Worked Examples and Solutions for the Book: Classification, Parameter Estimation, and State Estimation by F. van der Heijden, R. P. W. Duin, D. de Ridder, and D. M. J. Tax

John L. Weatherwax*

April 17, 2011

*wax@alum.mit.edu

Chapter 2: Detection and Classification

Exercise Solutions

Exercise 3 (sorting tomatoes)

In terms of the quality of the objects representing the three classes we are told to assume that $A > B > C$. Then the expression, $C(\hat{\omega}_i|\omega_k)$, represents the damage or loss of value if we pick the class $\hat{\omega}_i$ for a tomato when the true class is in fact ω_k. We will use these expressions to derive the *expected* cost under the distribution of classes given we have observed a certain feature z. If we assume that we select class $\hat{\omega}_i$ after taking or receiving measurement z the expected cost at making this decision is given by

$$R(\hat{\omega}_i|z) = E[C(\hat{\omega}_i|\omega_k)|z] = \sum_{k=1}^{K} C(\hat{\omega}_i|\omega_k)P(\omega_k|z)\,. \tag{1}$$

In the given hypothesis for this problem we are assuming that the cost of mistaking a class k tomato as a class i tomato where i is a lesser class (i.e. assigning a class B tomato to a class C tomato) will result in a given loss say l_1. At the same time assigning a class k tomato to the class i where i is a *greater* class (i.e. assigning a class B tomato to the class A) will result in a greater loss, say l_2, than previously. Thus we conclude that $l_2 > l_1$. With this formulation our cost matrix is given by something that looks like

$$C(\hat{\omega}_i|\omega_k) = \begin{bmatrix} 0 & l_2 & l_2 \\ l_1 & 0 & l_2 \\ l_1 & l_1 & 0 \end{bmatrix}.$$

Here we have assumed the "loss" due to correct classification is zero. The extra information (assumed here to be zero) is the actual reward (negative cost) received when we perform correct classification.

Exercise 4 (changing the prior)

Bayesian based classification, says to select the class that has the minimum risk associated with that decision, where the risk associated with selecting class $\hat{\omega}_i$, given an measurement z, is given by

$$\begin{aligned} R(\hat{\omega}_i|z) &= \sum_{k=1}^{K} C(\hat{\omega}_i|\omega_k)P(\omega_k|z) \\ &= \frac{1}{p(z)}\sum_{k=1}^{K} C(\hat{\omega}_i|\omega_k)p(z|\omega_k)P(\omega_k)\,. \end{aligned} \tag{2}$$

Here $C(\hat{\omega}_i|\omega_k)$ is the cost or loss associated with the classification of an object from the true class ω_k as an object from the class ω_i. Since the value of $p(z)$ is the same for all classes once

the measurement is taken its value makes no difference in the selection of the minimal risk class and thus does not need to be considered when performing classification. In the same way, if we estimate the prior probabilities of each class from a set of training samples as

$$P(\omega_k) \approx \frac{N_k}{N} .$$

Here N_k is the number of observed object from class k and N is the total number of objects from all classes. Then the optimal Bayesian classification can still be made by using this approximation in Equation 2 by selecting the class i that has the smallest value for

$$\sum_{k=1}^{K} C(\hat{\omega}_i|\omega_k) p(z|\omega_k) \frac{N_k}{N} ,$$

or since N is the same for all classes this is equivalent to selecting the class i that has the smallest value for

$$\sum_{k=1}^{K} C(\hat{\omega}_i|\omega_k) p(z|\omega_k) N_k . \tag{3}$$

With this expression we see how to answer the given question. If the number of objects in the scrap class was to double, then $N_{\text{scrap}} \to 2N_{\text{scrap}}$ and in order for the decision region to not change we require Equation 3 (with possibly modified cost functions $\tilde{C}(\hat{\omega}_i|\omega_k)$) evaluate to the same numerical value for each i. This implies that that the new set of cost functions $\tilde{C}(\hat{\omega}_i|\omega_k)$ be the same for all classes except when the summation index k corresponds to the scrap class. In that case we need to take new costs given by

$$\tilde{C}(\hat{\omega}_i|\omega_{\text{scrap}}) = \frac{1}{2} C(\hat{\omega}_i|\omega_{\text{scrap}}) ,$$

in order that the fact that we have doubled the number of scrap objects not influence our classification decision. Thus we make the scrap column in the entire cost matrix equal to one half of the cost of the scrap column in the old cost matrix.

Exercise 5 (inputs for Bayesian classification)

To compute a Bayes classification one needs three things

- A representation of how likely each object/class is given *no* other information. Mathematically this is represented by a prior probability, $P(\omega_k)$, for each class.
- A representation for how the observed features are related to the given classes. Mathematically this is given by the likelihood functions $p(z|\omega_k)$.
- A cost associated with the action of deciding an object is a member of class i when in-fact it is a member of class k. Mathematically this is given by a cost matrix $C(\hat{\omega}_i|\omega_k)$.

The first two items above would be estimated from past examples of the objects in each class, while the costs have to be determined by the preferences of the users of the classification algorithm.

Exercise 6 (decision regions for normally distributed features)

For any cost matrix $C(\hat{\omega}_i|\omega)$ the Bayes' decision boundary is given by selecting the conditional risk that is smallest, where the expression for the conditional risk, $R(\hat{\omega}_i|x)$, is defined in Equation 2. That is our classification rule is

$$\text{classify } z \text{ in } \omega_i \quad \text{if} \quad R(\hat{\omega}_i|z) < R(\hat{\omega}_j|x) \quad \forall j \neq i\,.$$

Using Bayes' rule to express $P(\omega_i|z)$ in the definition of the conditional risk in terms of $p(z|\omega_i)$ we can transform this rule into the following decision rule. We classify z as a member of ω_i if

$$\sum_{k=1}^{K} C(\hat{\omega}_i|\omega_k)p(z|\omega_k)P(\omega_k) < \sum_{k=1}^{K} C(\hat{\omega}_i|\omega_k)p(z|\omega_k)P(\omega_k) \quad \forall j \neq i\,.$$

For any given functional form for the conditional density $p(z|\omega_k)$ (including normal) the above expression can be evaluated and a classification decision made.

Exercise 7 (equal covariance matrices)

A feature that is a physical trait like the heights or weights of animals may be approximately normal and have approximately the same "spread" of values around its mean depending on whether one is considering males or females of the species. This equivalent spread implies equivalent class covariances.

Exercise 8 (some specific ROC curves)

The classical ROC analysis for a given detection problem is a plot of the probability of a false alarm $P_{\text{fa}} = P(\hat{\omega}_2|\omega_1)$ as the dependent variable and the probability of detection $P_{\text{d}} = P(\hat{\omega}_2|\omega_2)$ as the independent variable. Here we are assuming that the class ω_1 corresponds to "noise" and is uninteresting while the class ω_2 corresponds to an object of interest we would like to detect. In the case when the two distributions of x have *no* overlap we expect $P_{\text{fa}} \approx 0$ and $P_{\text{d}} \approx 1$ because we will never make a mistake when we perform the classification problem. Thus in this case the ROC curve becomes more of a point (rather than a curve) and we have

$$(P_{\text{fa}}, P_{\text{d}}) = (0, 1)\,.$$

This result can also be seen from the explicit expressions derived in the text for P_{fa} and P_{d} since in the no overlap case the discriminant expression d defined as

$$d^2 = (\mu_1 - \mu_2)^T C^{-1}(\mu_1 - \mu_2)\,,$$

is very large as there is no overlap so $\mu_1 \ll \mu_2$. Thus we see that

$$P_{\text{fa}} = \frac{1}{2} + \frac{1}{2}\text{erf}\left(\frac{T - \frac{1}{2}d^2}{d\sqrt{2}}\right) \to \frac{1}{2} + \frac{1}{2}\text{erf}(-\infty) = \frac{1}{2} + \frac{1}{2}(-1) = 0$$

$$
\begin{aligned}
P_{\text{miss}} &= \frac{1}{2} - \frac{1}{2}\text{erf}\left(\frac{T+\frac{1}{2}d^2}{d\sqrt{2}}\right) \rightarrow \frac{1}{2} - \frac{1}{2}\text{erf}(+\infty) = \frac{1}{2} - \frac{1}{2} = 0 \quad \text{so} \\
P_{\text{d}} &= 1 - P_{\text{miss}} = 1\,.
\end{aligned}
$$

If the two distributions have zero overlap then $d = 0$ since zero since $\mu_1 = \mu_2$ and we have no information as to the class from the measurement z and the class priors determine everything. Thus if $P(\omega_1) > P(\omega_2)$ we should aways pick the first class and if the opposite inequality holds we would always pick the second class. Since T is defined as $\log(\frac{P(\omega_2)}{P(\omega_1)})$ in terms of T, if $T < 0$ we would always pick class "one" while if $T > 0$ we would always pick class "two". Continuing this line of reasoning, if $T < 0$ we are always picking class one so our probability of false alarm is zero (since we never declare class two), while our probability of detection is always zero (for the same reason). If $T > 0$ as we are always selecting class two our probability of false alarm is one and our probability of detection is also one. We can verify these results using the expressions for $P_{fa}(T)$ and $P_d(T)$ given in the text. We have

$$
\begin{aligned}
P_{fa}(T) &= \begin{cases} \frac{1}{2} + \frac{1}{2}\text{erf}(-\infty) \\ \frac{1}{2} + \frac{1}{2}\text{erf}(+\infty) \end{cases} = \begin{cases} \frac{1}{2} + \frac{1}{2}(-1) = 0 & \text{when} \quad T < 0 \\ \frac{1}{2} + \frac{1}{2}(1) = 1 & \text{when} \quad T > 0 \end{cases} \\
P_d &= \begin{cases} \frac{1}{2} + \frac{1}{2}\text{erf}(-\infty) \\ \frac{1}{2} + \frac{1}{2}\text{erf}(+\infty) \end{cases} = \begin{cases} \frac{1}{2} + \frac{1}{2}(-1) = 0 & \text{when} \quad T < 0 \\ \frac{1}{2} + \frac{1}{2} = 1 & \text{when} \quad T > 0 \end{cases} ,
\end{aligned}
$$

which verify the arguments given above.

Exercise 9 (changes to the ROC curve depending on the priors)

The ROC curve of a detector is a plot of P_d as a function P_{fa} as we very the detection threshold T. In the case of Gaussian measurement vectors with equal class conditional covariance matrices (denoted C) and means μ_1 and μ_2 one can derive the following expressions for P_{fa} and P_d

$$
\begin{aligned}
P_{fa} &= \frac{1}{2} + \frac{1}{2}\text{erf}\left(\frac{T-\frac{1}{2}d^2}{d\sqrt{2}}\right) \\
P_d &= 1 - P_{miss} \\
&= \frac{1}{2} + \frac{1}{2}\text{erf}\left(\frac{T+\frac{1}{2}d^2}{d\sqrt{2}}\right) ,
\end{aligned}
$$

where $d^2 = (\mu_1 - \mu_2)^t C^{-1}(\mu_1 - \mu_2)$, is the Mahalanobis distance between the two means, and T is the threshold defined in terms of the class priors as

$$
T = \log\left(\frac{P(\omega_2)}{P(\omega_1)}\right) = \log\left(\frac{1-P(\omega_1)}{P(\omega_2)}\right) = \log\left(\frac{1}{P(\omega_1)} - 1\right) . \tag{4}
$$

In general one plots the probability of false alarm and detection as a function of this threshold T. When we do this as we change T we sweep over the points on the ROC curve. For *given* prior probabilities $P(\omega_1)$ a specific point on the ROC curve (corresponding to the computed T value) is determined. If we consider how the value of T changes as we change the prior probabilities we see that as a function of $P(\omega_1)$ since $0 < P(\omega_1) < 1$, that when $P(\omega_1) \sim 0$ we have

$$
T = \log\left(\frac{1}{P(\omega_1)} - 1\right) \sim +\infty\,.
$$

So that $P_{fa} \sim 1$ and $P_d \sim 1$, which is to be expected since when $P(\omega_1) \sim 0$ we will always classify as class two. In the case when $P(\omega_1) \sim 1$ we have

$$T = \log\left(\frac{1}{P(\omega_1)} - 1\right) \sim -\infty\,.$$

So that $P_{fa} \sim 0$ and $P_d \sim 0$, which is to be expected since when $P(\omega_1) \sim 1$ we will always classify as class one.

Exercise 10 (the class conditional probabilities from the ROC curve)

For this problem we assume that we are given the ROC curve $(P_{fa}(T), P_d(T))$ and we will compute the class conditional distribution of the log-likelihood ratio Λ i.e. $p_\Lambda(\Lambda|\omega_i)$ for both classes $i = 1, 2$ from it. Here Λ is the log-likelihood ratio given by

$$\Lambda = \log\left(\frac{p(z|\omega_1)}{p(z|\omega_2)}\right)\,.$$

Recalling the definitions of P_{fa} and P_d in terms of Λ we have

$$\begin{aligned} P_{fa}(T) &= P(\Lambda(z) < T|\omega_1) = \int_{-\infty}^{T} p_\Lambda(\Lambda|\omega_1)d\Lambda \\ P_{miss}(T) &= P(\Lambda(z) > T|\omega_2) = \int_{T}^{\infty} p_\Lambda(\Lambda|\omega_2)d\Lambda \\ P_d(T) &= 1 - P_{miss}(T)\,. \end{aligned}$$

Taking the derivatives of $P_{fa}(T)$ and $P_d(T)$ with respect to T we find

$$\begin{aligned} \frac{dP_{fa}}{dT} &= p_\Lambda(T|\omega_1) \\ \frac{dP_d}{dT} &= -p_\Lambda(T|\omega_2) = -(1 - p_\Lambda(T|\omega_1))\,. \end{aligned}$$

Dividing these two expressions we find

$$\frac{dP_d}{dP_{fa}} = -\frac{(1 - p_\Lambda(T|\omega_1))}{p_\Lambda(T|\omega_1)} = 1 - \frac{1}{p_\Lambda(T|\omega_1)}\,,$$

so solving for the conditional density function $p_\Lambda(T|\omega_1)$ we find

$$p_\Lambda(T|\omega_1) = \frac{1}{1 - \frac{dP_d}{dP_{fa}}}\,, \tag{5}$$

implying that $p_\Lambda(T|\omega_2)$ is given by a similar expression

$$p_\Lambda(T|\omega_2) = 1 - p_\Lambda(T|\omega_1) = \frac{\frac{dP_d}{dP_{fa}}}{\frac{dP_d}{dP_{fa}} - 1}\,. \tag{6}$$

Both of these are true provided that $\frac{dP_d}{dP_{fa}} \neq 1$. Thus given the ROC curve we can compute its slope $\frac{dP_d}{dP_{fa}}$ and use Equations 5 and 6 to derive the desired class conditional densities.

Chapter 3: Parameter Estimation

Notes on the Text

An estimator of the backscattering coefficient

We are told to assume that for fixed x the fraction $\frac{N_{probes}z}{x}$ is a Gamma distribution with parameter N_{probes}. Then if we define $u = \frac{N_{probes}z}{x}$, the probability of z given x is given by the transformation of probability distributions rule

$$\begin{aligned} p_Z(z|x) &= p_U(u(z)|x)\frac{du(z)}{dz} \\ &= \text{gamma\_pdf}\left(\frac{N_{probes}z}{x}; N_{probes}\right)\frac{N_{probes}}{x}\,, \end{aligned}$$

which is equation 3.5 in the book.

the MAP estimator

Our cost function in this case is given by

$$C(\hat{x}|x) = \begin{cases} 1 & ||\hat{x}-x||_1 > \Delta \\ 0 & ||\hat{x}-x||_1 < \Delta \end{cases} \tag{7}$$

Our conditional risk under this cost function is given by

$$\begin{aligned} R(\hat{x}|z) &= \int_X C(\hat{x}|x)p(x|z)dx \\ &= \int_X 1p(x|z)dx - \int_{||\hat{x}-x||_1<\Delta} 1p(x|z)dx \\ &\approx 1 - p(\hat{x}|z)\Delta \quad \text{as} \quad \Delta \to 0\,, \end{aligned}$$

which is the expression given in the book.

Exercise Solutions

Exercise 1 (a linear MMSE estimator)

If we assume a form for our estimator given by $\hat{x}_{\text{lMMSE}}(z) = Kz$, with K an unspecified (as of yet) matrix. We can begin by computing the overall risk R in terms of the conditional risk $R(\hat{x}|z)$ using such a decision rule as

$$R = \int_Z R(\hat{x}(z)|z)p(z)dz\,.$$

Using the definition of the conditional risk as the expectation of a mean square cost C as $R(\hat{x}|z) = E[C(\hat{x}|x)|z] = \int_X C(\hat{x}|x)p(x|z)dx$, we obtain

$$\begin{aligned} R &= \int_Z \int_X C(\hat{x}|x)p(x|z)p(z)dxdz \\ &= \int_Z \int_X (Kz - x)^T (Kz - x)p(x|z)p(z)dxdz\,. \end{aligned}$$

We now want to minimize this expression with respect to the parameter matrix K. Expanding out the quadratic term in the integrand we have

$$(Kz - x)^T(Kz - x) = z^T K^T Kz - z^T K^T x - x^T Kz + x^T x\,.$$

So that our overall risk R becomes

$$R = \int_Z \int_X (z^T K^T Kz - z^T K^T x - x^T Kz + x^T x)p(x|z)p(z)dxdz\,.$$

To minimize this expression with respect to K we take the K derivative of it, set the resulting expression equal to zero, and solve for K. To do this recall the matrix derivative identity that for matrices X and C and vectors a and b we have

$$\frac{\partial(a^T X^T CXb)}{\partial X} = C^T Xab^T + CXba^T\,. \tag{8}$$

From which we derive

$$\frac{\partial(z^T K^T Kz)}{\partial K} = Kzz^T + Kzz^T = 2Kzz^T\,.$$

Next recalling the identities

$$\frac{\partial(a^T Xb)}{\partial X} = ab^T \quad \text{and} \quad \frac{\partial(a^T X^T b)}{\partial X} = ba^T\,, \tag{9}$$

we have that

$$\frac{\partial(x^T Kz)}{\partial K} = xz^T \quad \text{and} \quad \frac{\partial(z^T K^T x)}{\partial K} = xz^T\,.$$

Thus we find the expression $\frac{\partial R}{\partial K} = 0$ becomes

$$\frac{\partial R}{\partial K} = \int_Z \int_X (2Kzz^T - xz^T - xz^T)p(x,z)dxdz = 0\,,$$

or

$$K \int_Z \int_X zz^T p(x,z)dxdz = \int_Z \int_X xz^T p(x,z)dxdz\,,$$

or

$$K \int_Z zz^T p(z)dz = E[xz^T]$$

or

$$KE[zz^T] = E[xz^T]\,,$$

so that K is given by $K = E[xz^T]E[zz^T]^{-1}$. Thus the optimal linear MMSE estimator is given by

$$\hat{x}_{\text{lMMSE}}(z) = E[xz^T]E[zz^T]^{-1}z\,. \tag{10}$$

As an aside we note that we can prove the stated matrix identities used in this problem and given in Equations 8 and 9 by considering the Einstein notation for the inner products involved. To demonstrate, consider the identity $\frac{\partial(z^TK^Tx)}{\partial K} = xz^T$. In Einstein notation we have the function we want to take the derivative of given by

$$z_i(K^T)_{ij}x_j = z_iK_{ji}x_j\,.$$

From which we see that the derivative of this expression becomes

$$\frac{\partial(z^TK^Tx)}{\partial K_{ij}} = x_iz_j = (xz^T)_{ij}\,,$$

the ij-th component of the product xz^T proving the stated identity.

Exercise 2 (non-singular covariance matrices)

If C_x is not invertible then some of the elements of x must be linearly dependent on the others. That is given some subset of the elements of x the remaining elements can be predicted exactly from them. This is assuming that the reason for C_x being singular is *not* because we simply don't have enough data to estimate it properly. The same comments hold for the covariance matrix C_v. If this situation holds (in that we have a singular covariance matrix) we must reduce the dimension of x and v until we are considering a subset of variables that is linearly independent. Principal component analysis will do this for you.

Exercise 3 (a proof that $p(x|z)$ is Gaussian)

In the Gaussian case with linear sensors we will assume that x is normal with mean μ_x and covariance C_x and the measurement $z = Hx + v$, where v is a Gaussian random variable with zero mean and a covariance given by C_v. To evaluate the conditional density $p(x|z)$ we will use Bayes' rule

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)}\,.$$

From the given assumptions on x and z we can conclude that

$$\begin{aligned} p(z|x) &= \mathcal{N}(z; Hx, C_v) \\ p(x) &= \mathcal{N}(x; \mu_x, C_x) \\ p(z) &= \mathcal{N}(z; H\mu_x, HC_xH^T + C_v)\,, \end{aligned}$$

as stated in the books equation 3.32 and derived in Exercise 4 below. In this case then the functional form for the $p(x|z)$ from Bayes' rule is given by

$$\frac{\frac{1}{(2\pi)^{N/2}|C_v|^{1/2}}\exp\left\{-\frac{1}{2}(z-Hx)^TC_v^{-1}(z-Hx)\right\}\frac{1}{(2\pi)^{M/2}|C_x|^{1/2}}\exp\left\{-\frac{1}{2}(x-\mu_x)^TC_x^{-1}(x-\mu_x)\right\}}{\frac{1}{(2\pi)^{N/2}|HC_xH^T+C_v|^{1/2}}\exp\left\{-\frac{1}{2}(z-H\mu_x)^T(HC_xH^T+C_v)^{-1}(z-H\mu_x)\right\}}\,,$$

or simplifying this expression some

$$\begin{aligned} p(x|z) &= \left(\frac{1}{(2\pi)^{M/2}}\right)\left(\frac{|HC_xH^T+C_v|^{1/2}}{|C_v|^{1/2}|C_x|^{1/2}}\right)e^{-\frac{1}{2}\mathcal{L}} \quad \text{with} \\ \mathcal{L} &= (z-Hx)^TC_v^{-1}(z-Hx)+(x-\mu_x)^TC_x^{-1}(x-\mu_x) \\ &- (z-H\mu_x)^T(HC_xH^T+C_v)^{-1}(z-H\mu_x) \end{aligned}$$

The argument of this exponential $\mathcal{L}$ (for log-likelihood) when we expand the quadratics to group by "powers" of the variable x becomes

$$\begin{aligned} \mathcal{L} &= z^TC_v^{-1}z - z^TC_v^{-1}Hx - (Hx)^TC_v^{-1}z + (Hx)^TC_v^{-1}Hx \quad (11) \\ &+ x^TC_x^{-1}x - x^TC_x^{-1}\mu_x - \mu_x^TC_x^{-1}x + \mu_x^TC_x^{-1}\mu_x \\ &- (z-H\mu_z)^T(HC_xH^T+C_v)^{-1}(z-H\mu_x) \\ &= x^TH^TC_v^{-1}Hx + x^TC_x^{-1}x - 2x^TH^TC_v^{-1}z - 2x^TC_x^{-1}\mu_x \quad (12) \\ &+ z^TC_v^{-1}z + \mu_x^TC_x^{-1}\mu_x \\ &- (z-H\mu_z)^T(HC_xH^T+C_v)^{-1}(z-H\mu_x) \\ &= x^T(H^TC_v^{-1}H+C_x^{-1})x - 2x^T(H^TC_v^{-1}z+C_x^{-1}\mu_x) \quad (13) \\ &+ z^TC_v^{-1}z + \mu_x^TC_x^{-1}\mu_x - (z-H\mu_z)^T(HC_xH^T+C_v)^{-1}(z-H\mu_x)\,. \end{aligned}$$

Where in going from Equation 11 to 12 are grouping by "powers" of x and in going from Equation 12 to 13 we are combining these powers. We now introduce an as yet unknown vector m such that the above can be written as

$$\begin{aligned} \mathcal{L} &= (x-m)^T(H^TC_v^{-1}H+C_x^{-1})(x-m) \\ &+ 2x^T(H^TC_v^{-1}H+C_x^{-1})m - m^T(H^TC_v^{-1}H+C_x^{-1})m \\ &- 2x^T(H^TC_v^{-1}z+C_x^{-1}\mu_x) \\ &+ z^TC_v^{-1}z + \mu_x^TC_x^{-1}\mu_x - (z-H\mu_z)^T(HC_xH^T+C_v)^{-1}(z-H\mu_x)\,. \end{aligned}$$

We can make the terms linear in x vanish if we take m such that

$$(H^TC_v^{-1}H+C_x^{-1})m = H^TC_v^{-1}z + C_x^{-1}\mu_x\,,$$

or

$$m = (H^TC_v^{-1}H+C_x^{-1})^{-1}(H^TC_v^{-1}z+C_x^{-1}\mu_x)\,. \quad (14)$$

so that the above becomes (with m replaced only in the $m^T(\cdot)m$ term)

$$\begin{aligned} \mathcal{L} &= (x-m)^T(H^TC_v^{-1}H+C_x^{-1})(x-m) \\ &- (H^TC_v^{-1}z+C_x^{-1}\mu_x)^T(H^TC_v^{-1}H+C_x^{-1})^{-1}(H^TC_v^{-1}z+C_x^{-1}\mu_x) \\ &+ z^TC_v^{-1}z + \mu_x^TC_x^{-1}\mu_x - (z-H\mu_z)^T(HC_xH^T+C_v)^{-1}(z-H\mu_x)\,. \end{aligned}$$

To further simplify this expression we recall the matrix identity given by Equation 21. Defining the matrix $C_{x|z}$ as

$$C_{x|z} \equiv (C_x^{-1}+H^TC_v^{-1}H)^{-1}\,, \quad (15)$$

we will use Equation 21 to write $(HC_xH^T+C_v)^{-1}$ in terms of $C_{x|z}$. We find

$$\begin{aligned} (C_v+HC_xH^T)^{-1} &= C_v^{-1} - C_v^{-1}H(H^TC_v^{-1}H+C_x^{-1})^{-1}H^TC_v^{-1} \quad (16) \\ &= C_v^{-1} - C_v^{-1}HC_{x|z}H^TC_v^{-1}\,. \quad (17) \end{aligned}$$

So we now have

$$\begin{aligned}\mathcal{L} &= (x-m)^T C_{x|z}^{-1}(x-m) \\ &- (H^T C_v^{-1} z + C_x^{-1}\mu_x)^T C_{x|z}(H^T C_v^{-1} z + C_x^{-1}\mu_x) \\ &+ z^T C_v^{-1} z + \mu_x^T C_x^{-1}\mu_x \\ &- (z - H\mu_x)^T \left[C_v^{-1} - C_v^{-1} H C_{x|z} H^T C_v^{-1}\right](z - H\mu_x)\,.\end{aligned}$$

expanding everything we find that the *non* quadratic parts of the above become

$$\begin{aligned}&- z^T C_v^{-1} H C_{x|z} H^T C_v^{-1} z - 2z^T C_v^{-1} H C_{x|z} C_x^{-1}\mu_x - \mu_x^T C_x^{-1} C_{x|z} C_x^{-1}\mu_x \\ &+ z^T C_v^{-1} z + \mu_x^T C_x^{-1}\mu_x \\ &- z^T C_v^{-1} z + 2z^T C_v^{-1} H\mu_x - \mu_x^T H^T C_v^{-1} H\mu_x \\ &+ z^T C_v^{-1} H C_{x|z} H^T C_v^{-1} z - z^T C_v^{-1} H C_{x|z} H^T C_v^{-1} H\mu_x - \mu_x^T H^T C_v^{-1} H C_{x|z} H^T C_v^{-1} z \\ &+ \mu_x^T H^T C_v^{-1} H C_{x|z} H^T C_v^{-1} H\mu_x \\ &= -2z^T \left(C_v^{-1} H C_{x|z} C_x^{-1} - C_v^{-1} H + C_v^{-1} H C_{x|z} H^T C_v^{-1} H\right)\mu_x \\ &- \mu_x^T \left(C_x^{-1} C_{x|z} C_x^{-1} - C_x^{-1} + H^T C_v^{-1} H - H^T C_v^{-1} H C_{x|z} H^T C_v^{-1} H\right)\mu_x\,.\end{aligned}$$

Now consider the matrix between z^T and μ_x or $C_v^{-1} H C_{x|z} C_x^{-1} - C_v^{-1} H + C_v^{-1} H C_{x|z} H^T C_v^{-1} H$. We see that it can be written by some simple factoring as

$$C_v^{-1} H(C_{x|z} C_x^{-1} - I + C_{x|z} H^T C_v^{-1} H) = C_v^{-1} H(C_{x|z}(C_x^{-1} + H^T C_v^{-1} H) - I) = 0\,.$$

Now consider the matrix between μ_x and μ_x which we define to be B. That is

$$B \equiv C_x^{-1} C_{x|z} C_x^{-1} - C_x^{-1} + H^T C_v^{-1} H - H^T C_v^{-1} H C_{x|z} H^T C_v^{-1} H\,.$$

Under some simple factorizations we can simplify this considerably

$$\begin{aligned}B &= C_x^{-1}(C_{x|z} C_x^{-1} - I) + H^T C_v^{-1} H - H^T C_v^{-1} H C_{x|z} H^T C_v^{-1} H \\ &= C_x^{-1} C_{x|z}(C_x^{-1} - C_{x|z}^{-1}) + H^T C_v^{-1} H - H^T C_v^{-1} H C_{x|z} H^T C_v^{-1} H \\ &= C_x^{-1} C_{x|z}(C_x^{-1} - C_x^{-1} - H^T C_v^{-1} H) + H^T C_v^{-1} H - H^T C_v^{-1} H C_{x|z} H^T C_v^{-1} H \\ &= -C_x^{-1} C_{x|z} H^T C_v^{-1} H + H^T C_v^{-1} H - H^T C_v^{-1} H C_{x|z} H^T C_v^{-1} H \\ &= (-C_x^{-1} C_{x|z} + I - H^T C_v^{-1} H C_{x|z}) H^T C_v^{-1} H \\ &= (-C_x^{-1} - H^T C_v^{-1} H + C_{x|z}^{-1}) C_{x|z} H^T C_v^{-1} H = 0\,.\end{aligned}$$

Thus we have shown that with the definitions of m and $C_{x|z}$ the density $p(x|z)$ is given by

$$p(x|z) = \left(\frac{1}{(2\pi)^{M/2}}\right)\left(\frac{|HC_x H^T + C_v|^{1/2}}{|C_v|^{1/2}|C_x|^{1/2}}\right) e^{-\frac{1}{2}(x-m)^T C_{x|z}^{-1}(x-m)}\,.$$

Since the only way this function can normalize to one when integrated over x and represent a valid probability density function is if the coefficient in front of the exponential satisfies

$$\frac{|HC_x H^T + C_v|^{1/2}}{|C_v|^{1/2}|C_x|^{1/2}} = \frac{1}{|C_{x|z}|^{1/2}}\,. \tag{18}$$

This is trivially shown if we use the following determinant identity

$$|A + XBX^T| = |B||A||B^{-1} + X^T A^{-1} X|\,. \tag{19}$$

Using this we have

$$|C_v + HC_xH^T| = |C_v||C_x||C_x^{-1} + H^T C_v^{-1} H| = |C_v||C_x||C_{x|z}^{-1}|\,,$$

which proves Equation 18 and completes the derivation of the fact that $p(x|z)$ is *Gaussian* with a mean m given by Equation 14 and a covariance matrix $C_{x|z}$ given by Equation 15.

Exercise 4 (covariance identities)

We consider a linear measurement model for z given by

$$z = Hx + v\,.$$

Here x is a random variable with mean μ_x with covariance matrix C_x, and v a random variable with zero mean and a covariance matrix given by C_v that is uncorrelated from x. By linearity of the expectation we directly have the mean of z given by

$$\mu_z = H\mu_x\,.$$

From the definition of the covariance we can compute the covariance matrix for z in terms of that of x and v. We find

$$\begin{aligned} C_z &= E[(z-\mu_z)(z-\mu_z)^T] \\ &= E[(Hx+v-H\mu_x)(Hx+v-H\mu_x)^T] \\ &= E[(H(x-\mu_x)+v)(H(x-\mu_x)+v)^T] \\ &= E[(H(x-\mu_x)+v)((x-\mu_x)^T H^T + v^T)] \\ &= E[H(x-\mu_x)(x-\mu_x)^T H^T] + E[H(x-\mu_x)v^T] + E[v(x-\mu_x)^T H^T] + E[vv^T]\,. \end{aligned}$$

Now since x and v are assumed uncorrelated the two cross terms vanish. For example

$$E[H(x-\mu_x)v^T] = HE[x-\mu_x]E[v^T] = 0\,.$$

We are left with

$$C_z = HC_xH^T + C_v\,,$$

the desired result.

Using the definitions of the cross covariances C_{xz} and C_{zx} we find

$$\begin{aligned} C_{xz} &= E[(x-\mu_x)(z-\mu_z)^T] \\ &= E[(x-\mu_x)(Hx-v-H\mu_z)^T] \\ &= E[(x-\mu_x)(x-\mu_x)^T]H^T - E[(x-\mu_x)v^T] \\ &= C_xH^T\,, \end{aligned}$$

and

$$\begin{aligned} C_{zx} &= E[(z-\mu_z)(x-\mu_x)^T] \\ &= E[(Hx-v-H\mu_z)(x-\mu_x)^T] \\ &= E[H(x-\mu_x)(x-\mu_x)^T] + E[v(x-\mu_x)^T] \\ &= HC_x\,, \end{aligned}$$

as claimed.

Exercise 5 (the equivalence of the Kalman form)

From Equation 3.20 in the book we have that

$$\hat{x}_{\text{MMSE}}(z) = (H^T C_v^{-1} H + C_x^{-1})^{-1}(H^T C_v^{-1} z + C_x^{-1}\mu_x)\,. \tag{20}$$

Recalling the matrix inversion lemma given by

$$(A^{-1} + H^T B^{-1} H)^{-1} = A - AH^T(HAH^T + B)^{-1}HA\,, \tag{21}$$

we see that on the matrix C_e defined as $(H^T C_v^{-1} H + C_x^{-1})^{-1}$ we obtain

$$\begin{aligned} C_e &= (C_x^{-1} + H^T C_v^{-1} H)^{-1} &(22)\\ &= C_x - C_x H^T (HC_x H^T + C_v)^{-1} HC_x\,. &(23)\end{aligned}$$

Using this the expression in Equation 20 and expanding we obtain

$$\begin{aligned} \hat{x}_{\text{MMSE}}(z) &= \left[C_x - C_x H^T (HC_x H^T + C_v)^{-1} HC_x\right](H^T C_v^{-1} z + C_x^{-1}\mu_x)\\ &= \mu_x + C_x H^T C_v^{-1} z\\ &- C_x H^T (HC_x H^T + C_v)^{-1} HC_x H^T C_v^{-1} z\\ &- C_x H^T (HC_x H^T + C_v)^{-1} H\mu_x\\ &= \mu_x + C_x H^T (HC_x H^T + C_v)^{-1}(HC_x H^T + C_v) C_v^{-1} z\\ &- C_x H^T (HC_x H^T + C_v)^{-1} HC_x H^T C_v^{-1} z\\ &- C_x H^T (HC_x H^T + C_v)^{-1} H\mu_x\\ &= \mu_x + C_x H^T (HC_x H^T + C_v)^{-1}\\ &\times \left[HC_x H^T C_v^{-1} z + z - HC_x H^T C_v^{-1} z - H\mu_x\right]\\ &= \mu_x + C_x H^T (HC_x H^T + C_v)^{-1}(z - H\mu_x)\,, \end{aligned}$$

which is the expression for $\hat{x}_{\text{MMSE}}(z)$ as we were to show.

Exercise 7 (the unbiased MMSE is indeed unbiased)

The unbiased linear MMSE estimator is given by

$$\hat{x}_{\text{ulMMSE}}(z) = Kz + a\,,$$

with $K = C_{xz} C_z^{-1}$ and $a = \mu_x - K\mu_z$. To show that this estimator is unbiased we take the expectation of $\hat{x}_{\text{ulMMSE}}(z)$. We find

$$\begin{aligned} E[\hat{x}_{\text{ulMMSE}}(z)] &= KE[z] + a\\ &= K\mu_z + \mu_x - K\mu_z = \mu_x\,, \end{aligned}$$

as required for an unbiased estimator.

Exercise 8 (ML estimation of a binomial distribution)

We are told that z is distributed as a binomial random variable with parameters (x, M). This means that the probability we observe the value of z after M trials is given by

$$p(z|x) = \left(\begin{array}{c} M \\ z \end{array} \right) x^z (1-x)^{M-z} \quad \text{for} \quad 0 \leq z \leq M \,.$$

We desire to estimate the probability of success, x, from the measurement z.

Part (a): To compute the maximum likelihood (ML) estimate of x we compute

$$\hat{x}_{\rm ML}(z) = {\rm argmax}_x p(z|x) = {\rm argmax}_x \left(\begin{array}{c} M \\ z \end{array} \right) x^z (1-x)^{M-z} \,.$$

To compute this maximum we can take the derivative of $p(z|x)$ with respect to x, set the resulting expression equal to zero and solve for x. We find the derivative equal to

$$\left(\begin{array}{c} M \\ z \end{array} \right) \left(z x^{z-1}(1-x)^{M-z} + x^z (M-z)(1-x)^{M-z-1}(-1) \right) = 0 \,.$$

Dividing by $x^{z-1}(1-x)^{M-z-1}$ to get

$$z(1-x) + x(M-z)(-1) = 0 \,,$$

and solving for x gives or ML estimate of

$$\hat{x}_{\rm ML}(z) = \frac{z}{M} \,. \tag{24}$$

Part (b): Lets compute the bias and variance of this estimate of x. The bias, $b(x)$, is defined as

$$\begin{aligned} b(x) &= E[\hat{x} - x|x] = E[\hat{x}|x] - x \\ &= E\left[\frac{z}{M} |x \right] - x = \frac{1}{M} E[z|x] - x \,. \end{aligned}$$

Now since z is drawn from a binomial random variable with parameters (M, x), the expectation of z is xM, from which we see that the above equals zero and our estimator is unbiased. To study the conditional variance of our error (defined as $e = \hat{x} - x$) consider

$$\begin{aligned} \sigma_e^2(x) &= E[(e - E[e])^2|x] = E[e^2|x] = E[(\hat{x} - x)^2|x] \\ &= E[\left(\frac{1}{M} z - x \right)^2 |x] = \frac{1}{M^2} E[(z - Mx)^2|x] \\ &= \frac{1}{M^2}(Mx(1-x)) = \frac{x(1-x)}{M} \,. \end{aligned} \tag{25}$$

In the above we have used the result that the variance of a binomial random variable with parameters (M, x) is $Mx(1-x)$. In developing a ML estimator x is not considered random and as such the above expression is the desired variance of our estimator.

To work Exercise 9 below, where we assume that x is a uniform random variable between $0 < x < 1$ we now compute the total variance of our estimator, C_e, we recall

$$\begin{aligned} C_e &= M_e - bb^T = M_e = E[M_e(x)] \\ &= E[b(x)b(x)^T + C_e(x)] = E[C_e(x)] \\ &= E[\frac{x(1-x)}{M}] = \frac{1}{M}E[x(1-x)]\,, \end{aligned}$$

where the expectation is taken over the randomness in x. Under the uniform random variable assumption for x, the above expression for, C_e, becomes

$$\frac{1}{M}E[x(1-x)] = \frac{1}{M}\int_0^1 x(1-x)dx = \frac{1}{6M}\,. \tag{26}$$

Exercise 9 (MMSE and MAP estimation of a binomial distribution)

For this exercise We are told that the prior distribution on x is uniform between 0 and 1. That is $p(x) = 1$ when $0 < x < 1$ and is 0 otherwise. After observing the measurement z (the number of success in M trials) we will have a posteriori estimate of x given by Bayes' rule

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)} = \frac{p(z|x)p(x)}{\int p(z|x)p(x)dx}\,.$$

From the information provided $p(z)$ can be calculated as

$$\begin{aligned} p(z) &= \int p(z|x)p(x)dx = \int_0^1 \left(\begin{array}{c} M \\ z \end{array} \right) x^z(1-x)^{M-z}1dx \\ &= \left(\begin{array}{c} M \\ z \end{array} \right) \int_0^1 x^z(1-x)^{M-z}dx\,. \end{aligned}$$

This will be evaluated with the help of the **Beta function** defined as

$$B(p,q) = \int_0^1 v^{p-1}(1-v)^{q-1}dv\,, \tag{27}$$

which can be evaluated when p and q have positive real parts to

$$B(p,q) = \frac{\Gamma(p)\Gamma(q)}{\Gamma(p+q)}\,, \tag{28}$$

where $\Gamma(p)$ is the **Gamma function** defined as

$$\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}dt\,. \tag{29}$$

When p and q are positive integers the above expression for the Gamma functions simplify and we obtain

$$\int_0^1 v^{p-1}(1-v)^{q-1}dv = B(p,q) = \frac{(p-1)!(q-1)!}{(p+q-1)!}\,. \tag{30}$$

In words this states that to evaluate the integral on the left we take the factorials of the powers ($p-1$ and $q-1$) of v directly in computing the numerator and then divide by the factorial of the sum of the two powers of v (plus one). With this background, our problem then gives for $p(z)$ the following expression

$$\begin{aligned} p(z) &= \left(\begin{array}{c} M \\ z \end{array} \right) B(z+1, M-z+1) = \left(\begin{array}{c} M \\ z \end{array} \right) \left(\frac{\Gamma(z+1)\Gamma(M-z+1)}{\Gamma(z+1+M-z+1)} \right) \\ &= \left(\frac{M!}{(M-z)!z!} \right) \left(\frac{z!(M-z)!}{(M+1)!} \right) = \frac{1}{M+1}\,, \end{aligned}$$

which in words states that the probability we obtain any measurement z, for $z \in 0, 1, \cdots, M$ is equally likely. Thus we have shown that

$$p(x|z) = \begin{cases} (M+1) \left(\begin{array}{c} M \\ z \end{array} \right) x^z (1-x)^{M-z} & 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

is the posteriori density $p(x|z)$ of x.

The MMSE estimator is given by the expectation of x under this density so $\hat{x}_{\text{MMSE}}(z) = E[x|z]$, and since we have an explicit expression for the density $p(x|z)$ we can compute this directly. We find

$$\begin{aligned} E[x|z] &= \int_0^1 x p(x|z) dx = (M+1) \left(\begin{array}{c} M \\ z \end{array} \right) \int_0^1 x^{z+1} (1-x)^{M-z} dx \\ &= (M+1) \left(\begin{array}{c} M \\ z \end{array} \right) \frac{(z+1)!(M-z)!}{(z+1+M-z+1)!} \\ &= \frac{z+1}{M+2}\,. \end{aligned} \tag{32}$$

The MAP estimator $\hat{x}_{\text{MAP}}$ for x is given by

$$\hat{x}_{\text{MAP}} = \text{argmax}_x p(x|z) = \text{argmax}_x p(z|x)p(x)\,.$$

Again, since we have an explicit representation for $p(x|z)$ we can find the maximum explicitly by taking the derivative of the expression $p(z|x)p(x)$ with respect to x, set the result equal to zero and solve for x. When we do this find the same solution as in Exercise 8 of $\hat{x}_{\text{MAP}}(z) = \frac{z}{M}$. From Exercise 8 we know that $\hat{x}_{\text{MAP}}(z)$ is unbiased and has a variance given by Equation 26.

For $\hat{x}_{\text{MMSE}}(z)$ we have a conditional bias given by

$$\begin{aligned} b(x) &= E[\hat{x} - x|x] = E[\hat{x}_{\text{MMSE}}(z)|x] - x \\ &= E\left[\frac{z+1}{M+2} | x \right] - x \\ &= \frac{1}{M+2}(E\left[z|x\right] + 1) - x \\ &= \frac{1}{M+2}(Mx+1) - x \\ &= \frac{1-2x}{M+2}\,. \end{aligned}$$

The total bias b is given by integrating this expression with respect to $p(x)$ i.e.

$$\begin{aligned} b &= \int b(x)p(x)dx = \int_0^1 \left(\frac{1-2x}{M+2}\right) dx \\ &= \frac{1}{M-2} - \frac{2}{M-1}\left(\frac{x^2}{2}\Big|_0^1\right. \\ &= \frac{1}{M-2} - \frac{1}{M-1} = 0\,, \end{aligned}$$

showing that this estimator $\hat{x}_{\rm MMSE}$ is unbiased.

The conditional variance of the error in our MMSE estimator is given by

$$\begin{aligned} C_e(x) &= E[(e - E[e])^2|x] = E[(\hat{x}(z) - (x + b(x)))^2|x] \\ &= E\left[\left(\frac{z+1}{M+2} - x - \frac{1-2x}{M+2}\right)^2 |x\right] \\ &= \left[\left(\frac{z}{M+2} - \frac{x(M+2)}{M+2} + \frac{2x}{M+2}\right)^2 |x\right] \\ &= \frac{1}{(M+2)^2} E[(z - xM)^2|x] \\ &= \frac{1}{(M+2)^2}(Mx(1-x))\,, \end{aligned}$$

using the known expression for the variance of a binomial random variable. Thus

$$M_e(x) = b(x)^2 + C_e(x) = \frac{(1-2x)^2}{(M+2)^2} + \frac{Mx(1-x)}{(M+2)^2}\,.$$

So we derive

$$\begin{aligned} M_e &= \int M_c(x)p(x)dx = \frac{1}{(M+2)^2}\left[\int_0^1 ((1-2x)^2 + Mx(1-x))dx\right] \\ &= \frac{1}{6(M+2)}\,, \end{aligned}$$

for the expression for the total variance of the MMSE estimator for x.

Exercise 10 (estimators for a Poisson distribution)

Part (a): Given the number of counts z is distributed as a Poisson random variable with rate λ we have

$$p(z|\lambda) = \frac{\lambda^z e^{-\lambda}}{z!} \quad \text{for} \quad z = 0, 1, 2, \cdots,$$

so that a maximum likelihood estimated for λ after having observed z counts is

$$\hat{\lambda}_{\rm ML}(z) = \text{argmax}_\lambda p(z|\lambda) = \text{argmax}_\lambda \left(\frac{\lambda^z e^{-\lambda}}{z!}\right)\,.$$

To find this maximum we take the derivative of $p(z|\lambda)$ with respect to λ, set the resulting expression equal to zero, and solve for λ. Taking the derivative we find

$$z\lambda^{z-1}e^{-\lambda} + \lambda^z(-e^{-\lambda}) = 0\,,$$

when we multiply by $\lambda^{-(z-1)}e^{\lambda}$ and solve for λ we get

$$\hat{\lambda}_{\rm ML} = z\,, \tag{33}$$

The conditional bias $b(\lambda)$ for the ML estimator is given by

$$b(\lambda) = E[\hat{\lambda}_{\rm ML}(z) - \lambda|\lambda] = E[z - \lambda|\lambda] = 0\,,$$

since the expectation of a Poisson random variable with parameter λ is given by λ. Thus we see that the ML estimator will be absolutely unbiased. To compute the variance of the ML estimator we first compute the conditional error variance $C_e(\lambda)$ as

$$\begin{aligned} C_e(\lambda) &= E[(e - E[e])^2|\lambda] \\ &= E[(\hat{\lambda}_{\rm ML}(z) - \lambda)^2|\lambda] = E[(z-\lambda)^2|\lambda] = \lambda\,, \end{aligned}$$

using the fact that the variance of a Poisson random variable with rate λ is λ. Then $M_e(\lambda) = b(\lambda)^2 + C_e(\lambda) = C_e(\lambda)$ as $b(\lambda)$ is zero. The total variance, C_e, of our estimator $\hat{\lambda}_{\rm ML}$ is given by

$$\begin{aligned} C_e &= M_e - bb^T = M_e = \int M_e(\lambda)p(\lambda)d\lambda \\ &= \int \lambda p(\lambda)d\lambda = \frac{1}{L}\int_0^L \lambda d\lambda = \frac{L}{2}\,. \end{aligned}$$

Part (b): The MAP estimation for λ is given by

$$\begin{aligned} \hat{\lambda}_{\rm MAP}(z) &= {\rm argmax}_\lambda p(\lambda|z) = {\rm argmax}_\lambda \frac{p(z|\lambda)p(\lambda)}{p(z)} \\ &= {\rm argmax}_\lambda \left(\frac{\lambda^z e^{-\lambda}}{z!}\frac{1}{L} \right)\,, \end{aligned}$$

which will have the same argument for its maximum as the ML estimate, but truncated at a finite value of L. That is

$$\hat{\lambda}_{\rm MAP}(z) = \begin{cases} z & z \le L \\ L & z > L \end{cases}\,. \tag{34}$$

From this expression, the conditional bias $b(\lambda)$ for the MAP estimator is given by

$$\begin{aligned} b(\lambda) &= E[\hat{\lambda}_{\rm MAP}(z) - \lambda|\lambda] = E[\hat{\lambda}_{\rm MAP}(z)|\lambda] - \lambda \\ &= \sum_{z=0}^{L} \hat{\lambda}_{\rm MAP}(z)P\{Z=z\} + \sum_{z=L+1}^{\infty} \hat{\lambda}_{\rm MAP}(z)P\{Z=z\} - \lambda \\ &= \sum_{z=0}^{L} zP\{Z=z\} + \sum_{z=L+1}^{\infty} LP\{Z=z\} - \lambda \\ &= \sum_{z=0}^{\infty} zP\{Z=z\} - \sum_{z=L+1}^{\infty} zP\{Z=z\} + \sum_{z=L+1}^{\infty} LP\{Z=z\} - \lambda \\ &= -\sum_{z=L+1}^{\infty} (z-L)P\{Z=z\}\,, \end{aligned}$$

Since $E[z|\lambda] = \sum_{z=0}^{\infty} zP\{Z = z\} = \lambda$. Now in this later sum $z > L$ so $z - L > 0$ showing that

$$-\sum_{z=L+1}^{\infty} (z - L)P\{Z = z\} < 0\,,$$

and never zero. Note that this sum does decrease in magnitude as we take the value of L larger. Then the expression for $b(\lambda)$ is never zero, and as $p(\lambda) > 0$ for all λ the total bias of the MAP estimator $b = \int b(\lambda)p(\lambda)d\lambda$ is a non-zero negative number, which shows that the ML estimator is biased.

Chapter 4: (State Estimation)

Notes on the Text

Linear-Gaussian state space models

For the given linear-Gaussian discrete system

$$x(i+1) = F(i)x(i) + L(i)u(i) + w(i)\,, \tag{35}$$

The expectation of $x(i)$ can be computed directly as $L(i)$ and $u(i)$ are deterministic as

$$E[x(i+1)] = F(i)E[x(i)] + L(i)u(i)\,. \tag{36}$$

Then computing $C_x(i+1)$ directly we find

$$\begin{aligned}
C_x(i+1) &= E\left[(x(i+1) - E[x(i+1)])(x(i+1) - E[x(i+1)])^T\right] \\
&= E\left[(F(i)(x(i) - E[x(i)]) + w(i))(F(i)(x(i) - E[x(i)]) + w(i))^T\right] \\
&= E\left[(F(i)(x(i) - E[x(i)]) + w(i))((x(i) - E[x(i)])^T F(i)^T + w(i)^T)\right] \\
&= E\left[F(i)(x(i) - E[x(i)])(x(i) - E[x(i)])^T F(i)^T\right] \\
&+ E\left[F(i)(x(i) - E[x(i)])w^T(i)\right] + E[w(i)(x(i) - E[x(i)])^T F(i)^T] \\
&+ E\left[w(i)w(i)^T\right] \\
&= F(i)C_x(i)F(i)^T + C_w(i)\,,
\end{aligned}$$

Since both middle terms $E[F(i)(x(i) - E[x(i)])w^T(i)]$ and $E[w(i)(x(i) - E[x(i)])^T F(i)^T]$ evaluate to zero. The two equations are the books equations 4.13.

Special State Space Models: The Random Walk

At the time i we have undergone $n(i)$ increments and $i - n(i)$ decrements since these are the only possibilities. Thus we will find ourself located at the position $x(i)$ given by

$$\begin{aligned}
x(i) &= dn(i) - d(i - n(i)) \\
&= 2dn(i) - di\,.
\end{aligned}$$

Now $n(i)$ has a binomial distribution with parameters $(i, \frac{1}{2})$ so has a mean $\frac{i}{2}$, so we find the expectation of $x(i)$ given by

$$\begin{aligned}
E[x(i)] &= 2dE[n(i)] - di \\
&= 2d\left(\frac{i}{2}\right) - di = 0\,.
\end{aligned}$$

The variance of our position, $x(i)$, is given by (since di is a constant at the ith timestep)

$$\begin{aligned}
\sigma_x^2(i) &= (2d)^2 \mathrm{Var}(n(i)) \\
&= 4d^2\left(\frac{1}{2}\right)\left(\frac{1}{2}\right) i = d^2 i\,.
\end{aligned}$$

Special State Space Models: Second Order Autoregressive Model

We consider a process $x(i)$ given by

$$x(i+1) = \alpha x(i) + \beta x(i-1) + w(i)\,,$$

We can compute the discrete Lyapunov equation form the vector state space representation for this process given in the book. From that representation the covariance matrix C_x (in steady-state) is given by

$$C_x = \begin{bmatrix} \sigma_x^2(\infty) & R_1 \\ R_1 & \sigma_x^2(\infty) \end{bmatrix}.$$

Where we have defined R_1 and r_1 as

$$R_1 \equiv \lim_{i\to\infty} E\left[(x(i+1) - E[x(i+1)])(x(i) - E[x(i)])\right] \equiv \sigma_x^2(\infty) r_1\,.$$

Using the shorthand $\sigma_x^2 \equiv \sigma_x^2(\infty)$ we find for the discrete Lyapunov equation 42 given by the following

$$\begin{bmatrix} \sigma_x^2 & \sigma_x^2 r_1 \\ \sigma_x^2 r_1 & \sigma_x^2 \end{bmatrix} = \begin{bmatrix} \alpha & \beta \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \sigma_x^2 & \sigma_x^2 r_1 \\ \sigma_x^2 r_1 & \sigma_x^2 \end{bmatrix} \begin{bmatrix} \alpha & 1 \\ \beta & 0 \end{bmatrix} + \begin{bmatrix} \sigma_w^2 & 0 \\ 0 & 0 \end{bmatrix}.$$

Dividing both sides by σ_x^2 and multiplying the three matrices on the right-hand side together we find

$$\begin{bmatrix} 1 & r_1 \\ r_1 & 1 \end{bmatrix} = \begin{bmatrix} \alpha^2 + 2\alpha\beta r_1 + \beta^2 & \alpha + \beta r_1 \\ \alpha + \beta r_1 & 1 \end{bmatrix} + \frac{\sigma_w^2}{\sigma_x^2} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}.$$

Equating the components corresponding to the $(1,2)$ element of the matrices on both sides of the above equation gives the following

$$r_1 = \alpha + \beta r_1 \quad \text{so} \quad r_1 = \frac{\alpha}{1-\beta}\,,$$

and is the same expression found in the book. Equating the components corresponding to the $(1,1)$ element of both sides gives the equation

$$1 = \alpha^2 + 2\alpha^2\beta r_1 + \beta^2 + \frac{\sigma_w^2}{\sigma_x^2}\,.$$

Which when we solve for σ_x^2 gives

$$\sigma_x^2 = \frac{\sigma_w^2}{1 - \alpha^2 - 2\alpha\beta r_1 - \beta^2}\,,$$

and can be shown to be equivalent to the expression for σ_x^2 presented in the book.

The derivation of the discrete algebraic Ricatti equation

Assuming convergence the Kalman Filtering (KF) equations the covariance estimates $C(i+1|i) = C(i|i-1)$ so putting the measurement induced covariance update equation

$$C(i|i) = C(i|i-1) - K(i)S(i)K(i)^T\,, \tag{37}$$

into the dynamic covariance update equation

$$C(i+1|i) = F(i)C(i|i)F(i)^T + C_w(i)\,, \tag{38}$$

and defining the limiting covariance matrix as P we have

$$\begin{aligned} P &= F(P - KSK^T)F^T + C_w \\ &= FPF^T + C_w - FKSK^TF^T \\ &= FPF^T + C_w - F(PH^TS^{-1})S(S^{-T}HP)F^T \\ &= FPF^T + C_w - FPH^T(HPH^T + C_v)^{-1}HPF^T\,, \end{aligned} \tag{39}$$

which is called the discrete Ricatti equation. Note we have used the definition of the Kalman matrix K and the innovation matrices S given by

$$S = HPH^T + C_v \tag{40}$$

$$K = PH^TS^{-1}\,. \tag{41}$$

Exercise Solutions

Exercise 1 (a state-space model of a random constant)

The given discrete system has $F = 1$ and $C_w = 0$ so the steady-state Lyapunov equation

$$C_x = FC_xF^T + C_w\,, \tag{42}$$

for this system becomes the identity statement

$$\sigma_x^2 = \sigma_x^2\,,$$

which implies that σ_x^2 can be anything. If we assume that the state estimate $\bar{x}(i|i)$ in the discrete Kalman filter converges to the unknown constant then the limiting steady-state covariance estimate must be zero.

Exercise 2 (the discrete Kalman filter for a random constant)

For this problem we specify the discrete Kalman filter equations given in the book in equations 4.27 to the specific scalar discrete-time model of a random constant given in this problem. For this discrete time system everything is a scalar, $F = 1$, $H = 1$, $C_v = \sigma_v^2$ and we find

$$\hat{z}(i) = \bar{x}(i|i-1) \tag{43}$$

$$s(i) = c(i|i-1) + \sigma_v^2 \tag{44}$$

$$k(i) = \frac{c(i|i-1)}{s(i)} = \frac{c(i|i-1)}{c(i|i-1) + \sigma_v^2} \tag{45}$$

$$\bar{x}(i|i) = \bar{x}(i|i-1) + k(i)(z(i) - \hat{z}(i)) \tag{46}$$

$$c(i|i) = c(i|i-1) - s(i)k(i)^2 = \sigma_v^2\left(\frac{c(i|i-1)}{c(i|i-1) + \sigma_v^2}\right) \tag{47}$$

After incorporating the measurement we then assign new estimates of our state x as equations 4.28

$$\begin{aligned} \bar{x}(i+1|i) &= \bar{x}(i|i) \\ c(i+1|i) &= c(i|i)\,. \end{aligned}$$

With $\bar{x}(i+1|i)$ and $c(i+1|i)$ specified we increment i and continue the iterative process with Equation 43.

We now explicitly iterate these equations for $i = 0, 1, 2$ and 3. Starting with the assumption that $\bar{x}(0|-1)$ and $c(0|-1)$ are initial state and covariance given as

$$\begin{aligned} \bar{x}(0|-1) &= E[x(0)|z(-1)] = E[x(0)|\{\cdot\}] = E[x(0)] = 0 \\ c(0|-1) &= \sigma^2_{x(0)} = +\infty\,. \end{aligned}$$

Then taking $i = 0$ and the initial conditions given above we find the measurement update steps become

$$\begin{aligned} \hat{z}(0) &= \bar{x}(0|-1) = 0 \\ s(0) &= c(0|-1) + \sigma_v^2 = +\infty \\ k(0) &= \frac{c(0|-1)}{s(0)} = \frac{c(0|-1)}{c(0|-1)+\sigma_v^2} = 1 \\ \bar{x}(0|0) &= 0 + 1(z(0) - 0) = z(0) \\ c(0|0) &= \sigma_v^2 \left(\frac{c(0|-1)}{c(0|-1)+\sigma_v^2} \right) = \sigma_v^2\,, \end{aligned}$$

and the state propagation update equations are given by

$$\begin{aligned} \bar{x}(1|0) &= \bar{x}(0|0) = z(0) \\ c(1|0) &= c(0|0) = \sigma_v^2\,. \end{aligned}$$

Incrementing our counter to $i = 1$ we follow the measurement update steps combined with the state propagation steps to obtain

$$\begin{aligned} \hat{z}(1) &= \bar{x}(1|0) = z(0) \\ s(1) &= c(1|0) + \sigma_v^2 = 2\sigma_v^2 \\ k(1) &= \frac{\sigma_v^2}{2\sigma_v^2} = \frac{1}{2} \\ \bar{x}(1|1) &= z(0) + \frac{1}{2}(z(1) - z(0)) = \frac{1}{2}(z(0) + z(1)) \\ c(1|1) &= \sigma_v^2 \left(\frac{\sigma_v^2}{2\sigma_v^2} \right) = \frac{1}{2}\sigma_v^2 \\ \bar{x}(2|1) &= \bar{x}(1|1) = \frac{1}{2}(z(0) + z(1)) \\ c(2|1) &= \frac{1}{2}\sigma_v^2\,. \end{aligned}$$

Incrementing our counter to $i = 2$ we follow the above steps to get

$$\begin{aligned}
\hat{z}(2) &= \frac{1}{2}(z(0) + z(1)) \\
s(2) &= \frac{1}{2}\sigma_v^2 + \sigma_v^2 = \frac{3}{2}\sigma_v^2 \\
k(2) &= \frac{1/2}{1 + 1/2} = \frac{1}{3} \\
\bar{x}(2|2) &= \frac{1}{2}(z(0) + z(1)) + \frac{1}{3}(z(2) - \frac{1}{2}(z(0) + z(1))) \\
&= \frac{1}{3}(z(0) + z(1) + z(2)) \\
c(2|2) &= \sigma_v^2 \left(\frac{1/2}{1/2 + 1}\right) = \frac{1}{3}\sigma_v^2 \\
\bar{x}(3|2) &= \frac{1}{3}(z(0) + z(1) + z(2)) \\
c(3|2) &= \frac{1}{3}\sigma_v^2 .
\end{aligned}$$

Finally, for $i = 3$ we obtain

$$\begin{aligned}
\hat{z}(3) &= \frac{1}{3}(z(0) + z(1) + z(2)) \\
s(3) &= \frac{4}{3}\sigma_v^2 \\
k(3) &= \frac{1/3}{1 + 1/3} = \frac{1}{4} \\
\bar{x}(3|3) &= \frac{1}{4}(z(0) + z(1) + z(2) + z(3)) \\
c(3|3) &= \sigma_v^2 \left(\frac{1/3}{1/3 + 1}\right) = \frac{1}{4}\sigma_v^2 \\
\bar{x}(4|3) &= \frac{1}{4}(z(0) + z(1) + z(2) + z(3)) \\
c(4|3) &= \frac{1}{4}\sigma_v^2 .
\end{aligned}$$

These equations can be solved in general, see the next problem.

Exercise 3 (the solution to the KF equations in exercise 2)

By mathematical induction, the general solution seems to be

$$\hat{z}(i) = \frac{1}{i}(z(0) + z(1) + \cdots + z(i-1)) \tag{48}$$

$$s(i) = \frac{i+1}{i}\sigma_v^2 \tag{49}$$

$$k(i) = \frac{1}{i+1} \tag{50}$$

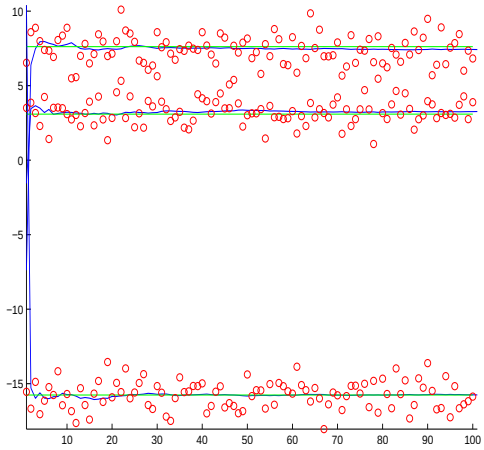

Figure 1: Three representative Kalman filtering results for Exercise 3, involving estimating a constant from noisy measurements. The x-axis represents is the iteration timestep (denoted in the text as i).

$$c(i|i-1) = \frac{1}{i}\sigma_v^2 \tag{51}$$

$$c(i|i) = \frac{1}{i+1}\sigma_v^2\,, \tag{52}$$

These expressions reproduce the results generated by iterating the Kalman Filter equations with $i = 0, 1, 2$ and 3 given above in exercise 2 as can be checked. To show that these give the general solution note that if Equation 51 is true then Equations 49, 50, and 52 follow from Equations 44, 45, and 47 respectively. Thus everything above is true if we are able to prove that Equation 51 is true. To do that we use Equation 48 to write it as

$$c(i+1|i) = \sigma_v^2 \left(\frac{c(i|i-1)}{c(i|i-1)+\sigma_v^2} \right)\,,$$

which is a first order difference equation for $c(i+1|i)$, considered as a function of the single argument i. The fact that Equation 51 *solves* this difference equation can be seen by directly substituting it in to the equation above and verifying that it indeed is a solution.

A MATLAB/OCTAVE implementation of the Kalman filter outlined above for this discrete system can be found in the script `ex_1_kalman_filter.m`. Three examples obtained when running this code are shown in Figure 1 where we represent the true constant we are attempting to estimate as a green line, the time varying estimate of this constant produced by the Kalman filter as a blue line and the observed random measurements as red circles. We note that the Kalman results converge quite nicely to the true constant values they are attempting to estimate. For many of the results the difference between the constant we are trying to estimate (the green curve) and our approximation of it (the blue curve) are too small to be distinguished by eye.

Exercise 4 (a scalar linear-Gaussian system)

The given system is a first order autoregressive model and the discrete steady-state Lyapunov equation 42 in this case is given by

$$\sigma_x^2 = \alpha\sigma_x^2\alpha + \sigma_w^2 \,.$$

Here $\sigma_x^2 \equiv \sigma_x^2(\infty)$. When we solve for σ_x^2 we find

$$\sigma_x^2 = \frac{\sigma_w^2}{1-\alpha^2} \,,$$

for the solution. The condition required for the existence of the solution is then that $\alpha \neq 1$. In addition, since $\sigma_x^2 > 0$ (variances are positive) we need to have $\alpha^2 < 1$ or $|\alpha| < 1$.

Exercise 5 (the steady-state covariance matrices)

When our system is operating in steady-state, the covariance matrix $C(i+1|i)$ is independent of time and its limiting value (denoted by P) must solve the discrete Ricatti Equation 39. For this scalar problem we have $F = \alpha$, $C_w = \sigma_w^2$, $H = 1$, and $C_v = \sigma_v^2$ so the Ricatti equation in this case becomes

$$P = \alpha^2 P + \sigma_w^2 - \alpha P(P + \sigma_v^2)^{-1} P\alpha \,,$$

or

$$P^2 + ((1-\alpha^2)\sigma_v^2 - \sigma_w^2)P - \sigma_w^2\sigma_v^2 = 0 \,.$$

Solving this quadratic equation we find

$$P = \frac{-((1-\alpha^2)\sigma_v^2 - \sigma_w^2) \pm \sqrt{((1-\alpha^2)\sigma_v^2 - \sigma_w^2)^2 + 4\sigma_v^2\sigma_w^2}}{2} \,. \tag{53}$$

Since P represents a variance we require that $P > 0$ and so we need to take the *positive* root in the above expression and we have the limiting value for $C(i|i-1)$ as $i \to \infty$. Using the above expression the limiting values for $S(i)$, $K(i)$, and $C(i|i)$ are given by

$$\begin{aligned} S &= HPH^T + C_v = P + \sigma_v^2 \\ &= \frac{((1+\alpha^2)\sigma_v^2 + \sigma_w^2) + \sqrt{((1-\alpha^2)\sigma_v^2 - \sigma_w^2)^2 + 4\sigma_v^2\sigma_w^2}}{2} \\ K &= PH^T S^{-1} \\ &= \frac{-((1-\alpha^2)\sigma_v^2 - \sigma_w^2) + \sqrt{((1-\alpha^2)\sigma_v^2 - \sigma_w^2)^2 + 4\sigma_v^2\sigma_w^2}}{((1+\alpha^2)\sigma_v^2 + \sigma_w^2) + \sqrt{((1-\alpha^2)\sigma_v^2 - \sigma_w^2)^2 + 4\sigma_v^2\sigma_w^2}} \end{aligned}$$

Finally since KSK^T is given by

$$KSK^T = \frac{1}{2}\left(\frac{((1-\alpha^2)\sigma_v^2 - \sigma_w^2) - \sqrt{((1-\alpha^2)\sigma_v^2 - \sigma_w^2)^2 + 4\sigma_v^2\sigma_w^2}}{((1+\alpha^2)\sigma_v^2 - \sigma_w^2) + \sqrt{((1-\alpha^2)\sigma_v^2 - \sigma_w^2)^2 + 4\sigma_v^2\sigma_w^2}}\right) \,,$$

the limiting value for $C(i|i)$ is given by the difference of P and KSK^T as

$$C(i|i) = P - KSK^T \,.$$

Exercise 6 (iterating the KF for a first order autoregressive model)

For this problem we consider the discrete Kalman filter equations given in the book in equations 4.27 as applied to the specific scalar first order autoregressive discrete-time model given in this problem. For the given discrete time model everything is a scalar, $F = \alpha$, $H = 1$, $C_v = \sigma_v^2$, $C_w = \sigma_w^2$ and we find

$$\begin{aligned}
\hat{z}(i) &= \bar{x}(i|i-1) && (54)\\
s(i) &= c(i|i-1) + \sigma_v^2 && (55)\\
k(i) &= \frac{c(i|i-1)}{s(i)} = \frac{c(i|i-1)}{c(i|i-1)+\sigma_v^2} && (56)\\
\bar{x}(i|i) &= \bar{x}(i|i-1) + k(i)(z(i) - \hat{z}(i)) && (57)\\
c(i|i) &= c(i|i-1) - s(i)k(i)^2 = \sigma_v^2 \left(\frac{c(i|i-1)}{c(i|i-1)+\sigma_v^2} \right) && (58)
\end{aligned}$$

Note that these are the *same* equations as in Exercise 2 above. The difference between this problem and the previous one comes when we assign a new estimates of our state x using the dynamic equations 4.28 as

$$\begin{aligned}
\bar{x}(i+1|i) &= \alpha \bar{x}(i|i) && (59)\\
c(i+1|i) &= \alpha^2 c(i|i) + \sigma_w^2 \,, && (60)
\end{aligned}$$

With $\bar{x}(i+1|i)$ and $c(i+1|i)$ specified as above we increment our counter variable i and continue the iterative process with Equation 54.

We will now explicitly iterate these equations for $i = 0, 1, 2$ and 3. Starting with the assumption that $\bar{x}(0|-1)$ and $c(0|-1)$ are initial state and covariance given as

$$\begin{aligned}
\bar{x}(0|-1) &= E[x(0)|z(-1)] = E[x(0)|\{\cdot\}] = E[x(0)] = 0\\
c(0|-1) &= \sigma_{x(0)}^2 = +\infty \,.
\end{aligned}$$

Then taking $i = 0$ and using the initial conditions given above we find the measurement update steps become

$$\begin{aligned}
\hat{z}(0) &= \bar{x}(0|-1) = 0\\
s(0) &= c(0|-1) + \sigma_v^2 = +\infty\\
k(0) &= \frac{c(0|-1)}{s(0)} = \frac{c(0|-1)}{c(0|-1)+\sigma_v^2} = 1\\
\bar{x}(0|0) &= 0 + 1(z(0) - 0) = z(0)\\
c(0|0) &= \sigma_v^2 \left(\frac{c(0|-1)}{c(0|-1)+\sigma_v^2} \right) = \sigma_v^2 \,,
\end{aligned}$$

and the state propagation update equations given by

$$\begin{aligned}
\bar{x}(1|0) &= \alpha \bar{x}(0|0) = \alpha z(0)\\
c(1|0) &= \alpha^2 c(0|0) + \sigma_w^2 = \alpha^2 \sigma_v^2 + \sigma_w^2 \,.
\end{aligned}$$

Incrementing our counter to $i = 1$ we follow the measurement update steps combined with the state propagation steps to obtain

$$\begin{aligned}
\hat{z}(1) &= \bar{x}(1|0) = \alpha z(0) \\
s(1) &= c(1|0) + \sigma_v^2 = (1+\alpha^2)\sigma_v^2 + \sigma_w^2 \\
k(1) &= \frac{c(1|0)}{c(1|0)+\sigma_v^2} = \frac{\alpha^2\sigma_v^2 + \sigma_w^2}{(1+\alpha^2)\sigma_v^2 + \sigma_w^2} \\
\bar{x}(1|1) &= \bar{x}(1|0) + k(1)(z(1) - \hat{z}(1)) \\
&= \alpha z(0) + \left(\frac{\alpha^2\sigma_v^2 + \sigma_w^2}{(1+\alpha^2)\sigma_v^2 + \sigma_w^2}\right)(z(1) - \alpha z(0)) \\
&= \alpha\left(\frac{\sigma_v^2}{(1+\alpha^2)\sigma_v^2 + \sigma_w^2}\right) z(0) + \left(\frac{\alpha^2\sigma_v^2 + \sigma_w^2}{(1+\alpha^2)\sigma_v^2 + \sigma_w^2}\right) z(1) \\
c(1|1) &= \sigma_v^2\left(\frac{\alpha^2\sigma_v^2 + \sigma_w^2}{(1+\alpha^2)\sigma_v^2 + \sigma_w^2}\right) \\
\bar{x}(2|1) &= \alpha\bar{x}(1|1) = \left(\frac{\alpha^2\sigma_v^2}{(1+\alpha^2)\sigma_v^2 + \sigma_w^2}\right) z(0) + \left(\frac{\alpha^2\sigma_v^2 + \sigma_w^2}{(1+\alpha^2)\sigma_v^2 + \sigma_w^2}\right)\alpha z(1) \\
c(2|1) &= \alpha^2\sigma_v^2\left(\frac{\alpha^2\sigma_v^2 + \sigma_w^2}{(1+\alpha^2)\sigma_v^2 + \sigma_w^2}\right) + \sigma_w^2 = \frac{\alpha^4\sigma_v^4 + (1+2\alpha^2)\sigma_v^2\sigma_w^2 + \sigma_w^4}{(1+\alpha^2)\sigma_v^2 + \sigma_w^2}\,.
\end{aligned}$$

At this point the algebra in these iterations starts to get tedious. Moving to Mathematica, in the script `ex_8_algebra.nb`, we implement the remaining iterations. The results are more complicated but in principle one could iterate and compute $\hat{z}(i)$, $s(i)$, $k(i)$, $\bar{x}(i|i)$, $c(i|i)$, $\bar{x}(i+1|i)$, and $c(i+1|i)$ to any desired order.

Exercise 7 (solving the discrete autoregressive KF equations)

For the first order autoregressive model given above, if we can explicitly compute a functional form for $c(i+1|i)$, then we can also explicitly compute $s(i)$, $k(i)$, and $c(i|i)$ using equations, 55, 56, and 58 respectively. Thus it seems valid to try to first determine $c(i+1|i)$. We begin by observing that a difference equation for $c(i+1|i)$, can be obtained using Equation 58 and 60 as follows

$$\begin{aligned}
c(i+1|i) &= \alpha^2 c(i|i) + \sigma_w^2 \\
&= \sigma_v^2\alpha^2\left(\frac{c(i|i-1)}{c(i|i-1)+\sigma_v^2}\right) + \sigma_w^2\,.
\end{aligned}$$

It is this difference equation we will solve for the functional form for $c(i+1|i)$. In the above expression define $d(i) \equiv \frac{c(i+1|i)}{\sigma_v^2}$, so that the difference equation for the unknown $d(i)$ then becomes

$$d(i+1) = \alpha^2\left(\frac{d(i)}{d(i)+1}\right) + r^2\,, \tag{61}$$

where we have defined r^2 as the ratio of variances $r^2 = \frac{\sigma_w^2}{\sigma_v^2}$, and $d(i)$ has an initial condition given by

$$d(0) = \frac{c(1|0)}{\sigma_v^2} = \frac{\alpha^2\sigma_v^2 + \sigma_w^2}{\sigma_v^2} = \alpha^2 + r^2\,.$$

To solve Equation 61, we multiply both sides by the expression $d(i)+1$ to obtain

$$d(i+1)d(i) + d(i+1) - \alpha^2 d(i) - r^2 = 0\,,$$

which is in the standard form of a **Ricatti difference equation** [6]. To solve this equation we derive an associated *linear* equation for a new function $z(i)$ (unrelated to any measurement function) when we use the substitution

$$d(i) = \frac{z(i+1)}{z(i)} - 1\,. \tag{62}$$

We find the function $z(i)$ must satisfy

$$z(i+2) + (-\alpha^2 - 1)z(i+1) + (-r^2 + \alpha^2)z(i) = 0\,.$$

Which is a second order *linear* difference equation for $z(i)$. The characteristic roots of this equation are given by solving the following quadratic equation for the root x

$$x^2 - (\alpha^2+1)x + (-r^2+\alpha^2) = 0\,. \tag{63}$$

Thus we find two roots given by $x_\pm$ of

$$\begin{aligned} x_\pm &= \frac{\alpha^2 + 1 \pm \sqrt{(\alpha^2+1)^2 + 4(r^2-\alpha^2)}}{2} \\ &= \frac{\alpha^2 + 1 \pm \sqrt{(\alpha^2-1)^2 + 4r^2}}{2}\,. \end{aligned} \tag{64}$$

With these two roots, $x_\pm$, the solution for $z(i)$ is thus given by

$$z(i) = Ax_+^i + Bx_-^i\,,$$

in terms of two as yet unknown constants A and B. From this and Equation 62, $d(i)$ is then given by

$$\begin{aligned} d(i) &= \frac{Ax_+^{i+1} + Bx_-^{i+1}}{Ax_+^i + Bx_-^i} - 1 \\ &= \frac{x_+^i(x_+ - 1) + Cx_-^i(x_- - 1)}{x_+^i + Cx_-^i}\,, \end{aligned} \tag{65}$$

where we have assumed that $A \neq 0$ and then divided the top and bottom of the above fraction by it, and finally defined $C = \frac{B}{A}$. The initial condition on $d(i)$ requires that C is determined by

$$\begin{aligned} d(0) &= \frac{x_+ - 1 + C(x_- - 1)}{1+C} = \alpha^2 + r^2 \quad \text{or} \\ C &= -\left(\frac{\alpha^2 + r^2 - x_+ + 1}{\alpha^2 - r^2 - x_- + 1}\right)\,. \end{aligned} \tag{66}$$

Where it should be remembered that $x_\pm$ are both known functions of r and α given by Equation 64. From this explicit solution for $d(i)$ given by Equations 65 and 66 we can derive an explicit solution for our a-priori variance $c(i|i-1)$ given by

$$c(i|i-1) = \sigma_v^2 d(i-1)\,.$$

Thus given the explicit form of the first order autoregressive model the above formulation provides an explicit solution for all the quantities of interest.

Note we can also obtain an explicit difference equation for the a-priori mean, $\bar{x}(i|i-1)$, in the case of a first order autoregressive model as follows

$$\begin{aligned}
\bar{x}(i+1|i) &= \alpha\bar{x}(i|i) \\
&= \alpha\bar{x}(i|i-1) + \alpha k(i)(z(i) - \hat{z}(i)) \\
&= \alpha\bar{x}(i|i-1) + \alpha\left(\frac{c(i|i-1)}{c(i|i-1)+\sigma_v^2}\right)(z(i) - \hat{x}(i|i-1)) \\
&= \alpha\left(\frac{\sigma_v^2}{c(i|i-1)+\sigma_v^2}\right)\bar{x}(i|i-1) + \alpha\left(\frac{c(i|i-1)}{c(i|i-1)+\sigma_v^2}\right)z(i)\,.
\end{aligned}$$

This is a first-order linear difference equation with known coefficients for the unknown a-priori mean $\bar{x}(i|i-1)$. These facts allow an explicit solution to be written down for $\bar{x}(i|i-1)$ in terms of the known functions $c(i|i-1)$ and $z(i)$. See reference [6] for a detailed description of the solution to difference equations like this.

Exercise 8 (a second order autoregressive model)

Following the discussion in the book, for a second order autoregressive processes we can define the system state vector $\mathbf{x}(i+1)$ as

$$\mathbf{x}(i+1) = \begin{bmatrix} x(i+1) \\ x(i) \end{bmatrix}.$$

Then the given linear state equation or linear plant equation for the given discrete time system can be written in terms of the vector state $\mathbf{x}$ as

$$\mathbf{x}(i+1) = \begin{bmatrix} x(i+1) \\ x(i) \end{bmatrix} = \begin{bmatrix} 1/2 & 1/4 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} x(i) \\ x(i-1) \end{bmatrix} + \begin{bmatrix} w(i) \\ 0 \end{bmatrix}.$$

The measurement equation for this system, in terms of the discrete vector state $\mathbf{x}$, is given by

$$\begin{aligned}
z(i) &= x(i) + v(i) \quad \text{or} \\
&= \begin{bmatrix} 1 & 0 \end{bmatrix}\begin{bmatrix} x(i) \\ x(i-1) \end{bmatrix} + v(i) = \begin{bmatrix} 1 & 0 \end{bmatrix}\mathbf{x}(i) + v(i)\,.
\end{aligned}$$

The discrete Lyapunov Equation 42 that the covariance of $\mathbf{x}$ must satisfy in steady-state for this system is then given by

$$\begin{bmatrix} \sigma_x^2 & \sigma_x^2 r_1 \\ \sigma_x^2 r_1 & \sigma_x^2 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/4 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} \sigma_x^2 & \sigma_x^2 r_1 \\ \sigma_x^2 r_1 & \sigma_x^2 \end{bmatrix}\begin{bmatrix} 1/2 & 1 \\ 1/4 & 0 \end{bmatrix} + \sigma_w^2\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}.$$

As this expression is the exact *same* functional form as the discrete Lyapunov equation model for the second order autoregressive process discussed in the book and considered on Page 21

we can just use the results from that section. We find

$$\begin{aligned} r_1 &= \frac{1/2}{1-1/4} = \frac{2}{3} = 0.66667 \\ r_2 &= \frac{1/4+1/4-1/16}{1-1/4} = \frac{7}{12} = 0.58333 \\ \sigma_x^2 &= \frac{1}{1-1/2(2/3)-1/4(7/12)} = 1.92\,. \end{aligned}$$

Thus the covariance matrix of our state, C_x, when in steady state is given by

$$C_x = \begin{bmatrix} 1.92 & 1.28 \\ 1.28 & 1.92 \end{bmatrix}.$$

These simple numerical calculations are done in the MATLAB script `ex_8.m`.

To find the long term limiting values for the innovations matrix, S, and the Kalman gain matrix, K, we will use Equations 40 and 41. We thus need to find the limiting values for P or $\lim_{i\to\infty} C(i|i-1)$ by using the discrete Ricatti Equation 39. To numerically solve for P in Equation 39 we will use a simple iterative scheme where we take $P^{(0)} = C_x$ (computed above) and iterate the following equation

$$P^{(n+1)} = FP^{(n)}F^T + C_w - FP^{(n)}H^T(HP^{(n)}H^T + C_v)^{-1}HP^{(n)}F^T\,.$$

We find $P \equiv P^{(\infty)}$ using this method in the script `ex_8.m` by

$$P = \begin{bmatrix} 1.2029 & 0.3080 \\ 0.3080 & 0.5461 \end{bmatrix}.$$

A MATLAB implementation of the Kalman filter for this discrete dynamical system can be found in the script `ex_8_kalman_filter.m`. Some results obtained when we run this script can be seen in Figure 2. In Figure 2 (left) we show a single filtering run, where we have taken a measurement noise variance of $\sigma_v^2 = 1$. In Figure 2 (right) we show three additional runs, shifted upwards from the origin by a constant amount (for visibility) and with noise variances σ_v^2 given by $1/2$ (for the bottom), 1 (for the middle), and $3/2$ (for the top). We see that the Kalman filter estimate does quite a good job tracking the truth value (drawn in green).

Exercise 9 (a moving average model)

For this exercise, if we define the state vector $\mathbf{x}(i)$ as simply the scalar $x(i)$ and define a lagged noise vector $\mathbf{w}(i)$ as $\mathbf{w}(i) = \begin{bmatrix} w(i) \\ w(i-1) \end{bmatrix}$, we then have a dynamic equation in state-space form given by

$$x(i+1) = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} w(i) \\ w(i-1) \end{bmatrix} \equiv G(i)\mathbf{w}(i)\,. \tag{67}$$

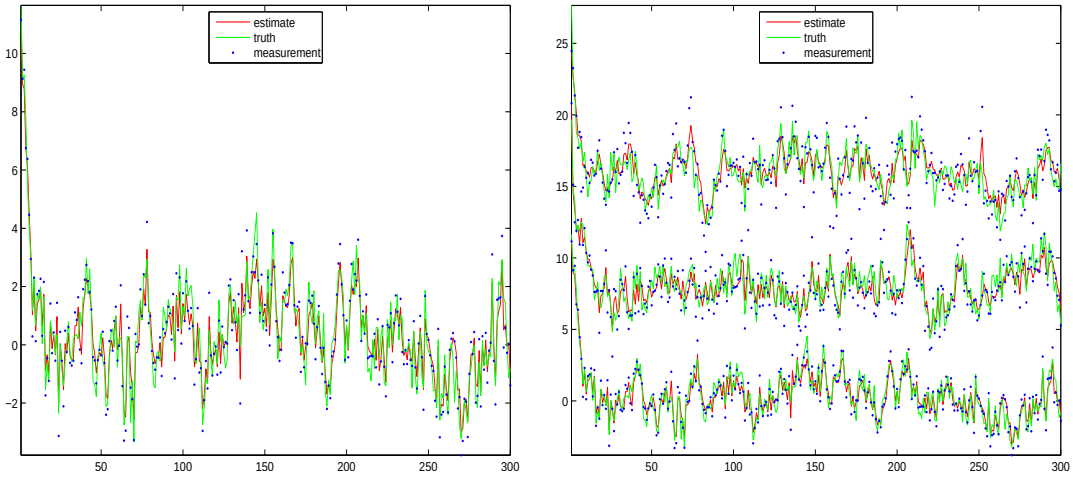

Figure 2: **Left:** A single representative Kalman filtering result for estimating the state $x(i)$ for a second order autoregressive process from noisy measurements. This is the discrete-time dynamical system described in Exercise 8. We plot the Kalman filtered computed estimate, $\bar{x}(i|i)$, along with the true value of $x(i)$ at the i-th step. The x-axis represents is the iteration timestep (denoted in the text as i). **Right:** Multiple examples of applying Kalman filtering to the second order autoregressive model. Note the oscillatory nature of the truth curve which is a characteristic of second order autoregressive systems.

Note that this is a slightly modified form of the dynamic state equation given in the text, in that it has a matrix $G(i) \equiv \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \end{bmatrix}$ multiplying the noise vector $\mathbf{w}(i)$, where as in the text we have the identity matrix. Notice that we have written the matrix G as a function of i (even thought in this specific dynamical system it is actually time independent) to indicate that the following results hold more generally for a time dependent G. To incorporate this generalization of the dynamic model, the only equation in the discrete Kalman filtering algorithm that is modified is the covariance propagation equation which was

$$C(i+1|i) = F(i)C(i|i)F(i)^T + C_w(i)\,.$$

This equation will now become

$$C(i+1|i) = F(i)C(i|i)F(i)^T + G(i)C_w(i)G(i)^T\,, \tag{68}$$

Note the presence of the $G(i)C_w(i)G(i)^T$ term, see the reference [4]. A MATLAB implementation of the Kalman filter for this discrete dynamical system can be found in the script `ex_9_kalman_filter.m`. Some results obtained when we run this script can be seen in Figure 3. In that figure we show a single filtering run, where we have taken a measurement noise variance of $\sigma_v^2 = 1$. Running experiments with this code is a good way to observe that the Kalman filter doing quite a good job tracking the truth state value $x(i)$ (drawn as a line in green).

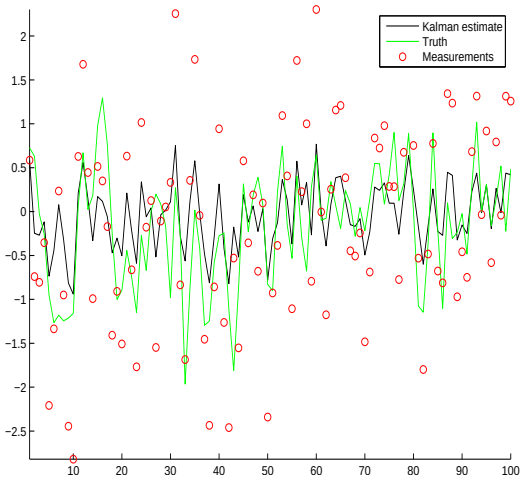

Figure 3: A single representative Kalman filtering result for estimating the state $x(i)$ for a first order moving average process. This is the discrete-time dynamical system described in Exercise 9. We plot the Kalman filtered computed estimate, $\bar{x}(i|i)$, (in black) along with the true value of $x(i)$ (in green) at the i-th step. The x-axis represents is the iteration timestep (denoted in the text as i). The noisy measurements are plotted as red circles.

Exercise 10 (a combined autoregressive/moving average model)

Following the discussion in the book and the two exercises above we will define the system state vector $\mathbf{x}(i+1)$ as

$$\mathbf{x}(i+1) = \begin{bmatrix} x(i+1) \\ x(i) \end{bmatrix},$$

and a lagged noise vector $\mathbf{w}(i)$ as $\mathbf{w}(i) = \begin{bmatrix} w(i) \\ w(i-1) \end{bmatrix}$. With these definitions we then have a dynamic equation for $\mathbf{x}(i)$ in state-space form given by

$$\begin{aligned} \mathbf{x}(i+1) &= \begin{bmatrix} x(i+1) \\ x(i) \end{bmatrix} \\ &= \begin{bmatrix} 1/2 & 1/2 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x(i) \\ x(i-1) \end{bmatrix} + \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} w(i) \\ w(i-1) \end{bmatrix} \\ &= \begin{bmatrix} 1/2 & 1/2 \\ 1 & 0 \end{bmatrix} \mathbf{x}(i) + \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{w}(i) . \end{aligned} \tag{69}$$

The measurement equation for this system, in terms of the discrete vector state $\mathbf{x}$, is given by exactly the same expression in exercise 8 of

$$z(i) = \begin{bmatrix} 1 & 0 \end{bmatrix} \mathbf{x}(i) + v(i) .$$

Thus with the process noise modification given in Exercise 9 the above, this discrete system has process and measurement matrices given by $F = \begin{bmatrix} 1/2 & 1/2 \\ 1 & 0 \end{bmatrix}$, $G = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}$, $H = \begin{bmatrix} 1 & 0 \end{bmatrix}$, $C_w = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and $C_v = \sigma_v^2$. From this discussion a MATLAB implemen-

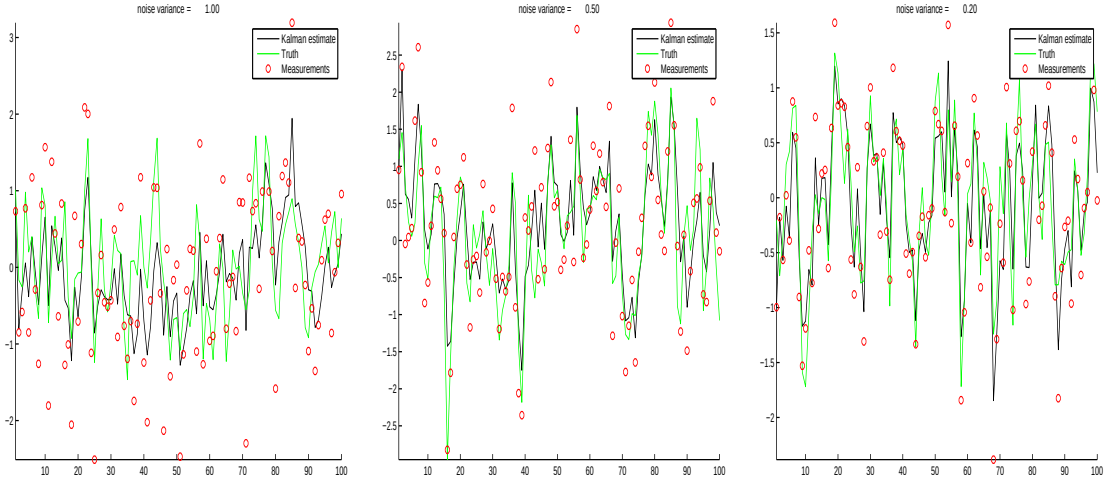

Figure 4: Some representative Kalman filtering results for estimating the state $x(i)$ for a ARMA(2,1) process. This is the discrete-time dynamical system described in Exercise 10. We plot the Kalman filtered computed estimate, $\bar{x}(i|i)$, (in black) along with the true value of $x(i)$ (in green) at the i-th step. The x-axis represents is the iteration timestep (denoted in the text as i). The noisy measurements are plotted as red circles. **Left:** With a noise variance $\sigma_v^2 = 1.0$. **Middle:** With a noise variance $\sigma_v^2 = 0.5$. **Right:** With a noise variance $\sigma_v^2 = 0.2$. Note that as the noise variance is reduced the filter results get better in that the black and green lines approach each other.

tation of the Kalman filter for this discrete dynamical system can be found in the script `ex_10_kalman_filter.m`. Some results obtained when we run this script can be seen in Figure 4.

Exercise 11 (discrete KF implementations)

MATLAB/OCTAVE implementations of the discrete Kalman filters for the various discrete systems represented in the exercises can be found in the scripts `ex_1_kalman_filter.m`, `ex_8_kalman_filter.m`, `ex_9_kalman_filter.m`, and `ex_10_kalman_filter.m`.

Chapter 5: (Supervised Learning)

Notes on the Text

The error associated with the elements of the covariance matrix

Given that we know the distribution of the random variables in $\hat{C}_k$ is given by a *Wishart* distribution and that the *variances* of each matrix element could be computed using

$$\mathrm{Var}[\hat{C}_{k_{i,j}}] = \frac{1}{N_k}(C_{k_{i,i}}C_{k_{j,j}} - C^2_{k_{i,j}})\,, \tag{70}$$

we could substitute the *estimated* values of $C_{k_{i,j}}$ into the right-hand-side of this formula to ensure that (or warn otherwise) the elements of our covariance matrix are computed accurately enough. That is we could estimate the relative error, η_{ij}, of the ij-th element of our covariance matrix as

$$\begin{aligned}\eta_{ij} &\equiv \frac{\sqrt{\mathrm{Var}[\hat{C}_{k_{i,j}}]}}{\hat{C}_{k_{i,j}}} = \frac{\sqrt{\frac{1}{N_k}(\hat{C}_{k_{i,i}}\hat{C}_{k_{j,j}} + \hat{C}^2_{k_{i,j}})}}{\hat{C}_{k_{i,j}}} \\ &= \frac{1}{\sqrt{N_k}}\sqrt{\left(\frac{\hat{C}_{k_{i,i}}\hat{C}_{k_{j,j}}}{\hat{C}^2_{k_{i,j}}} + 1\right)}\,.\end{aligned}$$

If these relative errors η_{ij} are too large relative to a fixed threshold then one needs to obtain more samples from class k.

Notes on regularizing the covariance matrix

In this section of the text several regularization techniques are discussed for reducing the sensitivity of of $\hat{C}$ to statistical errors. The first modifies $\hat{C}$ by adding a multiple of the identity as

$$\hat{C}_{\text{regularized}} = (1-\gamma)\hat{C} + \gamma\frac{\mathrm{trace}(\hat{C})}{N}I\,. \tag{71}$$

Here γ represents the amount of regularization to apply and is confined to be $0 \le \gamma \le 1$. Note that for all possible values of γ, the trace of $\hat{C}_{\text{regularized}}$ is equal to that of $\hat{C}$ and hence the regularized matrix has the same sum of the eigenvalues as the original matrix $\hat{C}$. When $\gamma = 0$ no regularization is applied, while when when $\gamma = 1$ we replace $\hat{C}$ with a diagonal matrix. This regularization technique is implemented in the C++ code `trace_regularization.cpp`.

An alternative regularization technique suggested in the book is to suppress the off diagonal elements of $\hat{C}$ by some factor. If again we let γ be such that $0 \le \gamma \le 1$ then this procedure is simply

$$\hat{C}_{\text{regularized},ij} = (1-\gamma)\hat{C}_{ij} \quad \text{for} \quad i \neq j\,. \tag{72}$$

As in the previous technique above $\gamma = 0$ is no regularization and $\gamma = 1$ corresponds to a diagonal matrix. This regularization technique is implemented in the C++ code `scale_regularization.cpp`.

An additional technique that is not discussed in the book but which might be helpful in practice is to compute the correlation matrix R from the estimated covariance matrix $\hat{C}$ by computing a matrix with elements

$$R_{ij} = \frac{\hat{C}_{ij}}{\sqrt{\hat{C}_{ii}\hat{C}_{jj}}} .$$

Since these numbers are estimates of the Pearson correlation between the ith and j features we can use statistical hypothesis testing to determine if this sample based estimate of the population correlation is sufficiently large to reject the hypothesis $H_0 : \rho = 0$. Pairs of features that don't reject this hypothesis are set equal to zero. Hypothesis testing of correlations is discussed in Chapter 8 of [8] and this hypothesis testing based regularization procedure is implemented in the C++ file `ht_regularization.cpp`. Again there is a regularization parameter γ that controls how many elements are set equal to zero. A modification (not implemented) of the above scheme would be to compute the p-values for each entry in the R matrix and then set the threshold for zeroing elements based on these p-values. I have not seen this method discussed else where. If anyone knows any references to such a procedure or has seen anything similar in the literature I would be interested in hearing about it.

Estimation of Prior Probabilities $P(\omega_k)$

Since the number N_k follows a multinomial distribution with parameters $(P(\omega_1), P(\omega_2), \cdots, P(\omega_k); N_S)$ the estimator of $P(\omega_k)$ denoted $\hat{P}(\omega_k)$ and defined as

$$\hat{P}(\omega_k) = \frac{N_k}{N_S} , \tag{73}$$

has an expectation given by

$$E[\hat{P}(\omega_k)] = \frac{1}{N_S} E[N_k] = \frac{N_S P(\omega_k)}{N_S} = P(\omega_k) ,$$

showing that the estimate $\hat{P}(\omega_k) = \frac{N_k}{N_S}$ is unbiased. The estimate $\hat{P}(\omega_k) = \frac{N_k}{N_S}$ has a variance given by

$$\mathrm{Var}[\hat{P}(\omega_k)] = \frac{1}{N_S^2} \mathrm{Var}[N_k] .$$

Since we can view N_k as a draw from a binomial distribution with parameters $(P(\omega_k), N_S)$ in this light it has a variance given by $\mathrm{Var}[N_k] = N_S P(\omega_k)(1 - P(\omega_k))$, so that the above expression becomes

$$\mathrm{Var}[\hat{P}(\omega_k)] = \frac{P(\omega_k)(1 - P(\omega_k))}{N_S} , \tag{74}$$

which is the books equation 5.19.

To get an order of magnitude estimate of the number of samples N_S required to estimate $P(\omega_k)$ with sufficient precision, we will determine N_S such that the percent error (or relative error) in our estimate of $P(\omega_k)$ is significantly small. Thus we desire

$$\frac{\sqrt{\text{Var}[\hat{P}(\omega_k)]}}{P(\omega_k)} \approx \eta \,, \tag{75}$$

with the value of $\eta \ll 1$. Using Equation 74 the above expression is equivalent to

$$N_S \approx \frac{1 - P(\omega_k)}{\eta^2 P(\omega_k)} \,. \tag{76}$$

Thus if we anticipate that some class, k has $P(\omega_k) = 0.01$ and we want to enforce a relative error of $\eta = 0.2$ (or 20 %) using Equation 76 we find $N_S \approx 2475$ required samples. Note that Equation 76 can be used to signal classes for which we may not have enough samples to estimate their probabilities sufficiently accurately. That is, one begins by computing $\hat{P}(\omega_k)$ from the unbiased estimate $\frac{N_k}{N_S}$ and $\text{Var}[\hat{P}(\omega_k)]$ from Equation 74 using the estimate $\hat{P}(\omega_k)$ for the population parameter $P(\omega_k)$. Next estimate the relative error η in the estimate of $P(\omega_k)$ using Equation 75. If this value is too large relative to the accuracy required then more samples N_S are needed.

Binary measurements

Problems using the estimator in Equation 73 can occur if we have a very large number k of classes to estimate probabilities for. As an example, if we assume that we have a state dimension of $N = 10$, then we have $2^{10} = 1024$ possible discrete measurements z and we expect our probabilities

$$P(z|\omega_k) = O(1/1024) = O(10^{-3}) \,.$$

To have a relative error of 10 % in our estimates of $\hat{P}(z|\omega_k)$ requires $\eta = 0.1$, using Equation 76 requires

$$N_k(z) = \frac{1 - 10^{-3}}{(0.1)^2(10^{-3})} \approx 10^5 \,,$$

which may be too large for many applications. A solution to this difficulty is to take as a *prior* estimate that $P(z|\omega_k) = 2^{-N}$ and then using the fact that we observe $N_k(z)$ elements that have feature measurements z from a set of $N_k = \sum_z N_k(z)$ elements to compute a *posteriori* estimate of $P(z|\omega_k)$. To derive the expression presented in the book we need to understand how to perform Bayesian estimation on the parameters in multinomial models (discussed in the next paragraph).

As some background to estimating parameters in a multinomial model, the reference [3] provides a description of how to perform Bayesian estimation using multinomial models. There it is discussed that the **conjugate prior** of a multinomial distribution is the **Dirichlet** distribution. Using the notation in that reference, this means that if the prior distribution of the parameters in a multinomial distribution are distributed as a Dirichlet distribution i.e.

$$\theta \sim \text{Dirichlet}(\alpha_1, \alpha_2, \cdots, \alpha_k) \,,$$

then the *posteriori* distribution of the parameters of our multinomial distribution $\theta|\mathbf{y}$ after observing y_i objects of type i from a total of $N = \sum_i y_i$ is given by another Dirichlet distribution such that

$$\theta|\mathbf{y} \sim \text{Dirichlet}(y_1 + \alpha_1, y_2 + \alpha_2, \cdots, y_k + \alpha_k)\,.$$

From this functional form for the posteriori distribution of our parameters $\theta|\mathbf{y}$ the posteriori expectation of a single probability parameter say θ_j (also known as the optimal mean-square estimator of θ_j) is given by

$$E(\theta_j) = \frac{y_j + \alpha_j}{\sum_{j=1}^{k}(y_j + \alpha_j)}\,. \tag{77}$$

Finally, as additional background needed to derive the expression for $\hat{P}(z|\omega_k)$ given in the book we need the information/fact that a uniform prior on the parameters of our multinomial distribution means that our Dirichlet prior is given by $\alpha_j = 1$ for all j.

Back to the notation used in the book. The uniform prior on the 2^N parameters we wish to estimate $P(z|\omega_k)$, means that they are initially distributed as a Dirichlet$(1, 1, 1, \cdots, 1)$ distribution (here there are 2^N values of α_j all of which have the same value 1. Then after having observed $N_k(z)$ values of the feature z from $\sum_z N_k(z) = N_k$ total values we can use Equation 77 to find the posteriori expectation of the parameters $P(z|\omega_k)$. We find

$$\begin{aligned}
\hat{P}(z|\omega_k) &= \frac{y_j + \alpha_j}{\sum_{j=1}^{2^N}(y_j + \alpha_j)} = \frac{y_j + \alpha_j}{\sum_{j=1}^{2^N} y_j + \sum_{j=1}^{2^N} \alpha_j} \\
&= \frac{N_k(z) + 1}{\sum_z N_k(z) + \sum_{j=1}^{2^N} 1} \\
&= \frac{N_k(z) + 1}{N_k + 2^N}\,,
\end{aligned} \tag{78}$$

which is the expression 5.22 given in the book. The variance of this expression can be computed using standard techniques. We find

$$\begin{aligned}
\text{Var}[\hat{P}(z|\omega_k)] &= \frac{\text{Var}[N_k(z) + 1]}{(N_k + 2^N)^2} = \frac{\text{Var}[N_k(z)]}{(N_k + 2^N)^2} \\
&= \frac{N_k P(z|\omega_k)(1 - P(z|\omega_k))}{(N_k + 2^N)^2}\,,
\end{aligned} \tag{79}$$

which agrees with the expression 5.23 given in the book.

Notes on the preceptron learning algorithm

From the definition of the sets Y_1 and Y_2 given in the book when $y \in Y_1$ the product $w^T y$ is negative, while if $y \in Y_2$ then the product $w^T y$ is positive. Thus $J_{\text{preceptron}}$ defined by

$$J_{\text{preceptron}} \equiv -\sum_{y \in Y_1} w^T y + \sum_{y \in Y_2} w^T y\,, \tag{80}$$

is a classification performance metric that gets larger the more elements are misclassified by the linear discriminant $g(y) = w^T y$. To minimize this metric we will use the gradient decent algorithm

$$w(i+1) = w(i) - \eta(i)\nabla J(w(i))\,, \tag{81}$$

where we compute a derivative for $J_{\text{preceptron}}$ as

$$\nabla J_{\text{preceptron}} = -\sum_{y\in Y_1} y + \sum_{y\in Y_2} y\,. \tag{82}$$

Note in fact for this problem $\nabla J_{\text{preceptron}}$ does not depend on the current vector $w(i)$. Implementing the derivative of $J_{\text{preceptron}}$ in this way is called the **batch** algorithm since we use all of the misclassified data points in computing summations needed in the derivative above. An alternative method would be to we consider only one sample, say y_n, at a time (imagining a data set with only one point) then if y_n is in the first class and is misclassified by our linear machine then we would update the weight vector now using Equation 82 but specified to only the point y_n as

$$w(i+1) = w(i) - \eta(i)(-y_n) = w(i) + \eta(i)y_n\,.$$

In the same way if y_n is in the *second* class and is misclassified by our linear machine then we would update the weight vector as

$$w(i+1) = w(i) - \eta(i)y_n = w(i) - \eta(i)y_n\,.$$

Thus if we introduce the variable c_n as $c_n = +1$ if y_n is misclassified and in the first class and $c_n = -1$ if y_n is misclassified and in the second class then the two update rules above can be combined as

$$w(i+1) = w(i) + c_n\eta(i)y_n\,, \tag{83}$$

which is the books equation 5.45. Note if y_n is classified *correctly* we could assign $c_n = 0$ and use Equation 83 on all of the data points.

Notes on the least squared error learning algorithm

In this subsection we comment on how to *use* the matrix W_{LS} once it is computed to classify a new sample $\mathbf{z}$. From the construction of the matrices T and Y given in this section we determine that T is of dimension $N_S \times K$, while Y is of dimension $N_s \times (N+1)$. From these two dimensions we conclude that $Y^T Y$ is of dimension $(N+1)\times K$ and so W_{LS} is of dimension $(N+1)\times K$ also. Thus given a new measurement vector $\mathbf{z}$ of dimension $N \times 1$ we compute the augmented vector $\mathbf{y}$ of dimension $(N+1)\times 1$. With this vector $\mathbf{y}$ we then compute a vector $\hat{\mathbf{t}}$ defined by

$$\hat{\mathbf{t}} = W_{\text{LS}}^T y = \begin{bmatrix} w_1^T y \\ w_2^T y \\ \vdots \\ w_K^T y \end{bmatrix}\,,$$

which will be a vector of size $K \times 1$. We then assign the object that generated the feature vector $\mathbf{z}$ to belong in the class k where k is chosen such that

$$k = \text{Argmin}_{1 \leq k \leq K} ||\mathbf{t}_k - \hat{\mathbf{t}}|| \,,$$

i.e. to the class with the closest target vector.

Notes on the kernel trick

We can greatly generalize the ability of support vector classifiers by recognizing that the classification decision made by a support vector classifier depends *only* on the evaluation of inner products. To see this note that in the training and the use of a support vector classifier we would perform the following steps

- Maximize the dual-form of L defined as

$$\sum_{n=1}^{N_S} \alpha_n - \frac{1}{2} \sum_{n=1}^{N_S} \sum_{m=1}^{N_S} c_n c_m \alpha_n \alpha_m z_n^T z_m \,, \tag{84}$$

  over the values of α using a "standard" quadratic optimization solver.

- With these values of α_n, the weight vector w used for classification is then given as

$$w = \sum_{n=1}^{N_S} \alpha_n c_n z_n \,. \tag{85}$$

- Classifications of a new feature vector z, are based on the sign of the expression $g(z) = w^T z$.

To generalize these procedures to nonlinear spaces we introduce a mapping $\mathbf{y}(\mathbf{z})$ that would take the given input vector $\mathbf{z}$ and map it into a higher dimensional nonlinear space. For example, given linear combinations of the simple features z_1 and z_2 in a two dimensional feature vector $\mathbf{z}$, the support vector classifier is capable of finding linear separating decision boundaries only. If we map these inputs into quadratic functions of the base inputs we can expand this linear space to include quadratic decision boundaries. For example, consider the mapping $\mathbf{y}(\mathbf{z})$ that takes an input vector $\mathbf{z}$ and returns the following larger vector $\mathbf{y}$

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \rightarrow \begin{bmatrix} z_1 \\ z_2 \\ z_1^2 \\ z_2^2 \\ z_1 z_2 \end{bmatrix} = \mathbf{y}(\mathbf{z}) \,.$$

To naively apply the above support vector classifier training procedure in the new higher dimensional space we would apply our nonlinear mapping $y(\cdot)$ to every sample point z_n and

then develop a classifier using these new points y_n. This could be very time and memory demanding if the space $\mathbf{y}$ is very large. A considerably simpler approach is to recognized that in solving the quadratic programming problem for α_n in Equation 84 all that are needed are the inner-product calculations of the mapped training set or $y(z_n)^T y(z_m)$. In addition, to classify a new feature $\mathbf{z}$ we would need to evaluate $g(z) = w^T y(z)$, which under the support vector classifier framework in the expanded space where $w = \sum_{n=1}^{N_S} \alpha_n c_n y(z_n)$ is given by

$$w^T y(z) = \sum_{n=1}^{N_S} \alpha_n c_n y(z_n)^T y(z) = \sum_{n=1}^{N_S} \alpha_n c_n K(z_n, z)\,,$$

which again only depends on the evaluation of the specific inner products $y(z_n)^T y(z)$ and in the above notation we have defined the "kernel" K function as $K(z_n, z_m) = z_n^T z_m$. Thus to train and evaluate a support vector classifier even in this higher dimensional space we only need to be able to compute the evaluation of the inner product kernel $K(z_n, z)$ easily. Another generalization of support vector classifiers that is simple involves replacing these inner product kernel's with another functional form, for example Gaussian kernels are often used.

Exercise Solutions

Exercise 1 (the covariance for the average of N realizations of z)

Lets consider, x to be the average of N realizations of the random variable Z. Then $x = \frac{1}{N}\sum_{k=1}^{N} z_k$ and the expectation of x is given by

$$E[x] = \frac{1}{N}\sum_{k=1}^{N} E[z_k] = E[Z]\,.$$

Using this, the covariance matrix of x is given by

$$\begin{aligned} E[(x - E[x])(x - E[x])^T] &= E\left[\left(\frac{1}{N}\sum_{k=1}^{N} z_k - E[Z]\right)\left(\frac{1}{N}\sum_{k=1}^{N} z_k - E[Z]\right)^T\right] \\ &= \frac{1}{N^2}\sum_{k=1}^{N}\sum_{j=1}^{N} E\left[(z_k - E[Z])(z_j - E[Z])^T\right]\,. \end{aligned}$$

Now the argument of the double summation above (by independence of the samples z_k) is equal to

$$\begin{aligned} E\left[(z_k - E[Z])(z_j - E[Z])^T\right] &= \begin{cases} E[z_k - E[Z]]E[z_j - E[Z]]^T & k \neq j \\ E[(z_k - E[Z])(z_k - E[Z])^T] & k = j \end{cases} \\ &= \begin{cases} 0 & k \neq j \\ C_z & k = j \end{cases}\,, \end{aligned}$$

since when $k \neq j$ we are assuming that z_k and z_j are independent draws, we have $E\left[(z_k - E[Z])(z_j - E[Z\right.$ $E[z_k - E[Z]]E[z_j - E[Z]]^T$, and $E[z_k - E[Z]] = 0$. When $k = j$ this pairwise expectation

is the covariance of the random variable Z or C_z. Thus we have for the covariance of x the following now

$$C_x = \frac{1}{N^2}\sum_{k=1}^{N}\sum_{j=1}^{N} C_z\delta_{kj} = \frac{C_z}{N^2}\sum_{k=1}^{N} 1 = \frac{1}{N}C_z\,,$$

as we were to show.

Exercise 2 (regularizing a covariance matrix)

Given that equation 5.16 in the book is

$$\hat{C}^{-1}_{\text{regularized}} = \hat{V}\left((1-\gamma)\hat{\Lambda} + \gamma\frac{\text{trace}(\hat{\Lambda})}{N}I\right)^{-1}\hat{V}^T\,, \tag{86}$$

we can use the eigenvector decomposition of the matrix $\hat{C}$ given by $\hat{C} = \hat{V}\hat{\Lambda}\hat{V}^T$ to "move" $\hat{V}$ inside the inverse above to find

$$\hat{C}^{-1}_{\text{regularized}} = \left(\hat{V}^{-T}(1-\gamma)\hat{\Lambda}\hat{V}^{-1} + \frac{\gamma}{N}\text{trace}(\hat{\Lambda})\hat{V}^{-T}\hat{V}^{-1}\right)^{-1}\,.$$

Since in the eigenvector expansion of a symmetric matrix $\hat{C}$ the matrix of eigenvectors $\hat{V}$ is orthogonal i.e. $\hat{V}^{-1} = \hat{V}^T$ we conclude that $\hat{V}^{-T}\hat{V}^{-1} = I$, and $\hat{V}^{-T}\hat{\Lambda}\hat{V}^{-1} = \hat{V}\hat{\Lambda}\hat{V}^T = \hat{C}$. With these substitutions and recalling the trace identity of

$$\text{trace}(\hat{C}) = \text{trace}(\hat{\Lambda})\,, \tag{87}$$

we have demonstrated equation 5.17 or

$$\hat{C}^{-1}_{\text{regularized}} = \left((1-\gamma)\hat{C} + \frac{\gamma}{N}\text{trace}(\hat{C})I\right)^{-1}\,. \tag{88}$$

Exercise 3 (Fisher's linear discriminant)

WWX: finish this problem

I think I need to apply gaussian ellimination to explicitly determine the scalar equation for W_{LS} below, but I ran out of time for this.

Equation 5.50 in the book that determines the weight matrix W that provides the optimal least squares classifier is

$$W_{\text{LS}} = (Y^TY)^{-1}Y^TT\,. \tag{89}$$

Here the matrix W is $W = [\mathbf{w}_1, \cdots, \mathbf{w}_K]$ or K columns of $\mathbf{w}$ vectors each of length $N+1$. The matrix Y is constructed as $[\mathbf{y}_1, \ldots \mathbf{y}_{N_s}]^T$ with $\mathbf{y}$ augmented feature vectors

$$\mathbf{y} = \begin{bmatrix} \mathbf{z} \\ 1 \end{bmatrix}\,.$$

Finally the matrix T is a matrix of target vectors $[\mathbf{t}_1, \ldots \mathbf{t}_{N_s}]^T$ with $\mathbf{t}_n$ a target vector with elements

$$t_{n,k} = \begin{cases} 1 & \text{if} \quad \theta_n = \omega_k \\ 0 & \text{otherwise} \end{cases} .$$

Now to specify the above description to the two-class case $K = 2$ we begin by constructing the matrix T (by rearranging the order of the samples if needed) so that it has the following form

$$T = \begin{bmatrix} 1 & 1 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & 1 & \cdots & 1 \end{bmatrix}^T ,$$

i.e. all of the class one objects are ordered before the class two objects. Next the matrix Y in terms of z is given by

$$Y = \begin{bmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,N_1} & z_{2,1} & z_{2,2} & \cdots & z_{2,N_2} \\ 1 & 1 & \cdots & 1 & 1 & 1 & \cdots & 1 \end{bmatrix}^T .$$

Here in the subscripts of $z_{k,n}$ the first element k is the *class* label, while the second element n is the sample *index*. Using this expression the required product $Y^T Y$ in Equation 89 becomes

$$\begin{aligned} Y^T Y &= \begin{bmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,N_1} & z_{2,1} & z_{2,2} & \cdots & z_{2,N_2} \\ 1 & 1 & \cdots & 1 & 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} z_{1,1}^T & 1 \\ z_{1,2}^T & 1 \\ \vdots & \vdots \\ z_{1,N_1}^T & 1 \\ z_{2,1}^T & 1 \\ z_{2,2}^T & 1 \\ \vdots & \vdots \\ z_{2,N_2}^T & 1 \end{bmatrix} \\ &= \begin{bmatrix} \sum_{n=1}^{N_1} z_{1,n} z_{1,n}^T + \sum_{n=1}^{N_2} z_{2,n} z_{2,n}^T & \sum_{n=1}^{N_1} z_{1,n} + \sum_{n=1}^{N_2} z_{2,n} \\ \sum_{n=1}^{N_1} z_{1,n}^T + \sum_{n=1}^{N_2} z_{2,n}^T & N_1 + N_2 \end{bmatrix} \\ &= \begin{bmatrix} \sum_{n=1}^{N_1} z_{1,n} z_{1,n}^T + \sum_{n=1}^{N_2} z_{2,n} z_{2,n}^T & N_1 \hat{\mu}_1 + N_2 \hat{\mu}_2 \\ N_1 \hat{\mu}_1^T + N_2 \hat{\mu}_2^T & N_S \end{bmatrix} , \end{aligned}$$

where we have used the definition $\hat{\mu}_1 = \frac{1}{N_1} \sum_{n=1}^{N_1} z_{1,n}$ and the same for $\hat{\mu}_2$. The next required product in Equation 89 is $Y^T T$ and is given by

$$\begin{aligned} Y^T T &= \begin{bmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,N_1} & z_{2,1} & z_{2,2} & \cdots & z_{2,N_2} \\ 1 & 1 & \cdots & 1 & 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ \vdots & \vdots \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ \vdots & \vdots \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} N_1 \hat{\mu}_1 & N_2 \hat{\mu}_2 \\ N_1 & N_2 \end{bmatrix} \end{aligned}$$

Consider now the representation for the two class projection matrix W denoted *Fisher's linear discriminant* and discussed in Chapter 6 of the book. There we find that $W = (\hat{\mu}_1 - \hat{\mu}_2)^T S_w^{-1}$, where

$$S_w = \frac{1}{N_S} \sum_{k=1}^{K} \sum_{n=1}^{N_k} (z_{k,n} - \hat{\mu}_k)(z_{k,n} - \hat{\mu}_k)^T . \tag{90}$$

Lets see how we can rewrite the above expression for S_w in the *two* class case. We find

$$\begin{aligned}
S_w &= \frac{1}{N_S} \sum_{k=1}^{2} \sum_{n=1}^{N_k} (z_{k,n} z_{k,n}^T - z_{k,n} \hat{\mu}_k^T - \hat{\mu}_k z_{k,n}^T + \hat{\mu}_k \hat{\mu}_k^T) \\
&= \frac{1}{N_S} \sum_{k=1}^{2} \sum_{n=1}^{N_k} z_{k,n} z_{k,n}^T - \frac{1}{N_S} \sum_{k=1}^{2} \left(\sum_{n=1}^{N_k} z_{k,n} \right) \hat{\mu}_k^T \\
&- \frac{1}{N_S} \sum_{k=1}^{2} \hat{\mu}_k \left(\sum_{n=1}^{N_k} z_{k,n}^T \right) + \frac{1}{N_S} \sum_{k=1}^{2} N_k \hat{\mu}_k \hat{\mu}_k^T \\
&= \frac{1}{N_S} \sum_{k=1}^{2} \sum_{n=1}^{N_k} z_{k,n} z_{k,n}^T - \frac{1}{N_S} \sum_{k=1}^{2} N_k \hat{\mu}_k \hat{\mu}_k^T \\
&- \frac{1}{N_S} \sum_{k=1}^{2} \hat{\mu}_k N_k \hat{\mu}_k^T + \frac{1}{N_S} \sum_{k=1}^{2} N_k \hat{\mu}_k \hat{\mu}_k^T \\
&= \frac{1}{N_S} \sum_{k=1}^{2} \sum_{n=1}^{N_k} z_{k,n} z_{k,n}^T - \frac{1}{N_S} \sum_{k=1}^{2} N_k \hat{\mu}_k \hat{\mu}_k^T .
\end{aligned} \tag{91}$$

Thus from this we see that the sum $\sum_{k=1}^{2} \sum_{n=1}^{N_k} z_{k,n} z_{k,n}^T$ is equal to

$$\sum_{k=1}^{2} \sum_{n=1}^{N_k} z_{k,n} z_{k,n}^T = N_S S_w + \sum_{k=1}^{2} N_k \hat{\mu}_k \hat{\mu}_k^T .$$

Note this involves inverting the matrix $Y^T Y$. From the calculations above we see that this is a block matrix as

$$Y^T Y = \begin{bmatrix} \Sigma & c \\ c^T & s \end{bmatrix} .$$

Exercise 4 (the behavior of two estimators for $P(z|\omega_k)$)

The two estimators for $P(z|\omega_k)$ discussed in the book were

$$\hat{P}(z|\omega_k) = \frac{N_k(z)}{N_k} , \tag{92}$$

and

$$\hat{P}(z|\omega_k) = \frac{N_k(z) + 1}{N_k + 2^N} . \tag{93}$$

The bias and variance for the estimator given by Equation 92 can be derived in exactly the same way as discussed on Page 36, for estimating $P(\omega_k)$. We find that this estimator is unbiased and to have a variance given by

$$\text{Var}[\hat{P}(z|\omega_k)] = \frac{P(z|\omega_k)(1-P(z|\omega_k))}{N_k}\,. \tag{94}$$

While this estimator is unbiased it can have a very large variance making it undesirable to use in practice especially in the "small sample" limit i.e. where $N_k \ll 2^N$. To show an example where this is true lets assume $N_k \ll 2^N$ so $\frac{N_k}{2^N} \ll 1$ and take $N = 10$ with $N_k \approx 50 \ll 2^{10} = 1024$. Then $P(z|\omega_k) = O(2^{-10}) = O(10^{-3})$ we when see that the variance of our estimator using Equation 94 would be

$$\text{Var}(\hat{P}(z|\omega_k)) = \frac{10^{-3}(1-10^{-3})}{50} = 1.99\,10^{-5}\,,$$

so the standard error would then be

$$\sqrt{\text{Var}(\hat{P}(z|\omega_k))} = 4.46\,10^{-3}\,.$$

From this the relative error in estimating $\hat{P}(z|\omega_k)$ using $\frac{N_k(z)}{N_k}$ is

$$\frac{\sqrt{\text{Var}(\hat{P}(z|\omega_k))}}{P(z|\omega_k)} = \frac{4.46\,10^{-3}}{10^{-3}} = 4.46 = 446\%\,,$$

a very poor estimate. If, however, we use the estimate given by Equation 93 of

$$\hat{P}(z|\omega_k) = \frac{N_k(z)+1}{N_k+2^N}\,,$$

then the expectation of this estimate becomes

$$\begin{aligned} E[\hat{P}(z|\omega_k)] &= \frac{N_k P(z|\omega_k)+1}{N_k+2^N} \\ &= \frac{(N_k+2^N-2^N)P(z|\omega_k)+1}{N_k+2^N} \\ &= P(z|\omega_k) + \frac{1-2^N P(z|\omega_k)}{N_k+2^N}\,, \end{aligned}$$

which can be seen to be a biased estimate. The variance of this estimator was derived earlier and is given by Equation 79. If we compute the value of these two expressions using the numeric quantities given above we find

$$\begin{aligned} E[\hat{P}(z|\omega_k)] &= P(z|\omega_k) + \frac{1-2^{10}(10^{-3})}{50+2^{10}} = P(z|\omega_k) - 2.234\,10^{-5} \approx P(z|\omega_k) \\ \text{Var}[\hat{P}(z|\omega_k)] &= \frac{50(10^{-3})(1-10^{-3})}{(50+2^{10})^2} = 4.33\,10^{-8} \quad \text{so} \\ \frac{\sqrt{\text{Var}[\hat{P}(z|\omega_k)]}}{P(z|\omega_k)} &= \frac{\sqrt{4.33\,10^{-8}}}{10^{-3}} = 0.20\,, \end{aligned}$$

which is not a stellar relative error but is certainly better than the previous estimate. In the opposite limit of N_k where $N_k \gg 2^N$ we have $\frac{2^N}{N_k} \ll 1$ and the variance of our two estimates become

$$\begin{aligned}
\mathrm{Var}[\hat{P}(z|\omega_k)] &= \frac{P(z|\omega_k)(1-P(z|\omega_k))}{N_k} \ll \frac{P(z|\omega_k)(1-P(z|\omega_k))}{2^N} \\
\mathrm{Var}[\hat{P}(z|\omega_k)] &= \frac{P(z|\omega_k)(1-P(z|\omega_k))}{\left(1+\frac{2^N}{N_k}\right)(N_k+2^N)} \approx \frac{P(z|\omega_k)(1-P(z|\omega))}{N_k+2^N} \\
&\ll \frac{P(z|\omega_k)(1-P(z|\omega_k))}{2^N}\,.
\end{aligned}$$

Thus both estimators have similar variances that improve the larger the value of N_k is relative to 2^N.

Exercise 5 (Bayesian learning of the error rate E)

Given a classifier system with an unknown error rate, E, initially assumed uniformly distributed between 0 and $1/K$ and the knowledge that when we attempt to classify N_{test} independent samples we obtain n_{error} classification errors allows us to refine our degree of believe as to the unknown error rate E. To do that we will use Bayes' rule as to update $p(E)$ after having observed n_{error} as

$$\begin{aligned}
p(E|n_{\text{error}}, N_{\text{test}}) &= \frac{p(n_{\text{error}}, N_{\text{test}}|E)p(E)}{p(n_{\text{error}}, N_{\text{test}})} \\
&= \frac{p(n_{\text{error}}, N_{\text{test}}|E)p(E)}{\int_e p(n_{\text{error}}, N_{\text{test}}|E=e)p(E=e)de}\,.
\end{aligned} \tag{95}$$

Because the initial probability of E was assumed uniform $p(E)$ is given by

$$p(E=e) = \begin{cases} K & 0 \le e \le 1/K \\ 0 & \text{otherwise} \end{cases}\,. \tag{96}$$

Now the expression for $p(n_{\text{error}}, N_{\text{test}}|E)$ can be derived as follows. Given the error rate E the density $p(n_{\text{error}}, N_{\text{test}}|E)$ is the probability of obtaining n_{error} errors when performing N_{test} trials. This is equivalent to stating that n_{error} is distributed as a *Binomial* distribution with parameters (E, N_{test}). Thus

$$p(n_{\text{error}}, N_{\text{test}}|E) = \begin{pmatrix} N_{\text{test}} \\ n_{\text{error}} \end{pmatrix} E^{n_{\text{error}}}(1-E)^{N_{\text{test}}-n_{\text{error}}}\,. \tag{97}$$

Now to compute $p(n_{\text{error}}, N_{\text{test}})$ we have to evaluate the following integral

$$\begin{aligned}
p(n_{\text{error}}, N_{\text{test}}) &= \int_0^{1/K} p(n_{\text{error}}, N_{\text{test}}|E=e)p(E=e)de \\
&= \begin{pmatrix} N_{\text{test}} \\ n_{\text{error}} \end{pmatrix} K \int_0^{1/K} e^{n_{\text{error}}}(1-e)^{N_{\text{test}}-n_{\text{error}}}de\,.
\end{aligned}$$

This expression, when viewed as a function of the upper limit of integration (which in this case is $1/K$) is an example of what is called the **Incomplete Beta function**. The **Beta** function was defined in Equation 27, and the Incomplete Beta function is defined as

$$B(x;p,q) = \int_0^x v^{p-1}(1-v)^{q-1}dv\,. \tag{98}$$

Note the upper limit of integration in the above is now x rather than the fixed value of 1 in the definition of the beta function. Using these three pieces given by Equations 96, 97 and 98 into Equation 95 we have derived the desired expression for $p(E|n_{\text{error}}, N_{\text{test}})$.

Exercise 6 (the dual formulation of the support vector classifier)

The Lagrangian we want to consider is given by

$$L = \frac{1}{2}||w||^2 - \sum_{n=1}^{N_S} \alpha_n(c_n(w^T z_n + b) - 1) \quad \text{for} \quad \alpha_n \geq 0\,. \tag{99}$$

Note that the sign of the second term in the above is correct but that the sign of this term in the in the book is *incorrect*, see [5] page 374 or [2] page 263. To minimize L by taking the required derivatives recall that

$$||w||^2 = w_1^2 + w_2^2 + \cdots + w_N^2\,,$$

so that the derivative of $||w||^2$ with respect to w is given by

$$\frac{\partial ||w||^2}{\partial w} = 2w\,.$$

Taking the needed derivative of L with respect to w and b and setting these equal to zero we obtain

$$\begin{aligned} \frac{\partial L}{\partial w} &= w - \sum_{n=1}^{N_S} \alpha_n c_n z_n = 0 \\ \frac{\partial L}{\partial b} &= \sum_{n=1}^{N_S} \alpha_n c_n = 0\,. \end{aligned} \tag{100}$$

Solving the first equation gives

$$w = \sum_{n=1}^{N_S} \alpha_n c_n z_n\,. \tag{101}$$

We now write L in a more "matrix" like form as

$$L = \frac{1}{2}w^T w - w^T\left(\sum_{n=1}^{N_S} \alpha_n c_n z_n\right) - b\sum_{n=1}^{N_S} \alpha_n c_n + \sum_{n=1}^{N_S} \alpha_n\,.$$

When we up Equation 101 and 100 into this expression for L we find

$$\begin{aligned} L &= \frac{1}{2}\left(\sum_{n=1}^{N_S}\alpha_n c_n z_n\right)^T\left(\sum_{n=1}^{N_S}\alpha_n c_n z_n\right) \\ &- \left(\sum_{n=1}^{N_S}\alpha_n c_n z_n\right)^T\left(\sum_{n=1}^{N_S}\alpha_n c_n z_n\right) + \sum_{n=1}^{N_S}\alpha_n \\ &= -\frac{1}{2}\left(\sum_{n=1}^{N_S}\alpha_n c_n z_n\right)^T\left(\sum_{n=1}^{N_S}\alpha_n c_n z_n\right) + \sum_{n=1}^{N_S}\alpha_n \\ &= \sum_{n=1}^{N_S}\alpha_n - \frac{1}{2}\sum_{n=1}^{N_S}\sum_{m=1}^{N_S}\alpha_n\alpha_m c_n c_m z_n^T z_m\,, \end{aligned}$$

which is the desired expression.

Exercise 7 (slack variables in support vector classifiers)

Following the book we note that our original constraints in the linearly separable problem which were

$$\begin{aligned} w^T z_n + b &\geq 1 \quad \text{when} \quad c_n = +1 \\ w^T z_n + b &\leq -1 \quad \text{when} \quad c_n = -1\,, \end{aligned}$$

become in the non-linearly separable case

$$\begin{aligned} w^T z_n + b &\geq 1 - \xi_n \quad \text{when} \quad c_n = +1 \\ w^T z_n + b &\leq -1 + \xi_n \quad \text{when} \quad c_n = -1\,, \end{aligned}$$

for some ξ_n and we seek a minimum of $||w||$ subject to the constraints that $\xi_n \geq 0$. Thus our problem becomes to minimize L where L is defined as

$$L = \frac{1}{2}||w||^2 + C\sum_{n=1}^{N_S}\xi_n - \sum_{n=1}^{N_S}\alpha_n(c_n(w^T z_n + b) - 1 + \xi_n) - \sum_{n=1}^{N_S}\gamma_n\xi_n\,,$$

with Lagrange multipliers taken $\alpha_n \geq 0$ and $\gamma_n \geq 0$. Taking the derivatives of L with respect to w, ξ, and b and setting them equal to zero gives.

$$\frac{\partial L}{\partial w} = \frac{1}{2}(2w) - \sum_{n=1}^{N_S}\alpha_n c_n z_n = 0 \tag{102}$$

$$\frac{\partial L}{\partial \xi_n} = C - \alpha_n - \gamma_n = 0 \tag{103}$$

$$\frac{\partial L}{\partial b} = -\sum_{n=1}^{N_S}\alpha_n c_n = 0\,. \tag{104}$$

Next put the above expressions into L we will write L in a "matrix" like representation as

$$\begin{aligned} L &= \frac{1}{2}w^T w + C\sum_{n=1}^{N_S}\xi_n - w^T\sum_{n=1}^{N_S}\alpha_n c_n z_n \\ &- b\sum_{n=1}^{N_S}\alpha_n c_n + \sum_{n=1}^{N_S}\alpha_n(1-\xi_n) - \sum_{n=1}^{N_S}\gamma_n\xi_n \end{aligned}$$

Solving Equation 103 for γ_n gives $\gamma_n = C - \alpha_n$. Using this expression and Equation 102 solved for w and Equation 104 directly we find L now looks like

$$\begin{aligned} L &= -\frac{1}{2}\left(\sum_{n=1}^{N_S}\alpha_n c_n z_n\right)^T\left(\sum_{n=1}^{N_S}\alpha_n c_n z_n\right) \\ &+ C\sum_{n=1}^{N_S}\xi_n + \sum_{n=1}^{N_S}\alpha_n(1-\xi_n) - \sum_{n=1}^{N_S}(C-\alpha_n)\xi_n \\ &= \sum_{n=1}^{N_S}\alpha_n - \frac{1}{2}\sum_{n=1}^{N_S}\sum_{m=1}^{N_S}\alpha_n\alpha_m c_n c_m z_n^T z_m\,, \end{aligned} \tag{105}$$

which is the same as the books equation 5.56. Note that the requirement that the Lagrange multipliers be non-negative $\gamma_n \geq 0$ is equivalent to $\gamma_n = C - \alpha_n \geq 0$ and thus requires the constraint that $\alpha_n \leq C$ when solving the quadratic programming problem to maximize Equation 105 over α_n.

Exercise 8 (deriving the neural network weight update rules)

Recall the error criterion function J_{SE} defined by

$$J_{\text{SE}} = \frac{1}{2}\sum_{n=1}^{N_S}\sum_{k=1}^{K}(g_k(y_n) - t_{n,k})^2\,, \tag{106}$$

where the output from the k-th element in the outer layer, $g_k(y)$, given by

$$g_k(y) = f\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y) + v_{k,H+1}\right)\,.$$

The k,h notation in $v_{k,h}$ means that the flow of information is *into* k from h. First, the update to the weights $\mathbf{v}$ that would be specified by a gradient-decent like procedure would be given by

$$\begin{aligned} \Delta v_{k,h} &= -\eta\frac{\partial J_{\text{SE}}}{\partial v_{k,h}} \\ &= -\eta\sum_{n=1}^{N_S}\sum_{k=1}^{K}(g_k(y_n) - t_{n,k})\frac{\partial g_k(y_n)}{\partial v_{k,h}}\,. \end{aligned} \tag{107}$$

To compute $\frac{\partial g_k(y_n)}{\partial v_{k,h}}$ we find

$$\frac{\partial g_k(y_n)}{\partial v_{k,h}} = \dot{f}\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y_n) - v_{k,H+1}\right) \frac{\partial}{\partial v_{k,h}}\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y_n) - v_{k,H+1}\right) .$$

The remaining derivative with respect to $v_{k,h}$ is different depending on the value of h. If we have $1 \leq h \leq H$ then

$$\frac{\partial}{\partial v_{k,h}}\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y_n) - v_{k,H+1}\right) = f(w_h^T y_h) ,$$

and Equation 107 becomes

$$\Delta v_{k,h} = -\eta \sum_{n=1}^{N_S}\sum_{k=1}^{K}(g_k(y_n) - t_{n,k})\dot{f}\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y_n) - v_{k,H+1}\right) f(w_h^T y_n) .$$

While if $h = H+1$ we obtain

$$\frac{\partial}{\partial v_{k,h}}\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y_n) - v_{k,H+1}\right) = -1 ,$$

and Equation 107 becomes

$$\Delta v_{k,H+1} = \eta \sum_{n=1}^{N_S}\sum_{k=1}^{K}(g_k(y_n) - t_{n,k})\dot{f}\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y_n) - v_{k,H+1}\right) .$$

These correspond to the equation 5.65 in the book. Second, the update to the weights $\mathbf{w}$ that would be specified by a gradient-decent like procedure would be given by

$$\begin{aligned} \Delta w_{h,i} &= -\eta \frac{\partial J_{\text{SE}}}{\partial w_{h,i}} \\ &= -\eta \sum_{n=1}^{N_S}\sum_{k=1}^{K}(g_k(y_n) - t_{n,k})\frac{\partial g_k(y_n)}{\partial w_{h,i}} . \end{aligned} \tag{108}$$

To compute $\frac{\partial g_k(y_n)}{\partial w_{h,i}}$ we find

$$\begin{aligned} \frac{\partial g_k(y_n)}{\partial w_{h,i}} &= \dot{f}\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y_n) - v_{k,H+1}\right) \frac{\partial}{\partial w_{h,i}}\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y_n)\right) \\ &= \dot{f}\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y_n) - v_{k,H+1}\right)\left(v_{k,h}\dot{f}(w_h^T y_n) y_{n,i}\right) . \end{aligned}$$

Thus in summary then we have

$$\Delta w_{h,i} = -\eta \sum_{n=1}^{N_S}\sum_{k=1}^{K}(g_k(y_n) - t_{n,k})\left(v_{k,h}\dot{f}(w_h^T y_n) y_{n,i}\right)\dot{f}\left(\sum_{h=1}^{H} v_{k,h} f(w_h^T y_n) - v_{k,H+1}\right) ,$$

which is the books equation 5.66.

Exercise 9 (limiting the nonlinearity in neural networks)

To avoid large values for the network weights one could add a penalty term to the cost function we seek to minimize and given in Equation 106 . For example we would attempt to minimize

$$J_{\text{SE}} = \frac{1}{2}\sum_{n=1}^{N_S}\sum_{k=1}^{K}(g_k(y_n) - t_{n,k})^2 + \frac{C}{2}||w||^2\,,$$

where C is a non-negative constant that could be chosen by cross-validation.

Exercise 10 (optimization techniques in neural network training)

Better optimization techniques like the Levenberg-Marquardt algorithm that in fact is coded in the Pattern Recognition Toolbox algorithm `lmnc.m` can speed up the time required until the algorithm converges. Thus second order algorithms may be quicker to run overall.

Chapter 6: (Feature Extraction and Selection)

Notes on the Text

Notes on the inter/intra distance

We will apply a linear transformation to each original feature vector z_n as $y_n = Az_n$, where A is chosen so that $AS_wA^T = I$, which when we diagonalize A as $S_w = V\Lambda V^T$ requires that $A = \Lambda^{1/2}V^T$. Then in the transformed space (denoted with a prime over each vector) we have new mean vectors given by

$$\begin{aligned}\hat{\mu}'_k &= \Lambda^{-1/2}V^T\hat{\mu}_k \\ \hat{\mu}' &= \Lambda^{-1/2}V^T\hat{\mu}\,,\end{aligned}$$

so that the between class scatter matrix S_b defined as

$$S_b = \frac{1}{N_S}\sum_{k=1}^{K} N_k(\hat{\mu}_k - \hat{\mu})(\hat{\mu}_k - \hat{\mu})^T\,, \tag{109}$$

transforms as

$$\begin{aligned}S'_b &= \frac{1}{N_S}\sum_{k=1}^{K} N_k(\hat{\mu}'_k - \hat{\mu}')(\hat{\mu}'_k - \hat{\mu}')^T \\ &= \frac{1}{N_S}\sum_{k=1}^{K} N_k\Lambda^{1/2}V^T(\hat{\mu}_k - \hat{\mu})(\hat{\mu}_k - \hat{\mu})^T(\Lambda^{-1/2}V^T)^T \\ &= \Lambda^{-1/2}V^T\left(\frac{1}{N_S}\sum_{k=1}^{K}(\hat{\mu}_k - \hat{\mu})(\hat{\mu}_k - \hat{\mu})^T\right)V\Lambda^{-1/2} \\ &= \Lambda^{-1/2}V^TS_bV\Lambda^{-1/2} = AS_bA^T\,.\end{aligned}$$

The the matrix S_w will transform in the same way under the linear operator A. Thus in the *transformed* space the trace of S'_w is a constant

$$\text{trace}(S'_w) = \text{trace}(AS_wA^T) = \text{trace}(I) = N\,,$$

the dimension of the state x. Because of this, the metric suggested in the book $\frac{J_{\text{INTER}}}{J_{\text{INTRA}}}$ depends only on the trace of S'_b. To compute this later expression we recall that taking the trace of a matrix product allows rotation inside the argument i.e. $\text{trace}(ABC) = \text{trace}(CAB)$ assuming all products are defined. Thus

$$\begin{aligned}\text{trace}(S'_w) &= \text{trace}(\Lambda^{-1/2}V^TS_bV\Lambda^{-1/2}) = \text{trace}(V\Lambda^{-1/2}\Lambda^{-1/2}V^TS_b) \\ &= \text{trace}(V\Lambda^{-1}V^TS_b) = \text{trace}(S_w^{-1}S_b)\,.\end{aligned}$$

This later expression is called the inter/intra distance.

Notes on Chernoff-Bhattacharyya distances

Here we will derive the integral expression used to prove the Bhattacharyya error rate bound. Recall that the smallest error $E_{\min}$ is obtained when we classify according to the Bayes' criterion in that case $E_{\min}$ becomes

$$E_{\min} = \int_z P(error, z)dz = \int_z P(error|z)p(z)dz\,. \tag{110}$$

where the conditional probability of error, $P(error|z)$, since this is a two class problem is given by

$$\begin{aligned} P(error|z) &= \min[P(\omega_1|z), 1 - P(\omega_1|z)] \\ &= \min[P(\omega_1|z), P(\omega_2|z)] \\ &= 1 - \max[P(\omega_1|z), P(\omega_2|z)]\,. \end{aligned}$$

Thus by Bayes' rule we have the product

$$\begin{aligned} \min[P(\omega_1|z), P(\omega_2|z)]p(z) &= \min[\frac{p(z|\omega_1)P(\omega_1)}{p(z)}, \frac{p(z|\omega_2)P(\omega_2)}{p(z)}]p(z) \\ &= \min[p(z|\omega_1)P(\omega_1), p(z|\omega_2)P(\omega_2)]\,. \end{aligned} \tag{111}$$

which when we integrate over z to get the entire error gives the books equation 6.12.

By generalizing the inequality $\min(a, b) \leq \sqrt{ab}$ for positive a and b, we can derive the so called Chernoff error rate bound which is more general than the Bhattacharyya bound and produces a distance metric (the Chernoff distance metric). Given the books expression for the Chernoff distance between two probability densities $p(z|\omega_k)$ for $k = 1, 2$ when these densities are multidimensional Gaussians we can derive an equivalent expression for the distance measure. Below we include a derivation of the Chernoff distance. It is quite detailed and could be skipped on first reading.

The derivation of the Chernoff distance

The generalization to $\min(a, b) \leq \sqrt{ab}$ we need is the identity that $\min(a, b) \leq a^s b^{1-s}$ for positive a and b and $0 < s < 1$. Then using Equation 110 and 111 we can bound $E_{\min}$ as

$$E_{\min} \leq P(\omega_1)^s P(\omega_2)^{1-s} \int p(x|\omega_1)^s p(x|\omega_2)^{1-s} dx \quad \text{for} \quad 0 \leq s \leq 1\,. \tag{112}$$

When the densities $p(x|\omega_k)$ for $k = 1, 2$ are d-dimensional multidimensional Gaussians then

$$p(x|\omega_i) = \frac{1}{(2\pi)^{d/2}|\mathbf{C}_i|^{1/2}} \exp\left\{-\frac{1}{2}(x - \mu_i)^T \mathbf{C}_i^{-1}(x - \mu_i)\right\}\,, \tag{113}$$

so the product in the integrand in Equation 112 is given by

$$\begin{aligned} p(x|\omega_1)^s p(x|\omega_2)^{1-s} &= \frac{1}{(2\pi)^{\frac{ds}{2}}(2\pi)^{\frac{d(1-s)}{2}}|\mathbf{C}_1|^{\frac{s}{2}}|\mathbf{C}_2|^{\frac{1-s}{2}}} \\ &\times \exp\left\{-\frac{s}{2}(x - \mu_1)^T \mathbf{C}_1^{-1}(x - \mu_1) - \frac{(1-s)}{2}(x - \mu_2)^T \mathbf{C}_2^{-1}(x - \mu_2)\right\}\,. \end{aligned}$$

Expanding the terms in the exponential we find (ignoring for now the factor $-\frac{1}{2}$)

$$sx^T\mathbf{C}_1^{-1}x - 2sx^T\mathbf{C}_1^{-1}\mu_1 + s\mu_1^T\mathbf{C}_1^{-1}\mu_1 + (1-s)x^T\mathbf{C}_2^{-1}x - 2(1-s)x^T\mathbf{C}_2^{-1}\mu_2 + (1-s)\mu_2\mathbf{C}_2^{-1}\mu_2\,.$$

Grouping the quadratic, linear, and constant terms we find

$$x^T(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})x - 2x^T(s\mathbf{C}_1^{-1}\mu_1 + (1-s)\mathbf{C}_2^{-1}\mu_2) + s\mu_1^T\mathbf{C}_1^{-1}\mu_1 + (1-s)\mu_2^T\mathbf{C}_2^{-1}\mu_2\,.$$

Using this expression the product we are considering then becomes

$$\begin{aligned} p(x|\omega_1)^s p(x|\omega_2)^{1-s} &= \frac{1}{(2\pi)^{\frac{d}{2}}|\mathbf{C}_1|^{\frac{s}{2}}|\mathbf{C}_2|^{\frac{1-s}{2}}}\exp\left\{-\frac{1}{2}\left(s\mu_1^T\mathbf{C}_1^{-1}\mu_1 + (1-s)\mu_2^T\mathbf{C}_2^{-1}\mu_2\right)\right\} \quad (114)\\ &\times \exp\left\{-\frac{1}{2}\left(x^T(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})x - 2x^T(s\mathbf{C}_1^{-1}\mu_1 + (1-s)\mathbf{C}_2^{-1}\mu_2)\right)\right\}. \end{aligned}$$

Thus we want to integrate this expression over all possible x values. The trick to evaluating an integral like this is to convert it into an integral that we know how to integrate. Since this involves the integral of a Gaussian like kernel we might be able to evaluate this integral by converting exactly it into the integral of a Gaussian. Then since it is known that the integral over all space of a Gaussians is one we may have evaluated indirectly the integral we are interested in. To begin this process we first consider what the argument of the *exponential* (without the $-1/2$) of a Gaussian with mean θ and covariance A would look like

$$(x-\theta)^T A^{-1}(x-\theta) = x^T A^{-1}x - 2x^T A^{-1}\theta + \theta^T A^{-1}\theta\,. \quad (115)$$

Using this expression to match the arguments of the quadratic and linear terms in the exponent in Equation 114 would indicate that

$$\begin{aligned} A^{-1} &= s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1} \quad \text{and}\\ A^{-1}\theta &= s\mathbf{C}_1^{-1}\mu_1 + (1-s)\mathbf{C}_2^{-1}\mu_2\,. \end{aligned}$$

Thus the Gaussian with a mean value θ and covariance A given by

$$\begin{aligned} A &= (s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1} \quad (116)\\ \theta &= A(s\mathbf{C}_1^{-1}\mu_1 + (1-s)\mathbf{C}_2^{-1}\mu_2)\\ &= (s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}(s\mathbf{C}_1^{-1}\mu_1 + (1-s)\mathbf{C}_2^{-1}\mu_2)\,, \quad (117) \end{aligned}$$

would evaluate to having exactly the *same* exponential terms (modulo the expression $\theta^T A^{-1}\theta$). The point of this is that with the definitions of A and θ we can write

$$x^T(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})x - 2x^T(s\mathbf{C}_1^{-1}\mu_1 + (1-s)\mathbf{C}_2^{-1}\mu_2) = (x-\theta)^T A^{-1}(x-\theta) - \theta^T A^{-1}\theta\,,$$

so that the integral we are attempting to evaluate can be written as

$$\begin{aligned} \int p(x|\omega_1)^s p(x|\omega_2)^{1-s}dx &= \frac{1}{(2\pi)^{\frac{d}{2}}|\mathbf{C}_1|^{\frac{s}{2}}|\mathbf{C}_2|^{\frac{1-s}{2}}}\\ &\times \exp\left\{-\frac{1}{2}\left(s\mu_1^T\mathbf{C}_1^{-1}\mu_1 + (1-s)\mu_2^T\mathbf{C}_2^{-1}\mu_2\right)\right\}\exp\left\{\frac{1}{2}\theta^T A^{-1}\theta\right\}\\ &\times \int \exp\left\{-\frac{1}{2}(x-\theta)^T A^{-1}(x-\theta)\right\}dx\,. \end{aligned}$$

In effect what we are doing is "completing the square" of the argument in the exponential. Since we know that multidimensional Gaussians integrate to one, this final integral becomes

$$\int \exp\left\{-\frac{1}{2}(x-\theta)^T A^{-1}(x-\theta)\right\} dx = (2\pi)^{d/2}|A|^{1/2}\,. \tag{118}$$

In addition, the argument in the exponential in front of the (now evaluated) integral is given by

$$s\mu_1^T\mathbf{C}_1^{-1}\mu_1 + (1-s)\mu_2^T\mathbf{C}_2^{-1}\mu_2 - \theta^T A^{-1}\theta\,. \tag{119}$$

When we put in the definition of A and θ given by Equations 116 and 117 we have that $\theta^T A^{-1}\theta$ is equivalent to three (somewhat complicated) terms

$$\begin{aligned}
\theta^T A^{-1}\theta &= (s\mu_1^T\mathbf{C}_1^{-1} + (1-s)\mu_2^T\mathbf{C}_2^{-1})(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}(s\mathbf{C}_1^{-1}\mu_1 + (1-s)\mathbf{C}_2^{-1}\mu_2) \\
&= s^2\mu_1^T\mathbf{C}_1^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_1^{-1}\mu_1 \\
&+ 2s(1-s)\mu_1^T\mathbf{C}_1^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_2^{-1}\mu_2 \\
&= (1-s)^2\mu_2^T\mathbf{C}_2^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_2^{-1}\mu_2\,.
\end{aligned}$$

Given that we still have to add the terms $s\mu_1^T\mathbf{C}_1^{-1}\mu_1 + (1-s)\mu_2^T\mathbf{C}_2^{-1}\mu_2$ to the negative of this expression we now stop and look at what our end result should look like in hopes of helping motivate the transformations to take next. Since we might want to try and factor this into an expression like $(\mu_1-\mu_2)^T B(\mu_1-\mu_2)$ by expanding this we see that we should try to get the expression above into a three term form that looks like

$$\mu_1^T B\mu_1 - 2\mu_1^T B\mu_2 + \mu_2^T B\mu_2\,, \tag{120}$$

for some matrix B. Thus lets add $s\mu_1^T\mathbf{C}_1^{-1}\mu_1 + (1-s)\mu_2^T\mathbf{C}_2^{-1}\mu_2$ to the negative of $\theta^T A^{-1}\theta$ and write the result in the three term form suggested by Equation 120 above. We find that Equation 119 then becomes when factored in this way

$$\begin{aligned}
& s\mu_1^T\left[\mathbf{C}_1^{-1} - s\mathbf{C}_1^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_1^{-1}\right]\mu_1 && (121)\\
- \;& 2s(1-s)\mu_1^T\left[\mathbf{C}_1^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_2^{-1}\right]\mu_2 && (122)\\
+ \;& (1-s)\mu_2^T\left[\mathbf{C}_2^{-1} - (1-s)\mathbf{C}_2^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_2^{-1}\right]\mu_2\,. && (123)
\end{aligned}$$

We now use the *inverse of inverse matrix sums lemma* (IIMSL) given by

$$(A^{-1}+B^{-1})^{-1} = A(A+B)^{-1}B = B(A+B)^{-1}A\,, \tag{124}$$

to write the matrix products in the middle term of the above expression as

$$\mathbf{C}_1^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_2^{-1} = ((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}\,. \tag{125}$$

Recognizing this matrix as one that looks familiar and that we would like to turn the others into lets now "hope" that the others can be transformed into a form that looks like that. To further see if this is possible, and to motivate the transformations done next, consider how the *desired* expression would look like expanded as in Equation 120. We have without the factor of $-\frac{1}{2}s(1-s)$ the following

$$\begin{aligned}
(\mu_1-\mu_2)^T((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}(\mu_1-\mu_2) &= \mu_1^T((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}\mu_1 && (126)\\
&- 2\mu_1^T((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}\mu_2 && (127)\\
&+ \mu_2^T((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}\mu_2\,. && (128)
\end{aligned}$$

Since as just shown the middle terms match as desired, looking at the terms Equation 121 and 126, to have the desired equality we want to show if we can prove

$$s\left[\mathbf{C}_1^{-1} - s\mathbf{C}_1^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_1^{-1}\right] = s(1-s)((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}\,, \tag{129}$$

and

$$(1-s)\left[\mathbf{C}_2^{-1} - (1-s)\mathbf{C}_2^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_2^{-1}\right] = s(1-s)((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}\,, \tag{130}$$

the similar expression for the terms Equation 123 and 128. To show that in fact this matrix difference is correct we will use another matrix identity lemma. This time we will use the *Woodbury* identity which can be written as

$$(A+UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}\,. \tag{131}$$

If we specialize this identity by taking C and V to both be identity matrices we obtain

$$\begin{aligned}(A+U)^{-1} &= A^{-1} - A^{-1}U(I + A^{-1}U)^{-1}A^{-1} \\ &= A^{-1} - A^{-1}(U^{-1} + A^{-1})^{-1}A^{-1}\,.\end{aligned}$$

Using this last expression with $A = s\mathbf{C}_1^{-1}$ and $U = (1-s)\mathbf{C}_2^{-1}$ we can derive that

$$\begin{aligned}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1} &= \frac{1}{s}\mathbf{C}_1 - \frac{1}{s}\mathbf{C}_1\left(\frac{1}{1-s}\mathbf{C}_2 + \frac{1}{s}\mathbf{C}_1\right)^{-1}\frac{1}{s}\mathbf{C}_1 \\ &= \frac{1}{s}\mathbf{C}_1 - \frac{(1-s)}{s}\mathbf{C}_1\left((1-s)\mathbf{C}_1 + s\mathbf{C}_2\right)^{-1}\mathbf{C}_1\,.\end{aligned}$$

Multiplying this last expression by $s\mathbf{C}_1^{-1}$ on the left and $\mathbf{C}_1^{-1}$ on the right to get

$$s\mathbf{C}_1^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_1^{-1} = \mathbf{C}_1^{-1} - (1-s)((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}\,.$$

This last expression gives that

$$\mathbf{C}_1^{-1} - s\mathbf{C}_1^{-1}(s\mathbf{C}_1^{-1} + (1-s)\mathbf{C}_2^{-1})^{-1}\mathbf{C}_1^{-1} = (1-s)((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}\,,$$

which is equivalent to the desired Equation 129. Using exactly the same steps one can prove Equation 130. In summary then we have shown that

$$\begin{aligned}\int p(x|\omega_1)^s p(x|\omega_2)^{1-s}dx &= \frac{|A|^{1/2}}{|\mathbf{C}_1|^{\frac{s}{2}}|\mathbf{C}_2|^{\frac{1-s}{2}}} \\ &\times \exp\left\{-\frac{1}{2}s(1-s)(\mu_1-\mu_2)^T((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}(\mu_1-\mu_2)\right\}\,.\end{aligned}$$

It remains to evaluate the coefficient $\frac{|A|^{1/2}}{|\mathbf{C}_1|^{\frac{s}{2}}|\mathbf{C}_2|^{\frac{1-s}{2}}}$. Taking the determinant of both sides of Equation 125 and solving for the expression A defined in Equation 116 we find

$$|A| = \frac{|\mathbf{C}_1||\mathbf{C}_2|}{|(1-s)\mathbf{C}_1 + s\mathbf{C}_2|}\,. \tag{132}$$

When we put this into what we have found for $\int p(x|\omega_1)^s p(x|\omega_2)^{1-s} dx$ we obtain

$$\begin{aligned}\int p(x|\omega_1)^s p(x|\omega_2)^{1-s} dx &= \frac{|\mathbf{C}_1|^{\frac{1-s}{2}}|\mathbf{C}_2|^{\frac{s}{2}}}{|(1-s)\mathbf{C}_1 + s\mathbf{C}_2|^{\frac{1}{2}}} \\ &\times \exp\left\{-\frac{1}{2}s(1-s)(\mu_1-\mu_2)^T((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}(\mu_1-\mu_2)\right\}.\end{aligned}$$

If we define the above expression equal to $e^{-k(s)}$ we see that $k(s)$ is given by

$$\begin{aligned}k(s) &= \frac{1}{2}s(1-s)(\mu_1-\mu_2)^T((1-s)\mathbf{C}_1 + s\mathbf{C}_2)^{-1}(\mu_1-\mu_2) \quad (133)\\ &+ \frac{1}{2}\log\left\{\frac{|(1-s)\mathbf{C}_1 + s\mathbf{C}_2|}{|\mathbf{C}_1|^{1-s}|\mathbf{C}_2|^s}\right\}. \quad (134)\end{aligned}$$

When this is combined with Equation 112 we have the expression given in the books equation 6.19 also known as the Chernoff distance.

Notes on feature extraction based on the Bhattacharyya distance

The Chernoff distance between to multidimensional Gaussians (derived above) is given by

$$J_C(s) = \frac{1}{2}s(1-s)(\mu_2-\mu_1)^T[(1-s)C_1 + sC_2]^{-1}(\mu_2-\mu_1) + \frac{1}{2}\ln\left[\frac{|(1-s)C_1 + sC_2|}{|C_1|^{1-s}|C_2|^s}\right]. \quad (135)$$

when $s = 1/2$ this becomes the Bhattacharyya distance and is given by

$$J_{\text{BHAT}} = \frac{1}{4}(\mu_2-\mu_1)^T[C_1 + C_2]^{-1}(\mu_2-\mu_1) + \frac{1}{2}\ln\left[\frac{|C_1 + C_2|}{2^N|C_1|^{1/2}|C_2|^{1/2}}\right], \quad (136)$$

where N is the dimension of the state space. If $C_1 = C_2 = C$ then this simplifies to give

$$J_{\text{BHAT}} = \frac{1}{8}(\mu_2-\mu_1)^T C^{-1}(\mu_2-\mu_1). \quad (137)$$

If we now desire to transform our initial vectors z as $y = Wz$ the means of our distribution transform as $W\mu_k$ and the covariance matrices will transform as WC_kW^T. Thus the Bhattacharyya distance in Equation 136 becomes

$$\begin{aligned}J_{\text{BHAT}} &= \frac{1}{4}(W\mu_2 - W\mu_1)^T\left[WC_1W^T + WC_2W^T\right]^{-1}(W\mu_2 - W\mu_1) \quad (138)\\ &+ \frac{1}{2}\ln\left[\frac{|WC_1W^T + WC_2W^T|}{2^N|WC_1W^T|^{1/2}|WC_2W^T|^{1/2}}\right],\end{aligned}$$

which is the books equation 6.35, when we define $m = \mu_1 - \mu_2$ and recall that N is the dimension of the state vector.

Exercise Solutions

Exercise 1 (simplifying $\overline{\rho^2}$)

Recall the books equation 6.4 or the average squared distance between points, $\overline{\rho^2}$, of

$$\overline{\rho^2} = \frac{1}{N_S^2} \sum_{n=1}^{N_S} \sum_{m=1}^{N_S} (z_n - z_m)^T (z_n - z_m) \,. \tag{139}$$

To solve this exercise lets follow the hint and write the difference $z_n - z_m$ as

$$z_n - z_m = z_{k,n} - z_{l,m} \,, \tag{140}$$

where we have assumed that the sample z_n is in class k and the sample z_m is in class l. Now by writing the sum over all samples as a paired sum first over classes and second over the points that are members of each class as

$$\sum_{n=1}^{N_S} = \sum_{k=1}^{K} \sum_{n=1}^{N_k} ,$$

we see that $\overline{\rho^2}$ becomes under this convention

$$\overline{\rho^2} = \frac{1}{N_S^2} \sum_{k=1}^{K} \sum_{n=1}^{N_k} \sum_{l=1}^{K} \sum_{m=1}^{N_l} (z_{k,n} - z_{l,m})^T (z_{k,n} - z_{l,m}) \,.$$

Now use the decomposition

$$z_{k,n} - z_{l,m} = (z_{k,n} - \hat{\mu}_k) + (\hat{\mu}_k - \hat{\mu}) + (\hat{\mu} - \hat{\mu}_l) + (\hat{\mu}_l - z_{l,m}) \,,$$

when we put all the "square" terms first and the cross terms second we can simplify the inner product in the summation above as

$$\begin{aligned}
(z_{k,n} - z_{l,m})^T (z_{k,n} - z_{l,n}) &= (z_{k,n} - \hat{\mu}_k)^T (z_{k,n} - \hat{\mu}_k) + (\hat{\mu}_k - \hat{\mu})^T (\hat{\mu}_k - \hat{\mu}) \\
&+ (\hat{\mu} - \hat{\mu}_l)^T (\hat{\mu} - \hat{\mu}_l) + (\hat{\mu}_l - z_{l,m})^T (\hat{\mu}_l - z_{l,m}) \\
&+ 2(z_{k,n} - \hat{\mu}_k)^T (\hat{\mu}_k - \hat{\mu}) + 2(z_{k,n} - \hat{\mu}_k)^T (\hat{\mu} - \hat{\mu}_l) \\
&+ 2(z_{k,n} - \hat{\mu}_k)^T (\hat{\mu}_l - z_{l,m}) \\
&+ 2(\hat{\mu}_k - \hat{\mu})^T (\hat{\mu} - \hat{\mu}_l) + 2(\hat{\mu}_k - \hat{\mu})^T (\hat{\mu}_l - z_{l,m}) \\
&+ 2(\hat{\mu} - \hat{\mu}_l)^T (\hat{\mu}_l - z_{l,m}) \,.
\end{aligned}$$

Using this decomposition we will now perform the summation $\sum_{k=1}^{K} \sum_{n=1}^{N_k} \sum_{l=1}^{K} \sum_{m=1}^{N_l} (\cdot)$ "on" each term. Remembering that $\sum_{k=1}^{K} \sum_{n=1}^{N_k} 1 = N_S$, we find the sums of the first term given by

$$N_S \sum_{k=1}^{K} \sum_{n=1}^{N_k} (z_{k,n} - \hat{\mu}_k)^T (z_{k,n} - \hat{\mu}_k) \,. \tag{141}$$

The sums of second term given by

$$N_S \sum_{k=1}^{K} N_k (\hat{\mu}_k - \hat{\mu})^T (\hat{\mu}_k - \hat{\mu}) \,. \tag{142}$$

The sums of the third term given by

$$N_S \sum_{l=1}^{K} N_l (\hat{\mu} - \hat{\mu}_l)^T (\hat{\mu} - \hat{\mu}_l) \,, \tag{143}$$

which equals the sums of the second term Equation 142 . The sums of the fourth term given by

$$N_S \sum_{l=1}^{K} \sum_{m=1}^{N_l} (\hat{\mu}_l - z_{l,m})^T (\hat{\mu}_l - z_{l,m}) \,, \tag{144}$$

which equals the sums of the first term Equation 141. For the sums of the fifth term we find

$$\begin{aligned} 2N_S \sum_{k=1}^{K} \sum_{n=1}^{N_k} (z_{k,n} - \hat{\mu}_k)^T (\hat{\mu}_k - \hat{\mu}) &= 2N_S \sum_{k=1}^{K} \left(\sum_{n=1}^{N_k} (z_{k,n} - \hat{\mu}_k)^T \right) (\hat{\mu}_k - \hat{\mu}) \\ &= 2N_S \sum_{k=1}^{K} \left(\sum_{n=1}^{N_k} (z_{k,n}^T - \hat{\mu}_k^T) \right) (\hat{\mu}_k - \hat{\mu}) \,, \end{aligned}$$

but this inner sum simplifies greatly

$$\sum_{n=1}^{N_k} (z_{k,n}^T - \hat{\mu}_k^T) = N_k \hat{\mu}_k^T - N_k \hat{\mu}_k^T = 0 \,,$$

and so this term vanishes. Using the same logic as in computing the fifth term we see that the sixth, seventh, ninth, and tenth terms also vanish. The only remaining term is then the eighth and we have

$$\begin{aligned} 2 \sum_{k=1}^{K} \sum_{n=1}^{N_k} \sum_{l=1}^{K} \sum_{m=1}^{N_l} (\hat{\mu}_k - \hat{\mu})^T (\hat{\mu}_l - \hat{\mu}) &= 2 \sum_{k=1}^{K} \sum_{l=1}^{K} N_k N_l (\hat{\mu}_k - \hat{\mu})^T (\hat{\mu}_l - \hat{\mu}) \\ &= 2 \sum_{l=1}^{K} N_l \left(\sum_{k=1}^{K} N_k (\hat{\mu}_k - \hat{\mu})^T \right) (\hat{\mu} - \hat{\mu}_l) \,. \end{aligned}$$

The inner sum in the above simplifies greatly and its transpose is given by

$$\begin{aligned} \sum_{k=1}^{K} N_k (\hat{\mu}_k - \hat{\mu}) &= \sum_{k=1}^{K} N_k \hat{\mu}_k - \hat{\mu} \sum_{k=1}^{K} N_k \\ &= \sum_{k=1}^{K} \sum_{n=1}^{N_k} z_{k,n} - N_S \hat{\mu} \\ &= \sum_{k=1}^{K} \sum_{n=1}^{N_k} z_{k,n} - \sum_{n=1}^{N_S} z_n = 0 \,. \end{aligned}$$

Thus using all of these relations we find

$$\begin{aligned}\overline{\rho^2} &= \frac{2}{N_S}\sum_{k=1}^{K}\sum_{n=1}^{N_k}(z_{k,n}-\hat{\mu}_k)^T(z_{k,n}-\hat{\mu}_k)+\frac{2}{N_S}\sum_{k=1}^{K}N_k(\hat{\mu}-\hat{\mu}_k)^T(\hat{\mu}-\hat{\mu}_k) \\ &= \frac{2}{N_S}\sum_{k=1}^{K}\left[\sum_{n=1}^{N_k}\left((z_{k,n}-\hat{\mu}_k)^T(z_{k,n}-\hat{\mu}_k)+(\hat{\mu}-\hat{\mu}_k)^T(\hat{\mu}-\hat{\mu}_k)\right)\right], \qquad (145)\end{aligned}$$

which is the books equation 6.4 and was we were to show.

Exercise 3 (l-takeaway-r with $l > r$ vs. plus-l-takeaway-r selection with $l < r$)

A problem where the features are highly or largely independent would do best with plus-l-takeaway-r selection since on each evaluation of the classifier when the features are independent we expect sequential forward search (SFS) to work quite well as each new feature adds novel information. If certain features are in fact *more* informative than others these features would be selected early on and keep during the entire search process.

In contrast, we would prefer backwards selection or l-takeaway-r if the features are expected to be highly correlated. In that case only when the features are considered in total (i.e. as a group) would we expect to get the maximal information out of them.

Exercise 4 ($W = m^T C^{-1}$ when we maximize the Bhattacharyyaa distance)

Before we begin this problem we note one point that is easy to become confused about. As suggested, if we assume our covariance matrices are equal than $C = C_1 = C_2$ and Equation 138 becomes

$$J_{\text{BHAT}}(W) = \frac{1}{8}m^T W^T (WCW^T)^{-1} Wm\,.$$

We might (*incorrectly*) think that we can simplify the matrix product in the above using

$$W^T(WCW^T)^{-1}W = W^T W^{-T} C^{-1} W^{-1} W = C^{-1}\,,$$

giving a result which is independent of W! The reason these transformation are not legal is because W is a feature projection matrix of dimension $D \times N$ with $D \ll N$ and so is *not* invertable. Thus the matrix W^{-1} is not defined and cannot be used as above. The apporporiate way to solve this problem is to consider $J_{\text{BHAT}}(W)$ as a function of W defined above, compute its first derivative $J'_{\text{BHAT}}(W)$ with respect to this argument, set it equal to zero and solve for W. We could compute the derivative of $J_{\text{BHAT}}(W)$ as expressed above but we can be slightly more general and compute the derivative of the *Chernoff* distance function under the linear transformation implied by W. In deriving the results of this derivative we note that they rely heavily on the results from the theory of matrix derivatives of scalars functions of matrices. Results on this topic are nicely explained and derived in [1]. The associated study guide [7], has additional algebra and comments on these derivations.

To begin the derivation of the derivative of the Chernoff distance metric $J_C(W)$, we will modify slighlty how we derive our linearly transformed vectors y are obtained from our raw input vectors z via the linear mapping $y = W^t z$. Note this definition just uses the *transpose* of W to perform our mapping rather than W directly. Now as the expression for $J_C(W)$ is a *scalar* we can take the trace of its expression and use the properties of the trace operator to simplify some resulting expressions. Taking this trace, the Chernoff separability measure can be transformed as

$$\begin{aligned} J_C(W) &= \frac{1}{2}s(1-s)\mathrm{tr}\left\{(\mu_2-\mu_1)^t W\left[(1-s)W^tC_1W + sW^tC_2W\right]^{-1}W^t(\mu_2-\mu_1)\right\} \quad (146)\\ &+ \frac{1}{2}\ln\left|(1-s)W^tC_1W + sW^tC_2W\right| \\ &- \frac{1-s}{2}\ln\left|W^tC_1W\right| - \frac{s}{2}\ln\left|W^tC_2W\right|. \end{aligned}$$

To simplify the notation we will define the matrix L as

$$L \equiv \left[(1-s)W^tC_1W + sW^tC_2W\right]^{-1}.$$

Since we can cyclically permute the arguments of a trace we have that the first term in Equation 146 can be written as

$$\mathrm{tr}\left\{(\mu_2-\mu_1)^tWLW^t(\mu_2-\mu_1)\right\} = \mathrm{tr}\left\{W^t(\mu_2-\mu_1)(\mu_2-\mu_1)^tWL\right\}.$$

Lets also define the matrix M as $M \equiv (\mu_2-\mu_1)(\mu_2-\mu_1)^t$. To evaluate the value of W at which we will maximize the value of $J_C(W)$ we need to compute the first derivative of the scalar $J_C(W)$ with respect to the matrix W. To compute this lets take the derivative of each term one at a time. Using the results from [1] and [7] we find this *matrix* derivative then becomes

$$\frac{\partial \mathrm{tr}\left(W^tMWL\right)}{\partial W} = 2MWL - 2((1-s)C_1W + sC_2W)LW^tMWL.$$

Next we need to take the derivatives with respect to W of the expressions

$$\ln\left|(1-s)W^tC_1W + sW^tC_2W\right|, \ln\left|W^tC_1W\right|, \text{and } \ln\left|W^tC_2W\right|.$$

Since we can write

$$(1-s)W^tC_1W + sW^tC_2W = W^t((1-s)C_1 + sC_2)W,$$

all of these expressions have similar derivatives. Taking the derivative of $\ln|W^tC_2W|$ we find

$$\frac{\partial \ln\left|W^tC_2W\right|}{\partial W} = 2C_2W(W^tC_2W)^{-1}.$$

All the other derivatives are similar. For example

$$\begin{aligned} \frac{\partial \ln\left|(1-s)W^tC_1W + sW^tC_2W\right|}{\partial W} &= 2((1-s)C_1 + sC_2)W(W^t((1-s)C_1 + sC_2)W)^{-1} \\ &= 2((1-s)C_1W + sC_2W)((1-s)W^tC_1W + sW^tC_2W)^{-1} \\ &= 2((1-s)C_1W + sC_2W)L. \end{aligned}$$

Thus combining everything we finally find that our first derivative of $J_C(W)$ is given by

$$\begin{aligned}\frac{\partial J_C(W)}{\partial W} &= s(1-s)\left\{MWL - ((1-s)C_1W + sC_2W)LW^tMWL\right\} \\ &+ ((1-s)C_1W + sC_2W)L \\ &- (1-s)C_1W(W^tC_1W)^{-1} - sC_2W(W^tC_2W)^{-1}\,.\end{aligned}$$

To find the maximum of $J_C(W)$ we set $J_C'(W)$ equal to zero to derive an equation for W which would then need to be solved for W. Assuming L is non-singular we can premultiply by L^{-1} and divide by $s(1-s)$ to get the following equation equivalent to $J_C'(W) = 0$

$$MW - [(1-s)C_1W + sC_2W]LW^tMW + \frac{1}{s(1-s)}[(1-s)C_1W + sC_2W] \quad (147)$$

$$- \frac{1}{s}C_1W(W^tC_1W)^{-1}[(1-s)W^tC_1W + sW^tC_2W] \quad (148)$$

$$- \frac{1}{1-s}C_2W(W^tC_2W)^{-1}[(1-s)W^tC_1W + sW^tC_2W] = 0\,. \quad (149)$$

Note that in the above the last two terms can be expanded and written as

$$-\frac{1}{s}C_1W[(1-s)I + s(W^tC_1W)^{-1}W^tC_2W] - \frac{1}{1-s}C_2W[(1-s)(W^tC_2W)^{-1}W^tC_1W + sI]\,.$$

We can further combine these terms with $\frac{1}{s(1-s)}[(1-s)C_1W + sC_2W]$ the third term on line 147 above to get

$$\begin{aligned}&C_1W + C_2W - C_1W(W^tC_1W)^{-1}W^tC_2W - C_2W(W^tC_2W)^{-1}W^tC_1W \\ =\ &C_1W[I - (W^tC_1W)^{-1}W^tC_2W] + C_2W[I - (W^tC_2W)^{-1}W^tC_1W]\,.\end{aligned}$$

Combining these gives the following

$$\begin{aligned}&MW - [(1-s)C_1W + sC_2W]LW^tMW \\ +\ &C_1W[I - (W^tC_1W)^{-1}W^tC_2W] + C_2W[I - (W^tC_2W)^{-1}W^tC_1W] = 0\end{aligned} \quad (150)$$

In the special case where both Gaussians have the *same* covariance matrix then $C_1 = C_2 = \Sigma$ and the last two terms in Equation 150 vanish so that Equation 150 becomes

$$MW - \Sigma W(W^t\Sigma W)^{-1}W^tMW = 0\,. \quad (151)$$

Performing an eigenvector-eigenvalue decomposition of $(W^t\Sigma W)^{-1}W^tMW$ to write this matrix as $U\Lambda U^{-1}$ we have

$$MW - \Sigma WU\Lambda U^{-1} = 0\,,$$

or

$$\Sigma^{-1}MWU - WU\Lambda = 0\,. \quad (152)$$

Now since we don't explicitly know the value W we don't explicitly know the values of Λ or U and they are effectively functions of the matrix W. We will see below how to compute these matrices. On grouping some terms together we find

$$\Sigma^{-1}M(WU) = (WU)\Lambda\,.$$

Since Λ is a *diagonal* matrix containing the eigenvalues of $(W^t\Sigma W)^{-1}W^tMW$, multiplication on the right by this matrix Λ is equivalent to multiplying each column of the matrix WU by the corresponding eigenvalue. Comparing each side of this equation column by column we see that the columns of WU must be the eigenvectors of the matrix $\Sigma^{-1}M$. From this we can compute the eigenvectors of $\Sigma^{-1}M$ and place them as columns of the matrix say V. Then since $V = WU$ for some as yet unknown U, W would be given by $W = VU^{-1}$. The point to note now is that in fact the multiplication of V by an invertible matrix U^{-1} does not in fact change the value of J_C and it can be ignored. The fact that multiplication by U^{-1} on the left does not change the value of J_C can be seen by first considering $J_C(W)$ with equal covariance matrices. We find

$$\begin{aligned} J_C(W) &= \frac{1}{2}s(1-s)\mathrm{tr}\left\{W^tMW(W^t\Sigma W)^{-1}\right\} \\ &+ \frac{1}{2}\ln|W^t\Sigma W| - \frac{1-s}{2}\ln|W^t\Sigma W| - \frac{s}{2}\ln|W^t\Sigma W| \\ &= \frac{1}{2}s(1-s)\mathrm{tr}\left\{W^tMW(W^t\Sigma W)^{-1}\right\} . \end{aligned}$$

If we consider $J_C(WU^{-1})$ we find

$$\begin{aligned} J_C(WU^{-1}) &= \frac{1}{2}s(1-s)\mathrm{tr}\left\{U^{-t}W^tMWU^{-1}(U^{-t}W^t\Sigma WU^{-1})^{-1}\right\} \\ &= \frac{1}{2}s(1-s)\mathrm{tr}\left\{U^{-t}W^tMWU^{-1}U(W^t\Sigma W)^{-1}U^t\right\} \\ &= \frac{1}{2}s(1-s)\mathrm{tr}\left\{U^tU^{-t}W^tMW(W^t\Sigma W)^{-1}\right\} = J_C(W) . \end{aligned}$$

Since the value of U^{-1} does not matter we can effectively ignore this matrix (take $U = I$). Then the matrix W has columns that are simply the eigenvectors of $\Sigma^{-1}M$. When we use the decomposition $(W^t\Sigma W)^{-1}W^tMW = U\Lambda U^{-1}$ in the above expression for $J_C(W)$ we find

$$\begin{aligned} J_C(W) &= \frac{1}{2}s(1-s)\mathrm{tr}\left\{W^tMW(W^t\Sigma W)^{-1}\right\} \\ &= \frac{1}{2}s(1-s)\mathrm{tr}\left\{U\Lambda U^{-1}\right\} = \frac{1}{2}s(1-s)\mathrm{tr}\left\{U^{-1}U\Lambda\right\} \\ &= \frac{1}{2}s(1-s)\mathrm{tr}\left\{\Lambda\right\} . \end{aligned}$$

Since M is of rank one the product $\Sigma^{-1}M$ will also be of rank one and only have *one* non-zero eigenvalue. Thus the matrix W will thus in fact be only a column vector and not a matrix. To find its value we could explicitly compute the non-zero eigenvector of $\Sigma^{-1}M$, but an easier method it to write Equation 152 when W is a column vector say v_1 as

$$\Sigma^{-1}(\mu_2-\mu_1)(\mu_2-\mu_1)^tv_1 = \lambda v_1 .$$

Since $(\mu_2-\mu_1)^tv_1$ is an inner product and is therefore a scalar we can factor it out to get

$$\left((\mu_2-\mu_1)^tv_1\right)\Sigma^{-1}(\mu_2-\mu_1) = \lambda v_1 .$$

Comparing vectors (and the corresponding multiplying scalars) on each side of this expression we see that

$$\begin{aligned} v_1 &= \Sigma^{-1}(\mu_2-\mu_1) \\ \lambda &= (\mu_2-\mu_1)^tv_1 = (\mu_2-\mu_1)^t\Sigma^{-1}(\mu_2-\mu_1) . \end{aligned} \tag{153}$$

Equation 153 is the optimal Chernoff feature transformation

$$y = v_1^t z = (\mu_2 - \mu_1)^t \Sigma^{-1} z \,, \tag{154}$$

for the case when the covariance matrices of the two densities are equal.

Exercise 7 (optimizing the number of features to retain on the training set)

The problem with optimizing the number of features to use for classification on the training set is that due to the finite nature of the samples in the training set we might incorrectly optimized based on the noise in the data set rather than on something statistically stable. Recall the books Figure 6.1 where the classification error rate is viewed as a function of the measurement space dimension for several values of the number of samples N_S. In a given classification problem N_S will be specified at the outset of the problem and picking the number of features based on attempting to achieve an minimum error rate will certainly not be optimal as we can see from the two examples with $N_S = 20$ and $N_S = 80$ in this figure. There, the finite sample minimum is not equal to the theoretical minimum (where $N \to \infty$). We can avoid this problem by selecting the features subsets using cross-validation by evaluating the classifier on a set of date *not* used for the feature selection process.

Chapter 7: (Unsupervised Learning)

Notes on the text

Notes on Multi-dimensional scaling

From the given *stress* measure E_S

$$E_S = \frac{\sum_{i=1}^{N_S}\sum_{j=i+1}^{N_S}(\delta_{ij}-d_{ij})^2}{\sum_{i=1}^{N_S}\sum_{j=i+1}^{N_S}\delta_{ij}^2}\,, \tag{155}$$

when d_{ij} is defined as a Euclidean distance on D dimensional vectors then

$$d_{ij} = \sqrt{\sum_{l=1}^{D}(y_l(z_i)-y_l(z_j))^2}\,,$$

and the derivative with respect to the l-th component of $y(z_i)$ is given by (these are scalar derivatives)

$$\frac{\partial d_{ij}}{\partial y_{il}} = \frac{2(y_l(z_i)-y_l(z_j))}{2\sqrt{\sum_{l=1}^{D}(y_l(z_i)-y_l(z_j))^2}} = \frac{y_l(z_i)-y_l(z_j)}{d_{ij}}\,.$$

Which in vector form looks like

$$\frac{\partial d_{ij}}{\partial \mathbf{y_i}} = \frac{\mathbf{y}_i-\mathbf{y}_j}{d_{ij}}\,. \tag{156}$$

Thus the $\mathbf{y}_i$ derivative of E_S is given by

$$\begin{aligned}\frac{\partial E_S}{\partial \mathbf{y}_i} &= -\left(\frac{2}{\sum_{i=1}^{N_S}\sum_{j=i+1}^{N_S}\delta_{ij}^2}\right)\sum_{j=i+1}^{N_S}(\delta_{ij}-d_{ij})\frac{\partial d_{ij}}{\partial \mathbf{y_i}} \\ &= -\left(\frac{2}{\sum_{i=1}^{N_S}\sum_{j=i+1}^{N_S}\delta_{ij}^2}\right)\sum_{j=i+1}^{N_S}\frac{(\delta_{ij}-d_{ij})}{d_{ij}}(\mathbf{y}_i-\mathbf{y}_j)\,,\end{aligned}$$

the result given in the book.

Notes on clustering via mixtures of Gaussians

Here we present some notes on the EM derivation from this section. Since the vector x_n has position coding (the K components of this vector $x_{n,k}$ for $k = 1, 2, \cdots K$ are 1 if k is the index of the cluster that the sample y_n belongs to and 0 otherwise) and the *prior* probability the k-th component of x_n will take the value of 1 is given by π_k as shorthand we can write this as

$$p(x_n|\Psi) = \prod_{k=1}^{K}\pi_k^{x_{n,k}}\,,$$

since $x_{n,k}$ is zero for all but one element say k' and that element is 1 i.e. $x_{n,k'} = 1$. In the same way the probability we obtain feature vector z_n depends on which cluster z_n originates from. Using the same shorthand as above we have

$$p(z_n|x_n, \Psi) = \prod_{k=1}^{K} (N(z_n|\mu_k, C_k))^{x_{n,k}} .$$

Thus the *joint* distribution $p(z_n, x_n|\Psi)$ is given by the product of these two expressions

$$\begin{aligned} p(z_n, x_n|\Psi) &= p(z_n|x_n, \Psi)p(x_n|\Psi) \\ &= \prod_{k=1}^{K} (N(z_n|\mu_k, C_k))^{x_{n,k}} \times \prod_{k=1}^{K} \pi_k^{x_{n,k}} \\ &= \prod_{k=1}^{K} (N(z_n|\mu_k, C_k)\pi_k)^{x_{n,k}} , \end{aligned}$$

in agreement with the equation 7.16 given in the book.

To derive the EM algorithm lets recall the definitions of the vectors $\mathbf{Z}$, $\mathbf{\Psi}^{(i)}$, $\mathbf{X}$, and $\mathbf{Y} = \begin{bmatrix} \mathbf{Z} \\ \mathbf{X} \end{bmatrix}$ that will be used in the notation that follows. We have

- $\mathbf{Z}$ the observed data for which we *have* realizations of.
- $\mathbf{\Psi}^{(i)}$ the parameter of the distribution $\mathbf{Z}$ under the assumption that $\mathbf{Z}$ is distributed as a Gaussian mixture model. Thus $\mathbf{\Psi}^{(i)}$ is represented by K mean vectors μ_k and K covariance matrices C_k.
- $\mathbf{X}$ the missing data that we *don't have* realizations of. In this case this can considered to be the class labels of the data samples z_n.
- $\mathbf{Y}$ the vector of combined observed data z_n and the missing data x_n.

With these definitions the EM algorithm specifies that we iterate the following two equations (an *expectation* step followed by an *maximization* step):

$$E_Y[L(Y|X)|Z] = \int \ln(p(Y|\Psi)p(Y|Z, \Psi^{(i)})dY \tag{157}$$

$$\Psi^{(i+1)} = \text{argmax}_\Psi\{E_Y[L(Y|X)|Z]\} . \tag{158}$$

In the case where the missing data represents the cluster labels (the cluster that each sample z_n is drawn from) then one way to represent this information is to provide x_n as "position coded" (see above). With this notation, the log likelihood of all of the data given our parameters of the model takes the form

$$L(Y|\Psi) = \sum_{k=1}^{K} \sum_{n=1}^{N_S} x_{n,k} \ln(N(z_n|\mu_k, C_k)) + x_{n,k} \ln(\pi_k) . \tag{159}$$

Since we have an expression for $L(Y|\Psi)$ we need an expression for $p(Y|Z,\Psi^{(i)})$ from which to take the expectation with as specified in Equation 157. As Y represents all of the possible data (Z and X combined) we see that conditioning on Z we have

$$p(Y|Z,\Psi^{(i)}) = p(Z,X|Z,\Psi^{(i)}) = p(X|Z,\Psi^{(i)})\,,$$

so the density in expectation step in the EM-algorithm (Equation 157) becomes

$$\begin{aligned}\int L(Y|\Psi)p(Y|Z,\Psi^{(i)})dY &= \int L(Y|\Psi)p(X|Z,\Psi^{(i)})dX \\ &= \int L(X,Z|\Psi)p(X|Z,\Psi^{(i)})dX\,. \end{aligned} \tag{160}$$

Now from Equation 159 we see that $L(Y|\Psi)$ is *linear* in the variable X, implying that the expectation of L in Equation 160 is simply evaluating L at the expected value of X as

$$\begin{aligned} E_Y[L(Y|X)|Z] &= \int L(X,Z|\Psi)p(X|Z,\Psi^{(i)})dX \\ &= L\left(\int Xp(X|Z,\Psi^{(i)}),Z|\Psi\right)dX \\ &= L\left(\overline{X},Z|\Psi\right)\,. \end{aligned} \tag{161}$$

Here $\overline{X}$ the expectation of the data we don't have under the assumption that we know Z and $\Psi^{(i)}$ i.e. the data cluster labels and the parameters. Computing this expectation since $x_{n,k}$ is an indicator random variable expectations become probabilities and we see that

$$\bar{x}_{n,k} = E[x_{n,k}|z_n,\Psi^{(i)}] = P(x_{n,k}|z_n,\Psi^{(i)})\,.$$

From Bayes' rule this probability can be written as

$$\begin{aligned} p(x_{n,k}|z_n,\Psi^{(i)}) &= \frac{p(z_n|x_{n,k},\Psi^{(i)})}{p(z_n|\Psi^{(i)})} \\ &= \frac{N(z_n|\mu_k^{(i)},C_k^{(i)})\pi_k^{(i)}}{\sum_{k=1}^K N(z_n|\mu_k^{(i)},C_k^{(i)})\pi_k^{(i)}}\,. \end{aligned}$$

Where the variable $\bar{x}_{k,n}$ is called the *ownership*. For the M-step we maximize over the parameters Ψ the expression

$$\begin{aligned} E[L(Y|\Psi)|Z] &= L[\bar{x},Z|\Psi] \\ &= \sum_{k=1}^K\sum_{n=1}^{N_S} x_{n,k}\ln(N(z_n|\mu_k,C_k)) + x_{n,k}\ln(\pi_k)\,. \end{aligned}$$

Here the parameters to maximize over are μ_k, C_k, and π_k, subject to the constraint that $\sum_k \pi_k = 1$. The classical way to solve this maximization is using the method of Lagrange multipliers. In that method we would extend $L(Y|\Psi)$, creating a new objective function $L'(Y|\Psi)$, to include a Lagrange multiplier (denoted by λ) to enforce the constraint that $\sum_k \pi_k = 1$ as

$$L'(Y|\Psi) = \sum_{k=1}^K\sum_{n=1}^{N_S} x_{n,k}\ln(N(z_n|\mu_k,C_k)) + x_{n,k}\ln(\pi_k) - \lambda\left(\sum_{k=1}^K \pi_k - 1\right)\,. \tag{162}$$

We then proceed to maximize this expression by taking derivatives with respect to the variables μ_k, C_k, and π_k, setting the resulting expressions equal to zero, and solving the resulting equations for them. We begin by taking $\frac{\partial}{\partial \mu_k}$ of $E[L(Y|\Psi)|Z]$. We find

$$\frac{\partial}{\partial \mu_k} E[L(Y|\Psi)|Z] = \sum_{k=1}^{K} \bar{x}_{n,k} \left(\frac{1}{N(z_n|\mu_k, C_k)} \right) \left(\frac{\partial}{\partial \mu_k} N(z_n|\mu_k, C_k) \right) .$$

This derivative required in the above is given by

$$\begin{aligned} \frac{\partial}{\partial \mu_k} N(z_n|\mu_k, C_k) &= N(z_n|\mu_k, C_k) \frac{\partial}{\partial \mu_k} \left(-\frac{1}{2}(z_n - \mu_k)^t C_k^{-1} (z_n - \mu_k) \right) \\ &= \frac{1}{2} N(z_n|\mu_k, C_k) \left(C_k^{-1} + C_k^{-T} \right) (z_n - \mu_k) \\ &= N(z_n|\mu_k, C_k) C_k^{-1} (z_n - \mu_k) . \end{aligned} \tag{163}$$

Thus

$$\frac{\partial}{\partial \mu_k} E[L(Y|\Psi)|Z] = \sum_{n=1}^{N_S} \bar{x}_{n,k} C_k^{-1} (z_n - \mu_k) , \tag{164}$$

which is the books equation 7.20. Setting this expression equal to zero and solving for μ_k we have

$$\mu_k = \frac{\sum_{n=1}^{N_S} \bar{x}_{n,k} z_n}{\sum_{n=1}^{N_S} \bar{x}_{n,k}} , \tag{165}$$

which is the books equation 7.21. Next we take the derivative of $E[L(Y|\Psi)|Z]$ with respect to C_k. Which we will evaluate using the chain rule transforming the derivative with respect to C_k into one with respect to C_k^{-1}. We have

$$\frac{\partial}{\partial C_k} E[L(Y|\Psi)|Z] = \frac{\partial}{\partial {C_k}^{-1}} E[L(Y|\Psi)|Z] \frac{\partial {C_k}^{-1}}{\partial C_k} .$$

Thus if $\frac{\partial}{\partial {C_k}^{-1}} E[L(Y|\Psi)|Z] = 0$, we have that $\frac{\partial}{\partial C_k} E[L(Y|\Psi)|Z] = 0$ also. From this we can look for zeros of the derivative by looking for values of C_k where the derivative of the *inverse* of C_k vanishes. Taking the derivative of $E[L(Y|\Psi)|Z]$ with respect to ${C_k}^{-1}$ we find

$$\begin{aligned} \frac{\partial}{\partial {C_k}^{-1}} E[L(Y|\Psi)|Z] &= \sum_{n=1}^{N_S} \bar{x}_{n,k} \frac{\partial}{\partial {C_k}^{-1}} \ln(N(z_n|\mu_k, C_k)) \\ &= \sum_{n=1}^{N_S} \bar{x}_{n,k} \left(\frac{1}{N(z_n|\mu_k, C_k)} \right) \frac{\partial}{\partial {C_k}^{-1}} N(z_n|\mu_k, C_k) . \end{aligned}$$

From which we see that as a sub problem we need to compute $\frac{\partial}{\partial {C_k}^{-1}} N(z_n|\mu_k, C_k)$, which we now do

$$\begin{aligned} \frac{\partial}{\partial {C_k}^{-1}} N(z_n|\mu_k, C_k) &= \frac{\partial}{\partial {C_k}^{-1}} \left(\frac{1}{(2\pi)^N |C_k|^{1/2}} \exp\left\{ -\frac{1}{2}(z_n - \mu_k)^T {C_k}^{-1} (z_n - \mu_k) \right\} \right) \\ &= \frac{1}{(2\pi)^N} \frac{\partial}{\partial {C_k}^{-1}} \left(\frac{1}{|C_k|^{1/2}} \right) \exp\left\{ -\frac{1}{2}(z_n - \mu_k)^T {C_k}^{-1} (z_n - \mu_k) \right\} \\ &+ \frac{1}{(2\pi)^N} \frac{1}{|C_k|^{1/2}} \frac{\partial}{\partial {C_k}^{-1}} \exp\left\{ -\frac{1}{2}(z_n - \mu_k)^T {C_k}^{-1} (z_n - \mu_k) \right\} , \end{aligned}$$

using the product rule. To evaluate the first derivative in the above we note that

$$\begin{aligned}\frac{\partial}{\partial {C_k}^{-1}}\left(\frac{1}{|C_k|^{1/2}}\right) &= \frac{\partial}{\partial {C_k}^{-1}}|{C_k}^{-1}|^{1/2} \\ &= \frac{1}{2}|{C_k}^{-1}|^{-1/2}\frac{\partial}{\partial {C_k}^{-1}}|{C_k}^{-1}|\,,\end{aligned}$$

but using the following matrix derivative of a determinant identity

$$\frac{\partial}{\partial \mathbf{X}}|\mathbf{AXB}| = |\mathbf{AXB}|(\mathbf{X}^{-1})^T = |\mathbf{AXB}|(\mathbf{X}^T)^{-1}\,, \tag{166}$$

with $A = B = I$ we have $\frac{\partial}{\partial X}|X| = |X|(X^{-1})^T$ and the derivative $\frac{\partial}{\partial {C_k}^{-1}}\left(\frac{1}{|C_k|^{1/2}}\right)$ becomes

$$\begin{aligned}\frac{\partial}{\partial {C_k}^{-1}}\left(\frac{1}{|C_k|^{1/2}}\right) &= \frac{1}{2}|{C_k}^{-1}|^{-1/2}|{C_k}^{-1}|{C_k}^T \\ &= \frac{1}{2}\frac{1}{|C_k|^{1/2}}C_k\,.\end{aligned}$$

Next using the matrix derivative of an inner product is given by

$$\frac{\partial}{\partial \mathbf{X}}(\mathbf{a}^T\mathbf{X}\mathbf{b}) = \mathbf{a}\mathbf{b}^T\,, \tag{167}$$

we have the derivative of the inner product expression

$$\frac{\partial}{\partial {C_k}^{-1}}\left\{-\frac{1}{2}(z_n-\mu_k)^T{C_k}^{-1}(z_n-\mu_k)\right\} = -\frac{1}{2}(z_n-\mu_k)(z_n-\mu_k)^T\,.$$

Putting everything together we find that

$$\begin{aligned}\frac{\partial}{\partial {C_k}^{-1}}N(z_n|\mu_k,C_k) &= \frac{1}{2}\frac{1}{(2\pi)^N}\frac{1}{|C_k|^{1/2}}\exp\left\{-\frac{1}{2}(z_n-\mu_k)^T{C_k}^{-1}(z_n-\mu_k)\right\}C_k \\ &- \frac{1}{2}N(z_n|\mu_k,C_k)(z_n-\mu_k)^T(z_n-\mu_k) \\ &= \frac{1}{2}N(z_n|\mu_k,C_k)\left(C_k-(z_n-\mu_k)(z_n-\mu_k)^T\right)\,.\end{aligned} \tag{168}$$

So combining these subproblems we finally find

$$\frac{\partial}{\partial {C_k}^{-1}}\ln(N(z_n|\mu_k,C_k)) = \frac{1}{2}\left(C_k-(z_n-\mu_k)(z_n-\mu_k)^T\right)\,, \tag{169}$$

or the books equation 7.22. Using this in the expression for $\frac{\partial}{\partial {C_k}^{-1}}E[L(Y|\Psi)|Z] = 0$, we find the equation

$$\sum_{n=1}^{N_S}\bar{x}_{n,k}C_k - \sum_{n=1}^{N_S}\bar{x}_{n,k}(z_n-\mu_k)(z_n-\mu_k)^T = 0\,.$$

Which when we solve for C_k we find

$$C_k = \frac{\sum_{n=1}^{N_S}\bar{x}_{n,k}(z_n-\mu_k)(z_n-\mu_k)^T}{\sum_{n=1}^{N_S}\bar{x}_{n,k}}\,, \tag{170}$$

which is the books equation 7.23. To complete a full maximization of $L'(Y|\Psi)$ with we still need to determine π_k the priori probabilities of the k-th cluster. Setting $\frac{\partial L'(Y|\Psi)}{\partial \pi_k} = 0$ gives

$$\sum_{n=1}^{N_S} \frac{\bar{x}_{n,k}}{\pi_k} - \lambda = 0 \,,$$

or

$$\lambda \pi_k = \sum_{n=1}^{N_S} \bar{x}_{n,k} \,.$$

Summing this equation over k for $k = 1$ to K since $\sum_{k=1}^{K} \pi_k = 1$ we have

$$\lambda = \sum_{k=1}^{K} \sum_{n=1}^{N_S} \bar{x}_{n,k} \,.$$

This can be simplified by observing that

$$\lambda = \sum_{k=1}^{K} \sum_{n=1}^{N_S} \bar{x}_{n,k} = \sum_{k=1}^{K} \sum_{n=1}^{N_S} p(x_{n,k}|z_n, \Psi^{(i)}) = \sum_{n=1}^{N_S} \sum_{k=1}^{K} p(x_{n,k}|z_n, \Psi^{(i)}) = \sum_{n=1}^{N_S} 1 = N_S \,.$$

Where we have used the fact that $p(x_{n,k}|z_n, \Psi^{(i)})$ is a probability that the sample z_n is from cluster k. Since there are $1, 2, \ldots, K$ clusters summing this probability gives one. Thus

$$\pi_k = \frac{1}{N_S} \sum_{n=1}^{N_S} \bar{x}_{n,k} \,, \tag{171}$$

which is the books equation 7.27. Combining this expression with Equations 165 and 170 gives the EM algorithm.

Exercise solutions

Exercise 4 (deriving the EM algorithm for mixtures of Gaussians)

See the notes that begin on page 65 for these derivations.

Exercise 5 (what data will be clustered and what data will be along a subspace)

Data that is generated from class specification that is which comes from class specific probability densities would be expect to be distributed in clusters. Clustering is used to group together the measurements which are most similar. This is like determining the classification of a set of features and one would expect that if the observed data was generated from different classes it would belong in different clusters. One would expect data that is subject to a constraint would like on a lower dimensional surface. This lower dimensional surface

can come from a physical constraint that must hold true on the object observed or from a mathematical constraint required by the representation of the data points. A very simple example of this later type of constraint would be if the object observed was constrained to lie on the unit sphere and was describe by

$$\mathbf{x} = \begin{bmatrix} \sin(\theta) \\ \cos(\theta) \end{bmatrix},$$

where θ is the angle with the object location on the unit circle and the x-axis. In this case the object in the representation above is *over-parameterized* and the components of each $\mathbf{x}$ vector must satisfy

$$x_1^2 + x_2^2 = 1\,,$$

i.e. belong on a lower dimensional subspace.

Exercise 6 (desirable properties of clustering algorithms)

One very simple desirable property of a cluster algorithm is repeatability. That is running the same algorithm on the same set of data should give the same set of cluster. For many clustering algorithms such as K-mean clustering which uses initial cluster centers distributed randomly this property is *not* explicitly satisfied.

Exercise 7 (K-means minimized $|S_w|$)

The K-means algorithm minimizes $|S_w|$ or the volume occupied by the within-class scatter matrix S_w, since if a point is too far from its currently assumed cluster center $\hat{\mu}_k$ it will be relocated/reassigned to another cluster effectively reducing the volume occupied by

$$S_w = \frac{1}{N_S}\sum_{k=1}^{K}\sum_{n=1}^{N_k}(z_{k,n} - \hat{\mu}_k)(z_{k,n} - \hat{\mu}_k)^T\,.$$

Exercise 8 (when does the EM algorithm degenerate to the K-means algorithm)

From Equations 165 and 170 the EM algorithm will become the K-means algorithm if the distance between cluster centers and the data points becomes the Euclidean distance, which will happen if the class covariance converge to the identity matrix (or a constant multiple of it). In addition, the ownership $\bar{x}_{n,k}$ needs to converge to an indicator function

$$\bar{x}_{n,k} \to \begin{cases} 1 & z_n \in \omega_k \\ 0 & \text{otherwise} \end{cases},$$

so that the update Equation 165 converges to the K-means update equation

$$\hat{\mu}_k = \frac{1}{N_k}\sum_{z_i \in C_k} z_i\,. \tag{172}$$

Exercise 9 (avoiding local minimums)

The easiest way to avoid local minimum is to perform multiple (or random) restarts of the clustering algorithm. With this simple technique one runs the same cluster algorithm several times with different initial conditions and ultimately returns the result that has the best fit with the data.

Exercise 10 (parameters controlling generalization)

For the self-organizing map (SOM) the parameters that most strongly control the generalization ability are the dimension of the grid and the number of neurons in the grid. For most neural net based techniques the more complicated the network the more capability the net has to generalize. The drawback is that the net also has a much stronger propensity to overfit the data and learn spurious relationships.

Chapter 8: (State Estimation in Practice)

Notes on the Text

Notes on Dynamic Stability and Steady-State Solutions

If we assume that our Kalman filter is operating on a linear time invariant system we can derive a recursive update equation for $\bar{x}(i|i)$ using the Kalman filtering equations. We find

$$\begin{aligned}\bar{x}(i|i) &= \bar{x}(i|i-1) + K(z(i) - H\bar{x}(i|i-1)) \\ &= (I-KH)\bar{x}(i|i-1) + Kz(i) \\ &= (I-KH)(F\bar{x}(i-1|i-1) + Lu(i-1)) + Kz(i) \\ &= (I-KH)F\bar{x}(i-1|i-1) + (I-KH)Lu(i-1) + Kz(i)\,,\end{aligned}$$

which is the book's equation 8.25. Since the stability of this system does not depend on the measurements $z(i)$ or control $u(i)$ we can take these these equal to zero in the above to obtain

$$\bar{x}(i|i) = (I-KH)F\bar{x}(i-1|i-1)\,,$$

which is the book's equation 8.21.

Now if we define $P(i)$ to be $P(i) \equiv C(i+1|i)$ or the covariance matrix of the predicted state we can derive a recursive equation for the predicted state covariance as

$$\begin{aligned}P(i) &= C(i+1|i) \\ &= F(i)C(i|i)F(i)^T + C_w(i) \\ &= F(i)(C(i|i-1) - K(i)S(i)K^T(i))F^T(i) + C_w(i) \\ &= F(i)P(i-1)F(i)^T + C_w(i) - F(i)K(i)S(i)K^T(i)F^T(i)\,, \qquad (173)\end{aligned}$$

which is the book's equation 8.22 and is known as the discrete *Riccati* equation.

A fundamental theorem presented in this section is the sufficient condition for system stability. The conclusion that our system will be stable depends on the two previously introduced notions of observability and controllability. The theorem presented in the book is as follows. If (F, H) is completely observable and (F, G) is completely controllable then for any initial condition $P(0) = C_x(0)$ then the solution to the Riccati equation converges to a unique, finite, invertible steady state covariance matrix denoted $P(\infty)$. This theorem is followed by several examples that show that systems can still be steady if some of the initial conditions above do not hold true. Some of these examples are worked in more detail below.

Notes on Example 8.7: Stability of a system that is not observable

This example shows that a system can be stable even if its not observable. See the MATLAB script `example_8_7.m` for the MATLAB code that duplicates this example.

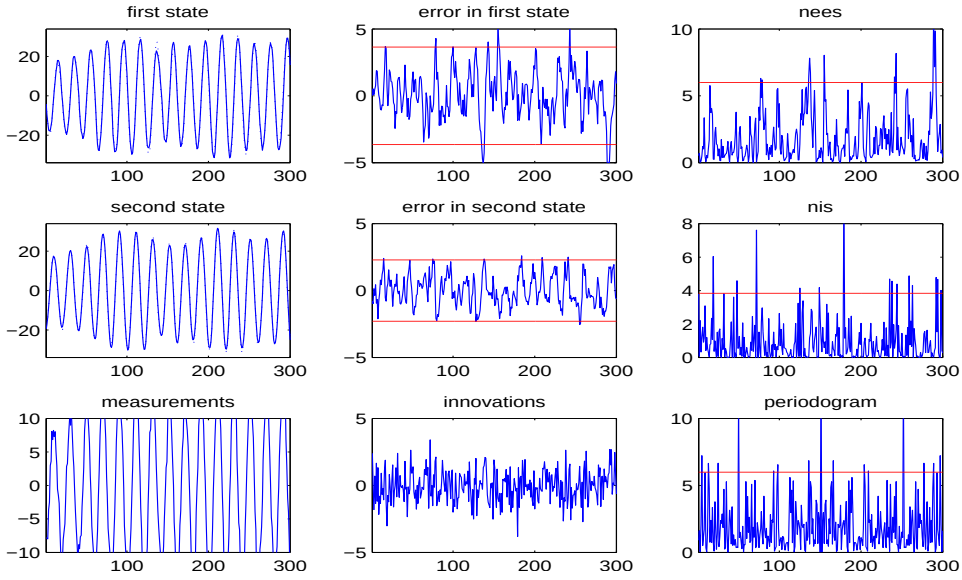

Figure 5: Innovations and normalized errors of a state estimator for a second order system. This figure is a duplication of the books Figure 8.12.

Notes on Example 8.15: Consistency checks applied to a second order system

In this example, we perform optimal Kalman filtering on a second order system. We then compute the three consistency checks for optimal filtering presented in the book. For review, the first is the **NEES** normalized estimation error squared or

$$Nees(i) = e^T(i|i)C^{-1}(i|i)e(i|i) = (x(i|i) - \hat{x}(i|i))^T C^{-1}(i|i)(x(i|i) - \hat{x}(i|i))\,.$$

The second is **NIS** the normalized innovation squared given by

$$Nis(i) = \tilde{z}^T S^{-1}(i)\tilde{z}(i)\,.$$

The third criterion is the whiteness of the residuals $\tilde{z}(i)$ which is determined based on a scaled version of the periodogram. All three of these items have known statistical properties. If the computed value of any one of them diverges from the known distributions this can be an indication that something has changed or that our model is not valid. In this example we apply these consistency checks to a perfectly matched filter. See the MATLAB script `example_8_15.m` for the MATLAB code that duplicates this example. When that script is run it produces the plot shown in Figure 5. Since the distributions from which the above residual errors arise are for filtering in *steady-state*, in the MATLAB script we generate some initial set of data to be discarded and then compute statistics over the second portion of the data where we hope the filtering is in steady-state. This procedure made the present example look quite different with the next one where we filter the measurements with an incorrect F matrix. Since in steady-state we know the distribution of from which $Nees(i)$ and $Nis(i)$ we can compute quantiles of the given distributions and count how many times the empirical measurements fall outside of these quantiles. Taking the 95% quantiles for this example we find

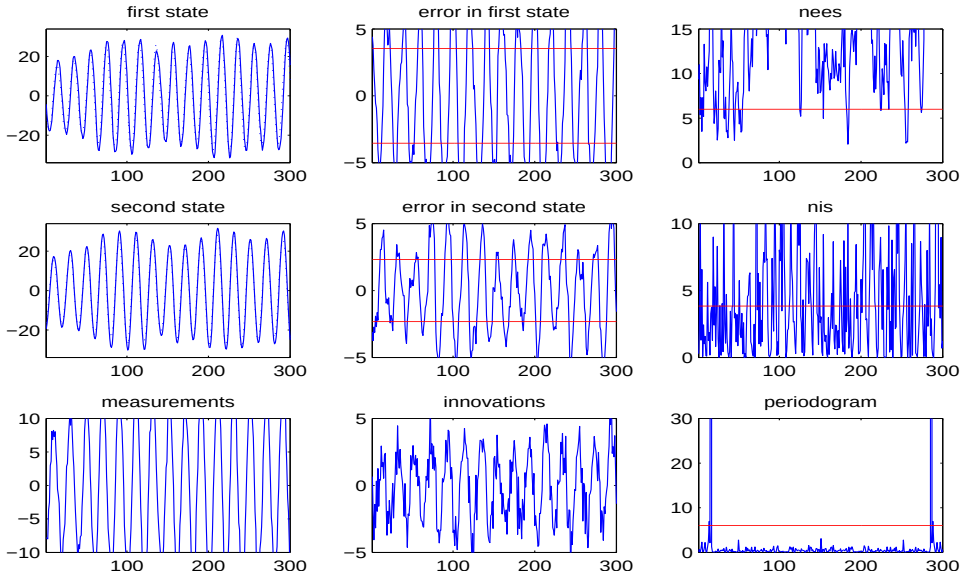

Figure 6: Innovations and normalized errors of a state estimator based on a *mismatched* model. This figure is a duplication of the books Figure 8.13. See the text for more details.

```
percentage of NESS points greater than c_95=  0.036667
percentage of NIS points greater than c_95=  0.046667
percentage of P_1k points greater than c_95=  0.050000
```

Since we expect that 5% of the points to fall outside of the computed quantile values these numbers are consistent with the given assumptions. Thus we can informally conclude that our filtering is "correct".

Notes on Example 8.17: Consistency checks on a mismatched system

This is a continuation of Example 8.15 but in this case we use a system matrix F for filtering the measurements z that is *incorrect*. We expect that the diagnostics procedures introduced above should show this as a mismatched model. In the MATLAB script `example_8_17.m` we perform the requested filtering. When we do that we get the result show in Figure 6. In the resulting figure we see that the three diagnostic tools $Nees$, Nis, and the periodogram all look to be well beyond expected values. We can again look and see how many empirical points from $Nees(i)$, $Nis(i)$, and $\frac{2P_n(k)}{\sigma_n^2}$ are beyond their expected thresholds. In the model mismatched case we find

```
percentage of NESS points greater than c_95=  0.886667
percentage of NIS points greater than c_95=  0.396667
percentage of P_1k points greater than c_95=  0.013333
```

A huge number of $Ness(i)$ and $Nis(i)$ data are too large and in addition the periodogram has a huge peak all of which indicate that the model we are using to filter is incorrect. Note that in this case even though the state estimates visually look similar to the true measurements, the actual errors in the first and second states are larger than specified by the components of the steady state covariance matrix.

Notes on autocorrelated measurement noise

If we assume a model for the measurement noise, $v(i)$ of the form

$$v(i+1) = F_v v(i) + \tilde{v}(i)\,,$$

by state augmentation, we can produce an alternative system that is amenable to a direct Kalman formulation. We form the augmented state given by $\begin{bmatrix} x(i) \\ v(i) \end{bmatrix}$ which has the following system and measurement equations

$$\begin{aligned} \begin{bmatrix} x(i+1) \\ v(i+1) \end{bmatrix} &= \begin{bmatrix} F & 0 \\ 0 & F_v \end{bmatrix} \begin{bmatrix} x(i) \\ v(i) \end{bmatrix} + \begin{bmatrix} I \\ 0 \end{bmatrix} w(i) + \begin{bmatrix} 0 \\ I \end{bmatrix} \tilde{v}(i) \\ z(i) &= \begin{bmatrix} H & I \end{bmatrix} \begin{bmatrix} x(i) \\ v(i) \end{bmatrix}. \end{aligned}$$

Now the process noise is given by the two terms $\begin{bmatrix} I \\ 0 \end{bmatrix} w(i) + \begin{bmatrix} 0 \\ I \end{bmatrix} \tilde{v}(i)$ and will have a covariance matrix given by

$$\begin{bmatrix} I \\ 0 \end{bmatrix} C_w \begin{bmatrix} I & 0 \end{bmatrix} + \begin{bmatrix} 0 \\ I \end{bmatrix} C_{\tilde{v}} \begin{bmatrix} 0 & I \end{bmatrix} = \begin{bmatrix} C_v & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & C_{\tilde{v}} \end{bmatrix} = \begin{bmatrix} C_v & 0 \\ 0 & C_{\tilde{v}} \end{bmatrix}.$$

Since there is no measurement noise, the book then says to consider as measurement the difference $z(i) - F_v z(i-1)$, where we find

$$y(i) = z(i) - F_z z(i-1) = Hx(i) + v(i) - F_v z(i-1)\,.$$

But using $z(i-1) = Hx(i-1) + v(i-1)$ in the above we find

$$\begin{aligned} y(i) &= Hx(i) + v(i) - F_v Hx(i-1) - F_v v(i-1) \\ &= Hx(i) - F_v Hx(i-1) + v(i) - F_v v(i-1) \\ &= Hx(i) - F_v Hx(i-1) + \tilde{v}(i-1)\,, \end{aligned}$$

which has a nontrivial covariance expression of $C_{\tilde{v}}$.

Exercise Solutions

Exercise 1 (an AR process)

The estimation of the parameters of an AR(M) model

$$x(i) = \sum_{n=1}^{M} \alpha_n x(i-n) + w(i)\,, \tag{174}$$

can be done by solving the Yule-Walker equations which are given by

$$\begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ \vdots \\ r_M \end{bmatrix} = \begin{bmatrix} 1 & r_1 & r_2 & r_3 & \cdots & r_{M-1} \\ r_1 & 1 & r_1 & r_2 & \cdots & r_{M-2} \\ r_2 & r_1 & 1 & r_1 & \cdots & r_{M-3} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{M-1} & r_{M-2} & r_{M-3} & r_{M-4} & \cdots & r_1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_M \end{bmatrix}, \tag{175}$$

where $r_k \equiv \frac{1}{\sigma_x^2} E[x(i)x(i-k)]$ is the sample autocorrelation of $x(i)$. Now σ_w^2 is given by multiplying Equation 174 by $x(i)$ and taking expectations. We get

$$\sigma_x^2 = \sum_{n=1}^{M} \alpha_n r_n \sigma_x^2 + E(w(i)x(i))\,.$$

We can evaluate this last term as

$$\begin{aligned} E(w(i)x(i)) &= E\left(w(i) \left(\sum_{n=1}^{M} \alpha_n x(i-n) + w(i) \right) \right) \\ &= \sigma_w^2 + \sum_{n=1}^{M} \alpha_n E(w(i)x(i-n)) = \sigma_w^2\,, \end{aligned}$$

where the summation vanishes since the terms $w(i)$ and $x(i-n)$ are independent when $n \geq 1$ and $w(i)$ is a zero mean process. Thus we get

$$\sigma_x^2 = \sum_{n=1}^{M} \alpha_n \sigma_x^2 r_n + \sigma_w^2\,,$$

as the equation that can be solved for σ_w^2 once we know α_n.

The previous discussion aims at determining the coefficients of the AR(M) model. This does not discuss how to determine the *order* of the AR(M) model that best fits the data. The order of an AR sequence $x(i)$ is obtained by the partial autocorrelation function. To compute this function we fit an AR($\hat{M}$) model for $\hat{M} = 1, 2, \cdots$ and obtaining coefficients $\hat{\alpha}_{n,\hat{M}}$ that are estimates of the AR coefficients in Equation 174 for $1 \leq n \leq \hat{M}$. We then produce a partial autocorrelation function plot by plotting the *last* of these coefficients or $\alpha_{\hat{M},\hat{M}}$ for each model. The selection of the correct AR model is based on the fact that a true

AR(M) model has the property that $\alpha_{\hat{M},\hat{M}} = 0$ when $\hat{M} > M$. This motivated the practical procedure often used where the index at which our partial autocorrelation function falls to zero is taken to be the order of the AR model.

In the `R` file `ex_1.R` we compute the sample partial autocorrelation function on this data set and find that it has a significant component at the $n = 1$ lag indicating that this data looks to be coming from an AR(1) model. We next fit an AR(1) model to this data and to verify the completeness of our model look at the resulting partial autocorrelation and autocorrelation function of the residuals. If the AR(1) model is sufficient to describe this data these two functions should be insignificant. From the sample autocorrelation plot of the residuals they look to be coming from a moving average process since they have a significant component at the $n = 1$ lag. Thus to the original model we include a MA(1) term. The final result is that this data seems to be coming from an ARIMA(1,0,1) model. In addition, we found that another potential model could arise by taking the first difference of the time series data and then fitting an MA(1) model to it. This second approach also seems to yield statistically negligible autocorrelation and partial autocorrelation terms.

Exercise 2-6 (observability and controllability of some linear systems)

To solve these exercises one needs to recall the definitions of observability and controllability of linear systems. These two notions can be determined for a given linear system by looking at the *Gramian* or *matrix* representation or observability or controllability. For observability, a time-dependent system has a Gramian given by

$$\mathcal{G} = H^T(i)H(i) + \sum_{j=1}^{n} \left(H(i+j) \prod_{k=0}^{j-1} F(i+k) \right)^T \left(H(i+j) \prod_{k=0}^{j-1} F(i+k) \right) , \tag{176}$$

The observability Gramian for a linear-time *invariant* system then simplifies to

$$\mathcal{G} = \sum_{j=0}^{n} (HF^j)^T (HF^j) , \tag{177}$$

For our given system to be observable we must have the observability Gramian have a rank M (the dimension of the state space) or equivalently for $\mathcal{G}$ to be positive definite. In this case we can extract all the parameters of $x(i)$. We can check the observability (after observing an infinite number of measurements $z(i)$ by letting $n \to \infty$). We can also construct the observability matrix M which has its definition motivated by stacking $z(i)$, $z(i+1)$, $z(i+2)$ etc. into one column and relating the entire sequence of measurements to the initial state vector $x(i)$. This gives the *observability matrix* or

$$M = \begin{bmatrix} H \\ HF \\ HF^2 \\ \vdots \\ HF^{M-1} \end{bmatrix} . \tag{178}$$

As with the observability Gramian the matrix M must have a rank of M. This definition of observability assumes that there is no observation noise. When there *is* observation noise $v(i)$ with a covariance matrix given by C_v the discussion in the text indicates that the matrix one wants to consider in that case is given by

$$\sum_{j=0}^{i}((F^{-1})^T)^j H^T C_v^{-1} H (F^{-1})^j \,.$$

To be observable in the presence of noise requires that this matrix have rank M. Note that if C_v is a scalar so that our measurement z is a scalar then the noise covariance does not really play a roll in the determination of the observability since it won't affect the rank of the above expression.

The controllability Gramian is not defined in this book and instead the focus presented here is on the *controllability matrix* which for linear time invariant systems is defined by

$$\begin{bmatrix} L & FL & F^2L & \cdots & F^{M-1}L \end{bmatrix}, \tag{179}$$

where the M in the final power of F above is the dimension of the state vector $\mathbf{x}$. If the controllability matrix has rank M then the system is said to be controllable. Note that for observability say one can look at the eigenvalues of the observability Gramian or the singular values of the observability matrix and take the ratio of the smallest value to the largest value. This ratio determines "how" observable a system is. The smaller this number (the smallest it can and still have the system observable be is zero) the less observable the system is. Similar comments hold for controllability.

For the given F and H we find numerically that the observability Gramian and observability matrix are given by

$$\mathcal{G} = \begin{bmatrix} 426.1918 & 213.0752 \\ 213.0752 & 106.5273 \end{bmatrix} \quad M = \begin{bmatrix} 10.0000 & 5.0000 \\ 9.0002 & 4.4999 \end{bmatrix}.$$

One eigenvalues in each matrices is near zero and the other one is considerable larger. The ratio of smallest eigenvalue to largest eigenvalue for each matrix is given by

$$9.5300\,10^{-11} \quad \text{and} \quad -1.1415\,10^{-5}\,.$$

The fact that these are so small indicates while this system is observable it is very weakly so. This means that we should expect to have to process many measurements to obtain reasonable estimate of the state.

Exercise 3 (examples with observability and controllability)

For this system definition in the MATLAB file `ex_3.m` we find that the system is both observable and controllable. From the theorem presented above this indicates that the steady-state Kalman filter *does* exist. We can then compute the steady state value of P

by iterating Equation 173. Once this is done we can determine the steady-state innovation matrix S and Kalman gain using Equations 40 and 41 respectively. We find

$$S = 1.5827 \quad \text{and} \quad K = \begin{bmatrix} 0.0313 \\ 0.3368 \end{bmatrix} .$$

We can compute the steady-state error covariance matrix $C_x(\infty)$ by solving Equation 42 repeated here for convenience

$$C_x(\infty) = FC_x(\infty)F^T + C_w \, ,$$

which when we do this we get

$$C_x(\infty) = \begin{bmatrix} 0.1459 & -0.1039 \\ -0.1039 & 0.7537 \end{bmatrix} .$$

Exercise 4 (more examples of observability and controllability)

This problem is worked in the MATLAB script `ex_4.m`. When it is run we see that F is has eigenvalue greater than one and thus this system will not be stable. We find that the system is not observable but is controllable. Thus we cannot conclude that a steady-state solution to the Riccati equation will exist. In any case, it seems to exist and can be computed by iteration. Thus the steady-state Kalman gain and innovation matrix also exist. Attempting to compute the discrete Lyapunov equation by iteration fails.

Exercise 5 (more examples of observability and controllability)

This problem is worked in the MATLAB script `ex_5.m`. When it is run we see that F is one eigenvalue less than one and one eigenvalue equal to one. We find this system is observable and controllable and therefore we know that a steady-state solution to the Lyapunov equation exists.

Exercise 8 (drift in the measurements)

To solve this problem with a discrete Kalman filter we will let our state be $\mathbf{x}(i) = \begin{bmatrix} x(i) \\ v(i) \end{bmatrix}$, then the dynamics for a vector state like this can be derived as

$$\begin{aligned} \mathbf{x}(i+1) &= \begin{bmatrix} x(i+1) \\ v(i+1) \end{bmatrix} = \begin{bmatrix} \alpha x(i) + w(i) \\ \beta v(i) + \tilde{v}(i) \end{bmatrix} \\ &= \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix} \begin{bmatrix} x(i) \\ v(i) \end{bmatrix} + \begin{bmatrix} w(i) \\ \tilde{v}(i) \end{bmatrix} . \end{aligned}$$

Now our vector noise is given $\begin{bmatrix} w(i) \\ \tilde{v}(i) \end{bmatrix} \sim N(0, C_w)$ where $C_w(i) = \begin{bmatrix} \sigma_w^2 & 0 \\ 0 & \sigma_{\tilde{v}}^2 \end{bmatrix} = \begin{bmatrix} 0.0975 & 0 \\ 0 & 0.002 \end{bmatrix}$ and the measurement equation is given by

$$z(i) = x(i) + v(i) = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} x(i) \\ v(i) \end{bmatrix},$$

which has no noise i.e. C_v the measurement noise covariance is zero. So for this system we have found

$$F = \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix}, H = \begin{bmatrix} 1 & 1 \end{bmatrix}, C_w(i) = \begin{bmatrix} 0.0975 & 0 \\ 0 & 0.002 \end{bmatrix}, C_v(i) = 0,$$

for the state space model.

The observability of this system is given by looking at the Gramian

$$\mathcal{G} = \sum_{j=0}^{n} (HF^j)^T (HF^j).$$

Because F is diagonal the values of α and β are the *same* as the eigenvalues of F. When $\alpha = 0.95$ and $\beta = 0.999$ since they are both less than one we see that our system is therefore stable. We can compute the observability Gramian by taking $n \to \infty$ in the above summation. Since the magnitude of the eigenvalues of F are so close to 1 we will need a large number of terms in the summation to guarantee convergence. Taking 20000 terms we find

$$\mathcal{G} = \begin{bmatrix} 9.2564 & 18.6271 \\ 18.6271 & 499.2501 \end{bmatrix}.$$

which has rank 2 showing that this system *is* observable. To determine the controllability of this system consider the deterministic system with the addition of a term $Lu(i)$ giving

$$\mathbf{x}(i+1) = F\mathbf{x}(i) + L\mathbf{u}(i).$$

the theory of controllability is general enough to consider an arbitrary matrix L but since it is not directly specified in the problem statement we may take it to be the identity. Then the system controllability is determined by the matrix

$$\begin{bmatrix} L & FL & F^2L & \cdots & F^{M-1}L \end{bmatrix} = \begin{bmatrix} I & F & F^2 & \cdots & F^{M-1} \end{bmatrix}.$$

Where M is the dimension of the state space $\mathbf{x}$ which in this case is 2. When we construct this matrix we find

$$\begin{bmatrix} 1 & 0 & 0.95 & 0 \\ 0 & 1 & 0 & 0.99 \end{bmatrix},$$

which has rank two showing that this system is controllable.

The discrete Lyapunov equation is given by

$$C_x(\infty) = FC_x(\infty)F^T + C_w,$$

which we will solve numerically by iteration. We find

$$C_x(\infty) = \begin{bmatrix} 1 & 0 \\ 0 & 1.0005 \end{bmatrix}.$$

The numerical computations for this problem are performed in the MATLAB script `ex_8.m`.

References

[1] P. A. Devijver and J. Kittler. *Pattern recognition: A statistical approach*. Prentice Hall, 1982.

[2] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience Publication, 2000.

[3] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis, Second Edition*. Chapman & Hall/CRC, July 2003.

[4] M. S. Grewal and A. P. Andrews. *Kalman Filtering : Theory and Practice Using MATLAB*. Wiley-Interscience, January 2001.

[5] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.

[6] W. G. Kelley and A. C. Peterson. *Difference Equations. An Introduction with Applications*. Academic Press, New York, 1991.

[7] J. Weatherwax. *A Solution Manual and Study Guide for: Pattern recognition: A statistical approach by Pierre A. Devijver and Josef Kittler*. 2009.

[8] R. R. Wilox. *Basic Statistics: Understanding Conventional Methods and Modern Insights*. Prentice Hall, 2002.