8.1 Working with the Hat matrix

8.1.1. Prove the results given by (8.8) and (8.9).

Solution: This problem uses the matrix algebra result that for any matrices A, B and C, $\text{tr}(ABC) = \text{tr}(BCA)$, where "tr" means trace of a matrix, or the sum of its diagonal elements. Then:

$$\sum h_{ii} = \text{tr}(\mathbf{H}) = \text{tr}(\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}') = \text{tr}((\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{X}) = \text{tr}(\mathbf{I}_{p'}) = p'$$

As in the text, $\mathbf{HX} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{X} = \mathbf{XI} = \mathbf{X}$, so if $\mathbf{X}_j$ is any column of $\mathbf{X}$, $\mathbf{HX}_j = \mathbf{X}_j$. Let $\mathbf{1}$ be the column of ones, which is included in $\mathbf{X}$ because the mean function has an intercept, and so $\mathbf{H1} = \mathbf{1}$, which is in scalar form the same as (8.9). ∎

8.1.2. Prove that $1/n \le h_{ii} \le 1/r$, where h_{ii} is a diagonal entry in $\mathbf{H}$, and r is the number of rows in $\mathbf{X}$ that are exactly the same as $\mathbf{x}_i$.

Solution: That $1/n \le h_{ii}$ follows directly from (8.11). To prove the upper bound, we use the properties that $\mathbf{H} = \mathbf{H}^2 = \mathbf{HH}'$, $h_{ij} = h_{ji}$, and, if $\mathbf{x}_i = \mathbf{x}_j$, then $h_{ij} = h_{ji} = h_{ii}$. We can write

$$h_{ii} = \sum_{i=1}^{n} h_{ij}h_{ji} = \sum_{i=1}^{n} h_{ij}^2 \ge rh_{ii}^2$$

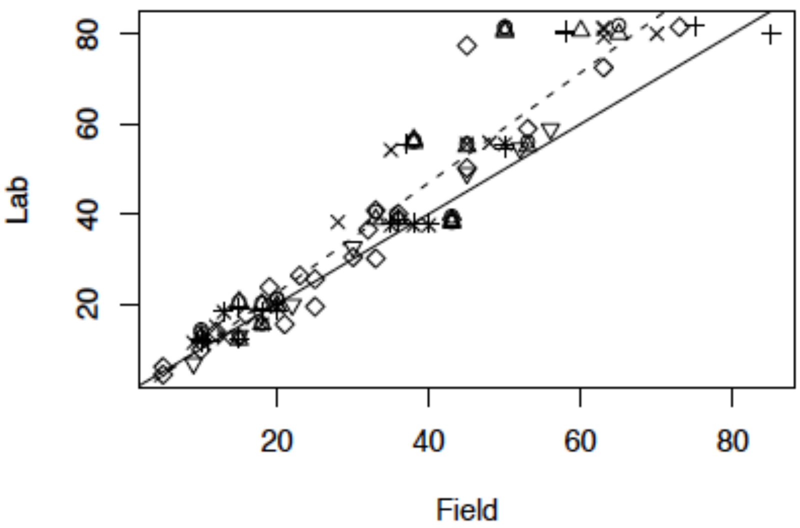
which simplifies to $h_{ii} \le 1/r$. ∎

8.3

This example compares in-field ultrasonic measurements of the depths of defects in the Alaska oil pipeline to measurements of the same defects in a laboratory. The lab measurements were done in six different batches. The goal is to decide if the field measurement can be used to predict the more accurate lab measurement. Use the Lab measurement as the response variable and the Field measurement as the predictor variable. The data, in the file `pipeline.txt`, were given at www.itl.nist.gov/div898/handbook/pmd/section6/pmd621.htm. The three variables are called *Field*, the in-field measurement *Lab* the more accurate in-lab measurement, and *Batch*, the batch number.

8.3.1. Draw the scatterplot of *Lab* versus *Field*, and comment on the applicability of the simple linear regression model.

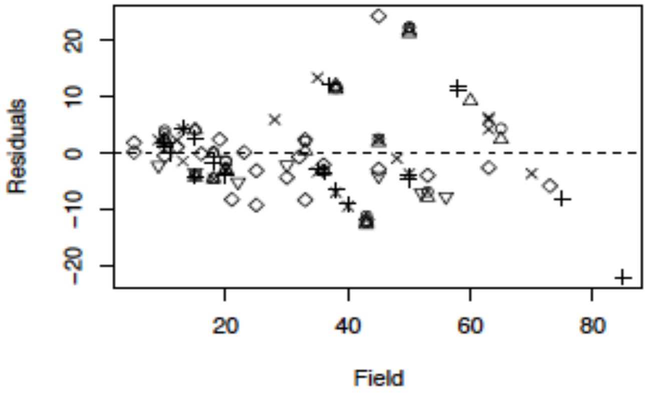
Solution:



Although not requested in the problem, a separate symbol has been used for each batch. A linear mean function seems plausible, but constant variance is unlikely. The solid line is the 45-degree line, and the dashed line is the OLS line. It appears that the field measurement underestimates depth for the deeper faults. ∎

8.3.2. Fit the simple regression model, and get the residual plot. Compute the score test for nonconstant variance and summarize your results.

Solution:



Here is the computer output for this problem using the car library in R/S-plus:

```
> m1 <- lm(Lab ~ Field, pipeline)
> ncvTest(m1)
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 29.586    Df = 1     p = 5.3499e-08
```

The score test for variance as a function of *Field* is $S = 29.59$ with 1 df, for a very small p-value. The conclusion is that variance increases with *Field*; deeper faults are less well measured. ∎

8.3.3. Fit the simple regression mean function again, but this time assume that $\text{Var}(Lab|Field) = \sigma^2 \times Field$. Get the score test for the fit of this variance function. Also test for nonconstant variance as a function of batch; since the batches are arbitrarily numbered, be sure to treat *Batch* as a factor. (Hint: Both these tests are extensions of the methodology outlined in the text. The only change required is to be sure that the residuals defined by (8.13) are used when computing the statistic.)

Solution:

```
> m2 <- update(m1, weights=1/Field)
> ncvTest(m2)
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 9.0315    Df = 1     p = 0.0026536
> ncvTest(m2, ~ factor(Batch))
Non-constant Variance Score Test
Variance formula: ~ factor(Batch)
Chisquare = 6.955    Df = 5     p = 0.22401
```

$S = 9.03$ with 1 df, and a tiny p-value, so this weighting scheme is not successful at modeling the nonconstant variance. The score statistics for *Batch* is $S = 6.96$ with 5 df, for a p-value of about 0.23, so there is no evidence that the variability differs between batches. ■

8.3.4. Repeat Problem 8.3.3, but with $\text{Var}(Lab|Field) = \sigma^2 \times Field^2$.

Solution:

```
> m3 <- update(m1, weights=1/Field^2)
> ncvTest(m3)
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 0.026989    Df = 1     p = 0.8695
```

The scores tests are $S = 0.027$ for *Field* with 1 df, and $S = 1.85$ with 5 df for *Batch*, both with large p-values. There is no evidence of an incorrectly specified variance function. ■

9.1 In an unweighted regression problem with $n = 54$, $p' = 5$, the results included $\hat{\sigma} = 4.0$, and the following statistics for four of the cases:

$\hat{e}_i$	h_{ii}
1.000	0.9000
1.732	0.7500
9.000	0.2500
10.295	0.185

For each of these four cases, compute r_i, D_i, and t_i. Test each of the four cases to be an outlier. Make a qualitative statement about the influence of each case on the analysis.

Solution:

```
> ehat <- c(1.000, 1.732, 9, 10.295)
> lev <- c(.9, .75, .25, .185)
> sig <- 4
> r <- ehat/(sig*sqrt(1-lev))
> D <- (1/5) * r^2 * (lev/(1-lev))
> ti <- r * sqrt((54-5-1)/(54-5-r^2))
> data.frame(ehat,lev,r,D,ti)
    ehat   lev       r       D      ti
1  1.000 0.900 0.79057 1.12500 0.78750

2  1.732 0.750 0.86600 0.44997 0.86375
3  9.000 0.250 2.59808 0.45000 2.76923
4 10.295 0.185 2.85094 0.36899 3.08954
```

Case 1 is likely to be most influential because of the large value of D. Cases 4 and 3 are most likely to be outliers because of the large values of t_i.■

9.3 The matrix $(\mathbf{X}'_{(i)}\mathbf{X}_{(i)})$ can be written as $(\mathbf{X}'_{(i)}\mathbf{X}_{(i)}) = \mathbf{X}'\mathbf{X} - \mathbf{x}_i\mathbf{x}'_i$, where $\mathbf{x}'_i$ is the ith row of $\mathbf{X}$. Use this definition to prove that (A.37) holds.

Solution: (A.37) asserts that

$$(\mathbf{X}'_{(i)}\mathbf{X}_{(i)})^{-1} = (\mathbf{X}'\mathbf{X})^{-1} + \frac{(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_i\mathbf{x}'_i(\mathbf{X}'\mathbf{X})^{-1}}{1 - h_{ii}}$$

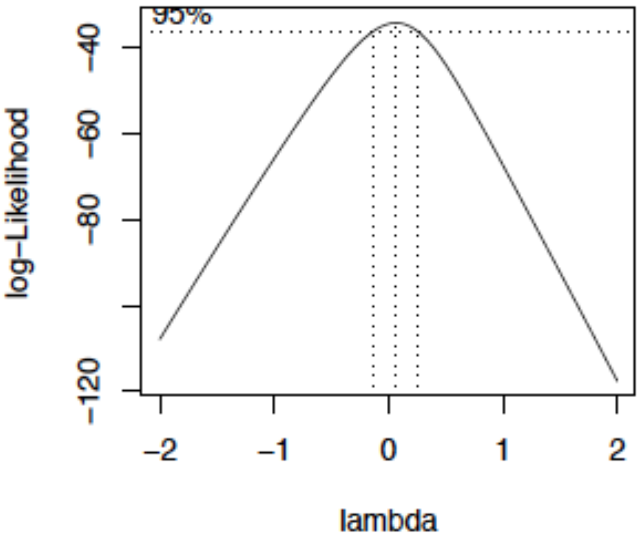
Multiply on the right by $(\mathbf{X}'_{(i)}\mathbf{X}_{(i)})$ and on the right by $\mathbf{X}'\mathbf{X} - \mathbf{x}_i\mathbf{x}'_i$, and simplify. The LHS equals $\mathbf{I}$, and the RHS is

$$\begin{aligned}\left((\mathbf{X}'\mathbf{X})^{-1} + \frac{(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_i\mathbf{x}'_i(\mathbf{X}'\mathbf{X})^{-1}}{1 - h_{ii}}\right)(\mathbf{X}'\mathbf{X} - \mathbf{x}_i\mathbf{x}'_i) = & \\ = & \ \mathbf{I} + \frac{(\mathbf{X}'\mathbf{X})^{-1}\mathbf{x}_i\mathbf{x}'_i}{1 - h_{ii}}(1 - 1 + h_{ii} - h_{ii}) \\ = & \ \mathbf{I}\end{aligned}$$

9.7 Refer to the lathe data in Problem 6.2.

9.7.1. Starting with the full second-order model, use the Box-Cox method to show that an appropriate scale for the response is the logarithmic scale.

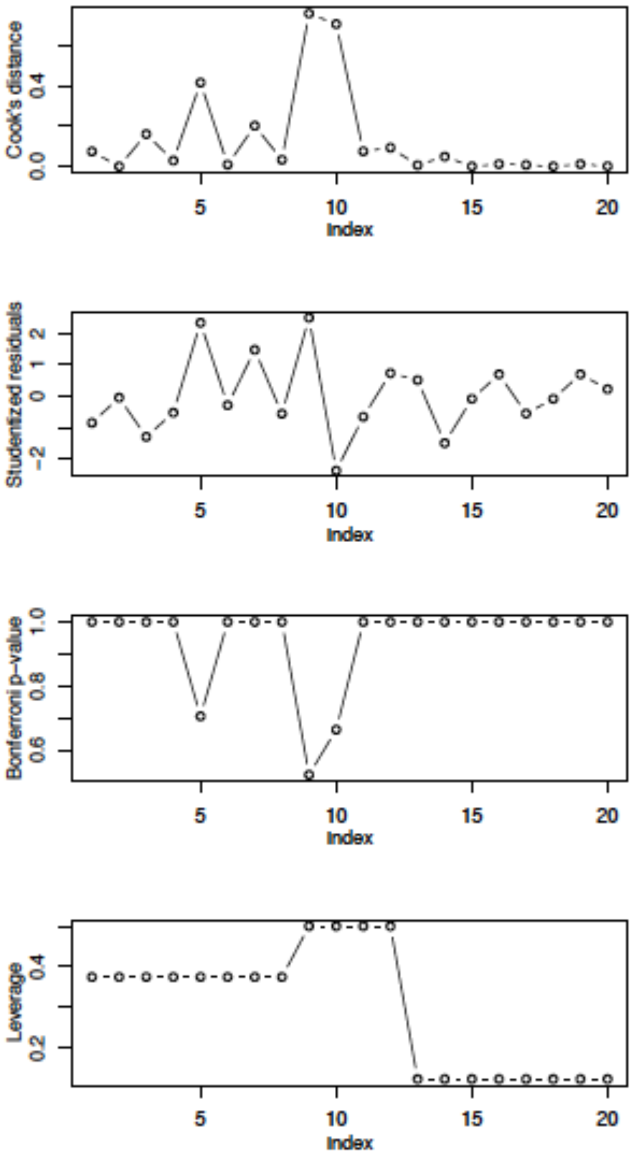
Solution:



This is the graph of the profile log-likelihood for the transformation parameter using the Box-Cox method for the second-order lathe model. The confidence interval for λ is very narrow and includes zero, suggesting a log transformation. ■

9.7.2. Find the two cases that are most influential in the fit of the quadratic mean function, and explain why they are influential. Delete these points from the data, refit the quadratic mean function, and compare to the fit with all the data.

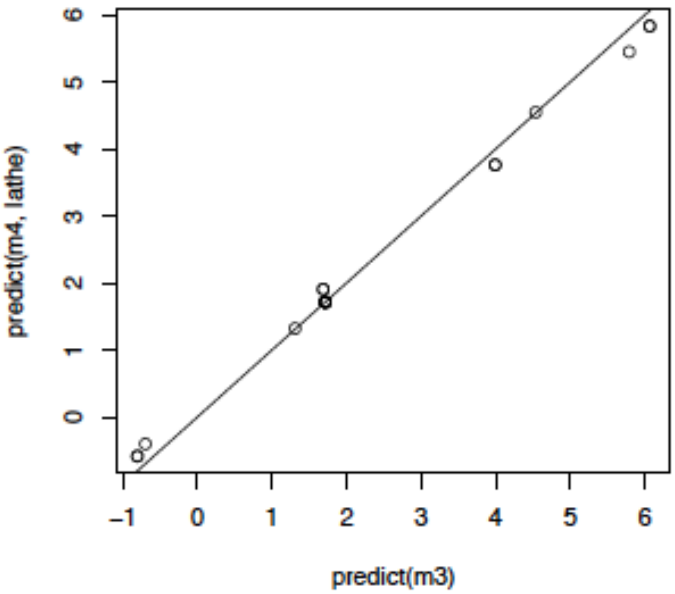
Solution:



Cases 9–12, the unreplicated "star points," have very high leverage. Two of these, numbers 9 and 10, also had large residuals, one positive and one negative, and these two cases have the largest Cook's distances. One way to assess their impact is to delete them, and refit to the smaller data set. We can then compare the fitted values:

```
> m4 <- update(m3, subset=-c(9,10))
> plot(predict(m3),predict(m4,lathe))
> abline(0,1)
```

The change in fitted values, including the fitted values for the two deleted cases, is generally not very large, and so the effect of deletion is minor by this measure. One could also look at changes on coefficients, in tests, and so on. ▮



10.1 Generate data as described for the two simulated data sets in Section 10.1, and compare the results you get to the results given in the text.

Solution: Here are the R commands that will reproduce the results given in the text.

```
> set.seed(1013185)
> case1 <- data.frame(x1=rnorm(100),x2=rnorm(100),
+                     x3=rnorm(100),x4=rnorm(100))
> e <- rnorm(100)
> case1$y <- 1 + case1$x1 + case1$x2 + e
> m1 <- lm(y~x1+x2+x3+x4,data=case1)
>
> X <- as.matrix(case1[,-5]) # change from data.frame to a matrix, drop y
> Var2 <- matrix(c(1,   0, .95,   0,
+                  0,   1,   0,-.95,
+                .95,   0,   1,   0,
+                  0,-.95,   0,   1), ncol=4)
> s1 <- chol(Var2)  # cholesky factor of Var2
> X <- X %*% s1
> dimnames(X)[[2]] <- paste("x",1:4,sep="")
> case2 <- data.frame(X)
> case2$y <- 1 + case2$x1 + case2$x2 + e
> m2 <- lm(y~x1+x2+x3+x4,data=case2)
```

The `set.seed` command initializes the random number generator to be sure the same numbers are used as in the book. For case 2, we have reused the same random numbers to make the results for the two cases correlated. *Var2* is the matrix (10.2). In the next line we found the Cholesky decomposition of *Var2*, so $Var2 = s1's1$, and so $X\,s1$ is like a sample from N(0, *Var2*). The next line makes sure the columns of X have the right names. Then y is recomputed, again using the same errors as for case 1, and the model is fit. ■

Table 10.12 Mantel's data for Problem 10.2.

	Y	X1	X2	X3
1	5.00	1.00	1004.00	6.00
2	6.00	200.00	806.00	7.30
3	8.00	−50.00	1058.00	11.00
4	9.00	909.00	100.00	13.00
5	11.00	506.00	505.00	13.10

10.2 Using the ŞdataŤ in Table 10.12 with a response Y and three predictors X_1, X_2 and X_3 from Mantel (1970) in the file `mantel.txt`, apply the BE and FS algorithms, using C_p as a criterion function. Also, find AIC and C_p for all possible models, and compare results. What is $X_{\mathcal{A}}$?

Solution: Using the `step` method in R/S-Plus, here is the result for forward selection:

```
> m0 <- lm(Y ~ 1, data=mantel)
> step(m0,scope=~X1+X2+X3,direction="forward")
Start:  AIC= 9.59
 Y ~ 1

       Df Sum of Sq   RSS   AIC
+ X3    1     20.69  2.11 -0.31
+ X1    1      8.61 14.19  9.22
+ X2    1      8.51 14.29  9.25
<none>              22.80  9.59

Step:  AIC= -0.31
 Y ~ X3

       Df Sum of Sq    RSS    AIC
<none>               2.112 -0.309
+ X2    1     0.066  2.046  1.532
+ X1    1     0.065  2.048  1.536

Call:
lm(formula = Y ~ X3, data = mantel)
```

```
Coefficients:
(Intercept)           X3
      0.798        0.695
```

This method uses AIC to select models, but since all the terms have a single df, the ordering of models with AIC and C_p is identical. Starting with the mean function with no predictors, at the first step we consider adding the one term that makes AIC as small as possible, which is X_3. At the second step, we consider adding another term after X_3 if it further reduces AIC; in this problem adding either X_1 or X_2 actually increases AIC, so we would select $X_{\mathcal{A}} = \{X_3\}$.

Using backward elimination,

```
> m1 <- lm(Y~X1+X2+X3, data=mantel)
> step(m1,scope=~1, direction="backward")
Start:  AIC= -314.77
 Y ~ X1 + X2 + X3

       Df Sum of Sq     RSS    AIC
<none>               4.6e-28 -314.8
- X3    1   1.7e-27 2.1e-27 -309.2
- X1    1       2.0     2.0    1.5
- X2    1       2.0     2.0    1.5

Call:
lm(formula = Y ~ X1 + X2 + X3, data = mantel)

Coefficients:
(Intercept)           X1           X2           X3
  -1.00e+03     1.00e+00     1.00e+00     4.11e-15
```

It appears that the backward elimination algorithm selects to remove *none* of the terms, as AIC is lowest for the mean function will all terms. *However, the residual sum of squares for both the full mean function, and the mean function without X_3, are zero, within rounding error.* Consequently, the difference in AIC between the full mean function and the mean function without X_3 is due to rounding error only. Consequently, X_3 can be deleted, and still give an exact fit. Using backward elimination, therefore, $X_{\mathcal{A}} = \{X_1, X_2\}$.

These two computational algorithms give different answers. We would certainly prefer the choice $X_{\mathcal{A}} = \{X_1, X_2\}$ from backward elimination because it gives an exact fit. ■