r_x(0)a_1 = -r_x(1)\,.$$

From the expression derived $r_x(k,m) = r_x(k-m)$ we see that $r_x(0) = 2$ and $r_x(1) = -1$ thus $a_1 = \frac{1}{2}$. To evaluate b_0 we use

$$\{\varepsilon_p\}_{\min} = r_x(0) + \sum_{k=1}^{p} a_k r^*(k) = 2 + a_1 r^*(1) = 2 + \frac{1}{2}(-1) = \frac{3}{2}\,.$$

Then $b_0 = \sqrt{\{\varepsilon_p\}_{\min}} = \sqrt{\frac{3}{2}}$.

Part (b): For the second order all-pole model we have $p = 2$ and we need to solve $\sum_{m=1}^{p} a_m r_x(k-m) = -r_x(k)$ for $k = 1, 2$. These two equations are

$$\begin{aligned} a_1 r_x(0) + a_2 r_x(-1) &= -r_x(1) \\ a_1 r_x(1) + a_2 r_x(0) &= -r_x(2)\,. \end{aligned}$$

or when we put in what we know about r_x this system is

$$\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}\,.$$

Solving we find $a_1 = \frac{2}{3}$ and $a_2 = \frac{1}{3}$ for the two AR coefficients.

Problem 4.9 (using the autocorrelation method)

For this problem we have a finite signal of length $N = 20$. That is the data

$$x[0], x[1], \cdots, x[N-2], x[N-1]\,.$$

We again use Equation 8 to solve for the elements of the vector a.

Part (a): For the all-pole *autocorrelation* method note we have $r_x(k,m) = r_x(k-m)$ so that when we consider the normal Equations 8 we have

$$\sum_{m=1}^{p} a_m r_x(k-m) = -r_x(k) \quad \text{for} \quad k = 1, 2, \cdots p\,.$$

We evaluate r_x using

$$r_x(k) = \sum_{n=k}^{N} x[n]x^*[n-k]\,,$$

for $k \geq 0$. Note the lower limit on the sum above is k. For the given signal $x[n]$ we find

$$\begin{aligned} r_x(1) &= \sum_{n=1}^{N} x[n]x^*[n-1] \\ &= x[1]x^*[0] + x[2]x^*[1] + x[3]x^*[2] + \cdots + x[N-2]x^*[N-3] + x[N-1]x^*[N-2] \\ &= -1 + -1 + -1 + \cdots + -1 = -(N-1)\,. \end{aligned}$$

I'm assuming that the last element of x is $x[N-1] = -1$. For $r_x(2)$ we find

$$\begin{aligned} r_x(2) &= \sum_{n=2}^{N} x[n]x^*[n-2] \\ &= x[2]x^*[0] + x[3]x^*[1] + x[4]x^*[2] + \cdots + x[N-2]x^*[N-4] + x[N-1]x^*[N-3] \\ &= 1 + 1 + 1 + \cdots + 1 = N-2\,. \end{aligned}$$

For $r_x(3)$ we find

$$\begin{aligned} r_x(3) &= \sum_{n=3}^{N} x[n]x^*[n-3] \\ &= x[3]x^*[0] + x[4]x^*[1] + \cdots + x[N-2]x^*[N-5] + x[N-1]x^*[N-4] \\ &= -1 + -1 + -1 + \cdots + -1 = -(N-3)\,. \end{aligned}$$

In general it looks like the pattern for $r_x(k)$ is

$$r_x(k) = (-1)^k(N-k) \quad \text{for} \quad k \geq 0\,.$$

Using these values we want to solve for a_1 and a_2 and

$$\begin{aligned} a_1 r_x(0) + a_2 r_x(-1) &= -r_x(1) \\ a_1 r_x(1) + a_2 r_x(0) &= -r_x(2)\,, \end{aligned}$$

or using the values for $r_x(k)$ we have

$$\begin{aligned} Na_1 - (N-1)a_2 &= +(N-1) \\ -(N-1)a_1 + Na_2 &= -(N-2)\,. \end{aligned}$$

When we solve these for a_1 and a_2 we get $a_1 = \frac{2(N-1)}{2N-1}$ and $a_2 = \frac{1}{2N-1}$ for the two AR coefficients. We could then take $N = 20$ to evaluate them numerically if desired.

Problem 4.10 (using the modified Yule-Walker equations)

The modified Yule-Walker equations follow from

$$r_x(k) + \sum_{m=1}^{p} a_m r_x(k-m) = \begin{cases} \sigma_v^2 c(k) & 0 \leq k \leq q \\ 0 & k > q \end{cases}\,, \tag{9}$$

where $c(k)$ is given by

$$c(k) = \sum_{m=k}^{q} b_m h^*[m-k] = \sum_{m=0}^{q-k} b_{m+k} h^*[m]\,. \tag{10}$$

In matrix form the left-hand-side of these equations looks like

$$\left[\begin{array}{c|ccccc} r_x(0) & r_x(-1) & r_x(-2) & \cdots & r_x(-(p-1)) & r_x(-p) \\ r_x(1) & r_x(0) & r_x(-1) & \cdots & r_x(-(p-2)) & r_x(-(p-1)) \\ \vdots & & & & & \vdots \\ r_x(q) & r_x(q-1) & r_x(q-2) & \cdots & r_x(q-p+1) & r_x(q-p) \\ \hline r_x(q+1) & r_x(q) & r_x(q-1) & \cdots & r_x(q-p+2) & r_x(q-p+1) \\ \vdots & & & & & \vdots \\ r_x(q+p-1) & r_x(q+p-2) & r_x(q+p-3) & \cdots & r_x(q) & r_x(q-1) \\ r_x(q+p) & r_x(q+p-1) & r_x(q+p-2) & \cdots & r_x(q+1) & r_x(q) \end{array}\right] \left[\begin{array}{c} 1 \\ a_1 \\ a_2 \\ \vdots \\ a_{p-1} \\ a_p \end{array}\right],$$

where I have partitioned the matrix to show the MA part (the first $q+1$ rows) and the AR part (the last rows). The modified Yule-Walker equations have a right-hand-side given by the vector

$$\sigma_v^2 \left[\begin{array}{c} c(0) \\ c(1) \\ \vdots \\ c(q-1) \\ c(q) \\ 0 \\ \vdots \\ 0 \end{array}\right].$$

When we extract out the AR(p) bottom part we need to solve for a in the matrix

$$\left[\begin{array}{cccc} r_x(q) & r_x(q-1) & \cdots & r_x(q-p+2) \; r_x(q-p+1) \\ r_x(q+1) & r_x(q) & \cdots & r_x(q-p+3) \; r_x(q-p+2) \\ \vdots & & & \vdots \\ r_x(q+p-2) & r_x(q+p-3) & \cdots & r_x(q) \quad r_x(q-1) \\ r_x(q+p-1) & r_x(q+p-2) & \cdots & r_x(q+1) \quad r_x(q) \end{array}\right] \left[\begin{array}{c} a_1 \\ a_2 \\ \vdots \\ a_{p-1} \\ a_p \end{array}\right] = - \left[\begin{array}{c} r_x(q+1) \\ r_x(q+2) \\ \vdots \\ r_x(q+p-1) \\ r_x(q+p) \end{array}\right]. \tag{11}$$

Once the coefficients in a_k are solved for we can evaluate b_k with $R_x a = c$. When $p = q = 1$ as we are asked to consider here when we consider the first three Equations in 9 we get the system

$$\left[\begin{array}{cc} r_x(0) & r_x(-1) \\ r_x(1) & r_x(0) \\ r_x(2) & r_x(1) \end{array}\right] \left[\begin{array}{c} 1 \\ a_1 \end{array}\right] = \sigma_v^2 \left[\begin{array}{c} c(0) \\ c(1) \\ 0 \end{array}\right].$$

Given what we are told about r_x the third equation means that $a_1 = -\frac{1}{2}$. Taking $\sigma_v^2 = 1$ and using what we just found for a_1 we find $c(0)$ and $c(1)$ given by

$$\begin{aligned} c(0) &= r_x(0) + r_x(-1)a_1 = 26 + 7(-1/2) = 22.5 \\ c(1) &= r_x(1) + r_x(0)a_1 = 7 + 26(-1/2) = -6\,. \end{aligned}$$

Then using Equation 10 we see that

$$\begin{aligned} c(0) &= b_0 h^*[0] + b_1 h^*[1] \\ c(1) &= b_1 h^*[1]\,. \end{aligned}$$

Given values for $h^*[0]$ and $h^*[1]$ we can use the above to solve for b_0 and b_1. **Warning:** I'm not sure what values to take for $h^*[0]$ and $h^*[1]$. If anyone knows what I should use for these please email me.

Problem 4.11 (a third-order all-pole model)

The AR coefficients are given by solving the Yule-Walker Equations 11 (when we take $q = 0$) and recall that $r_x(-n) = r_x^*(n) = r_x(n)$. For a AR(3) model this is the system

$$\begin{bmatrix} r_x(0) & r_x(1) & r_x(2) \\ r_x(1) & r_x(0) & r_x(1) \\ r_x(2) & r_x(1) & r_x(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = - \begin{bmatrix} r_x(1) \\ r_x(2) \\ r_x(3) \end{bmatrix} .$$

We can use the numbers given in this problem to solve for the values of a_k.

Problem 4.12 (predicting autocorrelations)

Given the values of $r_x(k)$ for $0 \leq k \leq 3$ and the coefficients a_k we can use Equation 9 to compute $r_x(k)$ for larger values of k. For example to compute $r_x(4)$ we would use

$$r_x(4) = -\sum_{m=1}^{3} a_m r_x(k-m) = -(a_1 r_x(3) + a_2 r_x(2) + a_3 r_x(1)) .$$

Everything on the right-hand-side is known.

Optimum Wiener Filter

Notes on the text

Notes on the derivation of the ideal continuous time Wiener filter

We start with an expansion of the objective function we seek to minimize, ξ, as

$$\begin{aligned}
\xi &= \int_{-\infty}^{\infty} \left| \frac{X(f)}{G(f)}\Phi(f) - \frac{S(f)}{G(f)} \right|^2 df \\
&= \int_{-\infty}^{\infty} \left| \frac{(S(f)+V(f))}{G(f)}\Phi(f) - \frac{S(f)}{G(f)} \right|^2 df \\
&= \int_{-\infty}^{\infty} \frac{1}{|G(f)|^2} \left\{ |V(f)\Phi(f) - (1-\Phi(f))S(f)|^2 \right\} df \\
&= \int_{-\infty}^{\infty} \frac{1}{|G(f)|^2} \left\{ |V(f)\Phi(f)|^2 - 2V(f)\Phi(f)(1-\Phi(f))^* S^*(f) + |1-\Phi(f)|^2 |S(f)|^2 \right\} df
\end{aligned}$$

If we assume (as does the book) that the middle term integrates to zero we get

$$\xi = \int_{-\infty}^{\infty} |G(f)|^{-2} \left\{ |S(f)|^2 |1-\Phi(f)|^2 + |V(f)|^2 |\Phi(f)|^2 \right\} df$$

Note there seems to be a type in the expression presented in the book. Taking the derivative of ξ with respect to the "object" $\Phi(f)$ and setting the result equal to zero gives

$$2|S(f)|^2(1-\Phi(f))(-1) + 2|V(f)|^2\Phi(f) = 0\,.$$

When we solve for $\Phi(f)$ in the above expression we get

$$\Phi(f) = \frac{|S(f)|^2}{|S(f)|^2 + |V(f)|^2}\,, \tag{12}$$

for the optimal Wiener solution.

Problem Solutions

Problem 5.1 (and example of FIR filtering)

For this problem we follow the section on general noise FIR Wiener filtering with $r_s(k) = \alpha^{|k|}$ and $q = 2$. This means that we need to solve $(R_s + R_v)w = r_s$ for the weight vector w, where $R_s = \begin{bmatrix} r_s(0) & r_s^*(1) \\ r_s(1) & r_s(0) \end{bmatrix}$, $R_v = \sigma_v^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and the right-hand-side vector is $r_s = \begin{bmatrix} r_s(0) \\ r_s(1) \end{bmatrix}$. We take $q = 2$ since we want a first order filter. For the signal generated with an

autocorrelation function of $r_s(k) = \alpha^{|k|}$ we have that $R_s = \begin{bmatrix} 1 & \alpha \\ \alpha & 1 \end{bmatrix}$, and $r_s = \begin{bmatrix} 1 \\ \alpha \end{bmatrix}$ so the solution to for the two weights $\begin{bmatrix} w[0] \\ w[1] \end{bmatrix}$ is given by

$$\begin{bmatrix} w[0] \\ w[1] \end{bmatrix} = \left(\begin{bmatrix} 1 & \alpha \\ \alpha & 1 \end{bmatrix} + \sigma_v^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right)^{-1} \begin{bmatrix} 1 \\ \alpha \end{bmatrix} = \begin{bmatrix} 1+\sigma_v^2 & \alpha \\ \alpha & 1+\sigma_v^2 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ \alpha \end{bmatrix}.$$

Thus performing the above inverse we find

$$\begin{aligned} \begin{bmatrix} w[0] \\ w[1] \end{bmatrix} &= \frac{1}{(1+\sigma_v^2)^2 - \alpha^2} \begin{bmatrix} 1+\sigma_v^2 & -\alpha \\ -\alpha & 1+\sigma_v^2 \end{bmatrix} \begin{bmatrix} 1 \\ \alpha \end{bmatrix} \\ &= \frac{1}{(1+\sigma_v^2) - \alpha^2} \begin{bmatrix} 1+\sigma_v^2 - \alpha^2 \\ -\alpha + (1+\sigma_v^2)\alpha \end{bmatrix}, \end{aligned} \tag{13}$$

for the two solutions. If $\alpha = 0.8$ and $\sigma_v^2 = 1$ we can use Equation 13 to get $w[0] = 0.40408$ and $w[1] = 0.2381$ so $W(z) = w[0] + w[1]z^{-1}$ is the optimal Wiener filter, the same as the expression given in the book.

Problem 5.3 (FIR Wiener prediction)

Following the section in the book entitled "FIR Wiener Linear Prediction" we need to compute $w[0]$ and $w[1]$ in the no noise case by solving $R_x w = r_{sx(1)} = r_{x(1)}$ or

$$\begin{bmatrix} r_x(0) & r_x^*(1) \\ r_x(1) & r_x(0) \end{bmatrix} \begin{bmatrix} w[0] \\ w[1] \end{bmatrix} = \begin{bmatrix} r_x(1) \\ r_x(2) \end{bmatrix}. \tag{14}$$

For the autocorrelation function given this becomes

$$\begin{bmatrix} 1 & \alpha \\ \alpha & 1 \end{bmatrix} \begin{bmatrix} w[0] \\ w[1] \end{bmatrix} = \begin{bmatrix} \alpha \\ \alpha^2 \end{bmatrix}.$$

Solving this we get

$$\begin{bmatrix} w[0] \\ w[1] \end{bmatrix} = \frac{1}{1-\alpha^2} \begin{bmatrix} 1 & -\alpha \\ -\alpha & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ \alpha^2 \end{bmatrix} = \frac{1}{1-\alpha^2} \begin{bmatrix} \alpha - \alpha^3 \\ -\alpha^2 + \alpha^2 \end{bmatrix} = \begin{bmatrix} \alpha \\ 0 \end{bmatrix}.$$

Thus the optimal first-order Wiener predictor is $\hat{x}[n+1] = \alpha x[n]$.

Problem 5.4-5.5 (FIR Wiener prediction with noise)

In this case the measurement of $x[n]$ is contaminated by zero-mean white noise as $y[n] = x[n] + \nu[n]$. here $y[n]$ is the measurement process. Then we want to estimate $x[n+1]$ using measurements of $y[n]$ as

$$\hat{x}[n+1] = \sum_{m=0}^{q-1} w[m] y[n-m]. \tag{15}$$

To do that we must solve the Wiener-Hopf equations or

$$\sum_{m=0}^{q-1} w[m] r_y(k-m) = r_{x(1)y}(k) \quad \text{for} \quad k = 0, 1, 2, \cdots, q-1 \,.$$

In this case given here we have

$$\begin{aligned} r_{x(1)y}(k) &= E\{x[n+1]y^*[n-k]\} = E\{x[n+1](x^*[n-k] + \nu[n-k])\} \\ &= E\{x[n+1]x^*[n-k]\} = r_{x(1)} \\ r_y(k) &= E\{y[n]y^*[n-k]\} = r_x(k) + r_v(k) \,. \end{aligned}$$

For the Markov autocorrelation signal given here this becomes the matrix equation $[R_x + R_v]w = r_{x(1)}$ or when we take $q = 2$ to get a first-order predictor we get

$$\begin{bmatrix} 1+\sigma_v^2 & \alpha \\ \alpha & 1+\sigma_v^2 \end{bmatrix} \begin{bmatrix} w[0] \\ w[1] \end{bmatrix} = \begin{bmatrix} \alpha \\ \alpha^2 \end{bmatrix} .$$

Solving for the vector w we get

$$\begin{aligned} \begin{bmatrix} w[0] \\ w[1] \end{bmatrix} &= \frac{1}{(1+\sigma_v^2)^2 - \alpha^2} \begin{bmatrix} 1+\sigma_v^2 & -\alpha \\ -\alpha & 1+\sigma_v^2 \end{bmatrix} \begin{bmatrix} \alpha \\ \alpha^2 \end{bmatrix} \\ &= \frac{1}{(1+\sigma_v^2)^2 - \alpha^2} \begin{bmatrix} (1+\sigma_v^2)\alpha - \alpha^3 \\ -\alpha^2 + (1+\sigma_v^2)\alpha^2 \end{bmatrix} = \frac{1}{(1+\sigma_v^2)^2 - \alpha^2} \begin{bmatrix} \alpha - \alpha^3 + \sigma_v^2\alpha \\ \sigma_v^2\alpha^2 \end{bmatrix} . \end{aligned}$$

Thus the optimum first order predictor in this case is given by

$$\hat{x}[n+1] = \left(\frac{\alpha - \alpha^3 + \sigma_v^2\alpha}{(1+\sigma_v^2)^2 - \alpha^2} \right) y[n] + \left(\frac{\sigma_v^2\alpha^2}{(1+\sigma_v^2)^2 - \alpha^2} \right) y[n-1] \,.$$

If we take $\sigma_v^2 \to 0$ in the above we get $\hat{x}[n+1] = \alpha y[n] = \alpha x[n]$, since $y[n] = x[n] + v[n] \to x[n]$ in this case. This is the same result seen earlier.

Problem 5.6 (FIR Wiener linear prediction of $x[n+3]$)

We want to estimate the signal $s[n]$ where $s[n] = x[n+3]$ using $x[n]$ and $x[n-1]$ as our measurements. From the section entitled "FIR Wiener linear prediction" we need solve $R_x w = r_{sx}(c) = r_x(c)$. With $c = 3$, this becomes

$$\begin{bmatrix} r_x(0) & r_x^*(1) \\ r_x(1) & r_x(0) \end{bmatrix} \begin{bmatrix} w[0] \\ w[1] \end{bmatrix} = \begin{bmatrix} r_x(3) \\ r_x(4) \end{bmatrix} . \tag{16}$$

With the expression for $r_x(k)$ given in this problem we have

$$\begin{bmatrix} 2 & 0.6364 \\ 0.6364 & 2 \end{bmatrix} \begin{bmatrix} w[0] \\ w[1] \end{bmatrix} = \begin{bmatrix} -0.5155 \\ -0.6561 \end{bmatrix} .$$

Solving this gives $w[0] = -0.17067$ and $w[1] = -0.27234$. The mean square error (MSE) corresponding to this solution is given by

$$\begin{aligned} \xi &= r_x(0) - r_{x(c)}^H w = r_x(0) - r_{x(3)}^H w \\ &= 2 - \begin{bmatrix} -0.5155 & -0.6561 \end{bmatrix} \begin{bmatrix} -0.17067 \\ -0.27234 \end{bmatrix} = 1.7324 \,. \end{aligned}$$

Problem 5.7 (FIR Wiener linear prediction of $x[n+c]$)

We repeat the previous problem for various values of $c = 1, 2, 3, 4, 5$ and look at the MSE for each. The only thing that changes in each case is the expression for the right-hand-side in Equation 16. See the Octave file `prob_5_7.m` where this is worked. When that script is executed we get the following for the MSE for each prediction lookahead c

```
  c =      1         2         3         4         5         6
MSE = 1.7747    1.8522    1.7324    1.7605    1.9030    1.9364
```

Optimum Kalman Filter

Notes on the text

Notes on the Kalman Filter Examples

We take for $\hat{x}[2]$ the following approximation based on measurements taken at the times $t = 1$ and $t = 2$

$$\hat{x}[2] \approx \begin{bmatrix} y_1[2] \\ \frac{1}{T}(y_1[2] - y_1[1]) \\ y_2[2] \\ \frac{1}{T}(y_2[2] - y_2[1]) \end{bmatrix} .$$

We want to evaluate $P(2|2)$. We first construct an approximation to $x[2] - \hat{x}[2]$ in that

$$x[2] - \hat{x}[2] \approx \begin{bmatrix} x_1[2] - y_1[2] \\ x_2[2] - \frac{1}{T}(y_1[2] - y_1[1]) \\ x_3[2] - y_2[2] \\ x_4[2] - \frac{1}{T}(y_2[2] - y_2[1]) \end{bmatrix} .$$

Using what we know from how the state maps to the measurements that

$$\begin{aligned} y_1[2] &= x_1[2] + v_1[2] \\ y_2[2] &= x_2[2] + v_2[2] \,, \end{aligned}$$

and how the state at the previous timestep maps to the state at the current timestep that

$$\begin{aligned} x_2[2] &= x_2[1] + u_1[1] \\ x_4[2] &= x_4[1] + u_2[1] \,, \end{aligned}$$

so that we get for $x[2] - \hat{x}[2]$

$$\begin{aligned} x[2] - \hat{x}[2] &= \begin{bmatrix} -v_1[2] \\ x_2[1] + u_1[1] - \frac{1}{T}(x_1[2] + v_1[2] - x_1[1] - v_1[1]) \\ -v_2[2] \\ x_4[1] + u_2[1] - \frac{1}{T}(x_2[2] + v_2[2] - x_2[1] - v_2[1]) \end{bmatrix} \\ &= \begin{bmatrix} -v_1[2] \\ u_1[1] - \frac{1}{T}(v_1[2] - v_1[1]) + x_2[1] - \frac{1}{T}(x_1[2] - x_1[1]) \\ -v_2[2] \\ u_2[1] - \frac{1}{T}(v_2[2] - v_2[1]) + x_4[1] - \frac{1}{T}(x_2[2] - x_2[1]) \end{bmatrix} . \end{aligned}$$

If we take

$$\begin{aligned} x_2[1] &\approx \frac{1}{T}(x_1[2] - x_1[1]) \quad \text{and} \\ x_4[1] &\approx \frac{1}{T}(x_2[2] - x_2[1]) \,, \end{aligned}$$

we get for $x[2] - \hat{x}[2]$ the following

$$\begin{bmatrix} -v_1[2] \\ u_1[1] - \frac{1}{T}(v_1[2] - v_1[1]) \\ -v_2[2] \\ u_2[1] - \frac{1}{T}(v_2[2] - v_2[1]) \end{bmatrix} .$$

The covariance we seek $P(2|2)$ is when we define $e \equiv x[2] - \hat{x}[2]$ is $P(2|2) = E[ee^T]$. This later matrix has components given by $P(2|2)_{ij} = E[e_i e_j]$. Thus when we compute several of these expectations we see that

$$\begin{aligned} P(2|2)_{11} &= E[e_1 e_1] = E\left[(-v_1[2])(-v_1[2])\right] = \sigma_\rho^2 \\ P(2|2)_{12} &= E[e_1 e_2] = E\left[(-v_1[2])(u_1[1] - \frac{1}{T}(v_1[2] - v_1[1]))\right] = \frac{\sigma_\rho^2}{T} \\ P(2|2)_{22} &= E[e_2 e_2] = E\left[(u_1[1] - \frac{1}{T}(v_1[2] - v_1[1]))(u_1[1] - \frac{1}{T}(v_1[2] - v_1[1]))\right] \\ &= \sigma_1^2 + 2\frac{\sigma_\rho^2}{T^2} . \end{aligned}$$

The remaining elements in $P(2|2)$ are computed in the same way.

Problem solutions

Problem 6.2 (Kalman filtering of a constant)

We begin with the estimation of a *constant* x from a series of uncorrelated corrupted noisy measurements. For this example we take our state $x[n]$ to be the estimate of our constant x and assume that there is no process dynamics. Thus the state equation is given by

$$x[n] = x[n-1] .$$

Thus in the notation of the book $Q_w(n) = 0$. If we assume that each measurement is a noised version of x we have that

$$y[n] = x[n] + v[n] .$$

with $v[n] \sim \mathcal{N}(0, \sigma_v^2)$ and $Q_v(n) = \sigma_v^2$. Then with this specification the Kalman equations for this problem we initialize our initial estimate of x and its uncertainty as $\hat{x}[0|0] = x_0$ and $P(0|0) = p_0$. Then for $n = 1, 2, \cdots$ we iterate

- Predict the state: $\hat{x}[n|n-1] = \hat{x}[n-1|n-1]$.
- Predict the uncertainty in the state: $P(n|n-1) = P(n-1|n-1)$.
- Compute the filter gain: $K(n) = \frac{P(n|n-1)}{P(n|n-1)+\sigma_v^2}$.

- Using $K(n)$ incorporate the measurement $y[n]$ to update the state estimate:

$$\hat{x}(n|n) = \hat{x}[n|n-1] + K(n)(y[n] - \hat{x}[n|n-1])$$

- Using $K(n)$ incorporate the measurement $y[n]$ to update the state uncertainty:

$$P(n|n) = (I - K(n))P(n|n-1)$$

Power Spectral Density Analysis

Problem solutions

Problem 7.1 (main approaches to spectral estimation)

The two main approaches to spectral estimation can be classified as the nonparametric and the parametric approach. In the nonparametric approach no model (or functional form) is assumed for the autocorrelation function $r_x(k)$ of the process $x[n]$ we are considering. It basically involves estimating the autocorrelation function and then taking its Fourier transform. The periodogram, the modified periodogram (windowing of the periodogram), Bartlett's (periodogram averaging), Welch's method, and the method of Blackman-Tukey are examples of the classical nonparametric approaches. In the nonclassical parametric approach a particular model is assumed for the signal $x[n]$. If the model is true this implies a particular form for the autocorrelation function. If the assumed model is correct these methods can generate better spectral estimates with fewer data points N. Typical models assume the process $x[n]$ is: autoregressive, moving average, autoregressive moving average or harmonic (complex exponential in random noise).

Problem 7.2 (the power spectrum of white noise)

The power spectrum of a white noise signal $x(t)$ of variance σ_x^2 is a *constant* that is

$$P_x(e^{j\theta}) = \sigma_x^2 \,.$$

Problem 7.3 (the minimum value of N to resolve two sinusoids)

Warning: I'm not entirely sure about this problem. See the comments at the end as to why. If anyone knows what I am doing incorrectly please let me know.

From the book the resolution for the nonparametric periodogram is given by

$$\Delta\theta = 0.89\frac{2\pi}{N} \,. \tag{17}$$

For us to be able to resolve these two different sinusoids using this method would then require a value of N such that

$$\Delta\theta = 0.89\frac{2\pi}{N} \ll 0.02\pi \,.$$

so

$$N \gg 89 \,.$$

I'm a bit worried about this results since I don't seem to use the white noise variance σ_x^2 or the sampling frequency F_s in this solution.

Problem 7.4 (the variance of the periodogram spectral estimator)

From the discussion in the book the variance of the periodogram spectral estimator is related to the true spectral density $P_x(e^{j\theta})$ as

$$\text{Var}\left\{\hat{P}_{\text{per}-x}(e^{j\theta})\right\} \approx P_x^2(e^{j\theta})\,. \tag{18}$$

If our signal $x(t)$ is white noise with variance σ_x^2 then we have $P_x(e^{j\theta}) = \sigma_x^2$ a constant. Thus $\text{Var}\left\{\hat{P}_{\text{per}-x}(e^{j\theta})\right\}$ above does not go to zero as the sample record length N increases.

Problem 7.5 (Bartlett's resolution and the number of data sections)

From the textbook, the resolution $\Delta\theta$ for Bartlett's power spectrum estimator is

$$\Delta\theta = 0.89K\frac{2\pi}{N} \tag{19}$$

and the variance of the estimator is

$$\text{Var}\left\{\hat{P}_B(e^{j\theta})\right\} \approx \frac{1}{K}P_x^2(e^{j\theta})\,. \tag{20}$$

For a fixed N as K, the number of nonoverlapping data sections increases the number of points in each interval L, must decrease. From Equation 19 we see that $\Delta\theta$ increases (gets worse), while from Equation 20 the variance gets better.

Problem 7.6 (a figure of merit)

The book defined a "figure of merit" or μ that is to capture the trade offs between reducing the variance vs. the associated cost in loss of resolution $\Delta\theta$. The figure of merit introduced is defined as the normalized variance ν, times the resolution $\Delta\theta$. For the four nonparametric methods considered in the book: periodogram, Bartlett, Welsh, and Blackman-Tukey, all the figures of merit are of the form $C\left(\frac{2\pi}{N}\right)$ for various numerical values for C. From this we see that the performance of all these nonparametric methods are directly limited by the length of the data sequence N.

Problem 7.7 (MV power spectrum estimation of white noise with variance σ_x^2)

Recall that for the the minimum variance (MV) method the spectral density estimate is given by

$$\hat{P}_{MV}(e^{j\theta}) = \frac{q+1}{\mathbf{e}^H R_x^{-1}\mathbf{e}}\,, \tag{21}$$

where the vector $\mathbf{e}$ is given by

$$\mathbf{e} = \begin{bmatrix} 1 \\ e^{j\theta} \\ e^{j2\theta} \\ \vdots \\ e^{j(q-1)\theta} \\ e^{jq\theta} \end{bmatrix} .$$

Also recall that for a white noise process with a variance of σ_x^2 we have an autocorrelation matrix of $R_x = \sigma_x^2 I$. Thus we have

$$\mathbf{e}^H R_x^{-1} \mathbf{e} = \frac{1}{\sigma_x^2} \mathbf{e}^H \mathbf{e} = \frac{q+1}{\sigma_x^2} ,$$

where we have used $\mathbf{e}_i^H \mathbf{e}_i = \sum_{i=0}^{q} 1^2 = q+1$. Using this in Equation 21 we find

$$\hat{P}_{MV}(e^{j\theta}) = \sigma_x^2 ,$$

as it should.

Problem 7.8 (MV spectral estimate of a harmonic model)

Note that the given signal is a specification of the **harmonic signal** given by

$$x[n] = \sum_{i=1}^{p} |A_i| e^{j\phi_i} e^{jn\theta_i} + w[n] , \tag{22}$$

in that the given signal for this problem is just one term in the above sum. In the book the autocorrelation matrix R_x for a harmonic model is computed and is given by

$$R_x = \sum_{i=1}^{p} P_i \mathbf{e}_i \mathbf{e}_i^H + \sigma_w^2 I , \tag{23}$$

where $P_i = |A_i|^2$ and $\mathbf{e}_i$ is given by

$$\mathbf{e}_i = \begin{bmatrix} 1 \\ e^{j\theta_i} \\ e^{j2\theta_i} \\ \vdots \\ e^{j(q-1)\theta_i} \\ e^{jq\theta_i} \end{bmatrix} .$$

We will compute the power spectral density using

$$\hat{P}_{\text{MV}}(e^{j\theta}) = \frac{q+1}{\mathbf{e}^H R_x^{-1} \mathbf{e}} .$$

For this problem we have that R_x is given by

$$R_x = \sigma_w^2 I + P_i \mathbf{e}_i \mathbf{e}_i^H = \sigma_w^2 \left(I + \left(\frac{P_i}{\sigma_w^2} \right) \mathbf{e}_i \mathbf{e}_i^H \right) .$$

Lets define the signal to noise ratio $\frac{P_i}{\sigma_w^2}$ as α then we get

$$R_x^{-1} = \frac{1}{\sigma_w^2} \left(I + \alpha \mathbf{e}_i \mathbf{e}_i^H \right)^{-1} .$$

Using the Sherman-Morrison formula we get

$$R_x^{-1} = \frac{1}{\sigma_w^2} \left(I - \frac{\alpha \mathbf{e}_i \mathbf{e}_i^H}{1 + \alpha \mathbf{e}_i^H \mathbf{e}_i} \right)$$

Note that $\mathbf{e}_i^H \mathbf{e}_i = q + 1$ and we get

$$R_x^{-1} = \frac{1}{\sigma_w^2} \left(I - \frac{\alpha}{1 + \alpha(q+1)} \mathbf{e}_i \mathbf{e}_i^H \right) .$$

Now consider the expression $\mathbf{e}^H R_x^{-1} \mathbf{e}$ need to evaluate $\hat{P}_{\text{MV}}(e^{j\theta})$ and we get

$$\mathbf{e}^H R_x^{-1} \mathbf{e} = \frac{1}{\sigma_w^2}(q+1) - \frac{\alpha}{\sigma_w^2 (1 + \alpha(q+1))} \mathbf{e}^H \mathbf{e}_i \mathbf{e}_i^H \mathbf{e} .$$

Note that the product in the second term is

$$(\mathbf{e}^H \mathbf{e}_i)(\mathbf{e}_i^H \mathbf{e}) = |\mathbf{e}^H \mathbf{e}_i|^2 = \left| \sum_{k=0}^{q} e^{jk(\theta_i - \theta)} \right|^2 .$$

Using this we get for $\hat{P}_{\text{MV}}(e^{j\theta})$ the following expression

$$\hat{P}_{\text{MV}}(e^{j\theta)}) = \frac{q+1}{\frac{q+1}{\sigma_w^2} + \frac{\alpha}{\sigma_w^2 (1+\alpha(q+1))} \left| \sum_{k=0}^{q} e^{jk(\theta_i - \theta)} \right|} .$$

This expression will have a single peak when $\theta = \theta_i$.

Adaptive Finite Impulse Response Filters

Notes on the Text

Notes on the time constant τ_m in the LMS algorithm

Pick τ_m such that when $k = \tau_m$ we have the initial error in the mth component reduced by a factor of e. That is we are looking for $k = \tau_m$ such that

$$v_m[k] \approx \frac{1}{e} v_m[0] \,.$$

We can write this as

$$(1 - \mu\lambda_m)^{\tau_m} \approx \frac{1}{e} = e^{-1} \,,$$

so

$$\tau_m \ln(1 - \mu\lambda_m) = -1 \,,$$

or taking logarithms we get

$$\tau_m = -\frac{1}{\ln(1 - \mu\lambda_m)} \,. \tag{24}$$

If $\mu \ll 1$ (or very small LMS step sizes) then

$$\ln(1 - \mu\lambda_m) \approx -\mu\lambda_m + O(\mu^2\lambda_m^2) \,,$$

so in this case we have

$$\tau_m = -\frac{1}{(-\mu\lambda_m)} = \frac{1}{\mu\lambda_m} \,. \tag{25}$$

For all possible values of λ_m the one where $\lambda_m = \lambda_{\min}$ is the most restrictive in that it results in the largest value of τ_m. This number represents the limiting number of iterations for the LMS algorithm.

Notes on the recursive least squares estimation

In this and the following section the definition of $\mathbf{x}$ is

$$\mathbf{x}[k] = \begin{bmatrix} x[k] & x[k-1] & x[k-2] & \cdots & x[k-p+1] \end{bmatrix}^T \,. \tag{26}$$

The definition of the vector $\mathbf{w}$ (with its p components) is

$$\mathbf{w}[n] = \begin{bmatrix} w_0[n] & w_1[n] & w_2[n] & \cdots & w_{p-1}[n] \end{bmatrix}^T \,. \tag{27}$$

The definition of ξ, or the objective function we will try to minimize is

$$\xi(n) = \sum_{k=0}^{n} \beta(n,k) |d[k] - \mathbf{w}^T[n]\mathbf{x}[k]|^2 \,. \tag{28}$$

When we specialize the "forgetting factor" $\beta(n,k)$ to be $\beta(n,k) = \lambda^{n-k}$ we get

$$\xi(n) = \sum_{k=0}^{n} \lambda^{n-k} |d[k] - \mathbf{w}^T[n]\mathbf{x}[k]|^2 \,, \tag{29}$$

Note that the term in the summation expresses how well the weight vector $\mathbf{w}$ times the input vector $\mathbf{x}$ matches the desired signal $d[n]$ over all of the times in the past i.e. for $0 \leq k \leq n$ and not just the most recent input data $\mathbf{x}[n]$ and $d[n]$.

Notes on the exponentially weighted recursive least squares algorithm

Warning: From what I have been able to duplicate, the book seems to make some errors in complex conjugation in the formulas it presents. I have tried to make sure that this derivations is correct but an error could have crept in. If anyone sees anything wrong with my arguments below (or finds them to be correct) please let me know.

The optimal solution to our exponentially weighted recursive least squares formulation was derived in Equation 45 and requires us to solve (for each n as new samples arrive)

$$\Phi(n)\hat{w}[n] = \theta(n) \,. \tag{30}$$

Here $\Phi(n)$ and $\theta(n)$ are defined in the book. This involves computing the inverse of $\Phi(n)$ for each n. To do this inverse we will use a special result from matrix algebra. In cases where our matrix A can be written as "almost an inverse", its inverse can sometimes be more easily computed. One case of this form is **Woodbury's identity**, which states that if A can be written as

$$\begin{aligned} A &= B^{-1} + CD^{-1}C^H \quad \text{then} \\ A^{-1} &= B - BC[D + C^H BC]^{-1} C^H B \,. \end{aligned} \tag{31}$$

Since we have the the recursive definition of $\Phi(n)$ of

$$\Phi(n) = \lambda\Phi(n-1) + \mathbf{x}^*[n]\mathbf{x}^T[n] \,,$$

we can use Equation 31 if we let $A = \Phi(n)$, $B^{-1} = \lambda\Phi(n-1)$, $C = \mathbf{x}^*[n]$ and $D = 1$. In that case we find

$$\Phi(n)^{-1} = \lambda^{-1}\Phi(n-1)^{-1} - \frac{\lambda^{-1}\Phi(n-1)^{-1}\mathbf{x}^*[n]\mathbf{x}[n]^T\lambda^{-1}\Phi(n-1)^{-1}}{(1 + \lambda^{-1}\mathbf{x}^T[n]\Phi(n-1)^{-1}\mathbf{x}^*[n])} \,. \tag{32}$$

Lets define $P(n) \equiv \Phi(n)^{-1}$ and the matrix $k(n)$ to be

$$k(n) \equiv \frac{\lambda^{-1}P(n-1)\mathbf{x}^*[n]}{1 + \lambda^{-1}\mathbf{x}^T[n]P(n-1)\mathbf{x}^*[n]} = \frac{\lambda^{-1}\Phi(n-1)^{-1}\mathbf{x}^*[n]}{1 + \lambda^{-1}\mathbf{x}^T[n]\Phi(n-1)^{-1}\mathbf{x}^*[n]} \,, \tag{33}$$

Then Equation 32 in terms of $P(n)$ then becomes

$$P(n) = \lambda^{-1}P(n-1) - \lambda^{-1}k(n)\mathbf{x}^T[n]P(n-1) \,. \tag{34}$$

Now multiply both sides of this expression by $\mathbf{x}^*[n]$ on the right to get

$$P(n)\mathbf{x}^*[n] = \lambda^{-1}P(n-1)\mathbf{x}^*[n] - \lambda^{-1}k(n)\mathbf{x}^T[n]P(n-1)\mathbf{x}^*[n]\,. \tag{35}$$

From the first expression in the definition of $k(n)$ given by Equation 33 when we multiply by the denominator we get

$$k(n)(1+\lambda^{-1}\mathbf{x}^T[n]P(n-1)\mathbf{x}^*[n]) = \lambda^{-1}P(n-1)\mathbf{x}^*[n]\,,$$

so writing the above as

$$\lambda^{-1}k(n)\mathbf{x}^T[n]P(n-1)\mathbf{x}^*[n] = \lambda^{-1}P(n-1)\mathbf{x}^*[n] - k(n)\,, \tag{36}$$

and then putting this into the right-hand-side of Equation 35 we get

$$P(n)\mathbf{x}^*[n] = k(n) \quad \text{or} \quad k(n) = \Phi(n)^{-1}\mathbf{x}^*[n]\,. \tag{37}$$

We will use the expression just developed to develop a recursive update algorithm for $\hat{w}[n]$. To begin, in Equation 30 or $\hat{w}[n] = P(n)\theta(n)$ first recursively replace $\theta(n)$ to get

$$\begin{aligned}\hat{w}[n] &= P(n)\theta(n) = P(n)(\lambda\theta(n-1) + d[n]\mathbf{x}^*[n]) \\ &= \lambda P(n)\theta(n-1) + P(n)d[n]\mathbf{x}^*[n]\,.\end{aligned}$$

Now in the first term above, we recursively expand the expression for $P(n)$ using Equation 34 and find $\hat{w}[n]$ given by

$$\begin{aligned}\hat{w}[n] &= \lambda(\lambda^{-1}P(n-1) - \lambda^{-1}k(n)\mathbf{x}^T[n]P(n-1))\theta(n-1) + P(n)d[n]\mathbf{x}^*[n] \\ &= P(n-1)\theta(n-1) - k(n)\mathbf{x}^T[n]P(n-1)\theta(n-1) + d[n]P(n)\mathbf{x}^*[n]\,.\end{aligned}$$

Replace $P(n)$ with $\Phi(n)^{-1}$ to get

$$\hat{w}[n] = \Phi(n-1)^{-1}\theta(n-1) - k(n)\mathbf{x}^T[n]\Phi(n-1)^{-1}\theta(n-1) + d[n]P(n)\mathbf{x}^*[n]\,.$$

Note that $\Phi(n-1)^{-1}\theta(n-1) = \hat{w}[n-1]$ and we get

$$\hat{w}[n] = \hat{w}[n-1] - k(n)\mathbf{x}^T[n]\hat{w}[n-1] + d[n]P(n)\mathbf{x}^*[n]\,.$$

From Equation 37 we can replace $P(n)\mathbf{x}^*[n]$ with $k(n)$ and get

$$\hat{w}[n] = \hat{w}[n-1] - k(n)\mathbf{x}^T[n]\hat{w}[n-1] + d[n]k(n)\,.$$

In that expression consider $k(n)\mathbf{x}^T[n]\hat{w}[n-1]$. Since $\mathbf{x}^T[n]\hat{w}[n-1]$ is a scalar we can write this as $\mathbf{x}^T[n]\hat{w}[n-1]k(n)$ and get

$$\hat{w}[n] = \hat{w}[n-1] + (d[n]\mathbf{x}^T[n]\hat{w}[n-1])k(n)\,.$$

When we define $\alpha[n] = d[n] - \mathbf{x}^T[n]\hat{w}[n-1]$ we get the result in the book.

Problem solutions

Problem 8.1 (finding the optimal second-order AR predictor)

We can solve this problem using Equation 14 to compute the optimal weights. In that equation we need to compute $r_x(k)$ for an AR(2) model. For an AR(2) model there is an exact formula for the autocorrelation function $r_x(k)$ (see [1]), which we now state. For the AR(2) model given by

$$z_t = \phi_1 z_{t-1} + \phi_2 z_{t-2} + a_t \,,$$

we first solve the quadratic equation

$$1 - \phi_1 B - \phi_2 B^2 = 0 \,,$$

for its two roots which we denote as G_1^{-1} and G_2^{-2}. Then the autocorrelation function for an AR(2) model is given by

$$r_x(k) = \frac{G_1(1-G_2^2)G_1^k - G_2(1-G_1^2)G_2^k}{(G_1 - G_2)(1+G_1G_2)} \,,$$

When we use that formula we find the autocorrelation function for $0 \leq k \leq 3$ given by

$$r_x(k) = 1.0000 \,, 0.7032 \,, 0.0850 \,, -0.4614 \,.$$

When we then put these into Equation 14 and solve for w we find $w = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} 1.2728 \\ -0.8100 \end{bmatrix}$ for the values that would be used for the estimation of $x[n]$ from $x[n-1]$ and $x[n-2]$ in the expression

$$\hat{x}[n] = w_0 x[n-1] + w_1 x[n-2] \,.$$

This part of the problem is worked in the MATLAB file `prob_8_1.m`.

To demonstrate how to generate these coefficients using the LMS algorithm we note that the vector $\mathbf{w}[k]$ and $\mathbf{x}[k]$ in the LMS algorithm are given by

$$\mathbf{w}[k] = \begin{bmatrix} w_0[k] \\ w_1[k] \end{bmatrix} \,, \quad \text{and} \quad \mathbf{x}[k] = \begin{bmatrix} x[k-1] \\ x[k-2] \end{bmatrix} \,.$$

The output from this filter is

$$y[k] = w_0[k]x[k-1] + w_1[k]x[k-2] \,.$$

We desire $y[k] \approx x[k]$ and define an error $e[k] = x[k] - y[k]$. With this notation the LMS weight update equation for $\mathbf{w}[k]$ becomes

$$\mathbf{w}[k+1] = \mathbf{w}[k] + 2\eta e[k]\mathbf{x}[k] \,.$$

We expect that if we have convergence that the components of the $\mathbf{w}[k]$ vector will approach their correct values. That is $w_0[k] \to 1.2728$ and $w_1[k] \to -0.81$ as $k \to +\infty$. For stability we must pick the parameter η such that $0 < \eta < \frac{1}{\lambda_{\max}}$ with

$$\lambda_{\max} \leq \sum_{k=0}^{p} \lambda_k = \mathrm{Tr}[R_x] = r_x(0) + r_x(1) = 1.0 + 0.7119 = 1.7119 \,.$$

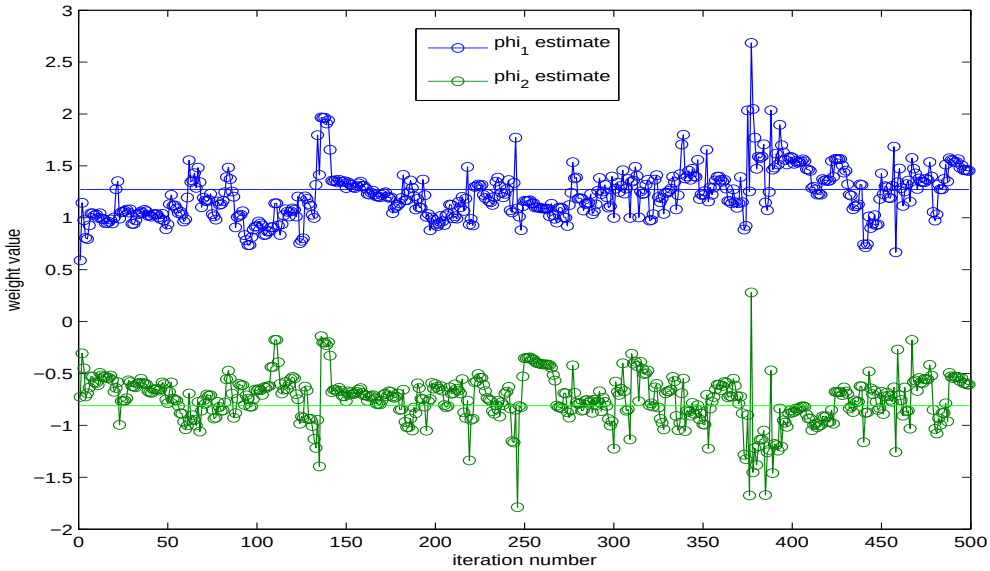

Figure 1: Convergence plots of the coefficient weights $w_0[k]$ and $w_1[k]$ when we run the LMS algorithm.

Computational Note: We will compute the one-dimensional signal using the MATLAB function `filter`. Once we have have the state $x[n]$ generated we will use the routine `aalms.m` from [2] which implements the LMS algorithm in MATLAB. When we run the MATLAB script `prob_8_1_LMS.m` we get the plot shown in Figure 1.

Problem 8.2 (bounds on the LMS step size)

For the autocovariance matrix for the previous problem we find eigenvalues given by 0.2968 and 1.7032. Thus the step size in the LMS algorithm

$$\mathbf{w}[k+1] = \mathbf{w}[k] + 2\eta e[k]\mathbf{x}[k]\,,$$

should be take

$$0 < \eta < \frac{1}{\lambda_{\max}} = \frac{1}{1.7032} = 0.5871\,. \tag{38}$$

In practice the step size should be at least 10 times this small giving a value of 0.058.

Problem 8.3 (the normalized LMS algorithm)

The normalized LMS algorithm uses the following to update the weights $\mathbf{w}[k]$

$$\mathbf{w}[k+1] = \mathbf{w}[k] + \frac{2\eta e[k]}{(a + ||\mathbf{x}[k]||^2)}\mathbf{x}[k]\,, \tag{39}$$

for $a > 0$. This is the same setup as problem 8.1 but using a different algorithm. See the MATLAB function `prob_8_1_NLMS.m` where we use the routine `aanlms.m` from [2] which implements the normalized LMS algorithm in MATLAB.

Problem 8.4 (the complex LMS algorithm)

If the process $\mathbf{x}[k]$ and the FIR filter coefficients are *complex* then $y[k] = \hat{x}[k] = \mathbf{w}^T\mathbf{x}$ as before but now $\mathbf{x}$ and $\mathbf{w}$ can be complex vectors. The LMS algorithm is based on the method of steepest descent in that the coefficients $\mathbf{w}[k]$ are updated as

$$\mathbf{w}[k+1] = \mathbf{w}[k] - \eta \nabla_{\mathbf{w}} E\{e^2[k]\} \,. \tag{40}$$

To derive the LMS algorithm we now need to evaluate $\nabla_{\mathbf{w}} E\{e^2[k]\}$. We find a derivative of the above given by (see equation 5.5 and 5.6 in the book in Chapter 5 for another example of this type of calculation) as

$$\begin{aligned}\nabla_{\mathbf{w}} E\{e^2[k]\} &= \frac{\partial}{\partial \mathbf{w}^*} e^2[k] = \frac{\partial}{\partial \mathbf{w}^*} e[k]e^*[k] = \frac{\partial}{\partial \mathbf{w}^*}(d[k] - y[k])(d^*[k] - y^*[k]) \\ &= (d[k] - y[k])\frac{\partial}{\partial \mathbf{w}^*}(d^*[k] - \mathbf{w}^H\mathbf{x}^*) = (d[k] - y[k])\frac{\partial}{\partial \mathbf{w}^*}(-\mathbf{w}^H\mathbf{x}^*) \,.\end{aligned}$$

By working out components we can see that

$$\frac{\partial}{\partial \mathbf{w}^*}(-\mathbf{w}^H\mathbf{x}^*) = -\mathbf{x}^* \,,$$

and thus

$$\nabla_{\mathbf{w}} E\{e^2[k]\} = -2e[k]\mathbf{x}^*[k] \,,$$

and the LMS algorithm takes the suggested form

$$\mathbf{w}[k+1] = \mathbf{w}[k] + 2\eta e[k]\mathbf{x}^*[k] \,. \tag{41}$$

Problem 8.6 (pick w to minimize exponentially weighted RLS)

The objective function we seek to minimize for this problem is

$$\xi(n) = \sum_{k=0}^{n} \lambda^{n-k} |d[k] - \mathbf{w}^T[n]\mathbf{x}[k]|^2 \,. \tag{42}$$

To determine a value of $\mathbf{w}[n]$ that minimizes $\xi(n)$ we take the derivative of $\xi(n)$ with respect to the scalars $w_m^*[n]$ (note the conjugate), set the result equal to zero and solve for $w_m[n]$. We find this derivative (see equations 5.5 and 5.6 in the book in Chapter 5 for another example

of this type of calculation) given by

$$\begin{aligned}\frac{\partial \xi(n)}{\partial w_m[n]^*} &= \sum_{k=0}^{n} \lambda^{n-k} \frac{\partial}{\partial w_m[n]^*} |d[k] - \mathbf{w}^T[n]\mathbf{x}[k]|^2 \\ &= \sum_{k=0}^{n} \lambda^{n-k} \frac{\partial}{\partial w_m[n]^*} (d[k] - \mathbf{w}^T[n]\mathbf{x}[k])(d[k] - \mathbf{w}^T[n]\mathbf{x}[k])^* \\ &= \sum_{k=0}^{n} \lambda^{n-k} (d[k] - \mathbf{w}^T[n]\mathbf{x}[k]) \frac{\partial}{\partial w_m[n]^*} (d^*[k] - \mathbf{w}^H[n]\mathbf{x}^*[k]) \\ &= \sum_{k=0}^{n} \lambda^{n-k} (d[k] - \mathbf{w}^T[n]\mathbf{x}[k]) \frac{\partial}{\partial w_m[n]^*} (-\mathbf{w}^H[n]\mathbf{x}^*[k]) \,. \end{aligned} \tag{43}$$

Note that this last derivative is given by

$$\frac{\partial}{\partial w_m[n]^*} (-\mathbf{w}^H[n]\mathbf{x}^*[k]) = -x_m^*[k] \,.$$

When we put this expression back into Equation 43 we get two terms

$$-\sum_{k=0}^{n} \lambda^{n-k} d[k] x_m^*[k] + \sum_{k=0}^{n} \mathbf{w}^T[n]\mathbf{x}[k] x_m^*[k] \,, \tag{44}$$

both of which are scalars. Since we need to evaluate $\frac{\partial \xi(n)}{\partial w_m[n]^*}$ for $m = 0, 1, \ldots, p-1$ we have p equations like the above expression. If we write these equations in rows for each of the m values we see that the first term in Equation 44 when written as a vector is

$$-\sum_{k=0}^{n} \lambda^{n-k} d[k]\mathbf{x}^*[k] = -\theta(n) \,.$$

For the second term in Equation 44 we first use $\mathbf{w}^T[n]\mathbf{x}[k] = \mathbf{x}[k]^T\mathbf{w}[n]$ to write it as

$$\sum_{k=0}^{n} \lambda^{n-k} x_m^*[k]\mathbf{x}[k]^T\mathbf{w}[n] \,.$$

This is again a scalar since $\mathbf{x}[k]^T\mathbf{w}[n]$ is the scalar inner product. If we write this in rows for each of the m values we see that we "fill in" the vector $x_m^*[k]$ and get

$$\sum_{k=0}^{n} \lambda^{n-k} \mathbf{x}^*[k]\mathbf{x}[k]^T\mathbf{w}[n] = \left(\sum_{k=0}^{n} \lambda^{n-k} \mathbf{x}^*[k]\mathbf{x}[k]^T \right) \mathbf{w}[n] = \Phi(n)\mathbf{w}[n] \,.$$

Thus we have shown that

$$\frac{\partial \xi(n)}{\partial w_m[n]^*} = -\theta(n) + \Phi(n)\mathbf{w}[n] \,. \tag{45}$$

Setting this result equal to zero and solving for $\mathbf{w}[n]$ gives the requested expression.

Frequency Domain Adaptive Filters

Problem solutions

Problem 9.1 (the number of multiplications in the block LMS algorithm (BLMS))

The block LMS algorithm has a weight update expression given by

$$\mathbf{w}[n+L] = \mathbf{w}[n] + 2\eta \sum_{m=0}^{L-1} e[n+m]\mathbf{x}[n+m]\,. \tag{46}$$

per block of $L \geq 1$ input points. Here $e[n+m]$ is the error made at sample $n+m$ and is given by

$$e[n+m] = d[n+m] - \mathbf{x}^T[n+m]\mathbf{w}[n]\,, \tag{47}$$

where we see that $\mathbf{w}[n]$ is held constant for the L error point estimations. For a fixed value of m, to evaluate $e[n+m]$ requires p multiplications to compute the inner product $\mathbf{x}^T[n+m]\mathbf{w}[n]$ and to evaluate the product $e[n+m]\mathbf{x}[n+m]$ another p multiplications. As this must be done for each value of $0 \leq m \leq L-1$. We get a total of $2pL$ multiplications to update $\mathbf{w}$ by the information contained in L input points. If we process $L = p$ points we get $2p^2$ multiplications, the same number as the normal LMS algorithm (see the text at the end of the chapter). This argument seems to indicate that we do *not* save in computational complexity with respect to multiplications when using the block LMS algorithm.

Problem 9.4 (frequency domain filtering)

The DFT F matrix is a $p \times p$ matrix that looks like

$$F = \begin{bmatrix} F_{00} & F_{01} & F_{02} & \cdots & F_{0(p-1)} \\ F_{10} & F_{11} & F_{12} & \cdots & F_{1(p-1)} \\ \vdots & & & & \vdots \\ F_{(p-2)0} & F_{(p-2)1} & F_{(p-2)2} & \cdots & F_{(p-2)(p-1)} \\ F_{(p-1)0} & F_{(p-1)1} & F_{(p-1)2} & \cdots & F_{(p-1)(p-1)} \end{bmatrix},$$

with the mn element given by

$$F_{mn} = e^{-j\frac{2\pi mn}{p-1}} \quad \text{for} \quad 0 \leq m, n \leq p-1\,,$$

and has an inverse given by $\frac{1}{p}F^H$ or the elements

$$\frac{1}{p} e^{j\frac{2\pi mn}{p-1}}\,.$$

Problem 9.5 (main purpose of the overlap-save and overlap-add methods)

The main purpose of the overlap-save and overlap-add methods is to turn circular convolutions into linear ones.

Problem 9.6 (disadvantages of the circular convolution method)

The circular convolution method had degraded performance.

Introduction to Neural Networks

Problem solutions

Problem 12.1 (aspects of artificial neural networks (ANN))

Following the two items from Haykin's formal definition of an artificial neural network we recall:

- In a neural network we acquire knowledge through a learning process
- The weights or connection strengths are used to store this knowledge.

Problem 12.2 (general characteristics of artificial neural networks)

Artificial neural networks are powerful techniques because they can deal with nonlinearities, high dimensions, noisy, complex, and imperfect sensor data. In addition artificial neural networks don't necessarily require a complete model of the problem to produce useful results.

Problem 12.3 (main types of artificial neural networks)

The book mentions three main types of artificial neural networks:

- Supervised learning
- Reinforcement learning
- Self-organizing (unsupervised learning) type

Problem 12.4 (artificial neural networks as black boxes)

Artificial neural networks can operate as black boxes in some situations. They should not be used when their performance gives the *same* results as other technologies for solving the same problem.

Problem 12.5 (possible applications of artificial neural networks)

The book mentions several applications of artificial neural networks. Some of them include

- Classification or pattern recognition
- Multivariate function mapping
- Time series analysis
- Nonlinear filtering or signal processing
- Intelligent or nonlinear control

The list of applications is only limited by ones imagination.

Problem 12.6 (an example artifical neural networks)

We can think of the output of a three layer neural network as the composition of linear operations and nonlinear activation function $f(\cdot)$ as

$$\mathbf{y} = \mathbf{f}(\mathbf{U}\mathbf{f}(\mathbf{W}\mathbf{x}))\,. \tag{48}$$

It can help to recall the definitions of the elements of $\mathbf{W}$ and $\mathbf{U}$ and then to place them into rectangular arrays such that the above multiplications are valid. The weight w_{ji} is the connection from the input node x_i to hidden node h_j. To facilitate the matrix product $\mathbf{W}\mathbf{x}$ the matrix $\mathbf{W}$ is of size $(M+1)\times(p+1)$ and is indexed with elements w_{ji} as

$$\mathbf{W} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ w_{10} & w_{11} & \cdots & w_{1p} \\ w_{20} & w_{21} & \cdots & w_{2p} \\ \vdots & \vdots & & \vdots \\ w_{j0} & w_{j1} & \cdots & w_{jp} \\ \vdots & \vdots & & \vdots \\ w_{M0} & w_{M1} & \cdots & w_{np} \end{bmatrix}.$$

Note following the book we are using the convention that the weights between the inputs $\mathbf{x}$ and the internal bias node h_0 are zero. This is why the first row in the matrix $\mathbf{W}$ is zero.

The matrix $\mathbf{U}$ is indexed u_{kj} which are the weights between the hidden layer h_j and the output layer y_k. To facilitate the matrix product $\mathbf{U}\mathbf{f}(\cdot)$ the matrix $\mathbf{U}$ is of dimension $K\times(M+1)$ and in terms of u_{kj} looks like

$$\mathbf{U} = \begin{bmatrix} u_{10} & u_{11} & u_{12} & \cdots & u_{1M} \\ u_{20} & u_{21} & u_{22} & \cdots & u_{2M} \\ \vdots & \vdots & \vdots & & \vdots \\ u_{k0} & u_{k1} & u_{k2} & \cdots & u_{kM} \\ \vdots & \vdots & \vdots & & \vdots \\ u_{K0} & u_{K1} & u_{K2} & \cdots & u_{KM} \end{bmatrix}.$$

Given the numerical values specified for this problem we can now evaluate the initial values for $\mathbf{W}$ and $\mathbf{U}$.

Part (a): The matrix $\mathbf{W}$ is of size $(M+1)\times(p+1) = 3\times 3$ and for the network given here we have

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

The matrix $\mathbf{U}$ is of dimension $K\times(M+1) = 1\times 3$ and for the network given we have $\mathbf{U} = \begin{bmatrix} -3 & 1 & -2 \end{bmatrix}$.

Part (b): The vector hidden layer output $\mathbf{h}$ is given by the first composition above or

$$\begin{bmatrix} h_0 \\ h_1 \\ h_2 \end{bmatrix} = \mathbf{f}(\mathbf{W}\mathbf{x}) = f\left(\begin{bmatrix} 0 & 0 & 0 \\ 3 & -1 & 2 \\ 1 & 2 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 3 \end{bmatrix} \right) = f\left(\begin{bmatrix} 0 \\ 8 \\ 3 \end{bmatrix} \right) = \begin{bmatrix} 0.5 \\ 0.9997 \\ 0.9526 \end{bmatrix} .$$

Note that due to the form of the activation function $f(\cdot)$ we have $h_0 = f(0) = \frac{1}{2} \neq 1$. In many places in the book the bias element h_0 is stated to be 1. We could enforce this if we change the form of the activation function (by multiplying by 2), the results are still consistent if we allow $h_0 = \frac{1}{2}$, rather than 1. The output $\mathbf{y}_1$ is then given in terms of the values of the hidden units as

$$y_1 = f(\mathbf{U}\mathbf{h}) = f\left(\begin{bmatrix} -3 & 1 & -2 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.9997 \\ 0.9526 \end{bmatrix} \right) = 0.0828 .$$

Part (c): We now want to calculate the error signals δy_1, δh_1, and δh_2. From the book we have

$$\delta y_k = (y_k - d_k) y_k (1 - y_k) = (0.0828 - 0.9)(0.0828)(1 - 0.0828) = -0.0620 ,$$

when we take $k = 1$. Again from the book we have the error signals δh_j for a general number K of output nodes is given by

$$\delta h_j = \left(\sum_{a=1}^{K} (y_a - d_a) y_a (1 - y_a) u_{aj} \right) h_j (1 - h_j) .$$

When we take $K = 1$ this becomes

$$\delta h_j = (y_1 - d_1) y_1 (1 - y_1) u_{1j} h_j (1 - h_j) .$$

In this later expression we take $j = 1$ and $j = 2$ and will need to use $u_{11} = 1$ and $u_{12} = -2$ with h_1 and h_2 computed above. We find $\delta h_1 = -2.0796\, 10^{-5}$ and $\delta h_2 = 0.0056$.

Part (d): Once we have the error signals δy_1, δh_1, and δh_2 we adjust the elements of $\mathbf{U}$ as

$$\Delta u_{kj} = -\eta \delta y_k h_j , \tag{49}$$

for $k = 1, \ldots, K$ and $j = 0, 1, \ldots, M$. Once we have these elements we update $\mathbf{U}$ as

$$\mathbf{U}^{\text{new}} = \mathbf{U}^{\text{old}} + \Delta\mathbf{U}\,.$$

Using the numbers from this problem we find

$$\begin{aligned}\Delta\mathbf{U} &= -\eta\delta y_1 \begin{bmatrix} h_0 & h_1 & h_2 \end{bmatrix} \\ &= -\eta(0.082755) \begin{bmatrix} 0.50000 & 0.99966 & 0.95257 \end{bmatrix} \\ &= \begin{bmatrix} 0.031017 & 0.062014 & 0.059093 \end{bmatrix}\end{aligned}$$

when we take $\eta = 1$. We adjust the elements of $\mathbf{W}$ as

$$\Delta w_{ji} = -\mu\delta h_j x_i\,, \tag{50}$$

for $j = 1, \ldots, M$ and $i = 0, 1, \ldots, p$. Since j starts at 1 we don't update the first row of $\mathbf{W}$ and it stays as a row of zeros. Once we have these elements we update $\mathbf{W}$ as

$$\mathbf{W}^{\text{new}} = \mathbf{W}^{\text{old}} + \Delta\mathbf{W}\,.$$

Using the numbers from this problem we find

$$\begin{aligned}\Delta\mathbf{W} &= -\mu \begin{bmatrix} \delta h_1 \\ \delta h_2 \end{bmatrix} \begin{bmatrix} x_0 & x_1 & x_2 \end{bmatrix} = -\mu \begin{bmatrix} -2.0796\,10^{-5} \\ 0.0056 \end{bmatrix} \begin{bmatrix} 1 & 1 & 3 \end{bmatrix} \\ &= \begin{bmatrix} 2.0796\,10^{-5} & 2.0796\,10^{-5} & 6.2389\,10^{-5} \\ -5.6050\,10^{-3} & -5.6050\,10^{-3} & -1.6815\,10^{-2} \end{bmatrix}.\end{aligned}$$

Now that we have new version of $\mathbf{W}$ and $\mathbf{U}$ we would accept another (randomly chosen) augmented input vector $\mathbf{x}$ and using them compute the needed update matrices $\Delta\mathbf{W}$ and $\Delta\mathbf{U}$ with this new vector. This process is repeated until the error on the training and testing set start to diverge. These numerical calculations are worked in the MATLAB file `prob_12_6.m`

References

[1] G. Box, G. M. Jenkins, and G. Reinsel. *Time Series Analysis: Forecasting & Control (3rd Edition)*. Prentice Hall, 3rd edition, Feb. 1994.

[2] A. Poularikas and Z. Ramadan. *Adaptive filtering primer with MATLAB*. Electrical Engineering Primer Series. CRC/Taylor & Francis, 2006.