3.3 OLS Regression on Simulated Data

Regression (or a line that fits the scatterplot) can be used for prediction. The function `lm()`, which stands for linear model, does run a regression in R. It fits a curve through a scatterplot, or a surface through higher-dimensional data.

```r
rm(list = ls(all = TRUE))  # Start from a clean slate.
set.seed(123)              # Ensures reproducible results.
x <- runif(100, 0, 1)      # x is uniform between 0 and 1.
error <- rnorm(100, 0, 1)  # Error is normal with mean = 0, sigma = 1.
y <- 10 + 2*x + error       # The real/true line is y = 10 + 2x.
plot(x, y)                 # Plot the scatterplot.
cor(x, y)                  # Correlation between x and y.
[1] 0.4916

model.1 <- lm(y ~ x)       # Fitting the regression.
model.1  # Note that the estimated coefficients are pretty close to the true ones

Call:
  lm(formula = y ~ x)

Coefficients:
  (Intercept)      x
   9.990      1.911

abline(model.1)  # Superimposes the fit on the scatterplot.

# To see what else is returned by lm(), use the following command:
names(model.1)
```

[1] "coefficients" "residuals" "effects" "rank"
[5] "fitted.values" "assign" "qr" "df.residual"
[9] "xlevels" "call" "terms" "model"
3.4 OLS Regression on “Real” Data

\[
x \leftarrow c(72, 70, 65, 68, 70) \quad \# \text{Enter data into R.}
y \leftarrow c(200, 180, 120, 118, 190) \quad \# \text{See 1.1 for alternative ways to enter data.}
\]

\[
\text{plot(x, y, cex = 0.5)}
\]

\[
\text{cor(x, y)}
\]

[1] 0.8692

\[
\text{model.1 } \leftarrow \text{lm(y } \sim \text{x)}
\]

\[
\text{abline(model.1)} \quad \# \text{Draws the fit}
\]

\[
\text{model.1} \quad \# \text{Returns the estimated intercept and slope.}
\]

\[
\text{Call:}
\text{lm(formula = y } \sim \text{x)}
\]

\[
\text{Coefficients:}
\text{(Intercept) x}
\]

\[
\begin{array}{cc}
-755.1 & 13.3
\end{array}
\]

\[
\text{summary(model.1)}
\]

\[
\text{Call:}
\text{lm(formula = y } \sim \text{x)}
\]
Residuals:
1  2  3  4  5
-1.46 5.11 11.54 -30.31 15.11

Coefficients:
          Estimate Std. Error t value Pr(>|t|)
(Intercept)  -755.11     272.53   -2.77   0.070  .
x            13.29      3.95    3.37  0.044   *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 20.9 on 3 degrees of freedom
Multiple R-squared: 0.791, Adjusted R-squared: 0.721
F-statistic: 11.3 on 1 and 3 DF,  p-value: 0.0436

![Graph showing regression line and data points]

**Example: Regression on Hail Data**

In practice, two quantities called “divergence” and “rotate” are measured by Doppler radar, while hail size is measured directly, i.e., on the ground. But if we can relate hail size to divergence and rotate, then we can predict hail size from Doppler radar. In regression lingo, size is the response (or dependent) variable, and the others are predictors (or independent variables, or covariates).

```r
dat <- read.table("hail_dat.txt", header=T)

plot(dat)
cor(dat)  # This shows the correlations between ALL the vars in the hail data

Divergence Rotational_velocity Hail_size
Divergence  1.0000     0.5496     0.5214
Rotational_velocity  0.5496     1.0000     0.5386
Hail_size      0.5214     0.5386     1.0000

size <- dat[, 3]  # Name the 3 columns in dat. Size is in 100th-of-an-inch.
```
rotate <- dat[, 2]
diverg <- dat[, 1]

model.1 <- lm(size ~ diverg)  # Regression of size and divergence.
plot(diverg, size)  # Viewing the scatterplot.
abline(model.1)  # Viewing the regression line.

model.2 <- lm(size ~ rotate)  # Regression on size and rotation.
plot(rotate, size)
abline(model.2)

Note that it looks like the line is not really going “through” the data; it seems like the line’s slope
should be larger. The fit is in fact correct. The line that intuitively (or visually) goes “through” the scatterplot is NOT the regression line, but something else called the “sd line.”
# Decomposing SST into SS_explained and SS_unexplained:

```r
anova(model.1)
```

## Analysis of Variance Table

<table>
<thead>
<tr>
<th>Response: size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Df</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>diverg</td>
</tr>
<tr>
<td>Residuals</td>
</tr>
</tbody>
</table>

---

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```r
summary(model.1)
```

## Call:

```r
lm(formula = size ~ diverg)
```

## Residuals:

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-126.1</td>
<td>-50.9</td>
<td>-19.8</td>
<td>44.8</td>
<td>262.6</td>
</tr>
</tbody>
</table>

## Coefficients:

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | 33.673     | 13.361  | 2.52     | 0.012 * |
| diverg    | 3.417      | 0.334   | 10.23    | <2e-16 *** |

---

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 76 on 280 degrees of freedom
Multiple R-squared: 0.272, Adjusted R-squared: 0.269
F-statistic: 105 on 1 and 280 DF,  p-value: <2e-16

# I.e., 27% of the variation in hail size can be attributed to (or explained by the linear relation with) divergence. The typical deviation of hail size about the regression line is 75.99 (100th-of-an-inch) ~ 0.76 (in) ~ 2 (cm).
3.5 Analysis of Variance (ANOVA) in Regression

ANOVA decomposes $SST$ (total sum of squares) into $SS_{explained}$ and $SS_{unexplained}$ (SSE).

$$SS_{explained} = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$$  \hspace{1cm} (1)

$$SS_{unexplained} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$  \hspace{1cm} (2)

$$SST = \sum_{i=1}^{n}(y_i - \bar{y})^2$$  \hspace{1cm} (3)

$SS_{explained}$ is converted to a proportion called R squared (a.k.a. coefficient of determination). It measures the proportion of the variability in $y$ that is explained by $x$. It’s a measure of goodness-of-fit.

```r
x <- c(72, 70, 65, 68, 70) # Enter data into R.
y <- c(200, 180, 120, 118, 190) # See 1.1 for alternative ways to enter data.
plot(x, y) # Plot the scatterplot.
cor(x, y) # Correlation between x and y.

[1] 0.8992

model.1 <- lm(y ~ x) # Fitting the regression.
anova(model.1) # Note that SS_explained = 4942 and SSE = 1309

Analysis of Variance Table

Response: y

<table>
<thead>
<tr>
<th></th>
<th>Df</th>
<th>Sum Sq</th>
<th>Mean Sq</th>
<th>F value</th>
<th>Pr(&gt;F)</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>1</td>
<td>4942</td>
<td>4942</td>
<td>11.3</td>
<td>0.044 *</td>
</tr>
<tr>
<td>Residuals</td>
<td>3</td>
<td>1309</td>
<td>436</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

summary(model.1) # R-squared = 0.7906

Call:
  lm(formula = y ~ x)

Residuals:
     1     2     3     4     5
-1.46  5.11 11.54 -30.31 15.11

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   -755.11     272.53    -2.77    0.070 .
x                13.29      3.95     3.37    0.044 *

---

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 20.9 on 3 degrees of freedom
Multiple R-squared: 0.791, Adjusted R-squared: 0.721
F-statistic: 11.3 on 1 and 3 DF, p-value: 0.0436

# The R^2 is reported as "Multiple R-squared".

# Note that the R-squared from summary() agrees with 1-(SSE/SST):
1 - (1308.9 / (4942.3 + 1308.9))

[1] 0.7906

$SS_{unexplained}$ (SSE) is converted to a standard deviation (of errors), and denoted as $se$. This standard deviation of errors is also called standard deviation about regression. Either way it is reported as “Residual standard error 20.9”. Note that it is equal to $\sqrt{\frac{SSE}{n-2}}$.

sqrt(1308.9 / (5 - 2))

[1] 20.89

In sum, R-squared and $se$ together tell you how good the model is. R-squared tells you what percent of the variance in $y$ can be attributed to $x$, and $se$ tells you the typical error, i.e., deviation of data from the line.

To get $R^2$ (and nothing else), use the following command:

summary(model.1)$r.squared

[1] 0.7906

# Do the following to check that R's SSE really is Sum of Squared Errors:
y_hat <- predict(model.1)  # This is a quick way of getting y_hat.
y_hat  # To see the predictions.

   1    2    3    4    5
201.5 174.9 108.5 148.3 174.9
3.6 Visual Assessment of Goodness-of-Fit

One way to access the goodness of fit is to examine the scatterplot of predicted $y$ versus actual $y$.  

```r
y_hat <- predict(model.1)

# Alternatively, you can use the predictions stored in lm() itself:
y_hat <- model.1$fitted.values

# Here is the scatterplot of predicted size vs. actual size:
plot(y, y_hat, cex = 0.5)

abline(0, 1, col = "red") # Add a diagonal line.
abline(h = mean(y)) # Add a horizontal line at the mean of y (i.e., size).
```

If the model were good, this scatterplot would be symmetrically spread about the red line. But, clearly our model is not good. This scatterplot (of predicted vs. actual) is often a great way of visualizing how well the model is doing. For example, we see that for smaller hail size (i.e., small $x$ value), the predictions of size are all above the diagonal indicating that the model over-predicts the size of small hail. Looking at larger $x$ values, it’s clear that the model under-predicts the size of large hail.

If a model is completely useless, then the predictions will be symmetrically spread about the horizontal line at the mean of $y$. Here, we can see that our model is nearly (but not completely) useless. This kind of model diagnosis can help in coming up with a better model.

Another visual assessment tool is the residual plot. This plot checks different facet of “goodness” (or quality) than the above plot.
In a good fit, these residuals (or errors) should NOT display any relationship with the predicted values. One way to confirm that there is no relationship is to compute the correlation:

```r
cor(y_hat, model.1$residuals)
[1] -1.237e-16
```

The fact that the correlation is zero is not a direct reflection of the goodness of fit, because that correlation is zero by construction. If it’s not zero, one has a bug! In fact, it is the identically-zero nature of this correlation which makes a plot of the residuals vs. \( \hat{y} \) a useful plot to examine. The correlation between the residuals and the observed \( y \) values is not identically zero; and for that reason the corresponding scatterplot is not readily interpretable.
3.7 Nonlinear Fits (Linear Regression with Higher Order Terms)

Linear regression is actually NOT linear when it comes to allowing nonlinear relationships between \( x \) and \( y \). (The term “linear” refers to the parameters of the model, i.e., the regression coefficients.) This is good news, because linear regression can fit any nonlinear data. But it’s also bad news, because the ability to fit nonlinear data also allows for overfitting. In developing regression models of data it is important to assure that the model is not overfitting the data, because such a model will have poor *predictive* capability. Toward the end of this book we will see how to assess the predictive capability of a regression model. Here, let’s first confirm that linear regression can *overfit* (memorize) data.

```r
set.seed(12)  # Set a seed to ensure reproducible results.
x <- seq(0, 0.9, 0.1)  # Pick 10 x's between 0 and 1.
y <- x + rnorm(10, 0, 0.3)  # x and y are "truly" linear, plus error.
plot(x,y)  # Look at the data.

lm.1 <- lm(y ~ x)  # Fit the simplest regression model
lines(x, lm.1$fitted.values)

lm.2 <- lm(y ~ x + I(x^2))  # Fit a regression model including the quadratic term.
lines(x, lm.2$fitted.values, col = 2)  # The I() is necessary. Don't ask why!

lm.3 <- lm(y ~ x + I(x^2) + I(x^3))  # Add a cubic term.
lines(x, lm.3$fitted.values, col = 3)  # Note that the fit is getting more curvy.

lm.4 <- lm(y ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5) + I(x^6) + I(x^7) + I(x^8) + I(x^9))  # Fit a 9th order polynomial.
lines(x, lm.4$fitted.values, col = 4)
summary(lm.4)$r.squared  # Examine the R-squared.

[1] 1

legend('bottomright', c('Linear', 'Quadratic', 'Cubic', '9th Order'),
text.col = c(1, 2, 3, 4), bty = 'n')

# Note that the last model will have no predictive power since it overfits the data.
```
3.8 Model Comparison

Example: Hail Data

Note that the closer $R^2$ is to 1, the “better” the fit and the closer it is to 0, the worse. Higher $R^2$ does not necessarily mean better predictions.

dat <- read.table("hail_dat.txt", header = T)

x_1 <- dat[, 1]    # Divergence.
x_2 <- dat[, 2]    # Rotate.
y <- dat[, 3]    # Hail size. Size is in 100th-of-an-inch.

# Renaming the columns in dat:
colnames(dat) <- c("x_1", "x_2", "y")

lm.1 <- lm(y ~ x_1)    # Predicting size from divergence (simple regression).
summary(lm.1)

Call:
  lm(formula = y ~ x_1)

Residuals:
  Min   1Q Median   3Q  Max
-126.1 -50.9  -19.8  44.8  262.6

Coefficients: