

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 decrease in winning time over the years.

EX 1.2. The following would do the job:

```
1      % Attributes are stored in Nx1 vector x
2      % Targets are stored in Nx1 vector t
3      xb = mean(x);
4      tb = mean(t);
5      x2b = mean(x.*x);
6      xtb = mean(x.*t);
7      w1 = (xtb - xt*xb) / (x2b-xb^2);
8      w0 = tb-w1*xb;
9      % Plot the data
10     plot(x,t,'b.','markersize',25);
11     % Plot the model
12     hold on;
13     plot(x,w0+w1*x,'r','linewidth',2);
```

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

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

Multiplying this by \mathbf{w} gives:

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

Finally, pre-multiplying this by \mathbf{w}^\top gives:

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

as required.

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

$$\mathbf{X}\mathbf{w} = \begin{bmatrix} w_0 x_{11} + w_1 x_{12} \\ w_0 x_{21} + w_1 x_{22} \\ \vdots \\ w_0 x_{N1} + w_1 x_{N2} \end{bmatrix}$$

Therefore

$$(\mathbf{X}\mathbf{w})^\top = [w_0 x_{11} + w_1 x_{12}, w_0 x_{21} + w_1 x_{22}, \dots, w_0 x_{N1} + w_1 x_{N2}]$$

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

$$\mathbf{w}^\top \mathbf{X}^\top = [w_0 x_{11} + w_1 x_{12}, w_0 x_{21} + w_1 x_{22}, \dots, w_0 x_{N1} + w_1 x_{N2}]$$

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} ,

$$\mathbf{X}^\top = \begin{bmatrix} x_{11}, x_{21}, \dots, x_{N1} \\ x_{12}, x_{22}, \dots, x_{N2} \end{bmatrix}$$

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

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

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

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

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

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

Multiplying this by \mathbf{w} gives:

$$\mathbf{x}_n \mathbf{x}_n^T \mathbf{w} = \begin{bmatrix} w_0 x_{n1}^2 + w_1 x_{n1} x_{n2} \\ w_0 x_{n2} x_{n1} + w_1 x_{n2}^2 \end{bmatrix}$$

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

EX 1.6. Code below:

```

1 %% Women's 100m data
2 % Load all Olympic data
3 load olympics;
4 % Copy the necessary variables
5 x = female100(:,1); % Olympic year
6 t = female100(:,2); % Winning time
7 % Augment x
8 X = [repmat(1,size(x)) x];
9 % Get solution
10 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`):

```

1 clear all;close all;
2 load synthdata
3
4 % Augment x
5 X = repmat(1,size(x));
6 for k = 1:4
7     X = [X x.^k];
8 end
9

```

```

10 % Fit the model
11 w = inv(X'*X)*X'*t;
12
13 % Randomise the data order
14 N = size(X,1);
15 order = randperm(N);
16 sizes = repmat(floor(N/10),1,10);
17 sizes(end) = sizes(end) + N-sum(sizes);
18 sizes = [0 cumsum(sizes)];
19
20 X = repmat(1,size(x));
21
22 loss = zeros(4,10);
23 for poly_order = 1:4
24     % Augment x
25     X = [X x.^poly_order];
26     for k = 1:10 % 10-fold CV
27         % Extract the train and test data
28         traindata = X(order,:);
29         traint = t(order);
30         testdata = X(order(sizes(k)+1:sizes(k+1)),:);
31         testt = t(order(sizes(k)+1:sizes(k+1)));
32         traindata(sizes(k)+1:sizes(k+1),:) = [];
33         traint(sizes(k)+1:sizes(k+1)) = [];
34
35         % Fit the model
36         w = inv(traindata'*traindata)*traindata'*traint;
37
38         % Compute loss on test data
39         predictions = testdata*w;
40         loss(poly_order,k) = sum((predictions - testt).^2);
41     end
42 end
43
44 % Plot the loss
45 plot([1:4],mean(loss,2));

```

EX 1.10. The total loss is

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

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

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

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

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

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:

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

Multiplying out, differentiating, equating to zero and solving:

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

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`):

```

1 clear all;close all;
2 load olympics;
3 % Extract men's 100m data
4 x = male100(:,1);
5 t = male100(:,2);
6
7 % Choose number of folds
8 K = 5;
9
10 % Randomise the data order
11 N = size(x,1);
12 order = randperm(N);
13 sizes = repmat(floor(N/K),1,K);
14 sizes(end) = sizes(end) + N-sum(sizes);
15 sizes = [0 cumsum(sizes)];
16
17 % Rescale x
18 x = x - x(1);
19 x = x./4;
20
21 X = [repmat(1,size(x)) x];
22 % Comment out the following line for linear
23 X = [X x.^2 x.^3 x.^4];

```

```
24
25 % Scan a wide range of values of the regularisation parameter
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
```