Formulas and R functions
STAT 450/870

## Chapter 1

- $Y_i = \beta_o + \beta_1 X_i + \varepsilon_i$ where $\varepsilon_i \sim$ independent $N(0,\sigma^2)$
- $\hat{Y}_i = b_0 + b_1 X_i$ where $b_1 = \dfrac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2} = \dfrac{\sum_{i=1}^{n} X_i Y_i - \frac{1}{n}\left(\sum_{i=1}^{n} X_i\right)\left(\sum_{i=1}^{n} Y_i\right)}{\left(\sum_{i=1}^{n} X_i^2\right) - \frac{1}{n}\left(\sum_{i=1}^{n} X_i\right)^2}$ and $b_0 = \bar{Y} - b_1 \bar{X}$
- $e_i = Y_i - \hat{Y}_i$
- SSE = $\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$
- $E(W_1 + c) = E(W_1) + c$
- $E(cW_1) = c * E(W_1)$
- $E(bW_1 + c) = b * E(W_1) + c$
- $E(W_1 + W_2) = E(W_1) + E(W_2)$
- $E(W_1 * W_2) \neq E(W_1) * E(W_2)$ – equality occurs when $W_1$ and $W_2$ are independent
- $MSE = \dfrac{SSE}{n-2}$

## Chapter 2

- $b_1 \sim N(\beta_1, \sigma^2 / \sum (X - \bar{X})^2)$
- $Var(\sum a_i Y_i) = \sum a_i^2 Var(Y_i)$
- $\hat{Var}(b_1) = MSE \Big/ \sum_{i=1}^{n}(X_i - \bar{X})^2$
- $t^* = (b_1 - \beta_1) \Big/ \sqrt{\hat{Var}(b_1)}$ and use a t(n-2) distribution
- $b_1 \pm t(1-\alpha/2; n-2) * \sqrt{\hat{Var}(b_1)}$
- $b_0 \sim N\left(\beta_0, \sigma^2\left(\dfrac{1}{n} + \dfrac{\bar{X}^2}{\sum (X_i - \bar{X})^2}\right)\right)$
- $\hat{Y}_h \pm t(1-\alpha/2; n-2)\sqrt{\hat{Var}(\hat{Y}_h)}$ where $\hat{Var}(\hat{Y}_h) = MSE\left(\dfrac{1}{n} + \dfrac{(X_h - \bar{X})^2}{\sum (X_i - \bar{X})^2}\right)$
- $\hat{Y}_h \pm t(1-\alpha/2; n-2)\sqrt{\hat{Var}(Y_{h(new)} - \hat{Y}_h)}$ where $\hat{Var}(Y_{h(new)} - \hat{Y}_h) = MSE\left(1 + \dfrac{1}{n} + \dfrac{(X_h - \bar{X})^2}{\sum (X_i - \bar{X})^2}\right)$
- $\hat{Y}_h \pm t(1-\alpha/2; n-2)\sqrt{\hat{Var}(\bar{Y}_{h(new)} - \hat{Y}_h)}$ for $\bar{Y}_{h(new)} = \dfrac{1}{m}\sum_{j=1}^{m} Y_{j,h(new)}$ where

  $\hat{Var}(\bar{Y}_{h(new)} - \hat{Y}_h) = MSE\left(\dfrac{1}{m} + \dfrac{1}{n} + \dfrac{(X_h - \bar{X})^2}{\sum (X_i - \bar{X})^2}\right)$
- SSTO = $\sum_{i=1}^{n}(Y_i - \bar{Y})^2$

- SSR = $\sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^2$
- SSTO = SSE + SSR
- MSR = SSR/1 = SSR when there is $\beta_0$ and $\beta_1$ in the regression model
- $F^* = \frac{MSR}{MSE}$ and use a F(1, n-2) distribution
- $R^2 = \frac{SSR}{SSTO} = \frac{SSTO - SSE}{SSTO}$

Chapter 3

- $e_i^* = \frac{e_i}{\sqrt{MSE}}$
- $Y_i^{(\lambda)} = \beta_0 + \beta_1 X_i + \varepsilon_i$ where $Y^{(\lambda)} = \begin{cases} Y^{\lambda} & \lambda \neq 0 \\ \log_e(Y) & \lambda = 0 \end{cases}$
- $d_{i1}=|e_{i1}-\tilde{e}_1|$ and $d_{i2}=|e_{i2}-\tilde{e}_2|$, median residual values are $\tilde{e}_1$ and $\tilde{e}_2$, $t_L^* = \frac{\bar{d}_1 - \bar{d}_2}{s\sqrt{\frac{1}{n_1}+\frac{1}{n_2}}}$ and use a t(n-2) distribution, $s^2 = \frac{\sum(d_{i1}-\bar{d}_1)^2 + \sum(d_{i2}-\bar{d}_2)^2}{n-2}$
- $\log(\sigma_i^2) = \gamma_0 + \gamma_1 X_i$, $X_{BP}^2 = \frac{SSR^*/2}{(SSE/n)^2}$ and use a $\chi^2$ distribution with 1 degree of freedom

Chapter 4

- $b_0 \pm t(1-\alpha/4;n-2) * \sqrt{\hat{Var}(b_0)}$, $b_1 \pm t(1-\alpha/4;n-2) * \sqrt{\hat{Var}(b_1)}$
- $\hat{Y}_h \pm t\left(1-\frac{\alpha}{2g};n-2\right)\sqrt{\hat{Var}(\hat{Y}_h)}$
- $\hat{Y}_h \pm t\left(1-\frac{\alpha}{2g};n-2\right)\sqrt{\hat{Var}(\hat{Y}_{h(new)})}$
- $\hat{X}_{h(new)} = \frac{Y_{h(new)} - b_0}{b_1}$
- $\hat{X}_{h(new)} \pm t(1-\alpha/2,n-2)\sqrt{\hat{Var}(\hat{X}_{h(new)})}$ where $\hat{Var}(\hat{X}_{h(new)}) = \frac{MSE}{b_1^2}\left[1+\frac{1}{n}+\frac{(\hat{X}_{h(new)}-\bar{X})^2}{\sum(X_i-\bar{X})^2}\right]$

Chapter 5

- $Cov(Z_1,Z_2) = E[(Z_1-\mu_1)(Z_2-\mu_2)]$
- $Corr(Z_1,Z_2) = \frac{Cov(Z_1,Z_2)}{\sqrt{Var(Z_1)}\sqrt{Var(Z_2)}}$ and $Cov(\mathbf{Z}) = \begin{bmatrix} Var(Z_1) & Cov(Z_1,Z_2) \\ Cov(Z_1,Z_2) & Var(Z_2) \end{bmatrix}$
- $\mathbf{Y} = E(\mathbf{Y}) + \boldsymbol{\varepsilon}$ where $E(\mathbf{Y}) = \mathbf{X}\boldsymbol{\beta}$

- Let **W** = **AY** where **Y** is a random vector and **A** is a matrix of constants. Then E(**A**) = **A**, E(**W**) = **A**E(**Y**), and Cov(**W**) = **A**Cov(**Y**)**A**′
- $\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$
- $\hat{\mathbf{Y}} = \mathbf{Xb} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} = \mathbf{HY}$ where $\mathbf{H}=\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$
- $\mathbf{e} = \mathbf{Y}(\mathbf{I}-\mathbf{H})$
- $\text{SSTO} = \mathbf{Y}'\mathbf{Y} - \frac{1}{n}\mathbf{Y}'\mathbf{JY}$, $\text{SSE} = \mathbf{e}'\mathbf{e}$, $\text{SSR} = \mathbf{b}'\mathbf{X}'\mathbf{Y} - \frac{1}{n}\mathbf{Y}'\mathbf{JY}$
- $\text{Cov}(\mathbf{b}) = \sigma^2(\mathbf{X}'\mathbf{X})^{-1}$
- $\widehat{\text{Var}}(\hat{Y}_h) = \text{MSE}(\mathbf{X}_h'(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}_h)$
- $\widehat{\text{Var}}(Y_{h(new)} - \hat{Y}_h) = \text{MSE}(1 + \mathbf{X}_h'(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}_h)$

Chapter 6

- $t^* = \frac{b_k - \beta_k}{\sqrt{\widehat{\text{Var}}(b_k)}}$ and use a t(n-p) distribution
- MSR = SSR/(p-1)
- MSE = SSE/(n-p)
- $F^*$ = MSR/MSE and use a F(p-1, n-p) distribution
- $R_a^2 = 1 - \frac{n-1}{n-p}(1-R^2)$

Chapter 7

- $\text{SSR}(X_2|X_1) = \text{SSE}(X_1) - \text{SSE}(X_1,X_2)$
- $F^* = \frac{(\text{SSE}(X_1,\ldots,X_g) - \text{SSE}(X_1,\ldots,X_g,X_{g+1},\ldots,X_{p-1}))/(p-1-g)}{\text{SSE}(X_1,\ldots,X_g,X_{g+1},\ldots,X_{p-1})/(n-p)} = \frac{\text{MSR}(X_{g+1},\ldots,X_{p-1} \mid X_1,\ldots,X_g)}{\text{MSE}(X_1,\ldots,X_g,X_{g+1},\ldots,X_{p-1})}$ and use a F(p-1-g, n-p) distribution
- $R^2_{Y3|12} = \frac{\text{SSR}(X_3 \mid X_1,X_2)}{\text{SSE}(X_1,X_2)} = \frac{\text{SSE}(X_1,X_2) - \text{SSE}(X_1,X_2,X_3)}{\text{SSE}(X_1,X_2)}$

Chapter 8

- Second order model for two predictor variables: $E(Y_i) = \beta_0 + \beta_1 Z_{i1} + \beta_2 Z_{i2} + \beta_3 Z_{i1} Z_{i2} + \beta_4 Z_{i1}^2 + \beta_5 Z_{i2}^2$

Chapter 9

- $C_p = \frac{\text{SSE}(X_1,\ldots,X_{p-1})}{\text{MSE}(X_1,\ldots,X_{P-1})} - (n-2p)$
- $AIC_p = n*\log(SSE_p) - n*\log(n) + 2p$
- $SBC_p = n*\log(SSE_p) - n*\log(n) + \log(n)*p$
- $\text{PRESS}_p = \sum_{i=1}^{n}(Y_i - \hat{Y}_{i(i)})^2 = \sum_{i=1}^{n}\frac{(Y_i - \hat{Y}_i)^2}{(1-h_i)^2}$

Chapter 10

- $r_i = \frac{e_i}{\sqrt{\hat{Var}(e_i)}} = \frac{e_i}{\sqrt{MSE(1-h_{ii})}}$ and $r_i \sim t(n-p)$
- $d_i = Y_i - \hat{Y}_{i(i)}$
- $t_i = \frac{d_i}{\sqrt{MSE_{(i)}(1+\mathbf{X}_i'(\mathbf{X}_{(i)}'\mathbf{X}_{(i)})^{-1}\mathbf{X}_i)}} = e_i\left[\frac{n-p-1}{SSE(1-h_{ii})-e_i^2}\right]^{1/2}$ and $t_i \sim t(n-p-1)$
- Hat matrix diagonal values criteria: $h_{ii} > 2p/n$; $h_{ii} > 0.5$ indicates very high, $0.2 < h_{ii} \leq 0.5$ indicate moderate
- $(DFFITS)_i = \frac{\hat{Y}_i - \hat{Y}_{i(i)}}{\sqrt{MSE_{(i)}h_{ii}}} = t_i\left(\frac{h_{ii}}{1-h_{ii}}\right)^{1/2}$ with $|(DFFITS)_i| > 1$ for "small to medium" sized data sets and $|(DFFITS)_i| > 2\sqrt{p/n}$ for "large" data sets as a criteria
- $D_i = \frac{\sum_{j=1}^{n}(\hat{Y}_j - \hat{Y}_{j(i)})^2}{pMSE} = \frac{e_i^2}{pMSE}\left[\frac{h_{ii}}{(1-h_{ii})^2}\right]$ and $D_i > F(0.50, p, n-p)$ as a criteria
- $(DFBETAS)_{k(i)} = \frac{b_k - b_{k(i)}}{\sqrt{MSE_{(i)}c_{kk}}}$ with $|(DFBETAS)_{k(i)}| > 1$ for "small to medium" sized data sets and $|(DFBETAS)_{k(i)}| > 2/\sqrt{n}$ for "large" data sets as a criteria
- $(VIF)_k = (1-R_k^2)^{-1}$ with $(VIF)_k > 10$ as a criteria

Chapter 11

- $\mathbf{b}_w = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}\mathbf{X}'\mathbf{W}\mathbf{Y}$ with $\mathbf{W} = \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_n \end{bmatrix} = \begin{bmatrix} 1/\sigma_1^2 & 0 & \cdots & 0 \\ 0 & 1/\sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1/\sigma_n^2 \end{bmatrix}$
- $Cov(\mathbf{b}_w) = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}$
- $d_i = \left[\frac{(X_{i1}-X_{h1})^2}{S_1} + \frac{(X_{i2}-X_{h2})^2}{S_2}\right]^{1/2}$ for two predictor variables
- $w_i = \begin{cases} \left[1-(d_i/d_q)^3\right]^3 & d_i < d_q \\ 0 & d_i \geq d_q \end{cases}$

**Test #1 R part:**

| Syntax | Description |
|---|---|
| *lm(formula = y ~ x, data = data.frame)* | Find the sample regression model (and various other measures) with the response variable y and predictor variable x |
| *summary(mod.fit)* | Summarizes information in *mod.fit* |
| *predict(object = mod.fit, newdata = data.frame, interval = "confidence", se.fit = TRUE, level = 1-alpha)* | Predicts the response variable at the predictor variable values in the *data.frame* using model information in *mod.fit*. The *data.frame* needs to have the same predictor variable as the original data set specified in *mod.fit*. The standard errors produced are $\sqrt{\hat{Var}(\hat{Y}_h)}$. A confidence interval is produced through the interval option. Prediction intervals are produced by changing *"confidence"* to *"prediction"*. The *level* option specifies the confidence level. |
| *anova(mod.fit)* | Finds the ANOVA table using information in an object called *mod.fit* |
| *boxcox(object = mod.fit, lambda = seq(from = -2, to = 2, by = 0.01))* | Finds $\hat{\lambda}$ for the Box-Cox transformation by looking in a region between -2 and 2 |
| *ifelse(test = condition to test, yes = yes value, no = no value)* | Returns a vector of values corresponding to the result of the test |
| *leveneTest(y = mod.fit$residuals, group = group)* | Perform Levene's test for residuals in *mod.fit* that have been split into *group*; this function is in the *car* package |
| *bptest(formula = y ~ x, data = data.frame, studentize = FALSE)* | Perform the Breusch-Pagan Test where y is the response variable, x is the predictor variable, and they are stored in a particular *data.frame*; this function is in the *lmtest* package |
| *examine.mod.simple(mod.fit.obj = mod.fit, const.var.test = TRUE, boxcox.find = TRUE)* | Runs my diagnostics function that is contained in *examine.mod.simple.R* and uses the model specified in *mod.fit*; constant variance tests are run and the Box-Cox transformation is found |
| *confint(object = mod.fit, level = 1 - alpha/g)* | Finds Bonferroni joint confidence intervals for g $\beta$'s in the model specified in *mod.fit* |

Additional general R functions and examples

| Syntax | Description |
|---|---|
| **Basic functions** | |
| *class(name)* | Finds the class of an object |
| *head(data.frame)* | Print the first six rows of the *data.frame* |
| *library(name)* | Make a library of functions called *name* ready to go in R |
| *mean(x = data)* | Find the mean of what is inside of *data* |

| Syntax | Description |
|---|---|
| *names(mod.fit)* | Allows one to see the elements in the list object called *mod.fit* |
| *set.seed(number)* | Set a seed number |
| *sum(x = data)* | Find the sum of what is inside of *data* |
| **Read in data** | |
| *read.table(file = "c:\\chris\\datafile.txt", header = TRUE, sep = "")* | Read in a text data file with variable names in the first row and spaces separating the variable names and their values. Comma delimited data files can be read in using the *sep=","* option (or *read.csv()* instead). If the first row does not contain the variable names, use *header = FALSE* and *col.names = c("var1", …, "var.c")* to name the variables yourself. |
| *write.table(x = set1, file = "C:\\out_file.csv", quote = FALSE, row.names = FALSE, sep=",")* | Save data in a data frame to a file on the hard drive. The data was in the data frame *set1* and it will be written as a comma delimited file named *out_file.csv*. |
| **Plotting** | |
| *abline(h = 0)* | Plots a horizontal line at 0 |
| *axis(side = 1, at = seq(from = 0, to = 4.5, by = 0.5), labels = true)* | Plot x-axis labels at 0 to 4.5 by 0.5. |
| *boxplot(x = x)* | Construct a box plot for data in a *vector* x |
| *curve(expr = x^2, xlim = c(-1, 1))* | Plots f(x) = $x^2$ for -1 < x < 1; the *add = TRUE* option can be used to add the curve to the current open plot |
| *hist(x = x)* | Construct a histogram for data in a *vector* x |
| *identify(x = x, y = y)* | This function allows you to interact with a plot to identify the observations on a y vs. x plot |
| *legend(locator(1), legend = vector of names, lty = vector of line types, col = vector of colors, bty = "n")* | Puts a legend on a plot where the insertion place is specified interactively with a left mouse click; no box is drawn around the legend |
| *plot(x = x, y = y)* | Plots y on the y-axis and x on the x-axis |
| *points(x =x, y = y, pch = 1, col = "red", cex = 2)* | Plots y on the y-axis and x on the x-axis with a plotting character of 1 (open circles), color of red, and symbol size that is twice as large as the default |
| *segments(x0 = $x_0$, y0 =$y_0$, x1 = $x_1$, y1 = $y_1$, lty=2)* | Draws a line segment between the points ($x_0$, $y_0$) and ($x_1$, $y_1$) using line type #2 (dashed line) |
| *stripchart(x = x, method = "jitter", vertical = TRUE)* | Construct a dot plot for data in a *vector* x with the points jittered; the points are plotted in a vertical manner |
| *win.graph(width = 6, height = 6, pointsize = 10)* | Opens a new graphics window that is 6"×6" with font size of 10 |
| **Probability distributions** | |

| Syntax | Description |
|---|---|
| *pchisq(q = y, df = degrees of freedom)* | Finds P(Y ≤ y) for Y that has a chi-square distribution with a given degrees of freedom |
| *qchisq(p = 1-$\alpha$, df = degrees of freedom)* | Finds 1-$\alpha$ quantile from a chi-square distribution for a given degrees of freedom |
| *pf(q = y, df1 = degrees of freedom 1, df1 = degrees of freedom 2)* | Finds P(Y ≤ y) for Y that has a F-distribution with a numerator (df1) and denominator (df2) degrees of freedom |
| *qf(p = 1-$\alpha$/2, df = degrees of freedom)* | Finds 1-$\alpha$/2 quantile from a F-distribution with a numerator (df1) and denominator (df2) degrees of freedom |
| *pnorm(q = y)* | Finds P(Y ≤ y) for Y that has a standard normal distribution |
| *qnorm(p = 1-$\alpha$/2)* | Finds 1-$\alpha$/2 quantile from a standard normal distribution |
| *pt(q = y, df = degrees of freedom)* | Finds P(Y ≤ y) for Y that has a t-distribution with a given degrees of freedom |
| *qt(p = 1-$\alpha$/2, df = degrees of freedom)* | Finds 1-$\alpha$/2 quantile from a t-distribution for a given degrees of freedom |
| **Code examples** | |
| *group<-ifelse(test = x < median(x), yes = 1, no = 2)*<br>*levene.test(y = mod.fit$residuals, group = group)* | Perform Levene's test for residuals in *mod.fit* that have been split into two groups based on the median of x; note that the *levene.test()* function is in the car package; this code is similar to what is in HS_college_GPA_ch3.R |

**Test #2 R part:**

| Syntax | Description |
|---|---|
| *matrix(data = c( ) , nrow = r , ncol = c, byrow = TRUE)* | Create a matrix of size r×c with data that is in the *c()* vector. The data is put into the matrix by rows. |
| *t(A)* | Find the transpose of a matrix A |
| *A%*%B* | Multiply the matrices A and B |
| *solve(A)* | Find the inverse of a matrix A |
| *J<-matrix(data = 1, nrow = n, ncol = n)* | Create a n×n matrix of 1's |
| *as.numeric(object)* | Treat an object's contents as numerical only |
| *diag(n)* | Create a n×n identity matrix |
| *vcov(mod.fit)* | Calculate the estimated covariance matrix for **b** using results from the *lm()* function stored in *mod.fit* |
| *cbind(1, c( ))* | Combine by columns the number 1 and a vector of data given by the *c()* function. The 1 value is "recycled" to match the same length as what is given by the *c()* function |
| *scatter3d(x = x, y = y, z = z, fit="linear", bg="white", grid=TRUE)* | Create a 3D scatter plot of the data in x, y, and z, and plot the multiple linear regression model using y as the response variable |
| *nba[nba$last.name == "Jordan" \| nba$last.name == "Stockton",]* | Extract rows of a data set called *nba* that match *last.name* values of Jordan **or** Stockton |
| *anova(mod.fit.red, mod.fit.comp, test = "F")* | Calculate the partial F-test statistic and p-value where *mod.fit.red* contains the reduced model results and *mod.fit.comp* contains the complete model results using separate implementations of the *lm()* function |
| *lm(formula = y ~ x + I(x^2), data = set1)* | This is an example of how to include a quadratic term in a regression model |
| *lm(formula = y ~ I(x-mean(set1$x)) + I((x-mean(set1$x))^2), data = set1)* | This is an example of how to include a mean adjusted quadratic term in a regression model |
| *library(Rcmdr)*<br>*library(rgl)*<br>*rgl.clear("all")*<br>*rgl.light()*<br>*rgl.bbox()*<br>*scatter3d(x = x, y = y, z = z, fit="quadratic", grid=TRUE, bg.col="black")* | This is an example of how to construct a 3D scatter plot with a 2nd order sample regression model |
| *lm(formula = y ~ x1 + x2 + x1:x2, data = set1)* | This is an example of how to include an interaction term in a regression model |
| *aggregate(formula = y ~ x, data = set1, FUN = mean)* | Finds the mean of y for each unique value of x. Other functions could be used to summarize y too by changing the *FUN* argument value. |
| *contrasts(x)* | Allows one to see how R will code a qualitative variable x by using indicator variables |
| *levels(x)* | Gives the levels of a qualitative variable x |

| Syntax | Description |
|---|---|
| *set1$x<-relevel(set1$x, ref = "b")* | Changes the reference level ("all 0 level") to "b" for the variable x in the data frame *set1*. |
| *library(package = multcomp)*<br>*K<-matrix(data = c(0,1,-1), nrow = 1, ncol = 3, byrow = TRUE)*<br>*compare.means<-glht(model = mod.fit, linfct = K)*<br>*summary(compare.means, test = adjusted("none"))* | This shows how to use the multcomp package to obtain a test for $H_0$: $\beta_1 - \beta_2 = 0$ vs. $H_a$: $\beta_1 - \beta_2 \neq 0$. The test argument of *summary()* can be changed to *adjusted("bonferroni")* to produce Bonferroni corrections to p-values when more than one test is performed. |
| *factor(x)* | This function creates a qualitative variable from the object *x*. |
| *fit.stat<-function(model, data) {*<br>*mod.fit<-lm(formula = model, data = data)*<br>*sum.fit<-summary(mod.fit)*<br>*aic.mod<-AIC(object = mod.fit, k = 2)*<br>*bic.mod<-AIC(object = mod.fit, k = log(length(mod.fit$residuals)))*<br>*sse<-anova(mod.fit)$"Sum Sq"[ length(anova(mod.fit)$"Sum Sq")]*<br>*p<-length(mod.fit$coefficients)*<br>*n<-length(mod.fit$residuals)*<br>*Cp<-sse/MSE.all - (n-2*p)*<br>*press<-sum(mod.fit$residuals^2/ (1 - lm.influence(mod.fit)$hat)^2)*<br>*data.frame(Rsq=sum.fit$r.squared, AdjRsq = sum.fit$adj.r.squared, AIC.stat = aic.mod, BIC.stat = bic.mod, PRESS = press, Cp = Cp) }* | Function used to calculate $R^2$, $R^2_a$, $AIC_p$, $SBC_p$, $PRESS_p$, and $C_p$. This function is in the nba_ch9.R program. |
| *text(x = x, y = y, labels = vector of labels)* | Put the text specified in labels on a plot at the desired (x, y) location |
| *regsubsets(x = model formula, data = set1, method = "exhaustive", nbest=4)* | Finds the best set of predictor variables for a regression model out of those given in the model formula. The four best models are given for possible sets of 1, 2, … predictor variables. The function is located in the *leaps()* package. The *summary()* and *plot()* functions can be used with the resulting object to summarize the results. The *subsets()* function in the *car* package can summarize the results as well. |

| Syntax | Description |
|---|---|
| *step(object = mod.fit, direction = "forward", scope=list(lower = model formula, upper = model formula), k = 2)* | Perform forward predictor variable selection between the models listed in the lower and upper list elements. One needs to fit a model first with *lm()* and use the *object* option to pass in a corresponding model fit object. The *k = 2* option corresponds to using the AIC (*k = log(sample size)* can be used for SBC or BIC). The *direction = "backward"* or *"both"* options can be used to perform backward elimination or stepwise forward, respectively. |
| *test.var<-function(Ho, Ha, data) {*<br>*Ho.mod<-lm(formula = Ho, data = data)*<br>*Ha.mod<-lm(formula = Ha, data = data)*<br>*anova.fit<-anova(Ho.mod, Ha.mod)*<br>*round(anova.fit$"Pr(>F)"[2], 4)*<br>*}* | This is a function that can be used to compare a complete and reduced model through the use of a partial F-test. The function is in the nba_ch9.R program. |

**Final exam R part:**

| Syntax | Description |
|---|---|
| *avPlots(model = mod.fit)* | Constructs added variable plots corresponding to the information in *mod.fit*. The function is in the *car* package. |
| *rstandard(model = mod.fit)* | Find the studentized residuals using information in *mod.fit* |
| *rstudent(model = mod.fit)* | Find the studentized deleted residuals using information in *mod.fit* |
| *outlier.test(model = mod.fit)* | Perform a test for outliers using information in *mod.fit* and with a Bonferroni adjustment. This function is in the *car* package. |
| *hatvalues(model = mod.fit)* | Find the diagonal values of the hat matrix using information in *mod.fit* |
| *dffits(model = mod.fit)* | Find the DFFITS values using information in *mod.fit* |
| *cooks.distance(model = mod.fit)* | Find the Cook's distance values using information in *mod.fit* |
| *dfbetas(model = mod.fit)* | Find the DFBETAS values using information in *mod.fit* |
| *symbols(x = x, y = y, circles = z, inches = 0.25)* | Plot y on the y-axis and x on the x-axis. The plotting point is proportional to z with the largest size being 0.25 inches. |
| *vif(mod.fit)* | Find the VIF values using information in *mod.fit*. This function is in the *car* package. |
| *examine.mod.multiple.final(mod.fit.obj = mod.fit, first.order = number, const.var.test = TRUE, boxcox.find = TRUE)* | Use this function to examine various plots and summary measures to evaluate a model. See the *examine.mod.multiple.final.R* program file for more information. |
| *runif(n = n, min = 0, max = 1)* | Simulate n observations from a uniform distribution between 0 and 1. This was used to split a data set in the *NBA_ch10_validation.R* program. |
| *quantile(x = x, probs = c(0.2, 0.4, 0.6, 0.8), type = 1)* | Find the 0.2, 0.4, 0.6, 0.8 observed quantiles for a vector of data |
| *tapply(X = x, INDEX = groups, FUN = var)* | This is an example of how to apply the *var()* function to a vector of data that has been put into a number of groups. See the *weighted_least_squares.R* program for an example. |
| *lm(formula = Y ~ X, data = data, weight = 1/w)* | Find a regression model using weights in w. |
| *lqs(formula = Y ~ X, data = data, method = "method name")* | Find the parameter estimates of a regression model. The method option of "lms" is for least median squares. The method option of "lts" is for least trimmed squares. This function is in the *MASS* package. |

| Syntax | Description |
|---|---|
| *persp(x = x, y = y, z = z, theta = 200, phi = 20, ticktype = "detailed", expand = 1, shade =0.5, col = "green3")* | Construct a 3D plot. The *theta* and *phi* options control the angle rotation of the plot. The other options can be used as needed to make the plot look nicer. |
| *loess(formula = Y ~ X, data = data, span = 0.5, degree = 1)* | Find the first order lowess regression model for a value of q specified in the *span* option. A higher order model can be found by changing the *degree* option. |