IPS9 in R: Inference for Regression (Chapter 10)

Bonnie Lin and Nicholas Horton (nhorton@amherst.edu)

July 19, 2018

Introduction and background

These documents are intended to help describe how to undertake analyses introduced as examples in the Ninth Edition of *Introduction to the Practice of Statistics* (2017) by Moore, McCabe, and Craig.

More information about the book can be found here. The data used in these documents can be found under Data Sets in the Student Site. This file as well as the associated R Markdown reproducible analysis source file used to create it can be found at https://nhorton.people.amherst.edu/ips9/.

This work leverages initiatives undertaken by Project MOSAIC (http://www.mosaic-web.org), an NSF-funded effort to improve the teaching of statistics, calculus, science and computing in the undergraduate curriculum. In particular, we utilize the `mosaic` package, which was written to simplify the use of R for introductory statistics courses. A short summary of the R needed to teach introductory statistics can be found in the mosaic package vignettes (http://cran.r-project.org/web/packages/mosaic). A paper describing the mosaic approach was published in the *R Journal*: https://journal.r-project.org/archive/2017/RJ-2017-024.

Chapter 10: Inference for regression

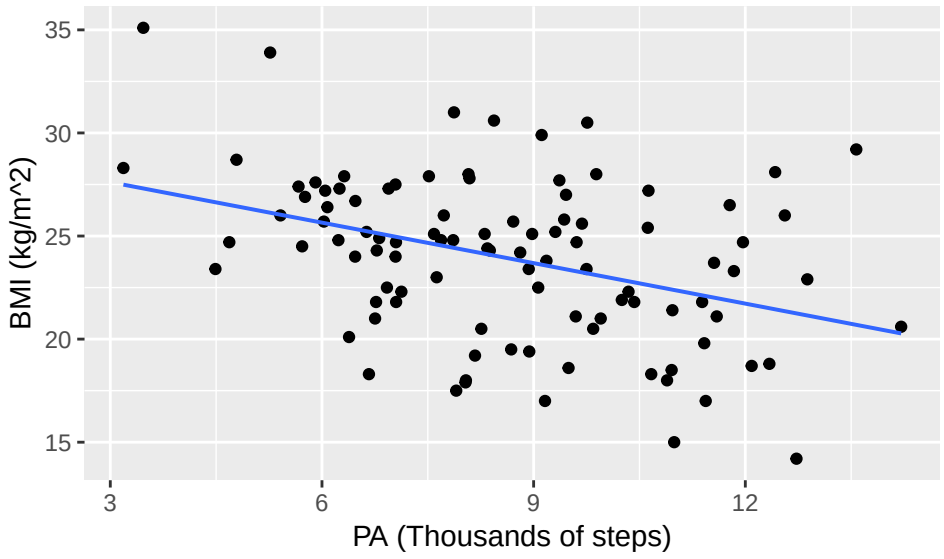
This file replicates the analyses from Chapter 10: Inference for regression.

First, load the packages that will be needed for this document:

```
library(mosaic)
library(readr)
```

Section 10.1: Simple linear regression

```
PABME <- read_csv("https://nhorton.people.amherst.edu/ips9/data/chapter10/EG10-01PABMI.csv")
### Figure 10.3, page 559
gf_point(BMI ~ PA, data = PABME) %>%
  gf_lm() %>%
  gf_labs(x = "PA (Thousands of steps)", y = "BMI (kg/m^2)")
```




```
# gf_lm() adds the least-squares line

## Creating the linear model with the lm() function
lm_PABME <- lm(BMI ~ PA, data = PABME)
## Displaying the output with the msummary() function
msummary(lm_PABME)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  29.5782     1.4120  20.948  < 2e-16 ***
## PA           -0.6547     0.1583  -4.135  7.5e-05 ***
##
## Residual standard error: 3.655 on 98 degrees of freedom
## Multiple R-squared:  0.1485, Adjusted R-squared:  0.1399
## F-statistic:  17.1 on 1 and 98 DF,  p-value: 7.503e-05
```

By default, the `read_csv()` function will output the types of columns, as we see above. To improve readability for future coding, we will suppress the "Parsed with column specification" message by adding `message = FALSE` at the top of the code chunks.

You would interpret this output by reporting the model as $\hat{BMI} = 29.578 - 0.655(PA)$, the same way as the textbook does on page 563. This equation also defines the straight line that was plotted in Figure 10.3, using the `gf_lm()` function.

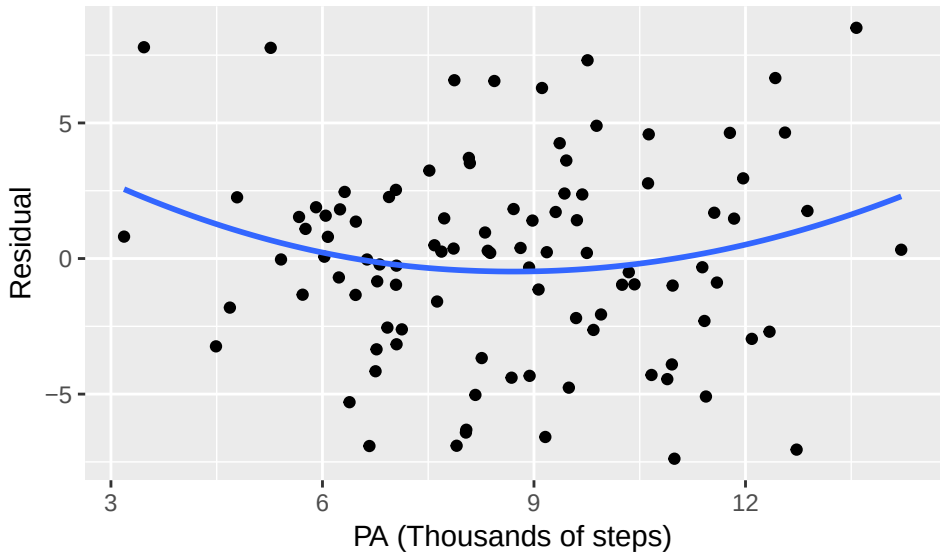
Suppose that a female college student averages 8000 steps per day. By making the linear model into a function, we can predict the BMI of the person.

```
### Example 10.4, page 564
PABME_mod <- makeFun(lm_PABME)
PABME_mod(8) # BMI estimate
```

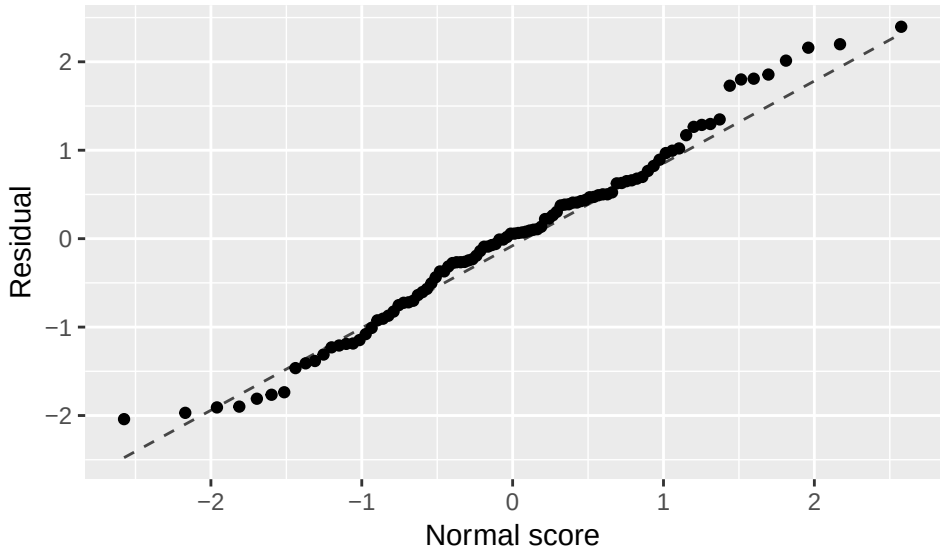
```
##        1
## 24.34076
```

```
### Figure 10.5, page 566
gf_point(resid(lm_PABME) ~ PA, data = PABME) %>%
  gf_smooth(span = 2) %>%
  gf_labs(x = "PA (Thousands of steps)", y = "Residual")
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



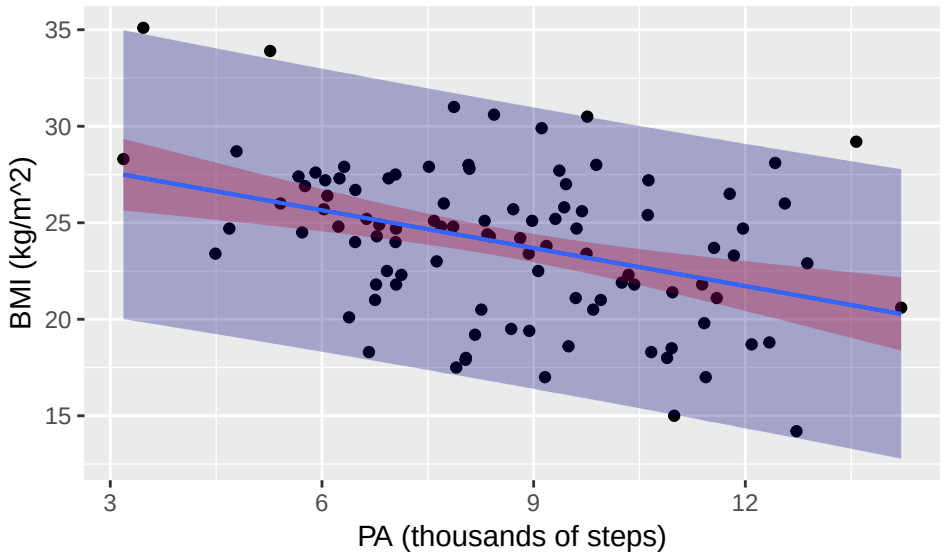

```
### Figure 10.6, page 566
gf_qq(~ rstandard(lm_PABME)) %>%
  gf_qqline() %>%
  gf_labs(x = "Normal score", y = "Residual")
```



If the student's actual BMI is 25.66, you can easily find the residual by calculating the difference between the actual and the predicted values. In this case, the residual would be 1.317.

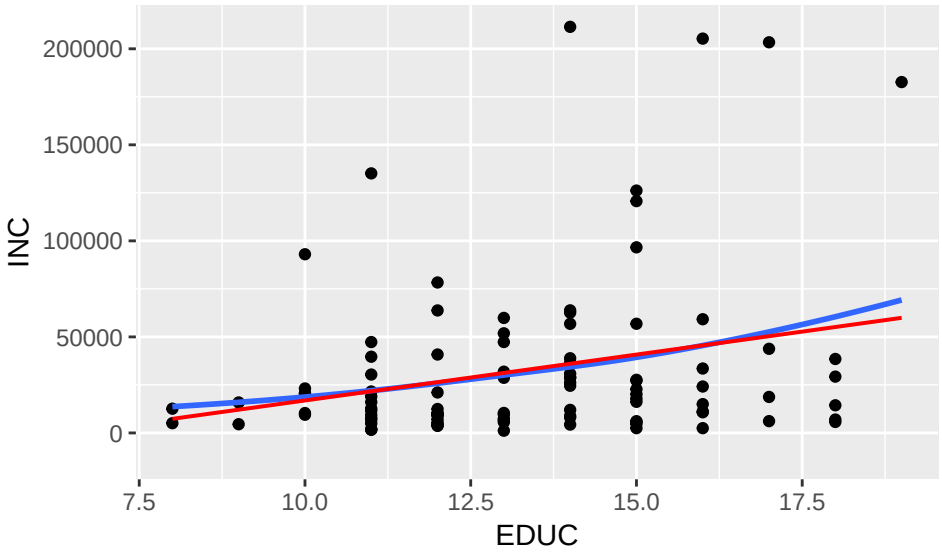
Note that plotting the Normal quantile plot of the residuals requires the `rstandard()` function call.

```
### Figure 10.7 and 10.8, page 571
gf_point(BMI ~ PA, data = PABME) %>%
  gf_lm(interval = "confidence", fill = "red") %>%
  gf_lm(interval = "prediction", fill = "navy") %>%
  gf_labs(x = "PA (thousands of steps)", y = "BMI (kg/m^2)")
```

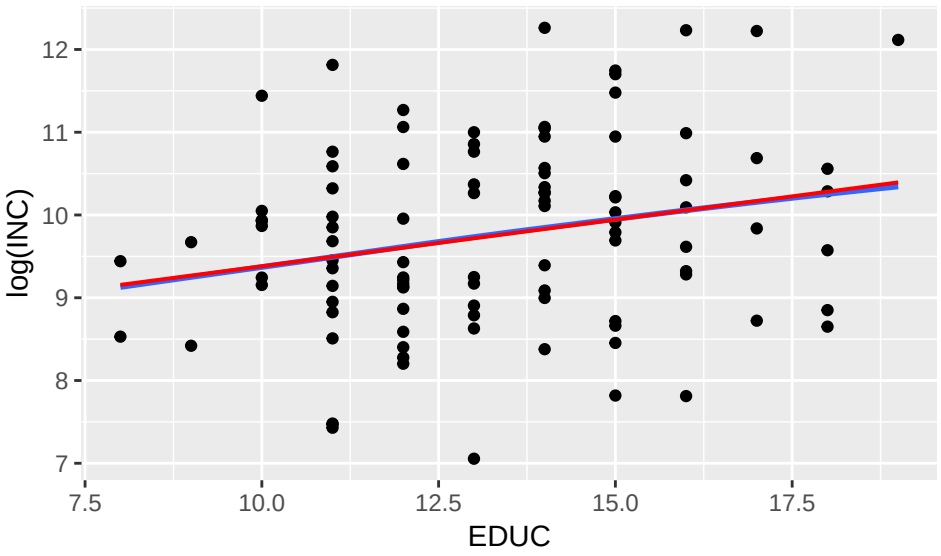


The red bands show the 95% confidence limits, while the navy bands show the 95% prediction limits.

```
ENTRE <- read_csv("https://nhorton.people.amherst.edu/ips9/data/chapter10/EG10-11ENTRE.csv")
### Figure 10.9, page 575
gf_point(INC ~ EDUC, data = ENTRE) %>%
  gf_smooth(span = 2) %>%
  gf_lm(color = "red")
```




```
### Figure 10.10, page 575
log_inc_ENTRE <- ENTRE %>%
  mutate(log_INC = log(INC))
gf_point(log_INC ~ EDUC, data = log_inc_ENTRE) %>%
  gf_smooth(span = 2) %>%
  gf_lm(color = "red") %>%
  gf_labs(y = "log(INC)")
```



On this scatterpot of income versus education, we have plotted the smooth function (blue) and the least-squares line (red).

Section 10.2: More details about simple linear regression

```
GADIA <- read_csv("https://nhorton.people.amherst.edu/ips9/data/chapter10/EG10-16GADIA.csv") %>%
  select(Diameter, GA) %>%
  na.omit()
```

```
## Warning: Missing column names filled in: 'X3' [3], 'X4' [4], 'X5' [5],
## 'X6' [6], 'X7' [7], 'X8' [8], 'X9' [9], 'X10' [10]
```

```
### Example 10.17, page 589
favstats(~ Diameter, data = GADIA)
```

```
##  min   Q1 median    Q3 max mean       sd n missing
##    2 6.75   11.5 19.25  23 12.5 8.360622 6       0
```

```
favstats(~ GA, data = GADIA)
```

```
##  min Q1 median    Q3 max     mean      sd n missing
##   16 20     27 31.75  39 26.66667 8.75595 6       0
```

```
cor(GA ~ Diameter, data = GADIA)
```

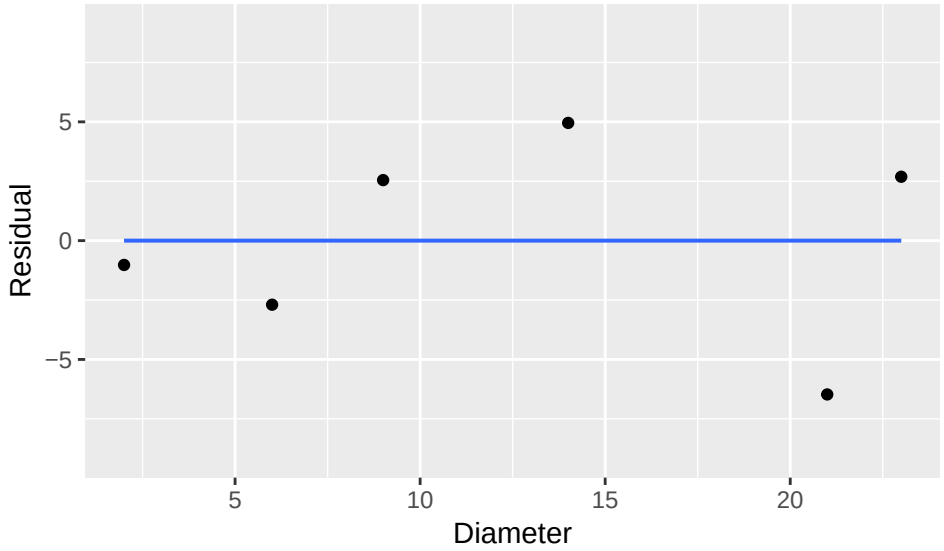
```
## [1] 0.876987
```

```
### Example 10.18, 10.20, and 10.21 page 589
lm_GADIA <- lm(GA ~ Diameter, data = GADIA)
msummary(lm_GADIA)
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  15.1860     3.6851   4.121   0.0146 *
## Diameter      0.9185     0.2516   3.650   0.0218 *
##
## Residual standard error: 4.704 on 4 degrees of freedom
## Multiple R-squared:  0.7691, Adjusted R-squared:  0.7114
## F-statistic: 13.32 on 1 and 4 DF,  p-value: 0.02177
```

From the msummary output, you can compute the least-squares regression line, estimate the standard deviation about the line and test the slope by using the t statistic and P-value by the 'Diameter' variable

```
### Example 10.19, page 590
gf_point(resid(lm_GADIA) ~ Diameter, data = GADIA) %>%
  gf_lm() %>%
  gf_labs(x = "Diameter", y = "Residual")
```




```
### Example 10.22, page 592
confint(lm_GADIA, 'Diameter')
```

```
##              2.5 %   97.5 %
## Diameter 0.2198527 1.617057
```

```
### Example 10.23, page 593
new.dat <- data.frame(Diameter = 10)
predict(lm_GADIA, newdata = new.dat, interval = 'confidence')
```

```
##        fit      lwr      upr
## 1 24.37053 18.75992 29.98114
```

Since there are only two columns in this dataset that are not filled with NA's, I have used the `select()` and `na.omit()` functions to select columns that we will use for further analysis.