Exam 2

David Lewis

May 1, 2025

Problem 1

```
load("mcmcSamples.RData")
```

Length

```
print(sprintf("The length of sample1Final is: %i", length(sample1Final)))
```

```
## [1] "The length of sample1Final is: 6008"
```

```
print(sprintf("The length of sample2Final is: %i", length(sample2Final)))
```

```
## [1] "The length of sample2Final is: 11413"
```

effective sample size

```
print(sprintf("The effective sample size of sample1Final is: %f", batchmeans::ess(sample1Final)))
```

```
## [1] "The effective sample size of sample1Final is: 937.959110"
```

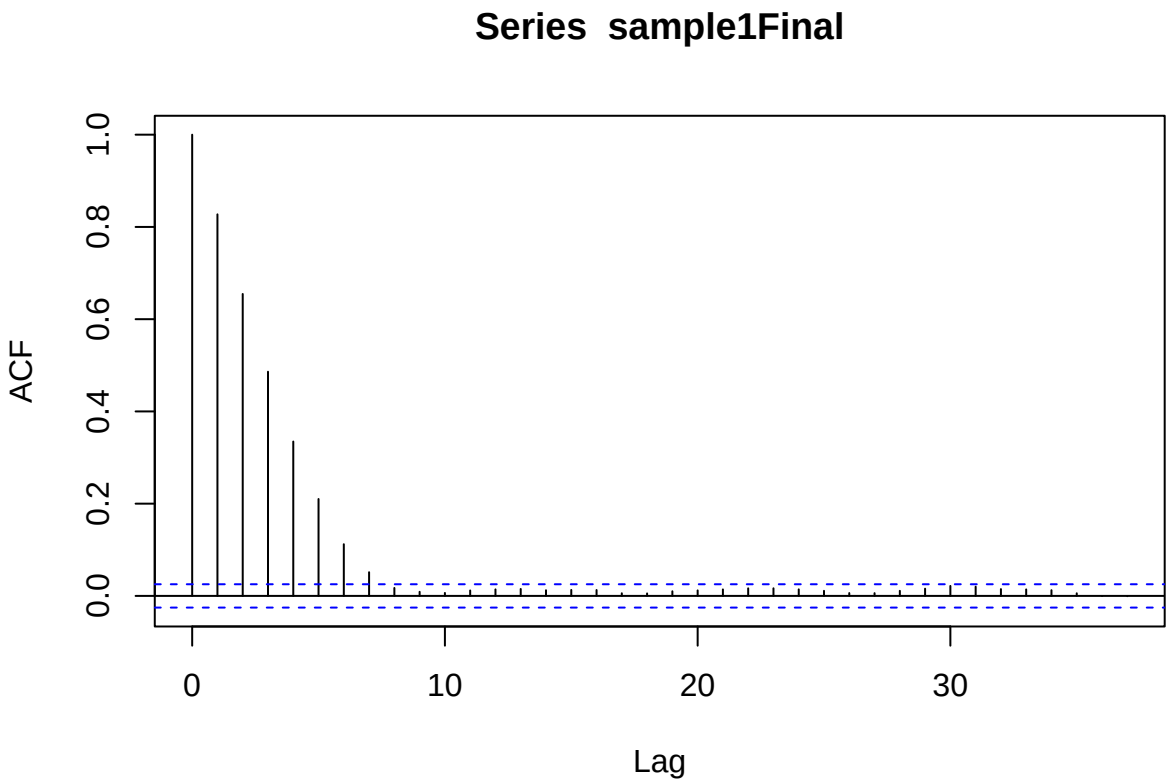
```
print(sprintf("The effective sample size of sample2Final is: %f", batchmeans::ess(sample2Final)))
```

```
## [1] "The effective sample size of sample2Final is: 508.357520"
```

auto correlation plots

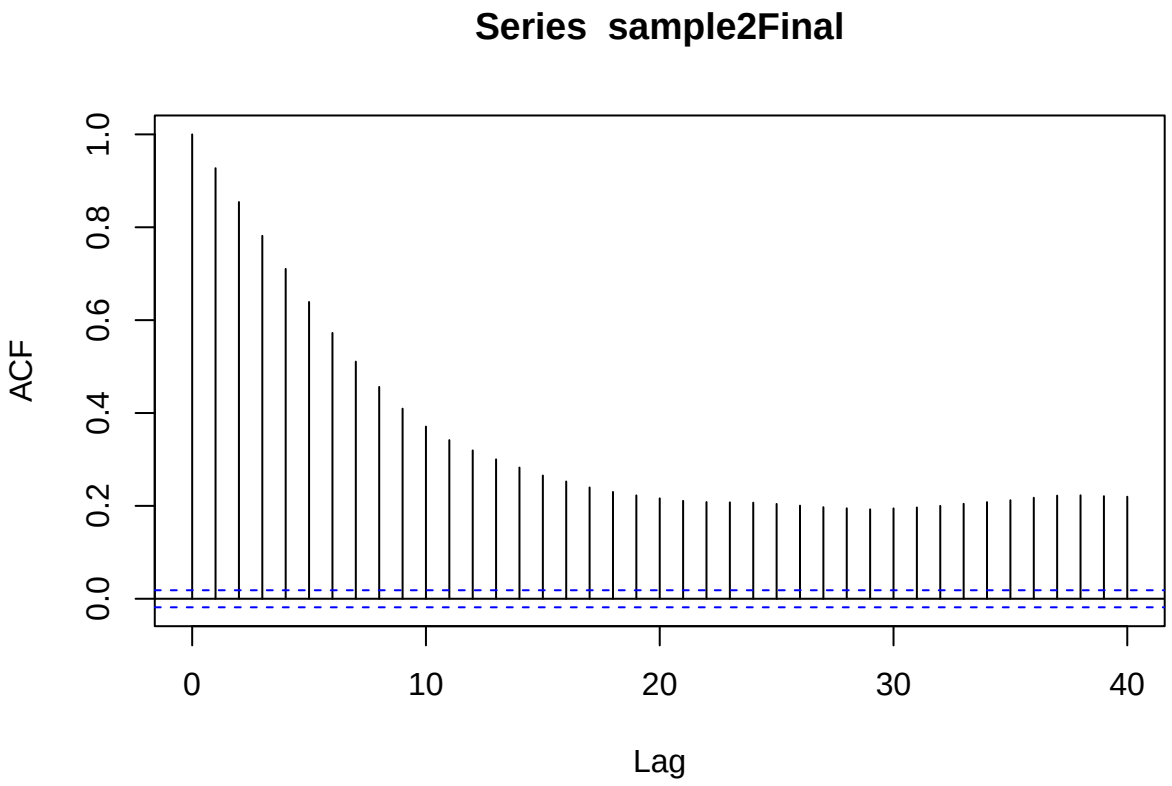
Sample1Final Autocorrelation

```
acf(sample1Final)
```



Sample2Final Autocorrelation

```
acf(sample2Final)
```

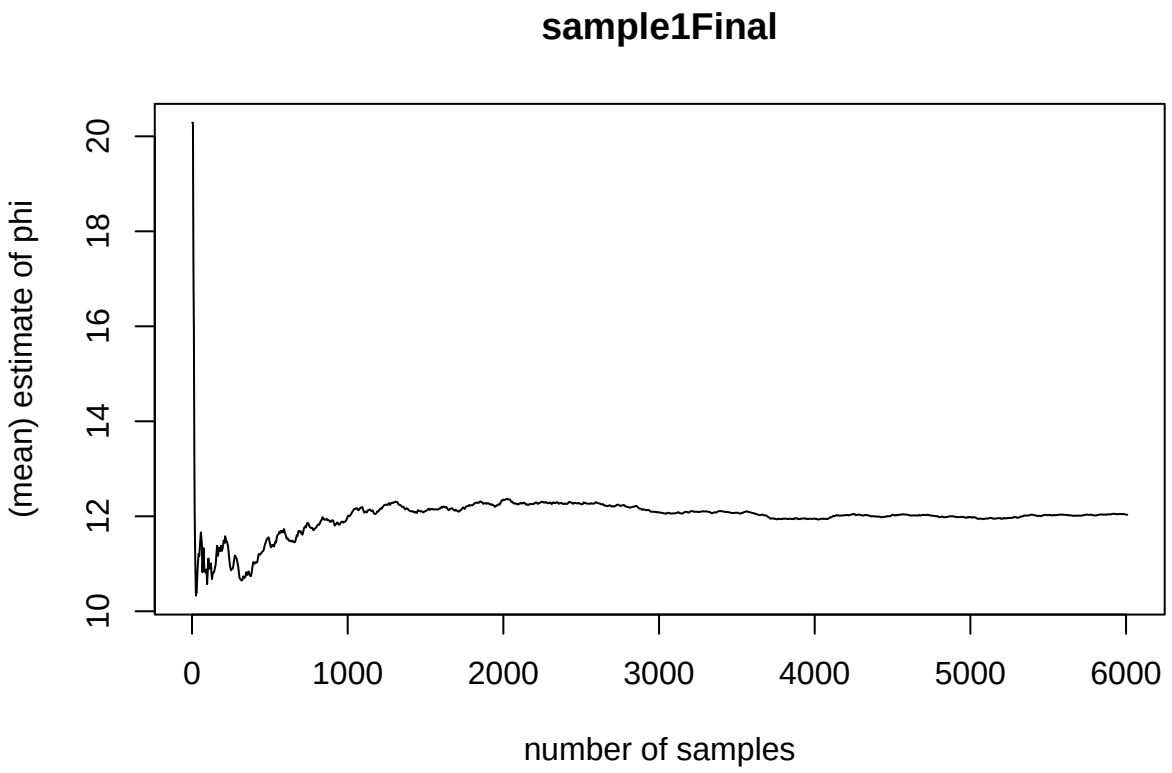


Estimate as samples increase

```
# plotting function
mean_by_sample_size <- function(sample, title) {
    plot(cumsum(sample) / seq_along(sample), type = "l", main = title, ylab = "(mean) estimate of phi",
}
```

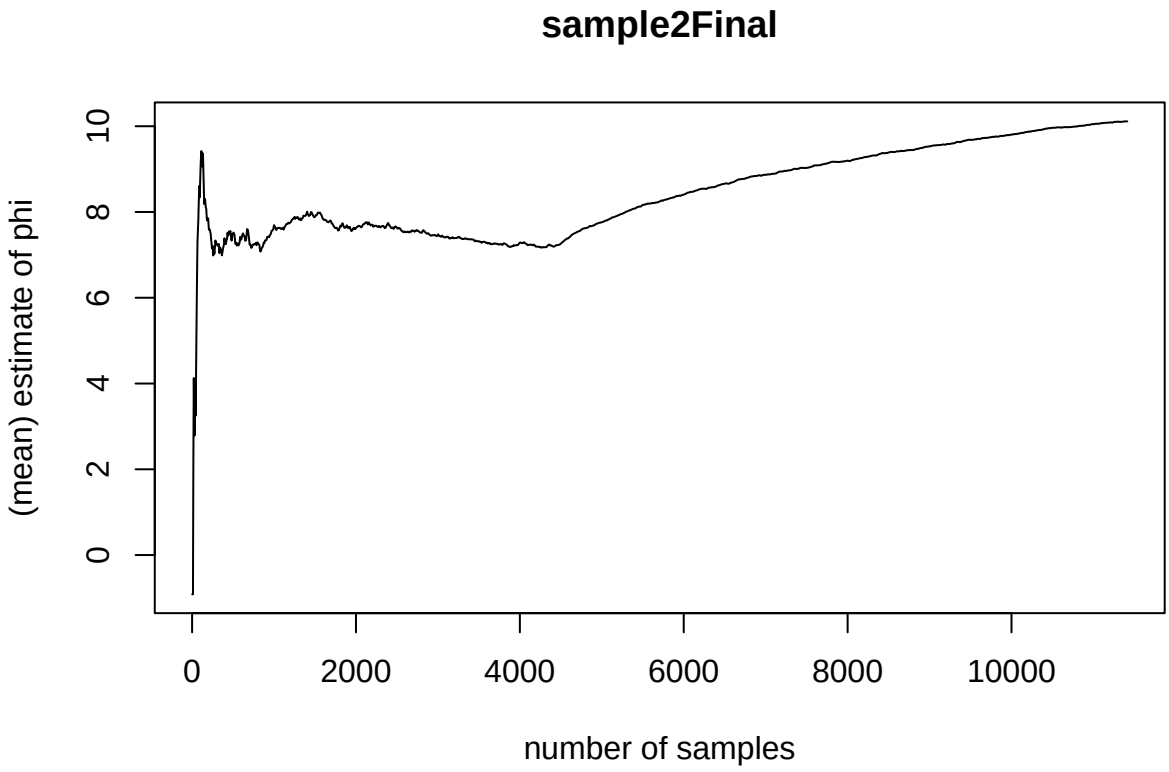
Sample1Final estimates

```
mean_by_sample_size(sample1Final, "sample1Final")
```



Sample2Final estimates

```
mean_by_sample_size(sample2Final, "sample2Final")
```



Which sample is better?

As we can see from the above estimates plot, Sample2Final does not appear to converge as the number of samples increases. Sample1Final on the other hand does not have this problem. The auto correlation plots also show a similar problem with sample2Final. Additionally, the ratio of effective sample size to length of sample is much lower in sample2Final, another indicator of low quality.

Final Verdict: Sample1Final is the better sample.

```
interval <- 0.95

lower <- (1 - interval) / 2
upper <- 1 - lower

quantile(sample1Final, c(lower, upper))
```

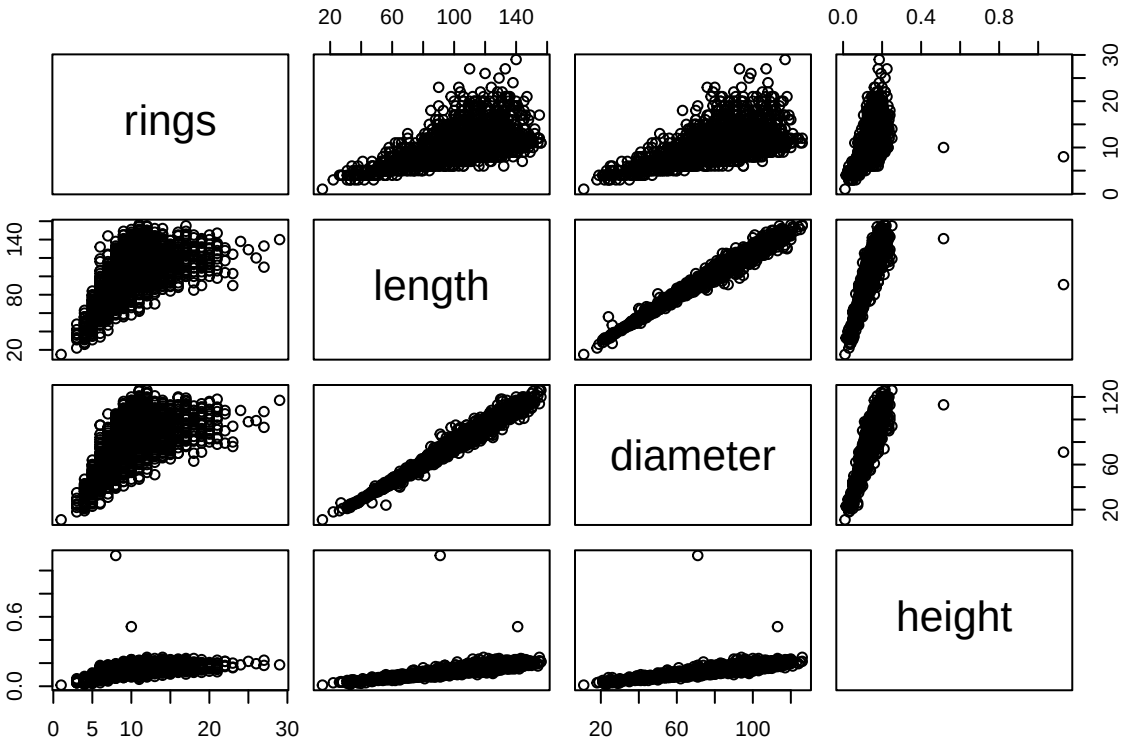
```
##      2.5%     97.5%
##  2.018822 21.429740
```

Problem 2

```
load("snail.RData")
```

What is odd about the height variable?

```
pairs(snail[, 1:4])
```



There is very little variability in the height variable aside from 2 outliers. The vast majority of the height measurements are between 0 and 0.2. By contrast, all of the other variables are spread mostly evenly across their axis. For example, length and diameter are about evenly correlated, resulting in a structure similar to a line with a slope of one. height and diameter, on the other hand, show a structure like a line with a much higher slope.

If the outliers are removed, however, I believe that the height diameter plot would look very similar to the diameter length plot.

The rings variable is actually the most different (lowest correlation) to the other variables.

correlation matrix

```
cor(snail[, 1:4])
```

```
##              rings    length  diameter    height
## rings    1.0000000 0.5659078 0.5820560 0.5330419
## length   0.5659078 1.0000000 0.9877781 0.7953226
## diameter 0.5820560 0.9877781 1.0000000 0.8007994
## height   0.5330419 0.7953226 0.8007994 1.0000000
```

We see that while height is less correlated with the other variables than length or diameter, rings are the least correlated with the other variable.

This is inline with the earlier assumptions about the correlation given the scatter plots.

Bootstrap

```
BootstrapCorrCI = function(x, y, reps) {
    n = length(x)
    thetahat = cor(x, y)
    thetahatbootstrap = rep(0, reps)
```

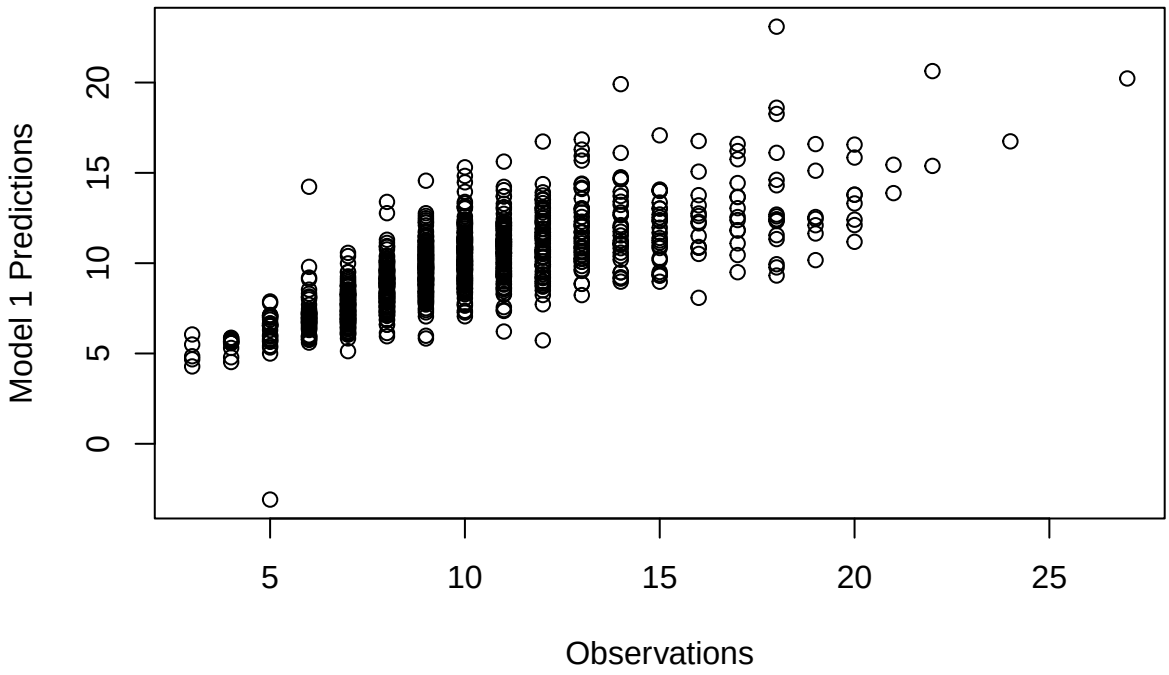
```
    for (i in 1:reps) {
        bootstrap_index = sample(1:n, n, replace = TRUE)
        xbootstrap = x[bootstrap_index]
        # bootstrap sample x
        ybootstrap = y[bootstrap_index]
        # bootstrap sample y
        if ((var(xbootstrap) != 0) & (var(ybootstrap) != 0)) {
            thetahatbootstrap[i] = cor(xbootstrap, ybootstrap)
            # sample corr for bootstrap sample
        }
    }
    quantile(thetahatbootstrap, prob = c(0.025, 0.975), na.rm = TRUE)
}

BootstrapCorrCI(snail$length, snail$rings, reps = 10000)
```

```
##      2.5%     97.5%
## 0.5401911 0.5907709
```

Bootstrap replicates are samples by preserving the index relationship between any length and rings values. Additionally, bootstrap uses replacement sampling, i.e. the same length-ring pair can be picked multiple times. `BootsrapCorrCI` repeatedly creates samples of the same length as the data, essentially creating new replicates from the existing data. In this case, 10000 replicates are created.

Problem 3

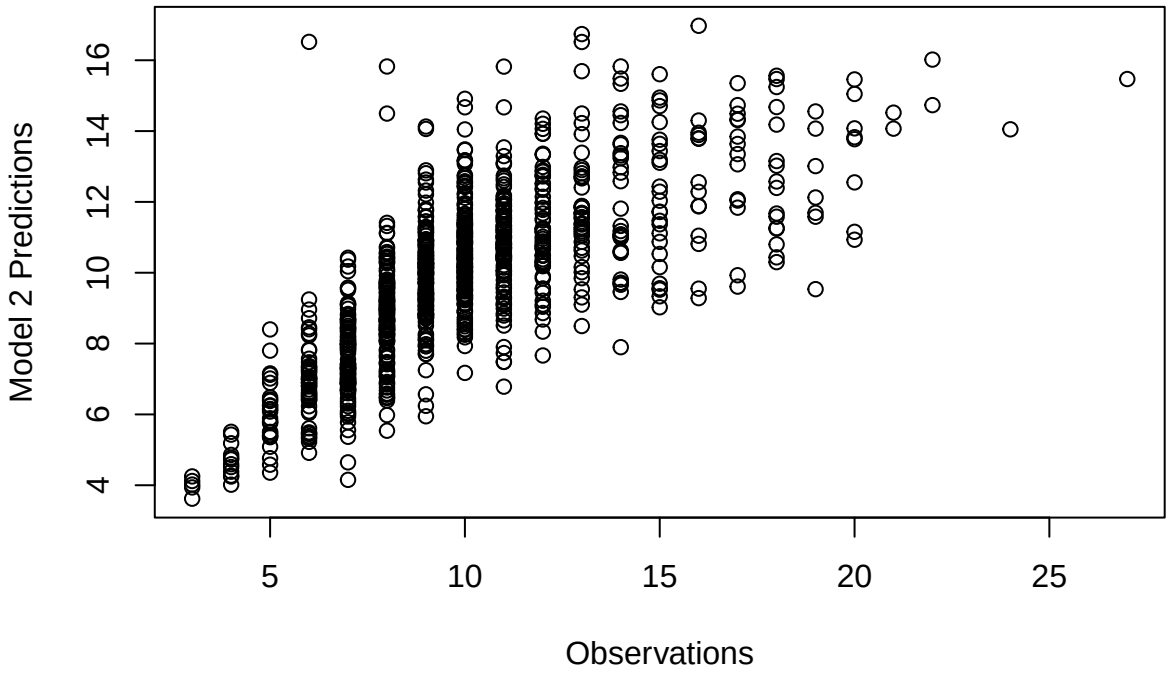
```
indexes <- caret::createDataPartition(snail$rings, p = 0.6, list = FALSE)
train <- snail[indexes, ]
test <- snail[-indexes, ]



model1 <- glm(rings ~ ., data = train)
model2 <- randomForest::randomForest(rings ~ ., data = train, ntree = 100)

model1_predict <- predict(model1, subset(test, select = -rings))
model2_predict <- predict(model2, subset(test, select = -rings))

plot(test$rings, model1_predict, xlab = "Observations", ylab = "Model 1 Predictions")
```




```
caret::postResample(model1_predict, test$rings)
```

```
##     RMSE Rsquared      MAE
## 2.185208 0.540513 1.558668
```

```
plot(test$rings, model2_predict, xlab = "Observations", ylab = "Model 2 Predictions")
```




```
caret::postResample(model2_predict, test$rings)
```

```
##      RMSE  Rsquared       MAE
## 2.1851327 0.5400217 1.5294121
```

Given only the plots, I would assume that model 1 performs better; however the postResample output shows

that model2 performs slightly beter (RMSE is lower, R^2 is higher).

Both models are underperforming, it is likely that there are many nonlinear effects or other variables needed to accurately predict the number of rings.

Problem 4

Packages for interactive plots

One can use **plotly** or **altair**. `Leaflet` can also be used for geospatial data.

What is dpylr for?

`dpylr` provides a grammar for the most common data manipulation tasks.

For example, it provides the `select` function which allows one to select variables based on their names.

What is leaflet for?

The `leaflet` R package provides an interface to the `leaflet` javascript visualization library for map visualization.

`leaflet` can be used to create interactive maps.

Here is a simple example from their documentation:

```
circles <- data.frame(
    lng = c(23.59, 34.95, 17.47),
    lat = c(-3.53, -6.32, -12.24)
)

leaflet() %>%
    addTiles() %>%
    addCircleMarkers(data = circles, color = "red")
```

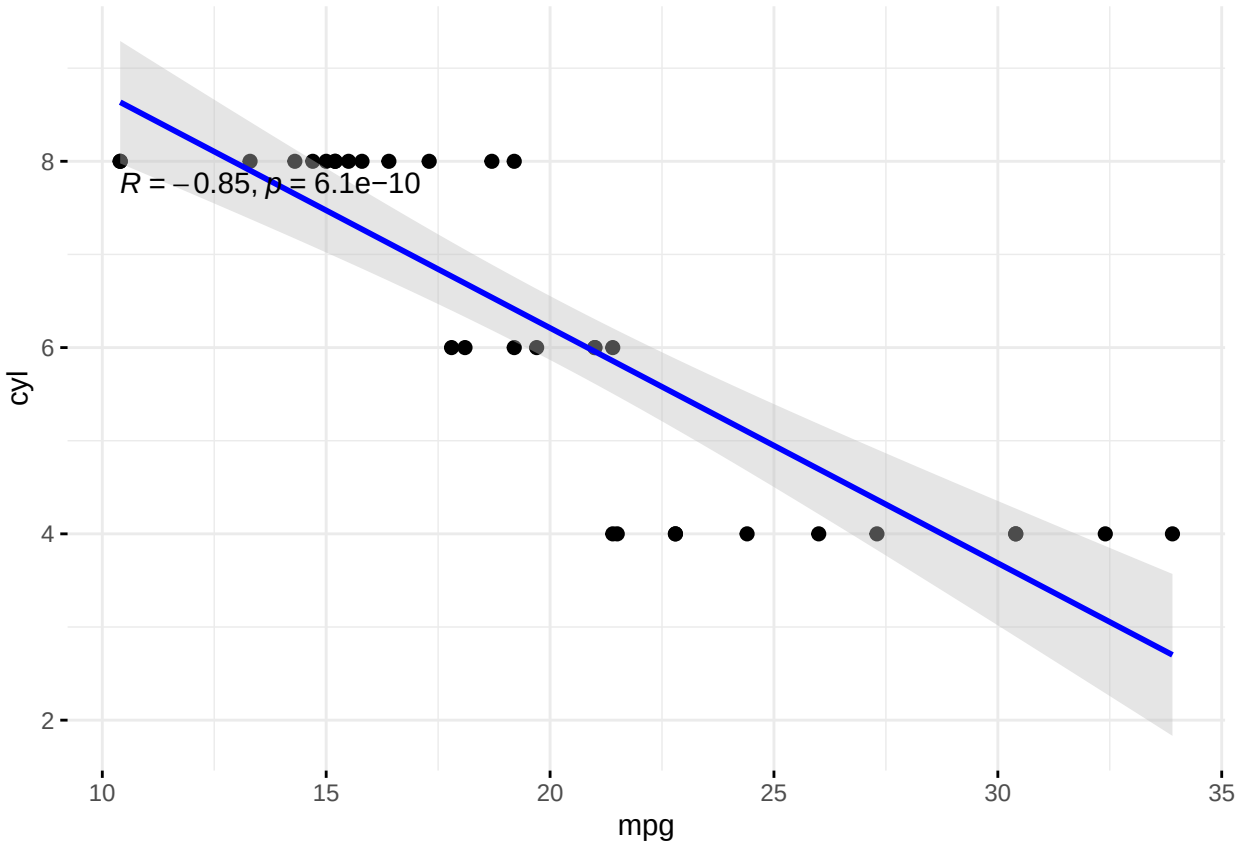
Examples of ggpubr

ggpubr can be used for publication ready plots. Here is an example of a linear regression plot:

```
library(ggpubr)
```

```
## Loading required package: ggplot2
```

```
ggscatter(mtcars,
    x = "mpg", y = "cyl",
    add = "reg.line",
    conf.int = TRUE,
    add.params = list(
        color = "blue",
        fill = "gray"
    ),
    cor.coef = TRUE, cor.method = "pearson",
    ggtheme = theme_minimal()
)
```



It can also be used for any ggplot2 related plot, such as histograms, correlation plots, etc.