SOLUTIONS MANUAL FOR

A First Course In Machine Learning (2Nd Edition) Exercise Solutions

by

Simon Rogers and Mark Girolami

CRC Press
Taylor & Francis Group

SOLUTIONS MANUAL FOR

A First Course In Machine Learning (2Nd Edition) Exercise Solutions

by

Simon Rogers and Mark Girolami

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

Printed on acid-free paper
Version Date: 20160404

International Standard Book Number-13: 978-1-4987-3859-0 (Ancillary)

Visit the Taylor & Francis Web site at
http://www.taylorandfrancis.com

and the CRC Press Web site at
http://www.crcpress.com

Chapter 1

EX 1.1. A high positive value of w_0 and a small negative value for w_1. These reflect the high intercept on the t axis (corresponding to the theoretical time winning time at $x = 0$ and the small decrese in winning time over the years.

EX 1.2. The following would do the job:

```
% Attributes are stored in Nx1 vector x
% Targets are stored in Nx1 vector t
xb = mean(x);
tb = mean(t);
x2b = mean(x.*x);
xtb = mean(x.*t);
w1 = (xtb - xt*xb)/(x2b-xb^2);
w0 = tb-w1*xb;
% Plot the data
plot(x,t,'b.','markersize',25);
% Plot the model
hold on;
plot(x,w0+w1*x,'r','linewidth',2);
```

EX 1.3. We need to find $\mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w}$. We'll start with $\mathbf{X}^\mathsf{T}\mathbf{X}$. Multiplying $\mathbf{X}^\mathsf{T}$ by $\mathbf{X}$ gives:

$$\mathbf{X}^\mathsf{T}\mathbf{X} = \begin{bmatrix} \sum_{n=1}^{N} x_{n1}^2 & \sum_{n=1}^{N} x_{n1}x_{n2} \\ \sum_{n=1}^{N} x_{n2}x_{n1} & \sum_{n=1}^{N} x_{n2}^2 \end{bmatrix}$$

Multiplying this by $\mathbf{w}$ gives:

$$\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w} = \begin{bmatrix} w_0 \sum_{n=1}^{N} x_{n1}^2 + w_1 \sum_{n=1}^{N} x_{n1}x_{n2} \\ w_0 \sum_{n=1}^{N} x_{n2}x_{n1} + w_1 \sum_{n=1}^{N} x_{n2}^2 \end{bmatrix}$$

Finally, pre-multiplying this by $\mathbf{w}^\mathsf{T}$ gives:

$$\begin{aligned}\mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w} &= w_0\left(w_0\sum_{n=1}^{N}x_{n1}^2 + w_1\sum_{n=1}^{N}x_{n1}x_{n2}\right) + \\ &\quad w1\left(w_0\sum_{n=1}^{N}x_{n2}x_{n1} + w_1\sum_{n=1}^{N}x_{n2}^2\right) \\ &= w_0^2\sum_{n=1}^{N}x_{n1}^2 + 2w_0w_1\sum_{n=1}^{N}x_{n1}x_{n2} + w_1^2\sum_{n=1}^{N}x_{n2}^2\end{aligned}$$

as required.

EX 1.4. Let's first work out $\mathbf{X}\mathbf{w}$:

$$\mathbf{X}\mathbf{w} = \begin{bmatrix} w_0x_{11} + w_1x_{12} \\ w_0x_{21} + w_1x_{22} \\ \vdots \\ w_0x_{N1} + w_1x_{N2} \end{bmatrix}$$

Therefore

$$(\mathbf{X}\mathbf{w})^\mathsf{T} = [w_0x_{11} + w_1x_{12},\ w_0x_{21} + w_1x_{22},\ \ldots\ ,\ w_0x_{N1} + w_1x_{N2}]$$

Finally, work out $\mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}$:

$$\mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T} = [w_0x_{11} + w_1x_{12},\ w_0x_{21} + w_1x_{22},\ \ldots\ ,\ w_0x_{N1} + w_1x_{N2}]$$

as required.

EX 1.5. Starting with $\sum_n \mathbf{x}_n t_n$. The result of this is a column vector of the same size as $\mathbf{x}$ (2×1). Now, using the definition of $\mathbf{X}$,

$$\mathbf{X}^\mathsf{T} = \begin{bmatrix} x_{11}, x_{21}, \ldots, x_{N1} \\ x_{12}, x_{22}, \ldots, x_{N2} \end{bmatrix}$$

(which is a $2 \times N$ vector). Multiplying this by $\mathbf{t}$ gives a 2×1 vector that looks like this:

$$\mathbf{X}^\mathsf{T}\mathbf{t} = \begin{bmatrix} \sum_{n=1}^{N} x_{n1}t_n \\ \sum_{n=1}^{N} x_{n2}t_n \end{bmatrix}$$

which is $\sum_n \mathbf{x}_n t_n$ as required. The second example, $\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w}$. We already know what $\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w}$ is (Exercise 1.3)

$$\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w} = \begin{bmatrix} w_0\sum_{n=1}^{N} x_{n1}^2 + w_1\sum_{n=1}^{N} x_{n1}x_{n2} \\ w_0\sum_{n=1}^{N} x_{n2}x_{n1} + w_1\sum_{n=1}^{N} x_{n2}^2 \end{bmatrix}$$

Now, $\mathbf{x}_n\mathbf{x}_n^\mathsf{T}$ is the following matrix:

$$\mathbf{x}_n\mathbf{x}_n^\mathsf{T} = \begin{bmatrix} x_{n1}^2 & x_{n1}x_{n2} \\ x_{n2}x_{n1} & x_{n2}^2 \end{bmatrix}$$

Multiplying this by $\mathbf{w}$ gives:

$$\mathbf{x}_n\mathbf{x}_n^\mathsf{T}\mathbf{w} = \begin{bmatrix} w_0x_{n1}^2 + w_1x_{n1}x_{n2} \\ w_0x_{n2}x_{n1} + w_1x_{n2}^2 \end{bmatrix}$$

Summing over the N terms leads us to the matrix we derived previously.

EX 1.6. Code below:

```
%% Women's 100m data
% Load all Olympic data
load olympics;
% Copy the necessary variables
x = female100(:,1); % Olympic year
t = female100(:,2); % Winning time
% Augment x
X = [repmat(1,size(x)) x];
% Get solution
w = inv(X'*X)*X'*t;
```

The fitted model is:

$$t = 40.9242 - 0.0151x$$

EX 1.7. Plugging 2012 and 2016 into the above expression yields winning times of 10.5997 and 10.5394 respectively.

EX 1.8. The men's model is:

$$t = 36.4165 - 0.0133x$$

The women's model is:

$$t = 40.9242 - 0.0151x$$

The women's time is decreasing faster than the men's. Therefore, the women will be faster at the first Olympics after the x that gives identical winning times:

$$\begin{aligned} 40.9242 - 0.0151x &= 36.4165 - 0.0133x \\ x &= 2589 \end{aligned}$$

The next Olympic year after 2589 is (assuming they continue to be held every four years) is the year 2592. The winning times are the unrealistically fast 1.8580 seconds and 1.8628 seconds for women and men respectively.

EX 1.9. Code below (`synthdata_cv.m`):

```
clear all;close all;
load synthdata

% Augment x
X = repmat(1,size(x));
for k = 1:4
    X = [X x.^k];
end

```

```
% Fit the model
w = inv(X'*X)*X'*t;

% Randomise the data order
N = size(X,1);
order = randperm(N);
sizes = repmat(floor(N/10),1,10);
sizes(end) = sizes(end) + N-sum(sizes);
sizes = [0 cumsum(sizes)];

X = repmat(1,size(x));

loss = zeros(4,10);
for poly_order = 1:4
    % Augment x
    X = [X x.^poly_order];
    for k = 1:10 % 10-fold CV
        % Extract the train and test data
        traindata = X(order,:);
        traint = t(order);
        testdata = X(order(sizes(k)+1:sizes(k+1)),:);
        testt = t(order(sizes(k)+1:sizes(k+1)));
        traindata(sizes(k)+1:sizes(k+1),:) = [];
        traint(sizes(k)+1:sizes(k+1)) = [];

        % Fit the model
        w = inv(traindata'*traindata)*traindata'*traint;

        % Compute loss on test data
        predictions = testdata*w;
        loss(poly_order,k) = sum((predictions - testt).^2);
    end
end

% Plot the loss
plot([1:4],mean(loss,2));
```

EX 1.10. The total loss is

$$\mathcal{L} = \sum_{n=1}^{N} (t_n - \mathbf{w}^\mathsf{T}\mathbf{x}_n)^2.$$

Writing this in matrix form, differentiating and solving gives us:

$$\begin{aligned} \mathcal{L} &= (\mathbf{t} - \mathbf{X}\mathbf{w})^\mathsf{T}(\mathbf{t} - \mathbf{X}\mathbf{w}) \\ &= \mathbf{t}^\mathsf{T}\mathbf{t} - 2\mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{t} + \mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w} \\ \frac{\partial \mathcal{L}}{\partial \mathbf{w}} &= -2\mathbf{X}^\mathsf{T}\mathbf{t} + 2\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w} = \mathbf{0} \\ \mathbf{w} &= (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{t}. \end{aligned}$$

This is identical to the value obtained for the average loss. It is not surprising as all we are doing is multiplying the loss by a constant and this will not change the value of $\mathbf{w}$ at the minimum.

EX 1.11. The loss is given by

$$\mathcal{L} = \frac{1}{N}\sum_{n=1}^{N} \alpha_n (t_n - \mathbf{w}^\mathsf{T}\mathbf{x}_n)^2$$

If we define the matrix:

$$\mathbf{A} = \begin{bmatrix} \alpha_1 & 0 & \dots & 0 \\ 0 & \alpha_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \alpha_N \end{bmatrix}$$

we can write the loss in vector/matrix form as:

$$\mathcal{L} = \frac{1}{N}\sum_{n=1}^{N} (\mathbf{t} - \mathbf{X}\mathbf{w})^\mathsf{T}\mathbf{A}(\mathbf{t} - \mathbf{X}\mathbf{w})$$

Multiplying out, differentiating, equating to zero and solving:

$$\begin{aligned} \mathcal{L} &= \frac{1}{N}\left(\mathbf{t}^\mathsf{T}\mathbf{A}\mathbf{t} - 2\mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{A}\mathbf{t} + \mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{A}\mathbf{X}\mathbf{w}\right) \\ \frac{\partial \mathcal{L}}{\partial \mathbf{w}} &= -\frac{2}{N}\mathbf{X}^\mathsf{T}\mathbf{A}\mathbf{t} + \frac{2}{N}\mathbf{X}^\mathsf{T}\mathbf{A}\mathbf{X}\mathbf{w} = \mathbf{0} \\ \mathbf{w} &= (\mathbf{X}^\mathsf{T}\mathbf{A}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{A}\mathbf{t}. \end{aligned}$$

Try this out in Matlab (set some α_n very low and some very high) to see the effect on the solution.

EX 1.12. Code below (`regls100m.m`):

```
clear all;close all;
load olympics;
% Extract men's 100m data
x = male100(:,1);
t = male100(:,2);

% Choose number of folds
K = 5;

% Randomise the data order
N = size(x,1);
order = randperm(N);
sizes = repmat(floor(N/K),1,K);
sizes(end) = sizes(end) + N-sum(sizes);
sizes = [0 cumsum(sizes)];

% Rescale x
x = x - x(1);
x = x./4;

X = [repmat(1,size(x)) x];
% Comment out the following line for linear
X = [X x.^2 x.^3 x.^4];
```

```
24
25 % Scan a wide range of values of the regularisation perameter
26 regvals = 10.^[-12:1:12];
27
28 for r = 1:length(regvals)
29    for k = 1:K
30         % Extract the train and test data
31         traindata = X(order,:);
32         traint = t(order);
33         testdata = X(order(sizes(k)+1:sizes(k+1)),:);
34         testt = t(order(sizes(k)+1:sizes(k+1)));
35         traindata(sizes(k)+1:sizes(k+1),:) = [];
36         traint(sizes(k)+1:sizes(k+1)) = [];
37
38         % Fit the model
39         w = inv(traindata'*traindata + regvals(r)*eye(size(X,2)))*...
40             traindata'*traint;
41
42         % Compute loss on test data
43         predictions = testdata*w;
44         loss(r,k) = sum((predictions - testt).^2);
45    end
46 end
```

Chapter 2

EX 2.1. The errors are real valued and hence a continuous random variable would be more appropriate.

EX 2.2. If all outcomes are equally likely, they have the same probability of occurring. Defining Y to be the random variable taking the value shown on a die, we can state the following:

$$P(Y = y) = r,$$

where r is a constant. From the definition of probabilities, we know that:

$$\sum_{y=1}^{6} P(Y = y) = 1.$$

Substituting r into this gives us the following:

$$\sum_{y=1}^{6} r = 1,\ 6r = 1,\ r = 1/6.$$

EX 2.3. (a) Y is a discrete random variable that can take any value from 0 to inf. The probability that $Y \leq 4$ is equal to the sum of all of the probabilities that satisfy $Y \leq 4, Y = 0, Y = 1, Y = 2, Y = 3, Y = 4$:

$$P(Y \leq 4) = \sum_{y=0}^{4} P(Y = y).$$

When $\lambda = 5$, we can compute these probabilities as:

$P(Y \leq 4) = 0.0067379 + 0.0336897 + 0.0842243 + 0.1403739 + 0.1754674 = 044049.$

(b) Because Y has to satisfy either $P(Y| \leq 4)$ or $P(Y > 4)$, we know that $P(Y > 4) = 1 - P(Y \leq 4)$:

$$P(Y > 4) = 0.5591.$$

EX 2.4. We require $\mathbf{E}_{p(y)}\{sin(y)\}$ where $p(y) = \mathcal{U}(a, b)$. The uniform density is given by:

$$p(y) = \begin{cases} \frac{1}{b-a} & a \leq y \leq b \\ 0 & \text{otherwise} \end{cases}$$

The required expectation is given by:

$$\begin{aligned}\mathbf{E}_{p(y)}\left\{sin(y)\right\} &= \int sin(y)p(y)\ dy \\ &= \int_{y=a}^{b} sin(y)\frac{1}{b-a}\ dy \\ &= \frac{1}{b-a}\left[-\cos(y)\right]_a^b \\ &= \frac{\cos(a)-\cos(b)}{b-a}.\end{aligned}$$

When $a = 0,\ b = 1$, this is equal to

$$\mathbf{E}_{p(y)}\left\{sin(y)\right\} = \frac{\cos(0)-\cos(1)}{1} = 0.45970.$$

Code to compute a sample-based approximation below (`sampleexpect.m`):

```
clear all;
close all;
% Compute a sample based approximation to the required expectation
u = rand(10000,1); % Take 10000 samples
su = sin(u);
% Plot how the approximation changes as more samples are used
ns = 10:100:10000;
stages = zeros(size(ns));
for i = 1:length(ns)
    stages(i) = mean(su(1:ns(i)));
end
plot(ns,stages)
% Plot the true value
hold on
plot([0 ns(end)],[0.4597 0.4597],'k--')
```

EX 2.5. The multivariate Gaussian pdf is given by:

$$p(\mathbf{w}) = \frac{1}{(2\pi)^{D/2}|\mathbf{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^{\mathsf{T}}\mathbf{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right\}.$$

Setting $\mathbf{\Sigma} = \sigma^2\mathbf{I}$ gives:

$$p(\mathbf{w}) = \frac{1}{(2\pi)^{D/2}|\sigma^2\mathbf{I}|^{1/2}}\exp\left\{-\frac{1}{2\sigma^2}(\mathbf{w}-\boldsymbol{\mu})^{\mathsf{T}}\mathbf{I}^{-1}(\mathbf{w}-\boldsymbol{\mu})\right\}.$$

Because it only has elements on the diagonal, the determinant of $\sigma^2\mathbf{I}$ is given by the product of these diagonal elements. As they are all the same, $|\sigma^2\mathbf{I}|^{1/2} = \left(\prod_{d=1}^{D}\sigma^2\right)^{1/2} = (\sigma^2)^{D/2}$. $\mathbf{I}^{-1} = \mathbf{I}$ and multiplying a vector/matrix by $\mathbf{I}$ leaves the matrix/vector unchanged. Therefore, the argument within the expectation can be written as $-\frac{1}{2\sigma^2}(\mathbf{w}-\boldsymbol{\mu})^{\mathsf{T}}(\mathbf{w}-\boldsymbol{\mu})$ and recalling that $\mathbf{b}^{\mathsf{T}}\mathbf{b} = \sum_i b_i^2$, we can rewrite the pdf as:

$$p(\mathbf{w}) = \frac{1}{(2\pi)^{D/2}(\sigma^2)^{D/2}}\exp\left\{-\frac{1}{2\sigma^2}\sum_{d=1}^{D}(w_d-\mu_d)^2\right\}.$$

Where w_d and μ_d are the dth elements of $\mathbf{w}$ and $\boldsymbol{\mu}$ respectively. The exponential of a sum is the same as a product of exponentials. Hence,

$$\begin{aligned} p(\mathbf{w}) &= \frac{1}{(2\pi)^{D/2}(\sigma^2)^{D/2}} \prod_{d=1}^{D} \exp\left\{-\frac{1}{2\sigma^2}(w_d - \mu_d)^2\right\} \\ &= \prod_{d=1}^{D} \frac{1}{(2\pi)^{1/2}\sigma} \exp\left\{-\frac{1}{2\sigma^2}(w_d - \mu_d)^2\right\} \\ &= \prod_{d=1}^{D} p(w_d|\mu_d, \sigma^2), \end{aligned}$$

where $p(w_d|\mu_d, \sigma^2) = \mathcal{N}(\mu_d, \sigma^2)$. Hence, the diagonal covariance is equivalent to assuming that the elements of $\mathbf{w}$ are distributed as independent, univariate Gaussians with mean μ_d and variance σ^2.

EX 2.6. Using the same methods as in the previous exercise, we see that the determinant of the covariance matrix is given by $\prod_{d=1}^{D} \sigma_d^2$ and we have the following:

$$p(\mathbf{w}) = \frac{1}{(2\pi)^{D/2}\left(\prod_{d=1}^{D} \sigma_d^2\right)^{1/2}} \exp\left\{-\frac{1}{2}\sum_{d=1}^{D} \frac{(w_d - \mu_d)^2}{\sigma_d^2}\right\}$$

Changing the sum to a product leaves us with

$$\begin{aligned} p(\mathbf{w}) &= \frac{1}{(2\pi)^{D/2}\left(\prod_{d=1}^{D} \sigma_d^2\right)^{1/2}} \prod_{d=1}^{D} \exp\left\{-\frac{1}{2\sigma_d^2}(w_d - \mu_d)^2\right\} \\ &= \prod_{d=1}^{D} \frac{1}{(2\pi)^{1/2}\sigma_d} \exp\left\{-\frac{1}{2\sigma_d^2}(w_d - \mu_d)^2\right\}. \end{aligned}$$

This is the product of D independent univariate Gaussian densities.

EX 2.7. The Hessian for a general model of our form is given by:

$$-\frac{1}{\sigma^2}\mathbf{X}^\mathsf{T}\mathbf{X}$$

For the linear model, $\mathbf{X}$ is defined as:

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix}$$

Therefore $-\frac{1}{\sigma^2}\mathbf{X}^\mathsf{T}\mathbf{X}$ is:

$$\mathbf{X}^\mathsf{T}\mathbf{X} = \begin{bmatrix} N & \sum_{n=1}^{N} x_n \\ \sum_{n=1}^{N} x_n & \sum_{n=1}^{N} x_n^2 \end{bmatrix}$$

The diagonal elements are $-N/\sigma^2$ and $-(1/sigma^2)\sum_{n=1}^{N} x_n^2$ which are equivalent (they differ only by multiplication with a negative constant) the expressions obtained in Chapter 1.

EX 2.8. We have N values, $x_1, \ldots, x_N$. Assuming that these values came from a Gaussian, we want to find the maximum likelihood estimate of the G and want to find the maximum likelihood estimates of the mean and variance of the Gaussian. The Gaussian pdf is:

$$\frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{1}{2\sigma^2}(x_n - \mu)^2\right\}$$

Assuming the IID assumption, the likelihood of all N points is given by a product over the N objects:

$$\prod_{n=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{1}{2\sigma^2}(x_n - \mu)^2\right\}.$$

We'll work with the log of the likelihood:

$$\log L = \sum_{n=1}^{N} \left(-\frac{1}{2}\log(2\pi) - \frac{1}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}(x_n - \mu)^2\right)$$

To find the maximum likelihood estimate for μ, we differentiate with respect to μ, equate to zero and solve:

$$\begin{aligned}
\frac{\partial \log L}{\partial \mu} &= \sum_{n=1}^{N} \frac{1}{\sigma^2}(x - \mu) \\
0 &= \frac{1}{\sigma^2}\sum_{n=1}^{N}(x_n - \mu) \\
0 &= \sum_{n=1}^{N} x_n - \sum_{n=1}^{N} \mu \\
&= \sum_{n=1}^{N} x_n - N\mu \\
\mu &= \frac{1}{N}\sum_{n=1}^{N} x_n
\end{aligned}$$

Similarly, for σ^2,

$$\begin{aligned}
\frac{\partial \log L}{\partial \sigma^2} &= \sum_{n=1}^{N} \left(-\frac{1}{2\sigma^2} + \frac{1}{2(\sigma^2)^2}(x_n - \mu)^2\right) = 0 \\
N\sigma^2 &= \sum_{n=1}^{N}(x_n - \mu)^2 \qquad (2.1) \\
\sigma^2 &= \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2 \qquad (2.2)
\end{aligned}$$

EX 2.9. The Bernoulli distribution is defined as:

$$P(X_n = x|r) = r^x(1-r)^{1-x}$$

where x is either 0 or 1. Using the IID assumption, we have:

$$L = \prod_{n=1}^{N} r^{x_n}(1-r)^{1-x_n}$$

and the log likelihood is:

$$\log L = \sum_{n=1}^{N} x_n \log r + (1-x_n)\log(1-r)$$

Differentiating with respect to r gives us:

$$\begin{aligned}
\frac{\partial \log L}{\partial r} &= \sum_{n=1}^{N}\left(\frac{x_n}{r} - \frac{1-x_n}{1-r}\right) = 0 \\
\sum_{n=1}^{N}\frac{x_n}{r} &= \sum_{n=1}^{N}\frac{1-x_n}{1-r} \\
\sum_{n=1}^{N} x_n - r\sum_{n=1}^{N} x_n &= rN - r\sum_{n=1}^{N} x_n \\
r &= \frac{1}{N}\sum_{n=1}^{N} x_n.
\end{aligned}$$

EX 2.10. The Fisher information is defined as the expectation of the negative second derivative. From the above expression, we can see that the second derivative of the Gaussian likelihood (assuming N observations, $x_1, \ldots, x_N$ is:

$$\frac{\partial^2 \log L}{\partial \mu^2} = -\frac{N}{\sigma^2}.$$

Hence the Fisher information is equal to N/σ^2.

EX 2.11. Starting from the second expression, we have

$$\widehat{\sigma^2} = \frac{1}{N}\left[\sum_{n=1}^{N} t_n^2 - 2\sum_{n=1}^{N} t_n \mathbf{x}_n^{\mathsf{T}}\widehat{\mathbf{w}} + \sum_{n=1}^{N}(\mathbf{x}_n\widehat{\mathbf{w}})^2\right].$$

Concentrating on the final term,

$$\begin{aligned}
\sum_{n=1}^{N}(\mathbf{x}^\mathsf{T}\widehat{\mathbf{w}})^2 &= \sum_{n=1}^{N}\mathbf{x}_n^\mathsf{T}\widehat{\mathbf{w}}\widehat{\mathbf{w}}^\mathsf{T}\mathbf{x}_n \\
&= \mathsf{Tr}(\mathbf{X}\widehat{\mathbf{w}}\widehat{\mathbf{w}}^\mathsf{T}\mathbf{X}^\mathsf{T}) \\
&= \mathsf{Tr}(\mathbf{X}(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{t}\mathbf{t}^\mathsf{T}\mathbf{X}(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}) \\
&= \mathsf{Tr}(\mathbf{X}^\mathsf{T}\mathbf{X}(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{t}\mathbf{t}^\mathsf{T}\mathbf{X}(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}) \\
&= \mathsf{Tr}(\mathbf{X}^\mathsf{T}\mathbf{t}\mathbf{t}^\mathsf{T}\mathbf{X}(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}) \\
&= \mathsf{Tr}(\mathbf{t}\mathbf{t}^\mathsf{T}\mathbf{X}(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}) \\
&= \mathsf{Tr}(\mathbf{t}^\mathsf{T}\mathbf{X}(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{t}) \\
&= \mathbf{t}^\mathsf{T}\mathbf{X}(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{t}) \\
&= \mathbf{t}^\mathsf{T}\mathbf{X}\widehat{\mathbf{w}} \\
&= \sum_{n=1}^{N} t_n\mathbf{x}_n^\mathsf{T}\widehat{\mathbf{w}}.
\end{aligned}$$

Therefore,

$$\widehat{\sigma^2} = \frac{1}{N}\left[\sum_{n=1}^{N} t_n^2 - \sum_{n=1}^{N} t_n\mathbf{x}_n^\mathsf{T}\widehat{\mathbf{w}}\right].$$

Now, $\sum_{n=1}^{N} t_n^2 = \mathbf{t}^\mathsf{T}\mathbf{t}$ and we already know that $\sum_{n=1}^{N} t_n\mathbf{x}_n\widehat{\mathbf{w}} = \mathbf{t}^\mathsf{T}\mathbf{X}\widehat{\mathbf{w}}$. So,

$$\widehat{\sigma^2} = \frac{1}{N}\left[\mathbf{t}^\mathsf{T}\mathbf{t} - \mathbf{t}^\mathsf{T}\mathbf{X}\widehat{\mathbf{w}}\right],$$

as required.

EX 2.12. Code below (`predvar.m`):

```
clear all;close all;
% Relevant code extraced from predictive_variance_example.m
x = rand(50,1)*10-5;
x = sort(x);
% Compute true function values
f = 5*x.^3 - x.^2 + x;
% Generate some test locations
testx = [min(x):0.2:max(x)]';
% Add some noise
t = f+randn(50,1)*sqrt(1000);
% Remove all training data between -1.5 and 1.5
pos = find(x>-1.5 & x<1.5);
x(pos) = [];
f(pos) = [];
t(pos) = [];

% Choose model order
K = 5;

X = repmat(1,size(x));
testX = repmat(1,size(testx));
```

```
for k = 1:K
    X = [X x.^k];
    testX = [testX testx.^k];
end


w_hat = inv(X'*X)*X'*t;
ss_hat = mean((t - X*w_hat).^2);
pred_va = ss_hat*diag(testX*inv(X'*X)*testX');
% Make a plot
figure(1);hold off
plot(x,t,'b.');
hold on
errorbar(testx,testX*w_hat,pred_va,'r');
```

EX 2.13. The Bernoulli distribution for a binary random variable x is:

$$p(x|\theta) = \theta^x(1-\theta)^{1-x}$$

The Fisher information is defined as the negative expected value of the second derivative of the log density evaluated at some paramater value:

$$\mathcal{F} = -\mathbf{E}_{p(x|\theta)}\left\{\left.\frac{\partial^2 \log p(x|\theta)}{\partial\theta^2}\right|_\theta\right\}$$

Differentiating $\log p(x|\theta)$ twice gives:

$$\begin{aligned}\frac{\partial \log p(x|\theta)}{\partial\theta} &= \frac{x}{\theta} - \frac{1-x}{1-\theta}\\ \frac{\partial^2 \log p(x|\theta)}{\partial\theta^2} &= -\frac{x}{\theta^2} - \frac{1-x}{(1-\theta)^2}.\end{aligned}$$

The Fisher information is therefore:

$$\mathcal{F} = \frac{1}{\theta^2}\mathbf{E}_{p(x|\theta)}\{x\} + \frac{1}{(1-\theta)^2}\mathbf{E}_{p(x|\theta)}\{1-x\}.$$

Substituing in the expectations (θ and $1-\theta$ respectively gives:

$$\mathcal{F} = \frac{\theta}{\theta^2} + \frac{1-\theta}{(1-\theta)^2} = \frac{1}{\theta(1-\theta)}$$

EX 2.14. The multivariate Gaussian pdf is given by:

$$p(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}.$$

Logging and removing terms not including $\boldsymbol{\mu}$:

$$\log p(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) \propto \boldsymbol{\mu}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}.$$

First and second derivatives are:

$$\begin{aligned}\frac{\partial \log p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})}{\partial \boldsymbol{\mu}} &= \boldsymbol{\Sigma}^{-1}\mathbf{x} - \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \\ \frac{\partial^2 \log p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})}{\partial \boldsymbol{\mu} \partial \boldsymbol{\mu}^{\mathsf{T}}} &= -\boldsymbol{\Sigma}^{-1}.\end{aligned}$$

Therefore, the Fisher information is:

$$\mathcal{F} = \boldsymbol{\Sigma}^{-1}.$$

Chapter 3

EX 3.1. Our Beta prior (with $\alpha = \beta = 1$) is defined as:

$$p(r) = 1 \quad (0 \leq r \leq 1)$$

The binomial likelihood is given by:

$$P(Y = y|r, N) = \left(\begin{array}{c} N \\ y \end{array} \right) r^y (1-r)^{N-y}$$

We know that the posterior density for r is proportional to the likelihood multiplied by the prior, and we also know, because our prior is a particular Beta density, and the Beta prior is conjugate to the binomial likelihood that the posterior must also be a Beta density:

$$\begin{aligned} p(r|Y, N) &\propto p(Y = y|r, N)p(r) \\ &\propto r^y (1-r)^{N-y} \times 1 \\ &= r^{\alpha'-1}(1-r)^{\beta'-1} \end{aligned}$$

suggesting that the posterior is a Beta density with parameters $\alpha' = y + 1$ and $\beta' = N - y + 1$.

EX 3.2. Using the same steps as the previous exercise:

$$\begin{aligned} p(r|Y, N) &\propto p(Y = y|r, N)p(r) \\ &\propto r^y (1-r)^{N-y} \times 2r \\ &\propto r^{y+1}(1-r)^{N-y} \\ &= r^{\alpha'-1}(1-r)^{\beta'-1} \end{aligned}$$

which suggests a Beta density with parameters $\alpha' = y + 2$ and $\beta' = N - y + 1$.
To find the prior parameters corresponding to the prior $p(r) = 2r$, consider the form of the Beta density (ignoring the constant term):

$$\begin{aligned} p(r) \propto r^{\alpha-1}(1-r)^{\beta-1} &\propto 2r \\ &\propto r^1(1-r)^0 \end{aligned}$$

From this, it is clear that the parameters are $\alpha = 2, \beta = 1$.

EX 3.3. Using the same steps as the previous exercise:

$$\begin{aligned} p(r|Y,N) &\propto p(Y=y|r,N)p(r) \\ &\propto r^y(1-r)^{N-y} \times 3r^2 \\ &\propto r^{y+2}(1-r)^{N-y} \\ &= r^{\alpha'-1}(1-r)^{\beta'-1} \end{aligned}$$

which suggests a Beta density with parameters $\alpha' = y+3$ and $\beta' = N-y+1$. To find the prior parameters corresponding to the prior $p(r) = 3r^2$, consider the form of the Beta density (ignoring the constant term):

$$\begin{aligned} p(r) \propto r^{\alpha-1}(1-r)^{\beta-1} &\propto 3r^2 \\ &\propto r^2(1-r)^0 \end{aligned}$$

From this, it is clear that the parameters are $\alpha = 3, \beta = 1$.

EX 3.4. The effective sample size is $\alpha - 1$ heads and $\beta - 1$ tails. In the first example, this is 0 of each. In the second, we have 1 head and 0 tails and in the third, 2 heads and 0 tails.

EX 3.5. From the definition of expectations,

$$\begin{aligned} \mathbf{E}_{p(r)}\{r\} &= \int_{r=0}^{r=1} rp(r)\, dr \\ &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_{r=0}^{r=1} r \times r^{\alpha-1}(1-r)^{\beta-1}\, dr \\ &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_{r=0}^{r=1} r^{\alpha}(1-r)^{\beta-1}\, dr \\ &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_{r=0}^{r=1} r^{\alpha'-1}(1-r)^{\beta-1}\, dr \\ \text{where} \quad & \alpha' = \alpha + 1. \end{aligned}$$

Now, the integrand is an unnormalised Beta density so its integral must be the inverse of the Beta normalisation constant. Therefore,

$$\begin{aligned} \mathbf{E}_{p(r)}\{r\} &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha')\Gamma(\beta)}{\Gamma(\alpha'+\beta)} \\ &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha+1)\Gamma(\beta)}{\Gamma(\alpha+\beta+1)}. \end{aligned}$$

Now we require the following Gamma identity, $\Gamma(n+1) = n\Gamma(n)$.

$$\begin{aligned} \mathbf{E}_{p(r)}\{r\} &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \frac{\alpha}{\alpha+\beta} \\ &= \frac{\alpha}{\alpha+\beta}. \end{aligned}$$

EX 3.6. We require

$$\mathsf{var}\{r\} = \mathbf{E}_{p(r)}\left\{r^2\right\} - \left(\mathbf{E}_{p(r)}\left\{r\right\}\right)^2.$$

From the previous Exercise, the second term is

$$\left(\mathbf{E}_{p(r)}\left\{r\right\}\right)^2 = \left(\frac{\alpha}{\alpha+\beta}\right)^2.$$

The first term is computed as follows:

$$\begin{aligned}
\mathbf{E}_{p(r)}\left\{r^2\right\} &= \int_{r=0}^{r=1} r^2 p(r)\, dr \\
&= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_{r=0}^{r=1} r^2 \times r^{\alpha-1}(1-r)^{\beta-1}\, dr \\
&= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_{r=0}^{r=1} r^{\alpha+1}(1-r)^{\beta-1}\, dr \\
&= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_{r=0}^{r=1} r^{\alpha'-1}(1-r)^{\beta-1}\, dr \\
\text{where} \qquad & \alpha' = \alpha + 2.
\end{aligned}$$

As in the previous Exercise, the integrand is just an unnormalised Beta density. Therefore,

$$\begin{aligned}
\mathbf{E}_{p(r)}\left\{r^2\right\} &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha')\Gamma(\beta)}{\Gamma(\alpha'+\beta)} \\
&= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha+1+1)\Gamma(\beta)}{\Gamma(\alpha+\beta+1+1)} \\
&= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha+1)\Gamma(\beta)}{\Gamma(\alpha+\beta+1)} \frac{\alpha+1}{\alpha+\beta+1} \\
&= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \frac{\alpha+1}{\alpha+\beta+1} \frac{\alpha}{\alpha+\beta} \\
&= \frac{\alpha(\alpha+1)}{(\alpha+\beta)(\alpha+\beta+1)},
\end{aligned}$$

where we used the Gamma identity twice. Combining this with the expression for $\left(\mathbf{E}_{p(r)}\left\{r\right\}\right)^2$ gives

$$\begin{aligned}
\mathsf{var}\{r\} &= \frac{\alpha(\alpha+1)}{(\alpha+\beta)(\alpha+\beta+1)} - \left(\frac{\alpha}{\alpha+\beta}\right)^2 \\
&= \frac{\alpha(\alpha+1)}{(\alpha+\beta)(\alpha+\beta+1)} - \frac{\alpha^2}{(\alpha+\beta)^2} \\
&= \frac{\alpha(\alpha+1)(\alpha+\beta)}{(\alpha+\beta)^2(\alpha+\beta+1)} - \frac{\alpha^2(\alpha+\beta+1)}{(\alpha+\beta)^2(\alpha+\beta+1)} \\
&= \frac{\alpha(\alpha+1)(\alpha+\beta) - \alpha^2(\alpha+\beta+1)}{(\alpha+\beta)^2(\alpha+\beta+1)} \\
&= \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}.
\end{aligned}$$

EX 3.7. Assuming that the probability of heads is given by r, we observe y heads in N tosses, and r has a Beta prior with parameters α and β, the posterior density is a Beta density with parameters $\delta = \alpha + y$ and $\gamma = \beta + N - y$. The marginal likelihood is given by:

$$p(y_N|\alpha,\beta) = \left(\begin{array}{c} N \\ y_N \end{array} \right) \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \frac{\Gamma(\alpha+y_N)\Gamma(\beta+N-y_N)}{\Gamma(\alpha+\beta+N)}.$$

The probability of winning is given by:

$$\mathbf{E}_{p(r|y_N)}\left\{P(Y_{\mathsf{new}} \leq 6|r)\right\} = 1 - \sum_{y_{\mathsf{new}}=7}^{y_{\mathsf{new}}=10} \mathbf{E}_{p(r|y_N)}\left\{P(Y_{\mathsf{new}} = y_{\mathsf{new}}|r)\right\}$$

where

$$\mathbf{E}_{p(r|y_N)}\left\{P(Y_{\mathsf{new}} = y_{\mathsf{new}}|r)\right\} = \left(\begin{array}{c} N_{\mathsf{new}} \\ y_{\mathsf{new}} \end{array} \right) \frac{\Gamma(\delta+\gamma)}{\Gamma(\delta)\Gamma(\gamma)} \frac{\Gamma(\delta+y_{\mathsf{new}})\Gamma(\gamma+N_{\mathsf{new}}-y_{\mathsf{new}})}{\Gamma(\delta+\gamma+N_{\mathsf{new}})}$$

From the question, $y_N = 9$, $N = 20$.
Scenario 1: $\alpha = \beta = 1$. The posterior density has parameters $\delta = \alpha + y_N = 10$, $\gamma = \beta + N - y_N = 1 + 20 - 9 = 12$. The marginal likelihood comes out as: 0.0476. The probability of winning as: 0.84812.
Scenario 2: $\alpha = \beta = 50$. The posterior density has parameters $\delta = \alpha + y_N = 59$, $\gamma = \beta + N - y_N = 50 + 20 - 9 = 61$. The marginal likelihood comes out as: 0.1486. The probability of winning as: 0.83162.
Scenario 3: $\alpha = 5, \beta = 1$. The posterior density has parameters $\delta = \alpha + y_N = 14$, $\gamma = \beta + N - y_N = 1 + 20 - 9 = 12$. The marginal likelihood comes out as: 0.0135. The probability of winning as: 0.7211.

EX 3.8. Code below (`scenarios.m`):

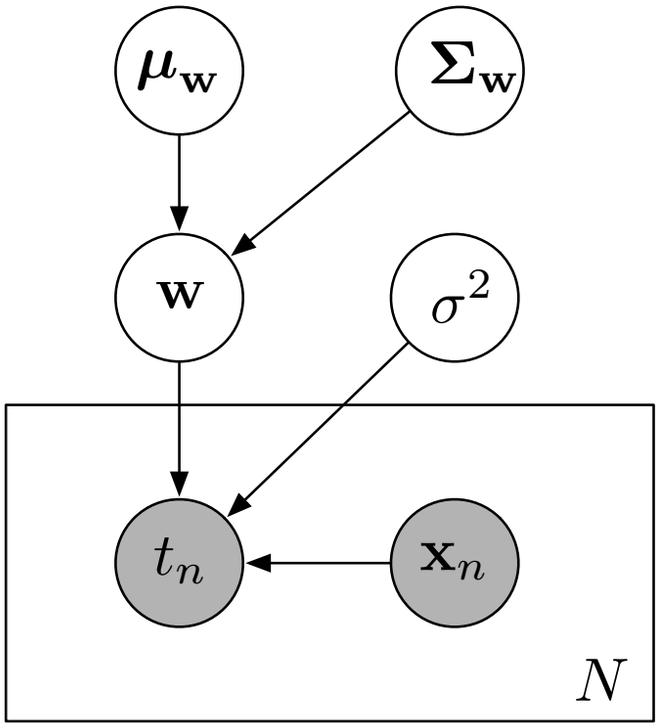
```
clear all;
close all;
% Define an array to hold the three scenarios
% Each row is alpha,beta for one scenario
hypers = [1,1;50,50;5,1];

scenario = 1; % Change this to look at different ones

% Generate the 100 tosses
N = 100;
toss = rand(N,1)<0.7;
Nheads = sum(toss);
Ntails = N - Nheads;

% Compute the posterior
alpha = hypers(scenario,1);
beta = hypers(scenario,2);
postalpha = alpha + Nheads;
postbeta = beta + Ntails;

```

```
% Plot the posterior and prior
figure(1);hold off
x = [0:0.01:1];
plot(x,betapdf(x,hypers(scenario,1),hypers(scenario,2)));
hold on
plot(x,betapdf(x,postalpha,postbeta),'r')
legend('Prior','Posterior');

% Compute the marginal likelihood (Equation 3.14 in book)
% Note we do it in log space first, because of numerical issues
ml = log(factorial(N)/(factorial(Nheads)*factorial(Ntails)))+...
    gammaln(alpha+beta)-gammaln(alpha)-gammaln(beta)+...
    gammaln(postalpha)+gammaln(postbeta)-gammaln(postalpha+postbeta+N);
ml = exp(ml);

% Probability of winning
% Compute prob of y=7,8,9,10
tot = 0;
for i = 7:10
    tot = tot + ...
        exp(...
        log(factorial(10)/(factorial(i)*factorial(10-i))) + ...
        gammaln(postalpha+postbeta) - gammaln(postalpha) - gammaln(postbeta) + ...
        gammaln(postalpha+i) + gammaln(postbeta + 10 - i) - ...
        gammaln(postalpha+postbeta+10)...
        )
end

% Compute probability of winning
prob = 1 - tot;
```

EX 3.9. Firstly equate the two expressions and then re-arrange to find $\boldsymbol{\Sigma}_0$:

$$\begin{aligned}
\frac{1}{\sigma^2}\left(\frac{1}{\sigma^2}\mathbf{X}^\mathsf{T}\mathbf{X}+\boldsymbol{\Sigma}_0^{-1}\right)^{-1}\mathbf{X}^\mathsf{T}\mathbf{t} &= \left(\mathbf{X}^\mathsf{T}\mathbf{X}+N\lambda\mathbf{I}\right)^{-1}\mathbf{X}^\mathsf{T}\mathbf{t} \\
\frac{1}{\sigma^2}\left(\frac{1}{\sigma^2}\mathbf{X}^\mathsf{T}\mathbf{X}+\boldsymbol{\Sigma}_0^{-1}\right)^{-1} &= \left(\mathbf{X}^\mathsf{T}\mathbf{X}+N\lambda\mathbf{I}\right)^{-1} \\
\left(\mathbf{X}^\mathsf{T}\mathbf{X}+N\lambda\mathbf{I}\right)\frac{1}{\sigma^2}\left(\frac{1}{\sigma^2}\mathbf{X}^\mathsf{T}\mathbf{X}+\boldsymbol{\Sigma}_0^{-1}\right)^{-1} &= \mathbf{I} \\
\left(\mathbf{X}^\mathsf{T}\mathbf{X}+N\lambda\mathbf{I}\right)\frac{1}{\sigma^2} &= \left(\frac{1}{\sigma^2}\mathbf{X}^\mathsf{T}\mathbf{X}+\boldsymbol{\Sigma}_0^{-1}\right) \\
\frac{1}{\sigma^2}\mathbf{X}^\mathsf{T}\mathbf{X}+\frac{N\lambda}{\sigma^2}\mathbf{I} &= \frac{1}{\sigma^2}\mathbf{X}^\mathsf{T}\mathbf{X}+\boldsymbol{\Sigma}_0^{-1} \\
\boldsymbol{\Sigma}_0 &= \frac{\sigma^2}{N\lambda}\mathbf{I} \qquad (3.1)
\end{aligned}$$

EX 3.10. See figure below:



EX 3.11. Code below (`marglike.m`):

```
clear all;close all;
% Generate the data
N = 100;
x = rand(N,1)*10-5;
x = sort(x);
ss = 100;
f = 5*x.^3 - x.^2 + x;
t = f + randn(size(f))*sqrt(ss);

% Choose \sigma_0^2
ss0 = 0.3; % Try increasing this

% Loop over model orders
order = [1:7];
X = repmat(1,size(x));
for i = 1:length(order)
    mu0 = zeros(order(i)+1,1);
    s0 = ss0*eye(order(i)+1);
    X = [X x.^order(i)];

    mlmean = X*mu0;
    mlcov = ss*eye(N) + X*s0*X';

    logml(i) = -(N/2)*log(2*pi) - (N/2)*log(det(mlcov)) - ...
        0.5*(t-mlmean)'*inv(mlcov)*(t-mlmean);
```

```
26
27      % Note that the non log value may be numerically instable.
28      ml(i) = exp(logml(i));
29  end
```

EX 3.12. Assuming $\mathbf{w}$ is fixed, and the following inverse gamma prior on σ^2:

$$p(\sigma^2|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}(\sigma^2)^{-\alpha-1}\exp\left\{-\frac{\beta}{\sigma^2}\right\}.$$

The likelihood is the standard Gaussian likelihood:

$$p(\mathbf{t}|\mathbf{w},\mathbf{X},\sigma^2) = \frac{1}{(2\pi)^{N/2}(\sigma^2)^{N/2}}\exp\left\{-\frac{1}{2\sigma^2}(\mathbf{t}-\mathbf{X}\mathbf{w})^\mathsf{T}(\mathbf{t}-\mathbf{X}\mathbf{w})\right\}.$$

The posterior density over σ^2 is therefore proportional to the product of these two densities. Collecting similar terms:

$$p(\sigma^2|\mathbf{w},\mathbf{X},\alpha,\beta) \propto (\sigma^2)^{-\alpha-N/2-1}\exp\left\{-\frac{1}{\sigma^2}\left(\beta+\frac{1}{2}(\mathbf{t}-\mathbf{X}\mathbf{w})^\mathsf{T}(\mathbf{t}-\mathbf{X}\mathbf{w})\right)\right\}.$$

This is another inverse gamma with parameters α^* and β^* given by:

$$\begin{aligned}\alpha^* &= \alpha+\frac{N}{2}\\ \beta^* &= \beta+\frac{1}{2}(\mathbf{t}-\mathbf{X}\mathbf{w})^\mathsf{T}(\mathbf{t}-\mathbf{X}\mathbf{w}).\end{aligned}$$

Chapter 4

EX 4.1. Assuming the following prior density for $\mathbf{w}$:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \mathbf{I}_D),$$

the posterior is:

$$p(\mathbf{w}|t_1, \ldots, t_N, \mathbf{x}_1, \ldots, \mathbf{x}_N) \propto \mathcal{N}(\mathbf{0}, \mathbf{I}_D) \prod_{n=1}^{N} p(t_n|\mathbf{w}, \mathbf{x}_n)$$

Combining all of the $D-$dimensional data objects into an $N \times D$ matrix $\mathbf{X}$ and the labels into a vector $\mathbf{t}$ allows us to write this as the product of two multi-variate Gaussians:

$$p(\mathbf{w}|\mathbf{t}, \mathbf{X}) \propto \exp\left\{-\frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{w}\right\} \exp\left\{-\frac{1}{2}(\mathbf{t} - \mathbf{X}\mathbf{w})^\mathsf{T}(\mathbf{t} - \mathbf{X}\mathbf{w})\right\}.$$

Re-arranging and equating coefficients gives the posterior:

$$\begin{aligned} p(\mathbf{w}|\mathbf{t}, \mathbf{X}) &\propto \exp\left\{-\frac{1}{2}\left(\mathbf{w}^\mathsf{T}\mathbf{w} + \mathbf{t}^\mathsf{T}\mathbf{t} + \mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{X}\mathbf{w} - \mathbf{w}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{t}\right)\right\} \\ &= \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ \boldsymbol{\Sigma} &= (\mathbf{X}^\mathsf{T}\mathbf{X} + \mathbf{I}_D)^{-1} \\ \boldsymbol{\mu} &= \boldsymbol{\Sigma}\mathbf{X}^\mathsf{T}\mathbf{t}. \end{aligned}$$

The Laplace approximation approximates the posterior with a Gaussian with mean equal to the posterior mode and a covariance matrix equal to the negative inverse of the second derivate of the log posterior. Removing constants, the log posterior is equal to:

$$g = -\frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{w} - \frac{1}{2}(\mathbf{t} - \mathbf{X}\mathbf{w})^\mathsf{T}(\mathbf{t} - \mathbf{X}\mathbf{w}).$$

To find the maximum value, we take partial derivatives with respect to $\mathbf{w}$:

$$\frac{\partial g}{\partial \mathbf{w}} = -\frac{1}{2}\left(2(\mathbf{I} + \mathbf{X}^\mathsf{T}\mathbf{X})\mathbf{w} - 2\mathbf{X}^\mathsf{T}\mathbf{t}\right).$$

Setting to zero and solving for $\mathbf{w}$ gives:

$$\mathbf{w} = (\mathbf{I} + \mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{t}.$$

This will be the mean of the Gaussian approximation and we can already see that it is equal to $\boldsymbol{\mu}$, the posterior mean.

To find the covariance matrix, we take second derivatives of g:

$$\frac{\partial^2 g}{\partial \mathbf{w} \partial \mathbf{w}^\mathsf{T}} = -(\mathbf{I}_D + \mathbf{X}^\mathsf{T}\mathbf{X}).$$

The covariance of the Laplace approximation is therefore:

$$(\mathbf{I} + \mathbf{X}^\mathsf{T}\mathbf{X})^{-1}.$$

This is identical to the posterior covariance matrix thus confirming that the approximation is, in this case, exact.

EX 4.2. We are interested in the Laplace approximation to the posterior $p(r|N, y, \alpha, \beta)$. The first step is to find the MAP value of r. The log posterior is (removing constants):

$$g = (\alpha - 1)\log r + (\beta - 1)\log(1 - r) + y \log r + (N - y)\log(1 - r).$$

Taking partial derivatives with respect to r leaves:

$$\frac{\partial g}{\partial r} = \frac{\alpha + y - 1}{r} - \frac{\beta + N - y - 1}{1 - r}.$$

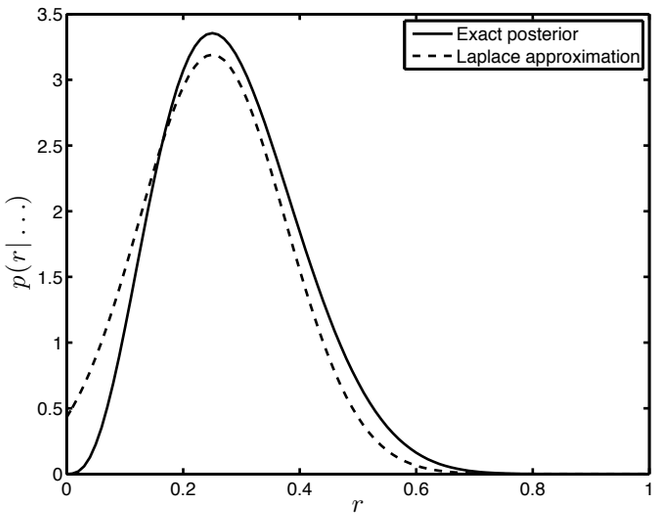
Re-arranging this expression gives the MAP solution for r:

$$\widehat{r} = \frac{\alpha + y - 1}{\alpha + \beta + N - 2}.$$

This will be the mean of the Gaussian approximation. The variance is given by:

$$\begin{aligned} \sigma^2 &= -\left(\frac{\partial^2 g}{\partial r^2}\right)^{-1} \\ &= -\left(-\frac{\alpha + y - 1}{r^2} - \frac{\beta + N - y - 1}{(1 - r)^2}\right)^{-1} \end{aligned}$$

EX 4.3. The plot below shows an example where $\alpha = \beta = 2$, $y = 2$, and $N = 10$.



EX 4.4. This is a nice example to illustrate the power of sampling. Imagine sampling pairs of values from uniform distributions between -0.5 and 0.5. These values would be uniformly distributed in the square shown below:

The probability that one of these points lies within the circle must be equal to the ratio of the area of the circle to the area of the square. We can obtain an empirical estimate of this value by say generating N pairs (i.e. points) and then computing the number of pairs that lie within the circle (s). Then:

$$\frac{s}{N} \approx \frac{\pi r^2}{1 \times 1} = 0.5^2 \pi.$$

Re-arranging this expression will give an approximate value of π:

$$\pi \approx \frac{s}{0.5^2 N}.$$

For example, one random sample of 100 points yields a value of:

$$\pi \approx 3.240.$$

As we would expect, this approximation will, on average, get better as we take more samples. An example after 10000 is:

$$\pi \approx 3.129.$$

EX 4.5. Re-arrange as follows:

$$\begin{aligned} P(T_{\mathsf{new}} = 1 | \mathbf{x}_{\mathsf{new}}, \widehat{\mathbf{w}}) &= \frac{1}{1} 1 + \exp(-\widehat{\mathbf{w}}^{\mathsf{T}} \mathbf{x}_{\mathsf{new}}) \\ 0.5 &= \frac{1}{2} = \frac{1}{1} 1 + \exp(-\widehat{\mathbf{w}}^{\mathsf{T}} \mathbf{x}_{\mathsf{new}}) \\ 2 &= 1 + \exp(-\widehat{\mathbf{w}}^{\mathsf{T}} \mathbf{x}_{\mathsf{new}}) \\ 1 &= \exp(0) = \exp(-\widehat{\mathbf{w}}^{\mathsf{T}} \mathbf{x}_{\mathsf{new}}) \\ \text{therefore} \quad & \widehat{\mathbf{w}}^{\mathsf{T}} \mathbf{x}_{\mathsf{new}} = 0. \end{aligned}$$

EX 4.6. Assuming a Gaussian prior:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{0}, \mathbf{I}_D),$$

and:

$$v_n = \exp\{\mathbf{w}^{\mathsf{T}} \mathbf{x}_n\}$$

the log posterior is proportional to:

$$g = -\frac{1}{2} \mathbf{w}^{\mathsf{T}} \mathbf{w} + \sum_n \left(t_n \log v_n - v_n \right).$$

The gradient is obtained by taking partial derivatives with respect to $\mathbf{w}$:

$$\frac{\partial g}{\partial \mathbf{w}} = -\mathbf{w} + \sum_n \frac{t_n}{v_n} \frac{\partial v_n}{\partial \mathbf{w}} - \frac{\partial v_n}{\partial \mathbf{w}}.$$

Now,

$$\frac{\partial v_n}{\partial \mathbf{w}} = \mathbf{x}_n v_n$$

and therefore:

$$\frac{\partial g}{\partial \mathbf{w}} = -\mathbf{w} + \sum_n \mathbf{x}_n (t_n - v_n).$$

The Hessian is obtained by differentiating a second time. You might find it worthwhile to work out each element of the matrix individually for (say) $D = 2$ if this step appears difficult:

$$\frac{\partial^2 g}{\partial \mathbf{w} \partial \mathbf{w}^\mathsf{T}} = -\mathbf{I} - \sum_n v_n \mathbf{x}_n \mathbf{x}_n^\mathsf{T}$$

Given some initial value of $\mathbf{w}$, the optimisation scheme repeatadly updates $\widehat{\mathbf{w}}$ as follows until it converges to a maximum:

$$\mathbf{w} \leftarrow \mathbf{w} - \left(-\mathbf{I} - \sum_n v_n \mathbf{x}_n \mathbf{x}_n^\mathsf{T} \right)^{-1} \left(-\mathbf{w} + \sum_n \mathbf{x}_n (t_n - v_n) \right)$$

EX 4.7. Given the result of the previous exercise, this is straightforward. The Laplace approximation is a Gaussian with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ where $\boldsymbol{\mu} = \widehat{\mathbf{w}}$, the maximum obtained using the Newton-Raphson scheme derived in the previous exercise and the covariance is:

$$\boldsymbol{\Sigma} = - \left(-\mathbf{I} - \sum_n \widehat{v_n} \mathbf{x}_n \mathbf{x}_n^\mathsf{T} \right)^{-1},$$

where:

$$\widehat{v_n} = \exp\{\widehat{\mathbf{w}}^\mathsf{T} \mathbf{x}_n\}.$$

EX 4.8. Example code below (`poisscount.m`):

```
%% Laplace and Metropolis–Hastings for a Poisson Counts model
% Simon Rogers, August 2011
clear all;
close all;

%% Generate the 'true' data.
N = 200; % Number of data objects
x = [repmat(1,N,1) randn(N,1)];
w = randn(2,1);

temp = exp(x*w);

t = poissrnd(temp);

truew = w;
%% Newton Raphson
% Initialise w
w = rand(2,1)*5;
wall = [];
```

```
change = inf;
temp = exp(x*w);
ll = -0.5*w'*w + sum(t.*log(temp) - temp);
while change > 1e-3
    % Compute gradient
    temp = exp(x*w);
    gra = -w + sum(repmat(t,1,2).*x - repmat(temp,1,2).*x,1)';
    % Compute Hessian
    hess = -eye(2) - x'*(x.*repmat(temp,1,2));
    % Update w
    neww = w - inv(hess)*gra;
    % Compute change
    change = sum((neww-w).^2);
    w = neww;

    % Store all of the w
    wall = [wall;w'];
    % Compute the likelihood (should always increase)
    temp = exp(x*w);
    ll = [ll;-0.5*w'*w + sum(t.*log(temp) - temp)];
end

%% Metropolis-Hastings
% Initialise w
wm = randn(2,1);
temp = exp(x*wm);
wall_m = [];
% Compute old log posterior
oldll = -0.5*wm'*wm + sum(t.*log(temp) - temp);
for s = 1:10000 % Generate 10000 samples
    % Propose a new candidate
    wnew = wm + randn(2,1)*0.1; % Gaussian proposal, std = 0.1
    % Compute new log likelihood
    temp = exp(x*wnew);
    newll = -0.5*wnew'*wnew + sum(t.*log(temp) - temp);
    % Accept of reject
    if rand<exp(newll-oldll)
        wm = wnew;
        oldll = newll;
    end
    % Store all w
    wall_m = [wall_m;wm'];
end

% Plot samples
% No thinning or ignoring of burn-in
figure(1);
hold off
plot(wall_m(:,1),wall_m(:,2),'k.','color',[0.6 0.6 0.6])


%% Plot the approximate posterior
% Compute posterior (negative inverse of Hessian)
gcov = -inv(-eye(2) - x'*(x.*repmat(temp,1,2)));
% Evaluate over a set of points
[W1,W2] = meshgrid([-1:0.01:1],[-1:0.01:1]);
gli = 1./(sqrt(2*pi)*det(gcov)).*repmat(1,size(W1));
for i = 1:size(W1,1)
    for j = 1:size(W1,2)
```

```
79          wv = [W1(i,j);W2(i,j)];
80          gli(i,j) = gli(i,j).*exp(-0.5*(wv-w)'*inv(gcov)*(wv-w));
81      end
82  end
83  % Plot the contours
84  hold on
85  contour(W1,W2,gli,'k')
86
87
88  % Plot the true value
89  plot(truew(1),truew(2),'ko','markersize',20,'linewidth',2)
```

Chapter 5

EX 5.1. The class conditional density is given by:

$$p(\mathbf{x}_n|t_n = c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) = \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c).$$

Assuming a Gaussian prior on $\boldsymbol{\mu}_c$:

$$p(\boldsymbol{\mu}_c|\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0),$$

the posterior density for $\boldsymbol{\mu}$ is given by:

$$p(\boldsymbol{\mu}_c|\mathbf{x}_1, \ldots, \mathbf{x}_{N_c}, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0, \boldsymbol{\Sigma}_c) \propto \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0) \prod_{n=1}^{N_c} p(\mathbf{x}_n|t_n = c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c).$$

Removing constant terms and re-arranging:

$$p(\boldsymbol{\mu}_c|\ldots) \propto \exp\left\{-\frac{1}{2}(\boldsymbol{\mu}_c - \boldsymbol{\mu}_0)^\mathsf{T}\boldsymbol{\Sigma}_0^{-1}(\boldsymbol{\mu}_c - \boldsymbol{\mu}_0)\right\} \exp\left\{-\frac{1}{2}\sum_{n=1}^{N_c}(\mathbf{x}_n - \boldsymbol{\mu}_c)^\mathsf{T}(\mathbf{x}_n - \boldsymbol{\mu}_c)\right\}.$$

Keeping only terms dependent on $\boldsymbol{\mu}_c$:

$$p(\boldsymbol{\mu}_c|\ldots) \propto \exp\left\{\frac{1}{2}\left(\boldsymbol{\mu}_c^\mathsf{T}\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_c + N_c\boldsymbol{\mu}_c^\mathsf{T}\boldsymbol{\mu}_c - 2\boldsymbol{\mu}_c^\mathsf{T}\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0 - 2\boldsymbol{\mu}_c^\mathsf{T}\sum_{n=1}^{N_c}\mathbf{x}_n\right)\right\}.$$

Therefore, the posterior is a Gaussian:

$$p(\boldsymbol{\mu}_c|\ldots) = \mathcal{N}(\mathbf{m}, \mathbf{S}),$$

where:

$$\begin{aligned}\mathbf{S} &= \left(\boldsymbol{\Sigma}_0^{-1} + N_c\mathbf{I}\right)^{-1} \\ \mathbf{m} &= \mathbf{S}\left(\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0 + \sum_{n=1}^{N_c}\mathbf{x}_n\right).\end{aligned}$$

EX 5.2. This requires the expectation of a Gaussian with respect to another Gaussian:

$$\begin{aligned}p(\mathbf{x}_{\mathsf{new}}|T_{\mathsf{new}} = c, \mathbf{X}, \mathbf{t}) &= \mathbf{E}_{p(\boldsymbol{\mu}_c|\mathbf{m},\mathbf{S})}\left\{p(\mathbf{x}_{\mathsf{new}}|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)\right\} \\ &= \int \mathcal{N}_{\boldsymbol{\mu}_c}(\mathbf{m}, \mathbf{S})\mathcal{N}_{\mathbf{x}_{\mathsf{new}}}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)\, d\boldsymbol{\mu}_c \\ &= \int \mathcal{N}_{\boldsymbol{\mu}_c}(\mathbf{m}, \mathbf{S})\mathcal{N}_{\boldsymbol{\mu}_c}(\mathbf{x}_n, \boldsymbol{\Sigma}_c)\, d\boldsymbol{\mu}_c.\end{aligned}$$

We now make use of the following standard result:

$$\mathcal{N}_{\mathbf{x}}(\mathbf{a}, \mathbf{A})\mathcal{N}_{\mathbf{x}}(\mathbf{b}, \mathbf{B}) = K\mathcal{N}_{\mathbf{x}}(\mathbf{d}, \mathbf{D}),$$

where:

$$K = \mathcal{N}_{\mathbf{a}}(\mathbf{b}, \mathbf{A} + \mathbf{B}).$$

The quantities $\mathbf{d}$ and $\mathbf{D}$ are not important as this density will disappear in our integral. Therefore, our expectation becomes:

$$p(\mathbf{x}_{\mathsf{new}}|T_{\mathsf{new}} = c, \mathbf{X}, \mathbf{t}) = \mathcal{N}_{\mathbf{x}_{\mathsf{new}}}(\mathbf{m}, \mathbf{S} + \mathbf{\Sigma}_c).$$

EX 5.3. The total likelihood for class c is:

$$\mathcal{L} = \prod_{n=1}^{N_c} \frac{1}{(2\pi)^{D/2}|\mathbf{\Sigma}_c|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}_n - \boldsymbol{\mu}_c)^{\mathsf{T}}\mathbf{\Sigma}_c^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_c)\right\}.$$

As normal, it is easiest to maximise the log likelihood. Logging and multiplying out gives:

$$\log \mathcal{L} = -\frac{DN_c}{2}\log(2\pi) - \frac{N_c}{2}\log|\mathbf{\Sigma}_c| - \frac{1}{2}\sum_{n=1}^{N_c}\left(\mathbf{x}_n^{\mathsf{T}}\mathbf{\Sigma}_c^{-1}\mathbf{x}_n - 2\boldsymbol{\mu}_c^{\mathsf{T}}\mathbf{\Sigma}_c^{-1}\mathbf{x}_n + \boldsymbol{\mu}_c^{\mathsf{T}}\mathbf{\Sigma}_c^{-1}\boldsymbol{\mu}_c\right).$$

Taking the partial derivative with respect to $\boldsymbol{\mu}_c$, equating to zero and solving gives a solution for $\boldsymbol{\mu}_c$:

$$\begin{aligned}
\frac{\partial \log \mathcal{L}}{\partial \boldsymbol{\mu}_c} = 0 &= \mathbf{\Sigma}_c^{-1}\sum_{n=1}^{N_c}\mathbf{x}_n - N_c\mathbf{\Sigma}_c^{-1}\boldsymbol{\mu}_c \\
\mathbf{\Sigma}_c^{-1}\sum_{n=1}^{N_c}\mathbf{x}_n &= N_c\mathbf{\Sigma}_c^{-1}\boldsymbol{\mu}_c \\
\boldsymbol{\mu}_c &= \frac{1}{N_c}\sum_{n=1}^{N_c}\mathbf{x}_n.
\end{aligned}$$

Similarly, for $\mathbf{\Sigma}_c$:

$$\frac{\partial \log \mathcal{L}}{\partial \mathbf{\Sigma}_c} = 0 = -\frac{N_c}{2}\mathbf{\Sigma}_c^{-1} + \frac{1}{2}\mathbf{\Sigma}_c^{-1}\sum_{n=1}^{N_c}(\mathbf{x}_n - \boldsymbol{\mu}_c)(\mathbf{x}_n - \boldsymbol{\mu}_c)^{\mathsf{T}}\mathbf{\Sigma}_c^{-1},$$

where we have used the following two results (for symmetric $\mathbf{A}$):

$$\begin{aligned}
\frac{\partial \log|\mathbf{A}|}{\partial \mathbf{A}} &= \mathbf{A}^{-1} \\
\frac{\partial \mathbf{a}^{\mathsf{T}}\mathbf{A}^{-1}\mathbf{a}}{\partial \mathbf{A}} &= -\mathbf{A}^{-1}\mathbf{a}\mathbf{a}^{\mathsf{T}}\mathbf{A}^{-1}.
\end{aligned}$$

Re-arranging gives the expression for $\mathbf{\Sigma}_c$:

$$\mathbf{\Sigma}_c = \frac{1}{N_c}\sum_{n=1}^{N_c}(\mathbf{x}_n - \mu_c)(\mathbf{x}_n - \mu_c)^{\mathsf{T}}$$

where $\boldsymbol{\mu}_c$ is the maximum likelihood estimate derived above.

EX 5.4. The data consists of N_c M-dimensional vectors of integer counts. The multinomial likelihood for the N_c data objects is:

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_{N_c} | q_{c1}, \ldots, q_{cM}) = \mathcal{L} \propto \prod_{n=1}^{N_c} \prod_{m=1}^{M} q_{cm}^{x_{nm}},$$

(where the constant that does not depend on q_{cm} has been omitted for brevity). Once again, we work with the log likelihood and take partial derivatives with respect to q_{cm}. We must also add a Lagrangian term due to the constraing that $\sum_m q_{cm} = 1$. Calling this new objective function g gives:

$$\begin{aligned}
\log \mathcal{L} &= \sum_{n=1}^{N_c} \sum_{m=1}^{M} x_{nm} \log q_{cm} \\
g &= \sum_{n=1}^{N_c} \sum_{m=1}^{M} x_{nm} \log q_{cm} - \lambda \left(\sum_m q_{cm} - 1 \right) \\
\frac{\partial \log \mathcal{L}}{\partial q_{cm}} = 0 &= \sum_{n=1}^{N_c} \frac{x_{nm}}{q_{cm}} - \lambda \\
q_{cm} &= \frac{\sum_{n=1}^{N_c} x_{nm}}{\lambda}.
\end{aligned}$$

To compute λ we sum both sides over m:

$$\begin{aligned}
\sum_m q_{cm} &= \frac{1}{\lambda} \sum_{m,n} x_{nm} \\
1 &= \frac{1}{\lambda} \sum_{m,n} x_{nm} \\
\lambda &= \frac{1}{\sum_{n,m} x_{nm}}.
\end{aligned}$$

Therefore, the maximum likelihood estimate of q_{cm} is:

$$q_{cm} = \frac{\sum_n x_{nm}}{\sum_{n,m'} x_{nm'}}.$$

EX 5.5. The likelihood (ignoring terms not involving q_{cm}) is:

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_{N_c} | \mathbf{q}_c) \propto \prod_{n=1}^{N_c} \prod_{m=1}^{M} q_{cm}^{x_{nm}}.$$

The prior over $\mathbf{q}_c$ is a Dirichlet with constant parameter α (again, ignoring terms not involving q_{cm}):

$$p(\mathbf{q}_c | \alpha) \propto \prod_{m=1}^{M} q_{cm}^{\alpha - 1}.$$

The Dirichlet is the conjugate prior to the multinomial likelihood. Therefore, we know the posterior will be another Dirichlet. Multiplying the prior and likelihood together and combing q_{cm} terms gives:

$$p(\mathbf{q}_c|\alpha, \mathbf{x}_1, \ldots, \mathbf{x}_{N_c}) \propto \prod_{m=1}^{M} q_{cm}^{\alpha-1+\sum_n x_{nm}}.$$

This is the form of a Dirichlet, where the mth parameter is $\beta_m = \alpha + \sum_n x_{nm}$.

EX 5.6. The required expectation is:

$$p(\mathbf{x}_{\mathsf{new}}|T_{\mathsf{new}} = c, \mathbf{X}, \mathbf{t}) = \mathbf{E}_{p(\mathbf{q}_c|\mathbf{X}^c,\alpha)}\left\{p(\mathbf{x}_{\mathsf{new}}|\mathbf{q}_c\right\}.$$

Writing out the expectation in full gives:

$$\begin{aligned} p(\mathbf{x}_{\mathsf{new}}|T_{\mathsf{new}} = c, \mathbf{X}, \mathbf{t}) &= \int \left[\frac{\Gamma(\sum_m \beta_m)}{\prod_m \Gamma(\beta_m)} \prod_m q_{cm}^{\beta_m - 1}\right] \left[\frac{(\sum_m x_{\mathsf{new},m})!}{\prod_m (x_{\mathsf{new},m}!)} \prod_m q_{cm}^{x_{\mathsf{new},m}}\right] d\mathbf{q_c} \\ &= \frac{(\sum_m x_{\mathsf{new},m})!}{\prod_m (x_{\mathsf{new},m}!)} \frac{\Gamma(\sum_m \beta_m)}{\prod_m \Gamma(\beta_m)} \int \prod_m q_{cm}^{\beta_m + x_{\mathsf{new},m} - 1} \, d\mathbf{q}_c \\ &= \frac{(\sum_m x_{\mathsf{new},m})!}{\prod_m (x_{\mathsf{new},m}!)} \frac{\Gamma(\sum_m \beta_m)}{\prod_m \Gamma(\beta_m)} \frac{\prod_m \Gamma(\beta_m + x_{\mathsf{new},m})}{\Gamma(\sum_m \beta_m + x_{\mathsf{new},m})}. \end{aligned}$$

EX 5.7. The MAP estimate is obtained by maximising the likelihood multiplied by the prior. Extracting only the terms that include q_{cm}, and logging give us:

$$\log \mathcal{L} \propto \sum_m (\alpha - 1) \log q_{cm} + \sum_m \left(\sum_n x_{nm}\right) \log q_{cm}.$$

Introducing the necessary Lagrangian (to ensure $\sum_m q_{cm} = 1$) we end up with the following objective function:

$$g = \sum_m \left(\sum_n x_{nm} + \alpha - 1\right) \log q_{cm} - \lambda \left(\sum_m q_{cm} - 1\right).$$

Taking partial derivatives, equating to zero and solving gives:

$$\begin{aligned} \frac{\partial g}{\partial q_{cm}} = 0 &= \frac{\alpha - 1 + \sum_n x_{nm}}{q_{cm}} - \lambda \\ q_{cm} &= \frac{\alpha - 1 + \sum_n x_{nm}}{M(\alpha - 1) + \sum_{n,m'} x_{nm'}}, \end{aligned}$$

where we summed both sides over m to obtain the value for λ.

EX 5.8. The starting point is:

$$\underset{\mathbf{w}}{\text{argmin}} \;\; \frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{w} + C \sum_n \xi_n$$

subject to:

$$\xi_n \geq 0, \quad t_n(\mathbf{w}^\mathsf{T}\mathbf{x}_n + b) \geq 1 - \xi_n.$$

Adding the constraints as Lagrangian terms gives:

$$\frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{w} + C\sum_n \xi_n - \sum_n \alpha_n(t_n(\mathbf{w}^\mathsf{T}\mathbf{x}_n + b) - 1 + \xi_n) - \sum_n \xi_n\gamma_n$$

subject to $\alpha_n \geq 0,\ \gamma_n \geq 0$, which must be minimised with respect to $\mathbf{w}$, b and ξ_n and maximised with respect to α_n and γ_n. At the solution:

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{w}} &= \mathbf{w} - \sum_n \alpha_n t_n = 0 \\
\frac{\partial}{\partial b} &= -\sum_n \alpha_n t_n = 0 \\
\frac{\partial}{\partial \xi_n} &= C - \alpha_n - \gamma_n = 0.
\end{aligned}$$

Substituing the first and third of these expressions back into the objective function and re-arranging gives:

$$-\frac{1}{2}\sum_{n,m} \alpha_n\alpha_m t_n t_m \mathbf{x}_n^\mathsf{T}\mathbf{x}_m + \sum_n \alpha_n$$

which must be maximised with respect to α subject to:

$$\sum_n \alpha_n t_n = 0,\ 0 \leq \alpha_n \leq C.$$

Where the first constraint comes from the partial derivative with respect to b and the second from the partial derivative with respect to ξ_n and the fact that $\gamma_n \geq 0$.

Chapter 6

EX 6.1. The log likelihood for all N data objects is:

$$\mathcal{L} = \sum_n \log \sum_k \pi_k \prod_d p(x_{nd}|\mu_{kd}, \sigma_{kd}^2).$$

Insersting a set of variational parameters, q_{nk} such that $\sum_k q_{nk} = 1$ and $q_{nk} \geq 0$ and subsequently applying Jensen's inequality results in:

$$\begin{aligned} \mathcal{L} &= \sum_n \log \sum_k \pi_k \prod_d \frac{q_{nk}}{q_{nk}} p(x_{nd}|\mu_{kd}, \sigma_{kd}^2) \\ &\geq \sum_n \sum_k q_{nk} \log \pi_k + q_{nk} \sum_d \log p(x_{nd}|\mu_{kd}, \sigma_{kd}^2) - q_{nk} \log q_{nk}. \end{aligned}$$

Only the second term involves σ_{kd}^2. Writing this term out in full leaves:

$$\sum_n \sum_k q_{nk} \sum_d \left(-\frac{1}{2} \log 2\pi - \frac{1}{2} \log \sigma_{kd}^2 - \frac{1}{2\sigma_{kd}^2} (x_{nd} - \mu_{kd})^2 \right).$$

Taking partial derivatives with respect to σ_{kd}^2 results in:

$$\frac{\partial}{\partial \sigma_{kd}^2} = -\frac{\sum_n q_{nk}}{2\sigma_{kd}^2} + \sum_n \frac{q_{nk}}{2(\sigma_{kd}^2)^2} (x_{nd} - \mu_{kd})^2.$$

Equating to zero and re-arranging results in:

$$\sigma_{kd}^2 = \frac{\sum_n q_{nk} (x_{nd} - \mu_{kd})^2}{\sum_n q_{nk}}.$$

EX 6.2. Starting from the log likelihood above and applying Jensen's inequality:

$$\begin{aligned} \mathcal{L} &= \sum_n \log \sum_k \pi_k \prod_d \frac{q_{nk}}{q_{nk}} p(x_{nd}|\mu_{kd}, \sigma_k^2) \\ &\geq \sum_n \sum_k q_{nk} \log \pi_k + q_{nk} \sum_d \log p(x_{nd}|\mu_{kd}, \sigma_k^2) - q_{nk} \log q_{nk}. \end{aligned}$$

σ_l^2 only appears in the second term. Expanding:

$$\sum_n \sum_k q_{nk} \left(-\frac{D}{2} \log 2\pi - \frac{D}{2} \log \sigma_k^2 - \frac{1}{2\sigma_k^2} \sum_d (x_{nd} - \mu_{kd})^2 \right).$$

Taking partial derivatives with respect to σ_k^2 gives:

$$\frac{\partial}{\partial \sigma_k^2} = -\frac{D}{2\sigma_k^2}\sum_n q_{nk} + \frac{1}{2(\sigma_k^2)^2}\sum_{d,n}(x_{nd}-\mu_{kd})^2.$$

Equating to zero and re-arranging gives:

$$\sigma_k^2 = \frac{\sum_n q_{nk}\sum_d (x_{nd}-\mu_{kd})^2}{D\sum_n q_{nk}}$$

EX 6.3. The log likelihood is given by:

$$\mathcal{L} = \sum_n \log \sum_k \pi_k \prod_d p_{kd}^{x_{nd}}(1-p_{kd})^{1-x_{nd}}.$$

Applying Jensen's inequality gives:

$$\mathcal{L} \geq \sum_n \sum_k \left[q_{nk}\log \pi_k + q_{nk}\sum_d \left(x_{nd}\log p_{kd} + (1-x_{nd})\log(1-p_{kd})\right) - q_{nk}\log q_{nk}\right]$$

Taking partial derivatives with respect to p_{kd} setting to zero and solving gives:

$$\begin{aligned}
\frac{\partial}{\partial p_{kd}} = 0 &= \sum_n q_{nk}\frac{x_{nd}}{p_{kd}} - \sum_n q_{nk}\frac{1-x_{nd}}{1-p_{kd}} \\
\sum_n q_{nk}\frac{x_{nd}}{p_{kd}} &= \sum_n q_{nk}\frac{1-x_{nd}}{1-p_{kd}} \\
\sum_n q_{nk}x_{nd} - p_{kd}\sum_n q_{nk}x_{nd} &= p_{kd}\sum_n q_{nk} - p_{kd}\sum_n q_{nk}x_{nd} \\
p_{kd} &= \frac{\sum_n q_{nk}x_{nd}}{\sum_n q_{nk}}
\end{aligned}$$

EX 6.4. This proceeds in exactly the same manner as the previous exercise except that after applying Jensen's inequality, the bound will also include a term from the Beta prior. Including only terms that involve q_{nk} (and removing summations over k and d):

$$(\alpha-1)\log p_{kd} + (\beta-1)\log(1-p_{kd}) + \sum_n q_{nk}\left(x_{nd}\log p_{kd} + (1-x_{nd})\log(1-p_{kd})\right).$$

Taking partial derivatives with respect to p_{kd}, equating to zero and solving gives:

$$\begin{aligned}
\frac{\partial}{\partial p_{kd}} = 0 &= \frac{\alpha-1}{p_{kd}} - \frac{\beta-1}{1-p_{kd}} + \sum_n q_{nk}\frac{x_{nd}}{p_{kd}} - \sum_n q_{nk}\frac{1-x_{nd}}{1-p_{kd}} \\
\sum_n q_{nk}\frac{x_{nd}}{p_{kd}} + \frac{\alpha-1}{p_{kd}} &= \sum_n q_{nk}\frac{1-x_{nd}}{1-p_{kd}} + \frac{\beta-1}{1-p_{kd}} \\
(\alpha-1) - p_{kd}(\alpha-1) + \sum_n q_{nk}x_{nd} &= p_{kd}(\beta-1) + p_{kd}\sum_n q_{nk} \\
p_{kd} &= \frac{\alpha-1+\sum_n q_{nk}x_{nd}}{\alpha+\beta-2+\sum_n q_{nk}}.
\end{aligned}$$

EX 6.5. The log likelihood (after applying Jensen's ineuqality and omitting terms not featuring μ_{kd}) is:

$$\mathcal{L} \geq -\frac{1}{2s^2}(\mu_{kd} - m)^2 - \sum_n \sum_k q_{nk} \sum_d \frac{1}{2\sigma_{kd}^2}(x_{nd} - \mu_{kd})^2 + \ldots$$

Taking partial derivatives with respect to μ_{kd}, equating to zero and re-arranging gives:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mu_{kd}} &= -\frac{1}{s^2}(\mu_{kd} - m) + \frac{1}{\sigma_{kd}^2} \sum_n q_{nk}(x_{nd} - \mu_{kd}) = 0 \\ \mu_{kd} &= \frac{\frac{1}{s^2} m + \frac{1}{\sigma_{kd}^2} \sum_n q_{nk} x_{nd}}{\frac{1}{s^2} + \frac{1}{\sigma_{kd}^2} \sum_n q_{nk}}. \end{aligned}$$

EX 6.6. The log likelihood is given by:

$$\mathcal{L} = \sum_n \log \sum_k \pi_k \frac{\lambda_k^{x_n} \exp\{-\lambda_k\}}{x_n!}$$

Incorporating variational parameteres q_{nk} and applying Jensen's inequality gives:

$$\mathcal{L} \geq \sum_n \sum_k q_{nk} \log \pi_k + \sum_n \sum_k q_{nk} \log \frac{\lambda_k^{x_n} \exp\{-\lambda_k\}}{x_n!} - \sum_n \sum_k q_{nk} \log q_{nk}.$$

We shall first derive an update for π_k. The relevant terms in the bound, incorporating a Lagrangian term to ensure $\sum_k \pi_k = 1$ are:

$$\sum_n \sum_k q_{nk} \log \pi_k - \lambda \left(\sum_k \pi_k - 1 \right).$$

Taking partial derivatives with respect to π_k gives:

$$\frac{\partial}{\partial \pi_k} = \frac{1}{\pi_k} \sum_n q_{nk} - \lambda.$$

Equating to zero:

$$\lambda \pi_k = \sum_n q_{nk}.$$

To compute λ we sum both sides over k:

$$\begin{aligned} \lambda \sum_k \pi_k &= \sum_n \sum_k q_{nk} \\ \lambda &= N. \end{aligned}$$

Therefore:

$$\pi_k = \frac{1}{N} \sum_n q_{nk}.$$

Now q_{nk}. Each term is relevant, and adding the Lagrangian:

$$\sum_n \sum_k q_{nk} \log \pi_k + \sum_n \sum_k q_{nk} \log p(x_n|\lambda_k) - \sum_n \sum_k q_{nk} \log q_{nk} - \lambda \left(\sum_k q_{nk} \right).$$

Taking partial derivatives with respect to q_{nk}:

$$\frac{\partial}{\partial q_{nk}} = \log \pi_k + \log p(x_n|\lambda_k) - (\log q_{nk} + 1) - \lambda.$$

Equating to zero, taking exponetials and grouping all constants into γ:

$$\gamma q_{nk} = \pi_k p(x_n|\lambda_k).$$

To find γ, we sum both sides over k:

$$\begin{aligned} \gamma \sum_k q_{nk} &= \sum_k \pi_k p(x_n|\lambda_k) \\ \gamma &= \sum_k \pi_k p(x_n|\lambda_k). \end{aligned}$$

Therefore:

$$q_{nk} = \frac{\pi_k p(x_n|\lambda_k)}{\sum_j \pi_j p(x_n|\lambda_j)}.$$

Finally, the update for λ_k. Expanding the relevant component of the bound:

$$\sum_n \sum_k q_{nk} x_n \log \lambda_k - \sum_n \sum_k q_{nk} \lambda_k.$$

Taking partial derivatives with respext to λ_k:

$$\frac{\partial}{\partial \lambda_k} = \frac{1}{\lambda_k} \sum_n q_{nk} x_n - \sum_n q_{nk}.$$

Equating to zero and solving for λ_k gives:

$$\lambda_k = \frac{\sum_n q_{nk} x_n}{\sum_n q_{nk}}.$$

Example code to run this mixture is given below (`poissmix.m`):

```
clear all;close all;

% Generate data from 3 poissons
N = 20;
x = poissrnd(10,[N 1]);
x = [x;poissrnd(2,[N,1])];
x = [x;poissrnd(20,[N,1])];
% Fit a two component mixture
K = 3;
% Initialise
```

```
pr = repmat(1/K,1,K);
mu = rand(K,1);
N = size(x,1);
allmu = [];
% Do 100 iterations
for it = 1:100
    % Update q
    temp = poisspdf(repmat(x,1,K),repmat(mu',N,1));
    q = temp.*repmat(pr,N,1);
    q = q./repmat(sum(q,2),1,K);

    % Update pr
    pr = mean(q,1);

    % Update means
    mu = (sum(repmat(x,1,K).*q,1)./sum(q,1))';
    allmu = [allmu;mu'];

    % Plot the evolution of the means, and show the q
    subplot(121);plot(allmu);
    subplot(122);imagesc(q);drawnow
end
```

Chapter 7

EX 7.1. The bound is given by:

$$\log p(Y) \geq \int Q(\theta) \log \frac{p(Y,\theta)}{Q(\theta)} \, d\theta.$$

To show that the bound is maximised if $Q(\theta)$ is the true posterior, we will substitute the posterior $p(\theta|Y)$ for $Q(\theta)$ and re-arrange the right hand side such that the inequality becomes an equality. Firstly, substituting and expanding:

$$\log p(Y) \geq \int p(\theta|Y) \log p(Y,\theta) \, d\theta - \int p(\theta|Y) \log p(\theta|Y) \, d\theta.$$

Next we expand the joint density: $p(Y,\theta) = p(\theta|Y)p(Y)$:

$$\log p(Y) \geq \int p(\theta|Y) \log p(\theta|Y) \, d\theta + \int p(\theta|Y) \log p(Y) \, d\theta - \int p(\theta|Y) \log p(\theta|Y) \, d\theta.$$

The first and third terms cancel. The second term is an expectation of a constant and hence the expectation disappears:

$$\int p(\theta|Y) \log p(Y) \, d\theta = \log p(Y) \int p(\theta|Y) \, d\theta = \log p(Y).$$

EX 7.2. We shall go through each term in the order they are given in Section 7.5.5 starting with $\mathbf{E}_{Q_\tau(\tau)} \{\log p(\tau|a,b)\}$:

$$\begin{aligned} \mathbf{E}_{Q_\tau(\tau)} \{\log p(\tau|a,b)\} &= \mathbf{E}_{Q_\tau(\tau)} \{a \log b - \log \Gamma(a) + (a-1) \log \tau - b\tau\} \\ &= a \log b - \log \Gamma(a) + (a-1) \langle \log \tau \rangle - b \langle \tau \rangle . \end{aligned}$$

Now $\sum_n \mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)} \{\log p(\mathbf{x}_n)\}$. Recall that $p(\mathbf{x}_n) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $Q_{\mathbf{x}_n}(\mathbf{x}_n) = \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_n}, \boldsymbol{\Sigma}_{\mathbf{x}_n})$:

$$\begin{aligned} \sum_n \mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)} \{\log p(\mathbf{x}_n)\} &= \sum_n \mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)} \left\{ -\frac{D}{2} \log 2\pi - \frac{1}{2} (\mathbf{x}_n^\mathsf{T} \mathbf{x}_n) \right\} \\ &= -\frac{ND}{2} \log 2\pi - \frac{1}{2} \sum_n \left(\mathsf{Tr}(\boldsymbol{\Sigma}_{\mathbf{x}_n}) + \boldsymbol{\mu}_{\mathbf{x}_n}^\mathsf{T} \boldsymbol{\mu}_{\mathbf{x}_n} \right). \end{aligned}$$

(Note that $\mathbf{E}_{\mathcal{N}_\mathbf{a}(\mathbf{m},\mathbf{S})} \left\{ \mathbf{a}^\mathsf{T} \mathbf{B} \mathbf{a} \right\} = \mathsf{Tr}(\mathbf{SB}) + \mathbf{m}^\mathsf{T} \mathbf{B} \mathbf{m}$).

The expression for $\sum_m \mathbf{E}_{Q_{\mathbf{w}_m}(\mathbf{w}_m)} \{\log p(\mathbf{w}_m)\}$ is arrived at in exactly the same manner as that for $\sum_n \mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)} \{\log p(\mathbf{x}_n)\}$.

Next, $\sum_n \sum_m \mathbf{E}_{Q(\cdot)} \{\log p(y_{nm}|\mathbf{x}_n, \mathbf{w}_m, \tau)\}$ (where the expectation is over all parameters):

$$\begin{aligned} &= \sum_n \sum_m \mathbf{E}_{Q(\cdot)} \left\{ -\frac{1}{2}\log 2\pi + \frac{1}{2}\log\tau - \frac{1}{2}\tau(y_{nm} - \mathbf{w}_m^\mathsf{T}\mathbf{x}_n)^2 \right\} \\ &= -\frac{NM}{2}\log 2\pi + \frac{NM}{2}\langle\log\tau\rangle - \frac{1}{2}\langle\tau\rangle \sum_n \sum_m \left\langle (y_{nm} - \mathbf{w}_m^\mathsf{T}\mathbf{x}_n)^2 \right\rangle . \end{aligned}$$

Now, we shall look at $\mathbf{E}_{Q_\tau(\tau)} \{\log Q_\tau(\tau)\}$. $Q_\tau(\tau)$ is Gamma with parameters e and f (see Equation 7.12). Therefore it follows exactly the same form as $\mathbf{E}_{Q_\tau(\tau)} \{\log p(\tau|a, b)\}$:

$$\mathbf{E}_{Q_\tau(\tau)} \{\log Q_\tau(\tau)\} = e\log f - \log\Gamma(e) + (e-1)\langle\log\tau\rangle - f\langle\tau\rangle .$$

The final two expressions are derived in exactly the same way and we will only show the one for $\mathbf{x}_n$, $\sum_n \mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)} \{\log Q_{\mathbf{x}_n}(\mathbf{x}_n)\}$. Recall that $Q_{\mathbf{x}_n}(\mathbf{x}_n) = \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_n}, \boldsymbol{\Sigma}_{\mathbf{x}_n})$:

$$\begin{aligned} \sum_n \mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)} \{\log Q_{\mathbf{x}_n}(\mathbf{x}_n)\} &= -\frac{ND}{2}\log 2\pi - \frac{1}{2}\sum_n \log|\boldsymbol{\Sigma}_{\mathbf{x}_n}| - \ldots \\ &\quad \frac{1}{2}\sum_n \mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)} \left\{ (\mathbf{x}_n - \boldsymbol{\mu}_{\mathbf{x}_n})^\mathsf{T} \boldsymbol{\Sigma}_{\mathbf{x}_n}^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_{\mathbf{x}_n}) \right\} . \end{aligned}$$

To progress further, we need the following identity:

$$\mathbf{E}_{\mathcal{N}_\mathbf{a}(\mathbf{b},\mathbf{C})} \left\{ (\mathbf{a}-\mathbf{d})^\mathsf{T}\mathbf{E}(\mathbf{a}-\mathbf{d}) \right\} = (\mathbf{b}-\mathbf{d})^\mathsf{T}\mathbf{E}(\mathbf{b}-\mathbf{d}) + \mathsf{Tr}(\mathbf{EC}).$$

Matching this to our expression, we can see that $\mathbf{b} = \mathbf{d} = \boldsymbol{\mu}_{\mathbf{x}_n}$ and so the first term on the RHS is zero. The second term is $\mathsf{Tr}(\boldsymbol{\Sigma}_{\mathbf{x}_n}^{-1}\boldsymbol{\Sigma}_{\mathbf{x}_n}) = \mathsf{Tr}(\mathbf{I}) = D$ (for D-dimensional $\mathbf{x}_n$. Therefore, our expectation becomes:

$$\sum_n \mathbf{E}_{Q_{\mathbf{x}_n}(\mathbf{x}_n)} \{\log Q_{\mathbf{x}_n}(\mathbf{x}_n)\} = -\frac{ND}{2}\log 2\pi - \frac{ND}{2} - \frac{1}{2}\sum_n \log|\boldsymbol{\Sigma}_{\mathbf{x}_n}|.$$

The expression for $\sum_m \mathbf{E}_{Q_{\mathbf{w}_m}(\mathbf{w}_m)} \{\log Q_{\mathbf{w}_m}(\mathbf{w}_m)\}$ follows in exactly the same way.

EX 7.3. Starting with $Q_{\mathbf{x}_n}(\mathbf{x}_n)$:

$$\begin{aligned} Q_{\mathbf{x}_n}(\mathbf{x}_n) &\propto \exp\left(\mathbf{E}_{Q_\mathbf{W}(\mathbf{W})Q_\tau(\tau)} \{\log p(\mathbf{Y}, \mathbf{W}, \mathbf{X}, \tau|\mathbf{Z})\}\right) \\ &\propto \exp\left(\mathbf{E}_{Q_\mathbf{W}(\mathbf{W})Q_\tau(\tau)} \left\{ \log p(\mathbf{x}_n) + \sum_m z_{nm}\log p(y_{nm}|\mathbf{x}_n, \mathbf{w}_m, \tau) \right\}\right) \\ &\propto \exp\left(-\frac{1}{2}\mathbf{x}_n^\mathsf{T}\mathbf{x}_n - \frac{1}{2}\sum_m z_{nm}\mathbf{E}_{Q_\mathbf{W}(\mathbf{W})Q_\tau(\tau)} \left\{ \tau(y_{nm} - \mathbf{w}_m^\mathsf{T}\mathbf{x}_n)^2 \right\}\right) \\ &\propto \exp\left(-\frac{1}{2}\mathbf{x}_n^\mathsf{T}\mathbf{x}_n - \frac{1}{2}\langle\tau\rangle \sum_m z_{nm}(\mathbf{x}_n^\mathsf{T}\left\langle \mathbf{w}_m\mathbf{w}_m^\mathsf{T} \right\rangle \mathbf{x}_n - 2y_{nm}\mathbf{x}_n^\mathsf{T}\langle\mathbf{w}_m\rangle)\right) . \end{aligned}$$

This has the form of a Gaussian, $Q_{\mathbf{x}_n}(\mathbf{x}_n) = \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}_n}, \boldsymbol{\Sigma}_{\mathbf{x}_n})$. Equating coefficients, we get the following:

$$\begin{aligned}
\boldsymbol{\Sigma}_{\mathbf{x}_n} &= \left(\mathbf{I} + \langle\tau\rangle \sum_m z_{nm} \left\langle \mathbf{w}_m \mathbf{w}_m^{\mathsf{T}} \right\rangle\right)^{-1} \\
\boldsymbol{\mu}_{\mathbf{x}_n} &= \langle\tau\rangle \, \boldsymbol{\Sigma}_{\mathbf{x}_n} \sum_m z_{nm} y_{nm} \left\langle \mathbf{w}_m \right\rangle .
\end{aligned}$$

The expression for $Q_{\mathbf{w}_m}(\mathbf{w}_m)$ follows an almost identical argument so we omit it here. Finally, the expression for $Q_\tau(\tau)$:

$$\begin{aligned}
Q_\tau(\tau) &\propto \exp\left(\mathbf{E}_{Q_{\mathbf{W}}(\mathbf{W})Q_{\mathbf{X}}(\mathbf{X})} \left\{\log p(\mathbf{Y}, \mathbf{W}, \mathbf{X}, \tau|\mathbf{Z})\right\}\right) \\
&\propto \exp\left(\mathbf{E}_{Q_{\mathbf{W}}(\mathbf{W})Q_{\mathbf{X}}(\mathbf{X})} \left\{\log\Gamma(a,b) + \sum_{n,m} z_{nm} \log p(y_{nm}|\mathbf{x}_n, \mathbf{w}_m, \tau)\right\}\right) \\
&\propto \exp\left(-b\tau + (a-1)\log\tau - \frac{\sum_{n,m} z_{nm}}{2}\log\tau - \frac{1}{2}\tau \sum_{n,m} z_{nm} \mathbf{E}_{Q_{\mathbf{W}}(\mathbf{W})Q_{\mathbf{X}}(\mathbf{X})} \left\{(y_{nm} - \mathbf{w}_m^{\mathsf{T}}\mathbf{x}_n)^2)\right\}\right) \\
&\propto \exp\left(-b\tau + (a-1)\log\tau - \frac{\sum_{n,m} z_{nm}}{2}\log\tau - \frac{1}{2}\tau \sum_{n,m} z_{nm} \left\langle (y_{nm} - \mathbf{w}_m^{\mathsf{T}}\mathbf{x}_n)^2 \right\rangle\right) .
\end{aligned}$$

This has the form of another Gamma distribution, with parameters:

$$\begin{aligned}
e &= a + \frac{1}{2}\sum_{n,m} z_{nm} \\
f &= b + \frac{1}{2}\sum_{n,m} z_{nm} \left\langle (y_{nm} - \mathbf{w}_m^{\mathsf{T}}\mathbf{x}_n)^2 \right\rangle .
\end{aligned}$$

Chapter 8

EX 8.1. The following code would do it:

```
x = [0:0.01:1];
gam = 10.0;
N = length(x);
K = zeros(N);
% Create the kernel matrix
for n = 1:N
    for m = 1:N
        K(n,m) = exp(-gam*(x(n)-x(m))^2);
    end
end
% Add a small constant to the diagonal for numerical stability
K = K + 1e-6*eye(N);
% Generate 10 samples
f_samps = gausssamp(repmat(0,N,1),K,10)
plot(x,f_samps)
```

EX 8.2. Use the code above, adding a linear term to K.

EX 8.3. The following code gives a function to compute the marginal likelihood. Note that we optimise the log of the parameters to avoid having to use a constrained optimisation:

```
function ML = marg_like(hyp,y,x)
% Function to compute the marginal likelihood
% hyp contains log of the hyperparameters
N = length(x);
% Compute covariance function
CSS = exp(-exp(hyp(1))*(repmat(x,1,N) - repmat(x',N,1)).^2);
CSS = CSS + exp(hyp(2))*eye(N);
% Compute marginal likelihood
ML = -(N/2)*log(2*pi) - 0.5*log(det(CSS)) - 0.5*y'*inv(CSS)*y;
% Return negative as optimiser minimises
ML = -ML;
```

and the following code calls Matlab's `fminunc` function for some generated data:

```
x = [0:0.05:1];
```

```
gam = 10.0;
N = length(x);
K = zeros(N);

% Create the kernel matrix
for n = 1:N
    for m = 1:N
        K(n,m) = exp(-gam*(x(n)-x(m))^2);
    end
end

K = K + 1e-6*eye(N);
% Generate a sample
f_samps = gausssamp(repmat(0,N,1),K,1)

% Add some noise
ss = 0.1;
true_f = f_samps(1,:) + randn(1,N)*sqrt(ss);


true_f = true_f';
x = x';

% Initial hyperparameters
inithyp = [0;0]

% Call optimisation code
hyp = fminunc(@(h) marg_like(h,true_f,x),inithyp)
```

EX 8.4. The function value at the nth data point if f_n. This function is then exponentiated to give the rate of a Poisson $\lambda_n = \exp(f_n)$. The integer count sampled from the Poisson is z_n. The gradient of the log posterior is given by:

$$\frac{\delta}{\delta \mathbf{f}} = \mathbf{z} - \exp(\mathbf{f}) - \mathbf{C}^{-1}\mathbf{f}$$

$$\frac{\delta^2}{\delta \mathbf{f} \delta \mathbf{f}^T} = -\mathbf{C}^{-1} - \mathbf{E}$$

where $\mathbf{E}$ is a matrix with zeros everywhere except the diagonal where the nth diagonal value is $\exp(f_n)$. This is demonstrated with the following code:

```
% Generate some data
x = [0:0.01:1];
gam = 10.0;
N = length(x);
C = zeros(N);

% Create the kernel matrix
for n = 1:N
    for m = 1:N
        C(n,m) = exp(-gam*(x(n)-x(m))^2);
    end
end

C = C + 1e-6*eye(N);
```

```
15          % Generate a sample
16          f_samps = gausssamp(repmat(0,N,1),C,1)
17
18          % Plot the function
19          figure(1);
20          hold off
21          plot(x,f_samps);
22
23          % Generate the counts
24          rate = exp(f_samps);
25          n = poissrnd(rate)';
26          hold on
27          plot(x,n,'ro')
28
29          % Use Newton-Raphson to optimise f
30          f = zeros(N,1);
31          iK = inv(C);
32          oldf = f;
33          for it = 1:100
34              gr = n - exp(f) - iK*f;
35              he = -iK - diag(exp(f));
36              f = f - inv(he)*gr;
37              ch = sum((f - oldf).^2);
38              oldf = f;
39              if ch < 1e-6
40                  break
41              end
42          end
43
44          plot(x,f,'g')
```

EX 8.5. The following code generates some data, uses M-H to sample from the posterior and uses the posterior samples to compute the predictions on a grid, in order to show the predictive contours.

```
1  % GP classification using M-H
2  x = [randn(20,2)-repmat(4,20,2);randn(20,2)+repmat(4,20,2)];
3  t = [repmat(0,20,1);repmat(1,20,1)];
4
5  N = size(x,1);
6
7  % Compute the covariance matrix
8  gam = 1;alp = 5;
9  C = zeros(N);
10 for n = 1:N
11     for m = 1:N
12         C(n,m) = alp*exp(-gam*(sum((x(n,:)-x(m,:)).^2)));
13     end
14 end
15
16 iC = inv(C);
17
18 nSamps = 10000;
19 pos0 = find(t==0);
20 pos1 = find(t==1);
21
22 % Initialise with f being -3 for the -ve class and +3 for the positive
```

```
%\ Could use anything, but this gets us to a good place a bit faster!
f = [repmat(-3,20,1);repmat(3,20,1)];
oldProb = -0.5*f'*iC*f - sum(t.*log(1+exp(-f)));
oldProb = oldProb + sum((1-t).*(-f - log(1+exp(-f))));

% Make a grid of prediction points and compute the
% test covariance matrix
[X,Y] = meshgrid(min(x(:,1))-0.5:0.1:max(x(:,1)),...
    min(x(:,2))-0.5:0.1:max(x(:,2)));

testData = [X(:),Y(:)];
n_test = size(testData,1);
testC = zeros(n_test,size(x,1));
for n = 1:n_test
    for m = 1:size(x,1)
        testC(n,m) = alp*exp(-gam*(sum((testData(n,:)-x(m,:)).^2)));
    end
end

% This term is used for predictions and doesn't change so we can just
% compute it once
testCovTerm = diag(testC*iC*testC');

% Run the sampler
testProbs = zeros(n_test,1);
allf = zeros(N,nSamps/100);
for s = 1:nSamps
    % Propose a new sample for each observation
    % Note that we could do this for all samples at once, but
    % it becomes very hard to get a sample accepted
    order = randperm(N);
    for n = 1:N
        newf = f;
        newf(order(n)) = f(order(n)) + randn(1,1)*0.1;
        newProb = -0.5*newf'*iC*newf - sum(t.*log(1+exp(-newf)));
        newProb = newProb + sum((1-t).*(-newf - log(1+exp(-newf))));
        u = rand;
        if u <= exp(newProb - oldProb)
            f = newf;
            oldProb = newProb;
        end
    end

    % Do the predictions
    testVar = alp - testCovTerm;
    testMu = testC*iC*f;

    f_samp = randn(n_test,1).*sqrt(testVar) + testMu;
    testProbs = testProbs + 1./(1+exp(-f_samp));

end


% Plot the output
figure(4)
hold off
pos = find(t==0);
plot(x(pos,1),x(pos,2),'ko');
pos = find(t==1);
```

```
82 hold on
83 plot(x(pos,1),x(pos,2),'ko','markerfacecolor','k')
84 testContour = reshape(testProbs./nSamps,size(X));
85 contour(X,Y,testContour,5);
```

Chapter 9

EX 9.1. The following code does both methods of sampling and then plots the autocorrelation:

```
% Generate some points data in 1-D
x = [0:0.1:1];
alp = 5;gam = 1;

N = length(x);
C = zeros(N);
for n = 1:N
    for m = 1:N
        C(n,m) = alp*exp(-gam*(x(n)-x(m))^2);
    end
end

C = C + 1e-6*eye(N);

nSamps = 10000;


% Sample 1000 values from the Gaussian
f_full = gausssamp(zeros(N,1),C,nSamps);

% Samples using Gibbs sampling
f = zeros(N,1);

all_f = zeros(nSamps,N);

for s = 1:nSamps
    for n = 1:N
        % Compute the conditional mean and covariance
        subC = C;
        subC(n,:) = [];
        subC(:,n) = [];
        subc = C(n,:);
        subc(n) = [];
        subf = f;
        subf(n) = [];
        % Note that the mean is zero so these
        % expressions are a bit simpler than in
        % the book

```

```
        % Also - the following expression is
        % much more accurate than subc*inv(subC)*subc'
        co = C(n,n) - subc/subC*subc';
        mu = subc*inv(subC)*subf;
        f(n) = randn.*sqrt(co) + mu;
    end
    all_f(s,:) = f';
end


% Compute the autocorrelation of the nth point
n = 5;

k_vals = [0:1:200];
ac = zeros(length(k_vals),2);
N = nSamps;

full_vals = f_full(:,n);
full_vals = full_vals - mean(full_vals);
all_vals = all_f(:,n);
all_vals = all_vals - mean(all_vals);

ss = [var(full_vals), var(all_vals)];


for k = 1:length(k_vals)
    ac(k,1) = sum(full_vals(1:end-k_vals(k)).*full_vals(1+k_vals(k):end))
    ac(k,1) = ac(k,1)/(ss(1)*(N-k_vals(k)));

    ac(k,2) = sum(all_vals(1:end-k_vals(k)).*all_vals(1+k_vals(k):end))
    ac(k,2) = ac(k,2)/(ss(2)*(N-k_vals(k)));
end

figure(1)
plot(ac)
xlabel('Lag');
ylabel('Autocorrelation');
```

EX 9.2. See the solution to Exercise 8.5. The only difference is changing the sigmoid likelihood to the probit likelihood.

EX 9.3. Code available on accompanying website.

EX 9.4. Code available on accompanying website.

EX 9.5. The following code will sample a regression dataset and then obtain an approximate posterior via ABC:

```
% ABC for GP regression
x = [0:0.1:1];

alp = 5;
gam = 1;

N = length(x);
C = zeros(N);

```

```
for n = 1:N
    for m = 1:N
        C(n,m) = alp*exp(-gam*(x(n)-x(m))^2);
    end
end

C = C + 1e-6*eye(N);
noise_var = 0.1;

t = gausssamp(repmat(0,N,1),C,1)' + randn(N,1).*sqrt(noise_var);

% ABC

nSamps = 5000;
accepted_samps = [];

err_thresh = 5.0;
figure(1);hold off
plot(x,t,'k','linewidth',2);hold on
for s = 1:nSamps
    % Propose a value
    proposal = gausssamp(repmat(0,N,1),C,1)' + randn(N,1).*sqrt(noise_var);
    err = sum((proposal - t).^2);

    if err < err_thresh
        accepted_samps = [accepted_samps proposal];
        plot(x,proposal,'k','color',[0.6 0.6 0.6]);drawnow
    end

end

plot(x,t,'k','linewidth',2);

%% Compare with the true posterior

postCov = inv((eye(n)*(1/noise_var) + inv(C)));
postMu = (1/noise_var) * postCov * t;

% Generate samples
samp_f = gausssamp(postMu,postCov,1000);
figure(2);hold off
plot(x,samp_f','k','color',[0.6 0.6 0.6]);
hold on
plot(x,t,'k','linewidth',2);
```

Chapter 10

EX 10.1. We have N observations, $x_1, \ldots, x_N$. Our model is defined as a set of K Gaussian components, with means μ_k and unit variances ($\sigma_k^2 = 1$). At any point within the Gibbs sampling procedure, each of the N observations is assigned to one of these components ($z_{nk} = 1$, $z_{nj} = 0\ \forall j \neq k$). Each mean value has a Gaussian prior density with its own mean and variance μ_0, σ_0^2. The conditional density for μ_k can therefore be computed as:

$$\begin{aligned}
p(\mu_k|\ldots) &\propto p(\mu_k|\mu_0,\sigma_0^2)\prod_{n=1}^{N}[p(x_n|\mu_k)]^{z_{nk}} \\
&\propto \exp\left\{-\frac{1}{2\sigma_0^2}(\mu_k-\mu_0)^2\right\}\exp\left\{-\frac{1}{2}\sum_{n=1}^{N}z_{nk}(x_n-\mu_k)^2\right\} \\
&= \mathcal{N}(a,b^2) \\
\frac{\mu_k^2}{b^2} &= \frac{\mu_k^2}{\sigma_0^2}+\mu_k^2\sum_n z_{nk} \\
b^2 &= \left(\sigma_0^{-2}+\sum_n z_{nk}\right)^{-1} \\
\frac{a\mu_k}{b^2} &= \frac{\mu_k\mu_0}{\sigma_0^2}+\mu_k\sum_{n=1}^{N}z_{nk}x_n \\
a &= b^2\left(\frac{\mu_0}{\sigma_0^2}+\sum_{n=1}^{N}z_{nk}x_n\right)
\end{aligned}$$

EX 10.2. The following code performs Gibbs sampling for the one-dimensional Gaussian mixture with fixed known variance:

```
% 1D Gaussian mixture
x = [randn(20,1);randn(20,1)+3];
x = sort(x);

N = length(x);

% Define the prior parameters
```

```
mu0 = 0;
ss0 = 1;

% Component variance
ss = 1;

% Number of components
K = 2;


nSamps = 1000;

% Parameters for the prior Dirichlet
alp = repmat(1,K,1);
prk = gamrnd(alp,1);
prk = prk./sum(prk);

Z = zeros(N,K);

% Structures to hold the output
allZZ = zeros(N);
allMu = zeros(nSamps,K);
allPrk = zeros(nSamps,K);

% Outer loop to run multiple chains for computing Rhat
nChain = 5;
samps = zeros(nSamps,nChain);
for chain = 1:nChain
    % Randomly initialise the means
    mu = randn(K,1).*sqrt(ss0) + mu0;


    for s = 1:nSamps
        % Update the Z
        for n = 1:N
            like = -0.5*(x(n) - mu).^2;
            like = like + log(prk);
            prob = exp(like - max(like));
            prob = prob./sum(prob);
            pos = find(rand<=cumsum(prob),1);
            Z(n,:) = 0;
            Z(n,pos) = 1;
        end

        % Update the mu

        for k = 1:K
            ssk = 1/(1/ss0 + sum(Z(:,k)));
            muk = ssk * (mu0/ss0 + sum(Z(:,k).*x));
            mu(k) = randn*sqrt(ssk) + muk;
        end

        % Update the prior

        prk = gamrnd(alp + sum(Z,1)',1);
        prk = prk./sum(prk);

        allPrk(s,:) = prk';
        allMu(s,:) = mu';
```

```
67          allZZ = allZZ + Z*Z';
68      end
69
70      figure(1);
71      plot(allMu);
72      figure(2);
73      plot(allPrk);
74
75
76      allZZ = allZZ./nSamps;
77      figure(3);
78      imagesc(allZZ);
79
80      % Compute the autocorrelation of the kth mu
81      lag_vals = [0:1:200];
82      ac = zeros(length(lag_vals),1);
83
84      k = 1;
85
86      full_vals = allMu(:,k);
87      full_vals = full_vals - mean(full_vals);
88
89      samp_ss = var(full_vals);
90
91
92      for l = 1:length(lag_vals)
93          ac(l) = sum(full_vals(1:end-lag_vals(l)).*full_vals(1+lag_vals(l):end))
94          ac(l) = ac(l)/(samp_ss*(nSamps-lag_vals(l)));
95      end
96
97      figure(4);
98      plot(lag_vals,ac,'ro');drawnow
99
100     % Store the values of the lower mean (they switch in different
101     % instances)
102     m = mean(allMu);
103     pos = find(m==min(m));
104     samps(:,chain) = allMu(:,pos);
105 end
106
107 % Compute Rhat
108 muc = zeros(nSamps,nChain);
109 vc = zeros(nSamps,nChain);
110 for s = 1:nSamps
111     muc(s,:) = mean(samps(1:s,:),1);
112     vc(s,:) = var(samps(1:s,:),1);
113 end
114 W = mean(vc,2);
115 count = [1:nSamps]';
116 B = (count./(nChain-1)).*sum((muc - repmat(mean(muc,2),1,nChain)).^2,2);
117 V = ((count-1)./count).*W + (1./count).*B;
118 Rhat = sqrt(V./W);
119
120 figure(5);
121 plot(count,Rhat);
```

EX 10.3. We observe N binary vectors, each of which has D dimensions. We model them as a mixture model with K components, each of which is parameterised by a a vector

of D probabilities, $\mathbf{p}_k = [p_{k1}, \ldots, p_{kD}]^T$. For each component of these vectors we assume a Beta prior with parameters α and β:

$$p(p_{kd}|\alpha, \beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} p_{kd}^{\alpha-1}(1-p_{kd})^{\beta-1}.$$

The vector of prior probabilities for each component $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_K]^T$ is given a uniform Dirichlet prior with parameter γ. The likelihood of an observation $\mathbf{x}_n = [x_{n1}, \ldots, x_{nD}]^T$ being in component k is:

$$P(\mathbf{x}_n|\mathbf{p}_k) = \prod_{d=1}^{D} p_{kd}^{x_{nd}}(1-p_{kd})^{1-x_{nd}}.$$

The posterior over all parameters is given by:

$$p(\mathbf{Z}, \boldsymbol{\pi}, \mathbf{p}_1, \ldots, \mathbf{p}_K|\mathbf{X}, \alpha, \beta, \gamma) \propto p(\boldsymbol{\pi}|\gamma) \left[\prod_{k=1}^{K} p(\mathbf{p}_k|\alpha, \beta)\right] \left[\prod_{n=1}^{N}\sum_{k=1}^{K} (\pi_k p(\mathbf{x}_n|\mathbf{p}_k))^{z_{nk}}\right].$$

To create a Gibbs sampler we require the conditional distributions for z_{nk}, p_{kd} and π_k. The conditional density for $\boldsymbol{\pi}_k$ is the same as in the Gaussian case – a Dirichlet with parameters $\gamma + \sum_{n=1}^{N} z_{nk}$:

$$p(\boldsymbol{\pi}|\ldots) = Dir\left(\gamma + \sum_{n=1}^{N} z_{n1}, \ldots, \gamma + \sum_{n=1}^{N} z_{nK}\right).$$

The conditional distribution for z_{nk} is:

$$P(z_{nk} = 1|\ldots) \propto \pi_k p(\mathbf{x}_n|\mathbf{p}_k),$$

where the normalisation constant will be the sum of these terms over $j = 1 \ldots K$. Finally, the conditional distribution for p_{kd} is derived as follows:

$$\begin{aligned} p(p_{kd}|\ldots) &\propto p_{kd}^{\alpha-1}(1-p_{kd})^{\beta-1} \prod_{n=1}^{N} \left(p_{kd}^{x_{nd}}(1-p_{kd})^{1-x_{nd}}\right)^{z_{nk}} \\ &\propto p_{kd}^{\alpha-1}(1-p_{kd})^{\beta-1} p_{kd}^{\sum_n z_{nk}x_{nd}}(1-p_{kd})^{\sum_n z_{nk}(1-x_{nd})} \\ &\propto p_{kd}^{\alpha-1+\sum_n z_{nk}x_{nd}}(1-p_{kd})^{\beta-1+\sum_n z_{nk}(1-x_{nd})} \end{aligned}$$

which is a Beta density with parameters $\alpha^* = \alpha + \sum_n z_{nk}x_{nd}$ and $\beta^* = \beta + \sum_n z_{nk}(1-x_{nd})$.

EX 10.4. The conditional density of the mean parameter given in the book is:

$$p(\boldsymbol{\mu}_k|\ldots) = \mathcal{N}(\mathbf{a}, \mathbf{B})$$

where:

$$\mathbf{B} = \left[\boldsymbol{\Sigma}_0^{-1} + \left(\sum_n z_{nk}\mathbf{I}\right)\right]^{-1},$$

and

$$\mathbf{a} = \mathbf{B}\left[\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0 + \sum_n z_{nk}\mathbf{x}_n\right].$$

We are interested in computing the conditional probability of $z_{nk} = 1$ with the mean parameter $\boldsymbol{\mu}_k$ collapsed. The posterior over $\boldsymbol{\mu}_k$ not including the nth observation is identical to that given above, with the summations omitting the nth component. From now on, assume that $\mathbf{a}$ and $\mathbf{B}$ do not include the contribution from the nth observation. The likelihood is given by:

$$p(\mathbf{x}|\boldsymbol{\mu}_k) = \mathcal{N}(\boldsymbol{\mu}_k, \mathbf{I}).$$

We therefore need to compute the following:

$$p(\mathbf{x}|\ldots) = \int \mathcal{N}(\boldsymbol{\mu}_k, \mathbf{I}) p(\boldsymbol{\mu}_k|\mathbf{a}, \mathbf{B})\ d\boldsymbol{\mu}_k.$$

As both components are Gaussian, this is another Gaussian:

$$p(\mathbf{x}|\ldots) = \mathcal{N}(\mathbf{a}, \mathbf{B} + \mathbf{I}).$$

Therefore, the conditional probability for $z_{nk} = 1$ is proportional to this term multiplied by the term derived in the book from collapsing $\boldsymbol{\pi}$. It is normalised by dividing by the same term summed over $j = 1 \ldots K$.

EX 10.5. The following code will perform collapsed Gibbs sampling for a Gaussian mixture with 1-dimensional data.

```
% 1D Gaussian mixture with collapsed Gibbs sampling
x = [randn(20,1);randn(20,1)+3];
x = sort(x);

N = length(x);

% Define the prior parameters
mu0 = 0;
ss0 = 1;

% Component variance
ss = 1;

% Number of components
K = 2;


nSamps = 1000;

% Parameters for the prior Dirichlet
alp = repmat(1,1,K);
prk = gamrnd(alp,1);
prk = prk./sum(prk);

% Randomly initialise Z
Z = rand(N,K);
```

```
Z = (Z == repmat(max(Z,[],2),1,K));

% Structures to hold the output
allZZ = zeros(N);
allMu = zeros(nSamps,K);
allPrk = zeros(nSamps,K);

% Outer loop to run multiple chains for computing Rhat
nChain = 5;
samps = zeros(nSamps,nChain);
for chain = 1:nChain


    for s = 1:nSamps
        % Update the Z
        for n = 1:N
            Z(n,:) = 0;
            sumZ = sum(Z,1);
            sumXZ = sum(Z.*repmat(x,1,K),1);
            postvar = (1./ss0 + sumZ).^(-1);
            postmean = postvar.*(mu0/ss0 + sumXZ);
            predvar = postvar + ss;

            like = -(1./(2*postvar)).*(x(n) - postmean).^2;
            like = like - 0.5*log(2*pi*postvar);

            % Add the collapsed prior term
            like = like + log(sumZ + alp);

            prob = exp(like - max(like));
            prob = prob./sum(prob);
            pos = find(rand<=cumsum(prob),1);
            Z(n,pos) = 1;
        end

        % Sample the mu (even though we don't need it)

        for k = 1:K
            ssk = 1/(1/ss0 + sum(Z(:,k)));
            muk = ssk * (mu0/ss0 + sum(Z(:,k).*x));
            mu(k) = randn*sqrt(ssk) + muk;
        end


        allMu(s,:) = mu';
        Z = 1.0*Z;
        allZZ = allZZ + Z*Z';
    end

    figure(1);
    plot(allMu);
    figure(2);
    plot(allPrk);


    allZZ = allZZ./nSamps;
    figure(3);
    imagesc(allZZ);

```

```
86      % Compute the autocorrelation of the kth mu
87      lag_vals = [0:1:200];
88      ac = zeros(length(lag_vals),1);
89
90      k = 1;
91
92      full_vals = allMu(:,k);
93      full_vals = full_vals - mean(full_vals);
94
95      samp_ss = var(full_vals);
96
97
98      for l = 1:length(lag_vals)
99          ac(l) = sum(full_vals(1:end-lag_vals(l)).*full_vals(1+lag_vals(l):end))
100         ac(l) = ac(l)/(samp_ss*(nSamps-lag_vals(l)));
101     end
102
103     figure(4);
104     plot(lag_vals,ac,'ro');drawnow
105
106     % Store the values of the lower mean (they switch in different
107     % instances)
108     m = mean(allMu);
109     pos = find(m==min(m));
110     samps(:,chain) = allMu(:,pos);
111 end
112
113 % Compute Rhat
114 muc = zeros(nSamps,nChain);
115 vc = zeros(nSamps,nChain);
116 for s = 1:nSamps
117     muc(s,:) = mean(samps(1:s,:),1);
118     vc(s,:) = var(samps(1:s,:),1);
119 end
120 W = mean(vc,2);
121 count = [1:nSamps]';
122 B = (count./(nChain-1)).*sum((muc - repmat(mean(muc,2),1,nChain)).^2,2);
123 V = ((count-1)./count).*W + (1./count).*B;
124 Rhat = sqrt(V./W);
125
126 figure(5);
127 plot(count,Rhat);
```

EX 10.6. The following code samples from a CRP. Vary alpha to see the effect to the histogram:

```
1  % Sample from a CRP
2  N = 100; % 100 customers
3
4  Z = ones(N,1); % Initialise with all customers at one table
5
6  nSamps = 1000;
7
8  alp = 0.1; % Concentration parameter to vary
9  nK = zeros(nSamps,1);
10
11 for s = 1:nSamps
```

```
12      for n = 1:N
13          thistable = find(Z(n,:));
14          Z(n,:) = 0;
15          sumZ = sum(Z,1);
16          if sumZ(thistable) == 0
17              sumZ(thistable) = [];
18              Z(:,thistable) = [];
19          end
20          probs = [sumZ alp];
21          probs = probs./sum(probs);
22          pos = find(rand<=cumsum(probs),1);
23          Z(n,pos) = 1;
24      end
25      nK(s) = size(Z,2);
26  end
27
28  hist(nK,unique(nK));
```

EX 10.7. It will be most efficient to use a collapsed Gibbs sampling scheme for this by marginalising the probabilities p_{kd}. As shown above, the conditional posterior density over these parameters is a Beta distribution with parameters:

$$\alpha^* = \alpha + \sum_n z_{nk} x_{nd}, \quad \beta^* = \beta + \sum_n z_{nk}(1 - x_{nd}).$$

(remember that when considering the collapsed update for the nth observation we would omit the nth data point from the summation). The update for z_{nk} is:

$$P(z_{nk} = 1|\ldots) \propto (\sum_{m \neq n} z_{nk}) \prod_{d=1}^{D} \int P(x_{nd}|z_{nk} = 1, p_{kd}) p(p_{kd}|\ldots)\, dp_{kd}.$$

Because of the conjugacy of the Beta prior and the binomial likelihood, we can analytically evaluate the integral:

$$\begin{aligned}
&= \int p_{kd}^{x_{nd}} (1 - p_{kd})^{1 - x_{nd}} \frac{\Gamma(\alpha^* + \beta^*)}{\Gamma(\alpha^*)\Gamma(\beta^*)} p_{kd}^{\alpha^* - 1} (1 - p_{kd})^{\beta^* - 1}\, dp_{kd} \\
&= \frac{\Gamma(\alpha^* + \beta^*)}{\Gamma(\alpha^*)\Gamma(\beta^*)} \int p_{kd}^{\alpha^* + x_{nd} - 1} (1 - p_{kd})^{\beta^* + (1 - x_{nd}) - 1}\, dp_{kd} \\
&= \frac{\Gamma(\alpha^* + \beta^*)}{\Gamma(\alpha^*)\Gamma(\beta^*)} \frac{\Gamma(\alpha^* + x_{nd})\Gamma(\beta^* + (1 - x_{nd}))}{\Gamma(\alpha^* + \beta^* + 1)} \\
&= \frac{(\alpha^*)^{x_{nd}} (\beta^*)^{1 - x_{nd}}}{\alpha^* + \beta^*}.
\end{aligned}$$

Where in the final step we have used the identity $\Gamma(z+1) = z\Gamma(z)$. This completes the definition of the sampler. The update for z_{nk} is proportional to the product of these terms over the dimensions (D) multiplied by the number of objects in the cluster k. The probability of a new cluster is proportional to the concentration parameter multiplied by the product of these terms where $\alpha^* = \alpha$ and $\beta^* = \beta$.

EX 10.8. The following code samples from a DP with a Gaussian base distribution:

```
% DP sample
basemu = 0;
basess = 1;
alp = 10; % Concentration parameter

N = 100; % Number of samples to draw

samples = [];

for n = 1:N
    if n==1
        samples = randn.*sqrt(basess) + basemu;
    else
        if rand < alp/(alp + n-1)
            % Sample a new value
            samples(end+1) = randn.*sqrt(basess) + basemu;
        else
            % Copy a previous one
            samples(end+1) = samples(randi(length(samples)));
        end
    end
end
```

EX 10.9. See script on accompanying webpage.

EX 10.10. We will derive the collapsed sampler. Assume that we observe $m = 1 \ldots M$ datasets, each of which has $n = 1 \ldots N_m$ observations, x_n^m. There are $k = 1 \ldots K$ top-level components and the top-level concentration parameter is α. The concentration parameter in the mth file is γ_m. In the mth file there are $v = 1 \ldots V_m$ components, each associated with one of the top-level components. Let $z_{nv}^m = 1$ if the nth observation in the mth file is associated with the vth component in that file. Let $w_{vk}^m = 1$ if the vth component in the mth file is associated with the kth top-level component.

For the collapsed sampler, we require the distributions for re-sampling z_{nv}^m. When we re-sample this assignment, there are three possibilities. 1: We put it into a current component in file m. 2: We put it into a new component that uses a pre-existing top-level component. 3: We put it into a new component that has a new top-level component. We will consider each of these in turn. Firstly, though we need the posterior density over μ_k (the mean of the kth top-level component). This is Gaussian and given by:

$$
\begin{aligned}
p(\mu_k|\ldots) &\propto \mathcal{N}(\mu_0, \sigma_0^2) \prod_{m=1}^{M} \prod_{n=1}^{N_m} \prod_{v=1}^{V_m} \mathcal{N}(x_n^m|\mu_k, \sigma^2)^{z_{nv}^m w_{vk}^m} \qquad (10.1)\\
&= \mathcal{N}(a, b^2)\\
b^2 &= \left[\frac{1}{\sigma_0^2 + \frac{1}{\sigma^2}\sum_{m,n,v} z_{nv}^m w_{vk}^m}\right]^{-1}\\
a &= b^2\left[\frac{\sum_{m,n,v} z_{nv}^m w_{vk}^m x_n^m}{\sigma^2} + \frac{\mu_0}{\sigma_0^2}\right]
\end{aligned}
$$

The predictive density is therefore (where summations will not include the data point in question):

$$p(x_n^m|z_{nv}^m = 1, w_{vk}^m = 1) = \mathcal{N}(a, b^2 + \sigma^2) = L(x_n^m|k)$$

Note that the expression for a brand new top-level component is identical but all the summations will equate to zero.

The probability of placing the object in component v is therefore proportional to:

$$P(z_{nv}^m = 1|w_{vk}^m = 1, \ldots) \propto L(x_n^m|k) \left(\sum_n z_{nv}^m \right)$$

To compute the probability of placing the point in a new component in file m we have to marginalise over the top level components (admitting the possibility of a new one). Let s_k be the number of components across all files that are assigned to top-level component k:

$$P(z_{nv^*}^m = 1|\ldots) \propto \gamma_m \left(\frac{\alpha L(x_n^m|k^*) + \sum_l s_l L(x_n^m|l)}{\alpha + \sum_j s_j} \right)$$

We can therefore use these two probabilities to decide if the nth object ought to go into one of the current components in the mth file, or a new component. If we choose a new component, we then need to choose which top-level component it ought to be associated with (or a new one). The probability of it being a current one is proportional to $s_l L(x_n^m|l)$ and the probability of a new one is proportional to $\alpha L(x_n^m|l^*)$.

This completes the sampler, although it might not be particularly efficient. To improve its efficiency, one could re-sample the w_{vk}^m for each file-level component. It is fairly straightforward to compute the required distribution but, be careful: if, as in this case, the mean parameters have all been marginalised, the joint density of all of the objects assigned to this component cannot be assumed to be independent in the standard way (they are all dependent via the marginalisation of the mean parameters). This can be overcome by, for example, computing $p(x_1, x_2, x_3, \ldots|\ldots) = p(x_1|\ldots)p(x_2|x_1, \ldots)p(x_3|x_1, x_2, \ldots)$ etc.

EX 10.11. The equation we are trying to obtain is the probability that the ith word in the nth document comes from topic k. This will be the product of two components: the probability of the kth topic in document n multiplied by the probability of word w (assuming that the ith word is word w) in topic k. Based on the definitions in the text, in the un-collapsed world, this would be given by:

$$P(z_{ni} = k|x_{ni} = w, \beta_{kw}, \theta_{nk}) = \theta_{nk}\beta_{kw}.$$

To obtain the collapsed version, we need to marginalise the parameters $\boldsymbol{\theta}_n$ and $\boldsymbol{\beta}_k$. Both are multinomials and have Dirichlet priors with parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\gamma}$ respectively. We know from our work in mixture models that if we have one sample from a multinomial (say the kth value has been sampled) whose parameters are Dirichlet distributed, the probability of that sample is equal to the kth Dirichlet

parameter divided by the sum of the parameters. We also know that the posterior Dirichlet for a Dirichlet prior and multinomial likelihood has parameters equal to the prior parameters plus the observation counts. Therefore, defining c_{nk}^{-i} as the count of the number of words (excluding the ith one) in the nth document currently assigned to topic k, we know that the term associated with the marginalisation of $\boldsymbol{\theta}_n$ is:

$$\frac{c_{nk}^{-i} + \alpha_k}{\sum_j c_{nj}^{-i} + \alpha_j}$$

and the term associated with the marginalisation of $\boldsymbol{\beta}_k$ is:

$$\frac{v_{kw} + \gamma_w}{\sum_{w'} v_{kw'} + \gamma_{w'}}$$

Multiplying the two together gives the update given in the text.