Inferences using Multiple Regression

A Nonlinear Example

We fit a nonlinear model to the following data in an earlier example, under the assumption that Y follows the model

$$Y = \beta_0 + \beta_1 x + \beta_2 \sqrt{x} + \beta_3 e^x + \epsilon.$$

x	0	1	2	3	4
y	-1.00	0.28	-2.56	-13.62	-46.60

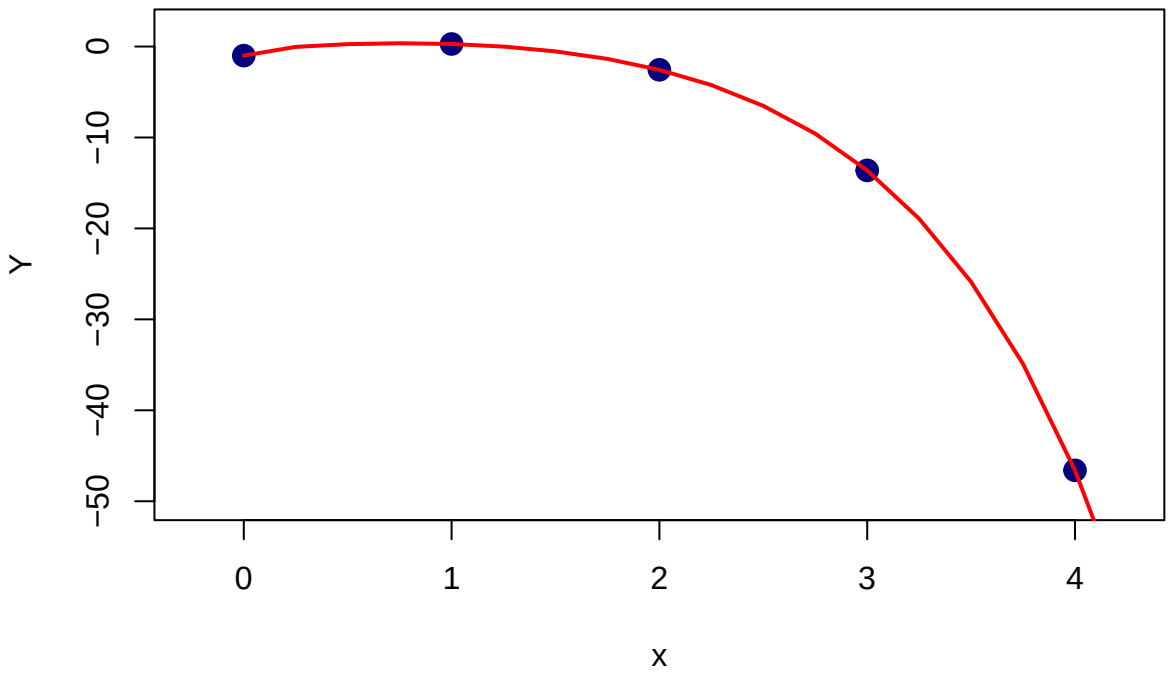
Specifically, we calculated the least squares estimator $\hat{\boldsymbol{\beta}}$ of the model coefficients by solving the linear system $(\mathbf{X}'\mathbf{X})\hat{\boldsymbol{\beta}} = \mathbf{X}'\mathbf{Y}$, where

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 & \sqrt{x_1} & e^{x_1} \\ 1 & x_2 & \sqrt{x_2} & e^{x_2} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_n & \sqrt{x_n} & e^{x_n} \end{bmatrix}, \qquad \mathbf{Y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \qquad \hat{\boldsymbol{\beta}} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix}.$$

This yields the model

$$\hat{\mathbf{Y}} = 2.139406 \times 10^{-4} + 1.0062727x + 1.9924763\sqrt{x} - 1.0002221e^x.$$

We plot the estimate of $E(Y)$ given by the least-squares coefficients below.



A Hypothesis Test

The observed value of $\hat{\beta}_0 = 2.139406 \times 10^{-4}$ is rather small. A reasonable question to ask is whether or not $\beta_0 = 0$.

We can test the null hypothesis $H_0 : \beta_0 = 0$ using the test statistic

$$t = \frac{\boldsymbol{a}'\hat{\boldsymbol{\beta}} - \boldsymbol{a}'\boldsymbol{\beta}}{S\sqrt{\boldsymbol{a}'(\mathbf{X}'\mathbf{X})^{-1}\boldsymbol{a}}}.$$

where

$$\boldsymbol{a} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

The formula for the test statistic simplifies to

$$t = \frac{\hat{\boldsymbol{\beta}}_{\mathbf{0}}}{S\sqrt{c_{00}}},$$

where c_{00} denotes the $(0,0)$-entry of the matrix $(\mathbf{X}'\mathbf{X})^{-1}$. We can calculate the observed value of test statistic using the following code.

```
# Load data.
x <- 0:4
Y <- c(-1, 0.28, -2.56, -13.62, -46.60)
```

```
n <- length(x)
# Create X.
X <- cbind(rep(x = 1, times = n), x, sqrt(x), exp(x))

# Create XX, XY.
XX <- t(X) %*% X
XY <- t(X) %*% Y

# Calculate S.
# Note that we divide by n - (k+1) = n - 4.
SSE <- t(Y) %*% Y - t(hB) %*% XY
V <- SSE/(n-4)
S <- sqrt(V)

# Calculate c00.
C <- solve(XX)
c00 <- C[1,1]

# Calculate t.
tval <- hB[1]/(sqrt(V*c00))
```

This yields

$$SSE = 4.0045052 \times 10^{-8}, \qquad S^2 = 4.0045052 \times 10^{-8}, \qquad c_{00} = 1.0188697, \qquad t = 1.0591548.$$

We use the rejection region

$$|t| > t_{\alpha/2} = t_{0.025} = 12.7062047$$

for $\alpha = 0.05$. Since the observed value of t is outside the rejection region, we do not have sufficient evidence to reject H_0.

Confidence and Prediction Intervals

Recall that a $100(1-\alpha)\%$-confidence interval for

$$E(Y) = \beta_0 + \beta_1 x + \beta_2 \sqrt{x} + \beta_3 e^x + \epsilon$$

at $x = x^*$ is given by

$$\boldsymbol{a'}\hat{\boldsymbol{\beta}} \pm t_{\alpha/2} S \sqrt{\boldsymbol{a'}(\mathbf{X'X})^{-1}\boldsymbol{a}},$$

where $\boldsymbol{a} = (1, x^*, \sqrt{x^*}, e^{x^*})$.

The following code creates a function for calculating the endpoints for the confidence intervals for a given x^*. The function takes as input $\hat{\boldsymbol{\beta}}$ as argument B, $(\mathbf{X^T X})^{-1}$ as argument C, sample standard deviation S (as argument S), number of observations n (as argument n), α (as argument *alpha*), and value x (as argument x).

```
CI <- function(B, C, S, n, alpha, x){
  # Forms vector a.
  a <- c(1, x, sqrt(x), exp(x))

  # Calculates radius of confidence interval.
  rad <- qt(p = alpha/2, df = n - 4, lower.tail = FALSE)*S*sqrt(t(a) %*% C %*% a)

  # Calculates endpoints of the confidence interval.
  lb <- t(a) %*% B - rad
```

```
  ub <- t(a) %*% B + rad

  # Return the interval.
  return(c(lb, ub))
}
```

For example, we can calculate the end points of a 95%-confidence interval for $E(Y)$ at $x = 1$ using the following code.

```
xstar <- 1
ci <- CI(B = hB, C = C, S = S, n = n, alpha = 0.05, x = xstar)
```

The function **CI** returns a 2-dimensional vector with first entry equal to the left endpoint of the confidence interval and second entry equal to the right endpoint:

$$0.2777317 < E(Y) < 0.2824228.$$

Similarly, a $100(1-\alpha)\%$-prediction interval for the value of Y at x^* is given by

$$\boldsymbol{a}'\hat{\boldsymbol{\beta}} \pm t_{\alpha/2} S \sqrt{1 + \boldsymbol{a}'(\mathbf{X}'\mathbf{X})^{-1}\boldsymbol{a}},$$

We can create a function for calculating a $100(1-\alpha)\%$ prediction interval at a given $x = x^*$ using the following code. Not that we use the same arguments as for the confidence interval function we calculated earlier.

```
PI <- function(B, C, S, n, alpha, x){
  # Forms vector a.
  a <- c(1, x, sqrt(x), exp(x))

  # Calculates radius of confidence interval.
  rad <- qt(p = alpha/2, df = n - 4, lower.tail = FALSE)*S*sqrt(1 + t(a) %*% C %*% a)

  # Calculates endpoints of the confidence interval.
  lb <- t(a) %*% B - rad
  ub <- t(a) %*% B + rad

  # Return the interval.
  return(c(lb, ub))
}
```

We can evaluate this function at $x = 1$ using the following code to obtain a 95% prediction interval for Y at $x^* = 1$.

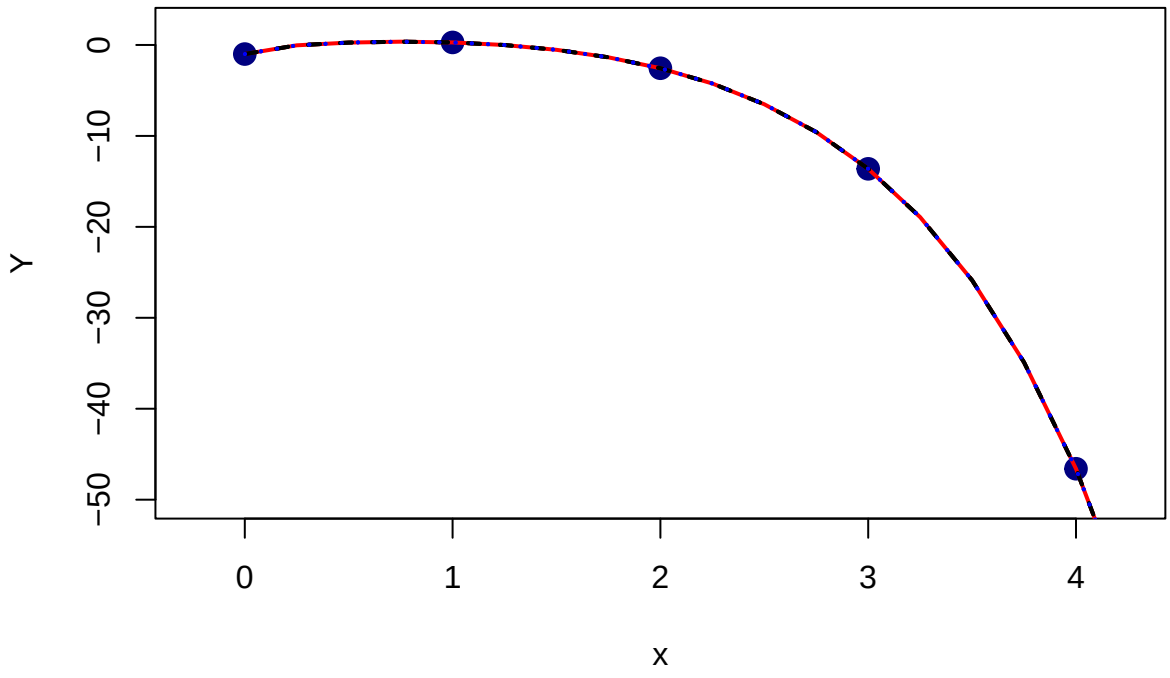
```
PI(B = hB, C = C, S = S, n = n, alpha = 0.05, x = xstar)
```

```
## [1] 0.2766180 0.2835365
```

We can use the **sapply** function in R to apply the **CI** and **PI** functions we just created to each element in a vector of x-values. This allows us to generate confidence and prediction bands for the least-squares curve. For example, the following code generates and plots 95% confidence and prediction for a sequence of x in the interval $[0, 4.25]$.

```
# Plot observations.
plot(x = 0:4, xlab = "x",
     y = c(-1, 0.28, -2.56, -13.62, -46.60), ylab = "Y",
     xlim = c(-1/4, 4 + 1/4),
     ylim = c(-50, 2), col = "navy",
     pch = 19, cex = 1.5)
```

```r
# Generate sequence of independent variables.
xs <- seq(from = 0, to = 4.25, by = 0.25)

# Predict points on curve using least-squares model
ys <- hB[1] + hB[2]*xs + hB[3]*sqrt(xs) + hB[4]*exp(xs)

# Plot least-squares curve
lines(xs, ys, col = "red", lwd = 2)

# Calculate 95% confidence intervals.
cis <- sapply(X = xs, FUN = CI, B = hB, C = C, S = S, n = n, alpha = 0.05)

# Calculate 95% prediction bands.
pis <- sapply(X = xs, FUN = PI, B = hB, C = C, S = S, n = n, alpha = 0.05)

# Plot curve indicating lower/upper bounds on confidence intervals.
lines(xs, cis[1,], col = "black", lty = 2, lwd = 2)
lines(xs, cis[2,], col = "black", pch = "o", lty = 2, lwd = 2)

# Plot prediction band.
lines(xs, pis[1,], col = "blue", pch = "o", lty = 3, lwd = 2)
lines(xs, pis[2,], col = "blue", pch = "o", lty = 3, lwd = 2)
```



Note that the prediction and confidence intervals are extremely close to the curve of predicted values. This suggests that these intervals are very narrow, which in turn suggests that the least-squares curve is an especially good estimator of the actual model coefficients.

A Test for Linearity

We can test whether the observed data follows a *linear* model. This occurs when $\beta_2 = \beta_3 = 0$, i.e., the model coefficients corresponding to nonlinear terms are equal to 0.

Formally, we want to test $H_0 : \beta_2 = \beta_3 = 0$ against the alternative hypothesis H_a : at least one of β_2 and β_3 are nonzero. We can test H_0 using the test statistic

$$F = \frac{(SSE_R - SSE_C)/2}{SSE_C/(n-4)},$$

where SSE_C is the sum of squares of deviations for the complete model, which we've already calculated to be $SSE_C = 4.0045052 \times 10^{-8}$, and SSE_R is the sum of squares of deviations for the reduced model

$$Y = \beta_0 + \beta_1 x + \epsilon.$$

We can calculate the least-squares line for the reduced model and the corresponding SSE_R using the following code.

```
# Create matrix of independent variable observations for reduced model.
XR <- cbind(rep(x = 1, times = n), x)

# Calculate the reduced model.
BR <- solve(t(XR) %*% XR, t(XR) %*% Y)

# Calculate SSE for the reduced model.
SSER <- t(Y) %*% Y - t(BR) %*% t(XR) %*% Y
```

This gives the least squares line

$$\hat{Y} = 8.32 + (-10.51)x$$

with $SSE_R = 453.6454$.

We are now ready to calculate the observed value of the test statistic.

```
fval <- ((SSER - SSE)/2)/(SSE/(n-4))
```

This gives $f = 5.664188 \times 10^9$. We use the rejection region

$$f > f_{0.05,2,n-4} = 0.0540166.$$

Since the observed value of F is in the rejection region, we can reject H_0.