Week 7 - Homework

STAT 420, Summer 2023, D. Unger

Scott Girten
NetID: sgirten2

Directions

Students are encouraged to work together on homework. However, sharing, copying or providing any part of a homework solution or code is an infraction of the University's rules on Academic Integrity. Any violation will be punished as severely as possible.

- Be sure to remove this section if you use this `.Rmd` file as a template.
- You may leave the questions in your final document.

Exercise 1 (EPA Emissions Data)

For this exercise, we will use the data stored in `epa2017.csv`. It contains detailed descriptions of vehicles manufactured in 2017 that were used for fuel economy testing as performed by the Environment Protection Agency. The variables in the dataset are:

- `Make` - Manufacturer
- `Model` - Model of vehicle
- `ID` - Manufacturer defined vehicle identification number within EPA's computer system (not a VIN number)
- `disp` - Cubic inch displacement of test vehicle
- `type` - Car, truck, or both (for vehicles that meet specifications of both car and truck, like smaller SUVs or crossovers)
- `horse` - Rated horsepower, in foot-pounds per second
- `cyl` - Number of cylinders
- `lockup` - Vehicle has transmission lockup; N or Y
- `drive` - Drivetrain system code
    - A = All-wheel drive
    - F = Front-wheel drive
    - P = Part-time 4-wheel drive
    - R = Rear-wheel drive
    - 4 = 4-wheel drive
- `weight` - Test weight, in pounds
- `axleratio` - Axle ratio
- `nvratio` - n/v ratio (engine speed versus vehicle speed at 50 mph)
- `THC` - Total hydrocarbons, in grams per mile (g/mi)

- CO - Carbon monoxide (a regulated pollutant), in g/mi
- CO2 - Carbon dioxide (the primary byproduct of all fossil fuel combustion), in g/mi
- mpg - Fuel economy, in miles per gallon

We will attempt to model CO2 using both horse and type. In practice, we would use many more predictors, but limiting ourselves to these two, one numeric and one factor, will allow us to create a number of plots.

Load the data, and check its structure using str(). Verify that type is a factor; if not, coerce it to be a factor.

```
library(tidyverse)

df = read_csv('epa2017.csv')
#str(df)

df = df %>%
  mutate(type = as.factor(type))
```

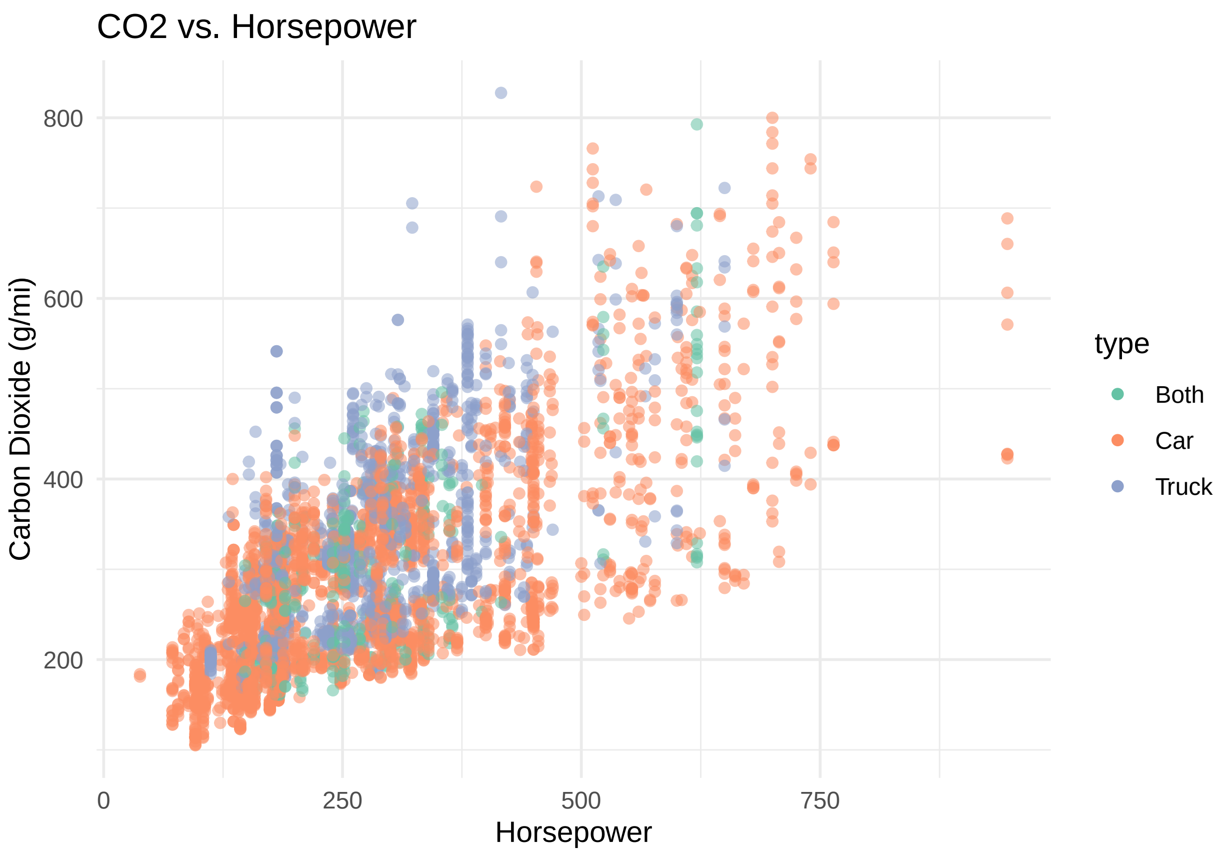
(a) Do the following:

- Make a scatterplot of CO2 versus horse. Use a different color point for each vehicle type.
- Fit a simple linear regression model with CO2 as the response and only horse as the predictor.
- Add the fitted regression line to the scatterplot. Comment on how well this line models the data.
- Give an estimate for the average change in CO2 for a one foot-pound per second increase in horse for a vehicle of type car.
- Give a 90% prediction interval using this model for the CO2 of a Subaru Impreza Wagon, which is a vehicle with 148 horsepower and is considered type Both. (Interestingly, the dataset gives the wrong drivetrain for most Subarus in this dataset, as they are almost all listed as F, when they are in fact all-wheel drive.)

Solution:

```
# Initial scatterplot
p = ggplot(df, aes(x = horse, y = CO2, color = type, alpha = 0.5)) +
  geom_point() +
  scale_color_manual(values = c('#66c2a5', '#fc8d62', '#8da0cb')) +
  theme_minimal() +
  labs(title = 'CO2 vs. Horsepower',
       x = 'Horsepower',
       y = 'Carbon Dioxide (g/mi)') +
  guides(alpha = FALSE)

p
```

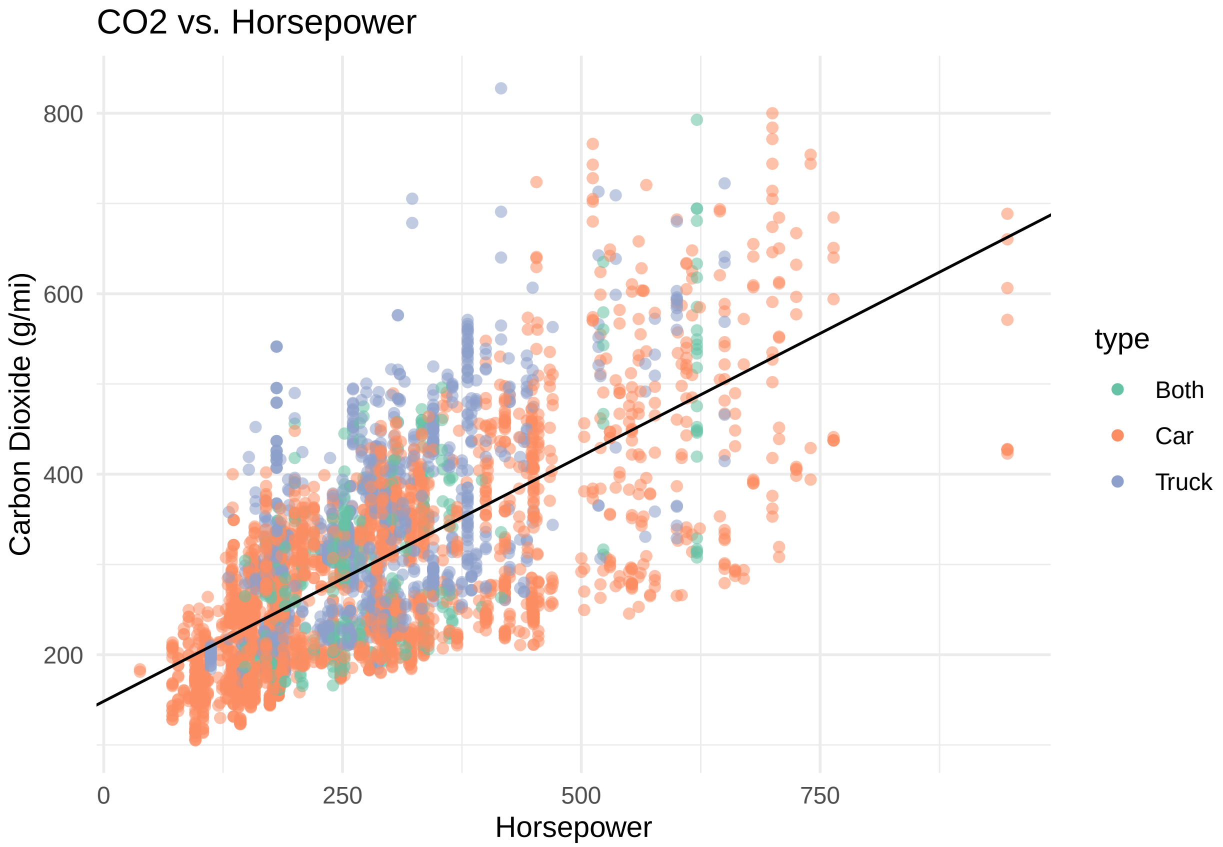


```
# Simple linear regression
mod1 = lm(CO2 ~ horse, data = df)

# Get slope and intercept
intercept = coef(mod1)[1]
slope = coef(mod1)[2]

# Add slope and intercept to initial scatter
p2 = p +
  geom_abline(slope = slope, intercept = intercept)

p2
```



The line from the regression model does fit the data, but there is a lot of variability in the data that one line has a difficult time modeling.

```r
# Coefficients for the first model
#coef(mod1)
change1 = coef(mod1)[2]
```

The average change in `CO2` for a one foot-pound per second increase in `horse` for a vehicle of type `car` is 0.5436.

```r
predict1 = predict(mod1, newdata = data.frame(horse = 148), interval = 'prediction', level = 0.90)
```

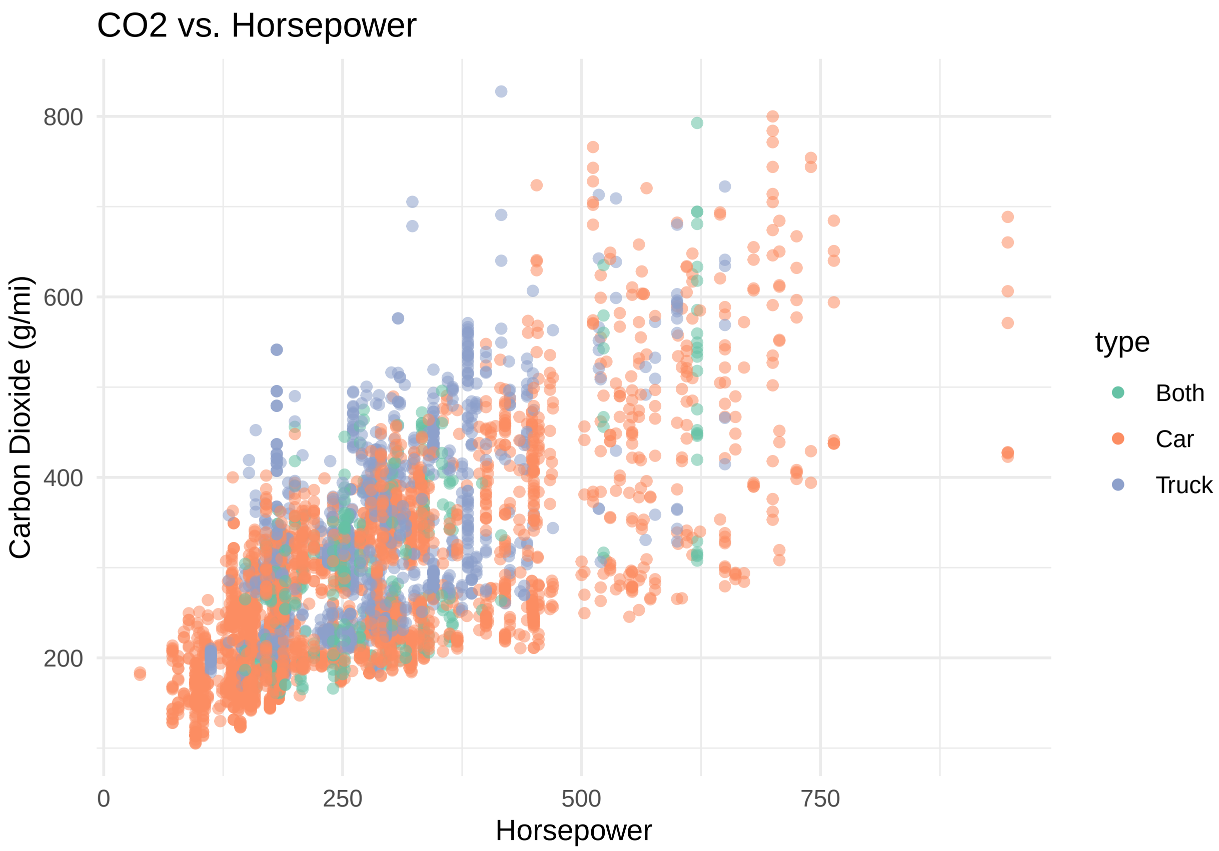
For a Subaru Impreza Wagon having a horsepower of 148 and categorized as `Both` a car and truck, the 90% prediction interval for the mean `CO2` emissions is (91.5033, 366.0446).

(b) Do the following:

- Make a scatterplot of `CO2` versus `horse`. Use a different color point for each vehicle `type`.
- Fit an additive multiple regression model with `CO2` as the response and `horse` and `type` as the predictors.
- Add the fitted regression “lines” to the scatterplot with the same colors as their respective points (one line for each vehicle type). Comment on how well this line models the data.
- Give an estimate for the average change in `CO2` for a one foot-pound per second increase in `horse` for a vehicle of type `car`.
- Give a 90% prediction interval using this model for the `CO2` of a Subaru Impreza Wagon, which is a vehicle with 148 horsepower and is considered type `Both`.

Solution:

```
# Re-use scatter plot from part A
p
```



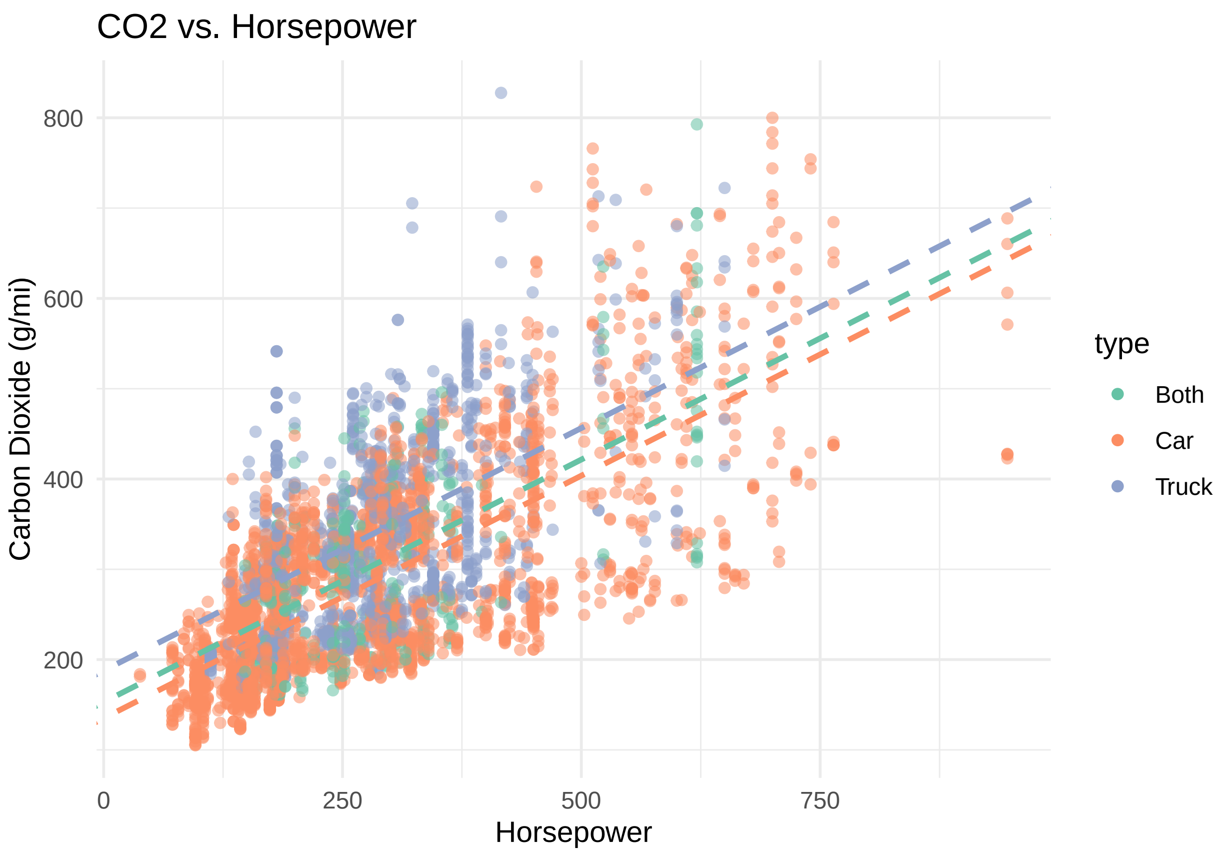
```
# Simple linear regression
mod2 = lm(CO2 ~ horse + type, data = df)

# Get slope and intercept
intercept_both = coef(mod2)[1]
intercept_car = coef(mod2)[1] + coef(mod2)[3]
intercept_truck = coef(mod2)[1] + coef(mod2)[4]

slope = coef(mod2)[2]

# Add slope and intercept to initial scatter plot
p3 = p +
  # Both
  geom_abline(slope = slope, intercept = intercept_both, color = '#66c2a5', linetype = 'dashed', size =
  # Car
  geom_abline(slope = slope, intercept = intercept_car, color = '#fc8d62', linetype = 'dashed', size =
  # Truck
  geom_abline(slope = slope, intercept = intercept_truck, color = '#8da0cb', linetype = 'dashed', size =


p3
```



The lines for each of the vehicle types model the data a little better than the previous model with did not account for vehicle types, but the lines still do not model the differences in variability of the vehicle types well.

```
#summary(mod2)

change2 = coef(mod2)[2] + coef(mod2)[3]
```

The average change in `CO2` for a one foot-pound per second increase in `horse` for a vehicle of type `car` is -17.3048.

```
predict2 = predict(mod2, newdata = data.frame(horse = 148, type = 'Both'), interval = 'prediction', leve
```

For a Subaru Impreza Wagon having a horsepower of 148 and categorized as `Both` a car and truck, the 90% prediction interval for the mean `CO2` emissions is (100.0012, 364.8952).

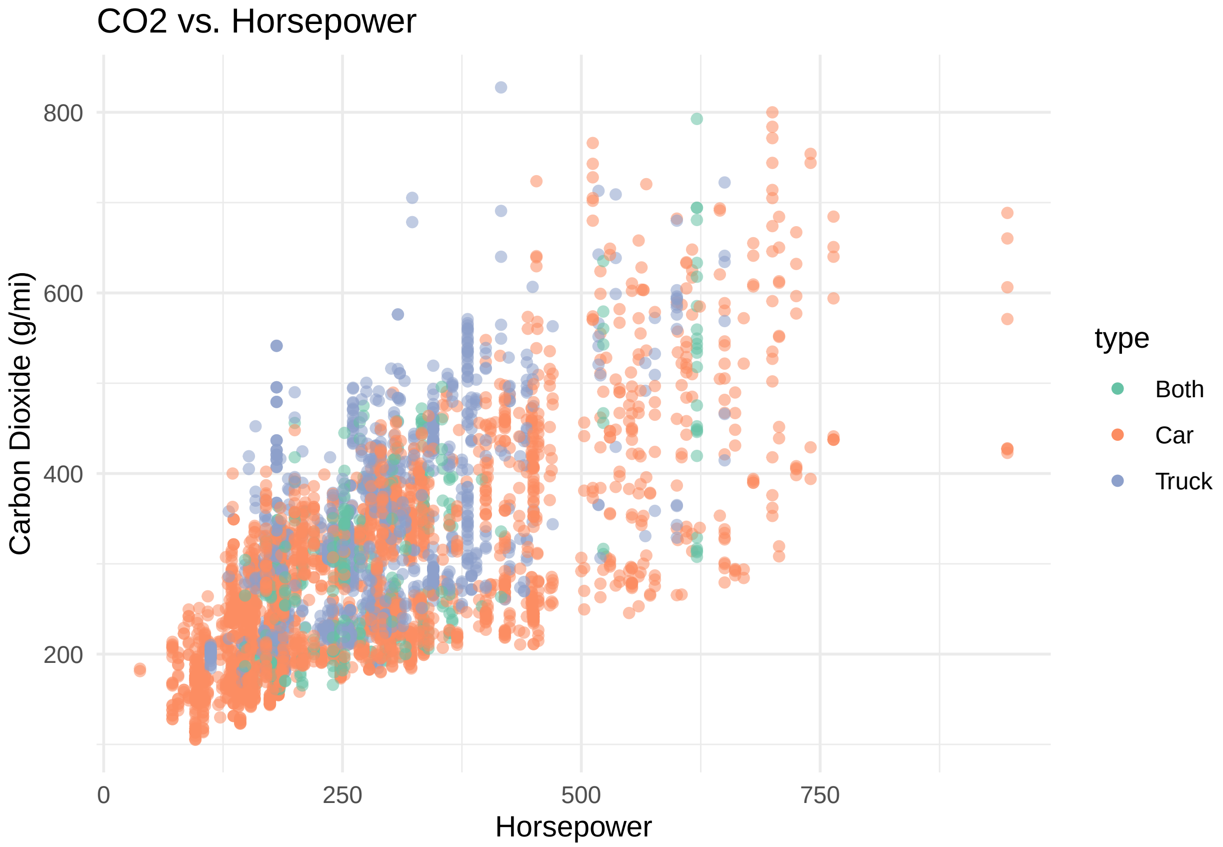
(c) Do the following:

- Make a scatterplot of `CO2` versus `horse`. Use a different color point for each vehicle `type`.
- Fit an interaction multiple regression model with `CO2` as the response and `horse` and `type` as the predictors.
- Add the fitted regression "lines" to the scatterplot with the same colors as their respective points (one line for each vehicle type). Comment on how well this line models the data.
- Give an estimate for the average change in `CO2` for a one foot-pound per second increase in `horse` for a vehicle of type `car`.

- Give a 90% prediction interval using this model for the `CO2` of a Subaru Impreza Wagon, which is a vehicle with 148 horsepower and is considered type `Both`.

Solution:

```r
# Re-use plot from part A
p
```



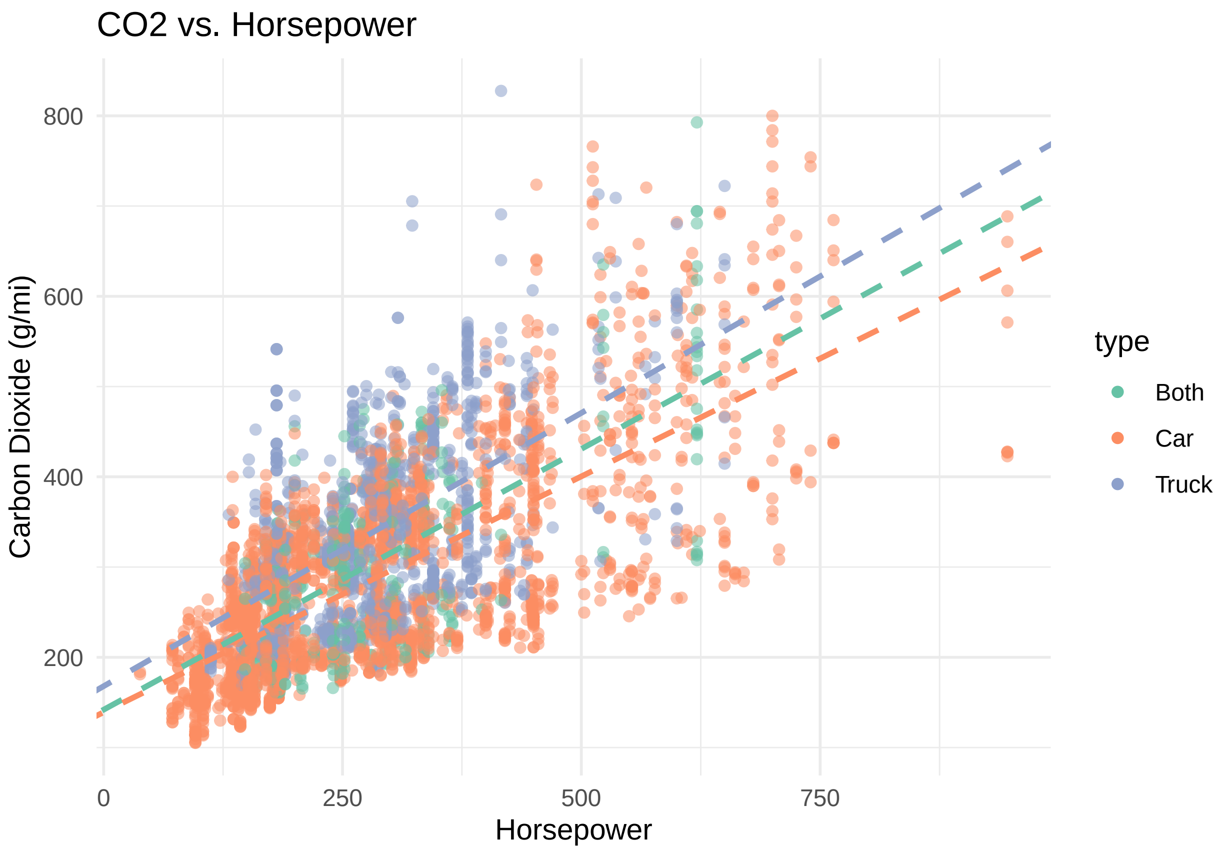
```r
# Interaction model
mod3 = lm(CO2 ~ horse + type + horse:type, data = df)

# Get slope and intercept
intercept_both3 = coef(mod3)[1]
intercept_car3 = coef(mod3)[1] + coef(mod3)[3]
intercept_truck3 = coef(mod3)[1] + coef(mod3)[4]

slope_both3 = coef(mod3)[2]
slope_car3 = coef(mod3)[2] + coef(mod3)[5]
slope_truck3 = coef(mod3)[2] + coef(mod3)[6]

# Add slope and intercept to initial scatter plot
p4 = p +
  # Both
  geom_abline(slope = slope_both3, intercept = intercept_both3, color = '#66c2a5', linetype = 'dashed',
  # Car
  geom_abline(slope = slope_car3, intercept = intercept_car3, color = '#fc8d62', linetype = 'dashed', s
  # Truck
  geom_abline(slope = slope_truck3, intercept = intercept_truck3, color = '#8da0cb', linetype = 'dashed


p4
```



The interaction model is a slight improvement over the additive model in terms of modeling `CO2` emissions, but the interaction terms did not drastically change the slope of the line modeling car and truck.

```
#summary(mod3)
change3 = coef(mod3)[2] + coef(mod3)[3] + coef(mod3)[5]
```

The average change in `CO2` for a one foot-pound per second increase in `horse` for a vehicle of type `car` is -2.2985

```
predict3 = predict(mod3, newdata = data.frame(horse = 148, type = 'Both'), interval = 'prediction', leve
```

For a Subaru Impreza Wagon having a horsepower of 148 and categorized as `Both` a car and truck, the 90% prediction interval for the mean `CO2` emissions is (95.0055, 359.9619).

(d) Based on the previous plots, you probably already have an opinion on the best model. Now use an ANOVA F-test to compare the additive and interaction models. Based on this test and a significance level of $\alpha = 0.10$, which model is preferred?

```r
anova_test = anova(mod2, mod3)

p_val = anova_test[2,6]
```

At a significance level of $\alpha = 0.10$, the p-value of 0.0059 for the F-test comparing the interaction model to the additive model would provide evidence there is a significant difference between the models and allow us to select the interaction model as the preferred model.

Exercise 2 (Hospital SUPPORT Data, White Blood Cells)

For this exercise, we will use the data stored in `hospital.csv`. It contains a random sample of 580 seriously ill hospitalized patients from a famous study called "SUPPORT" (Study to Understand Prognoses Preferences Outcomes and Risks of Treatment). As the name suggests, the purpose of the study was to determine what factors affected or predicted outcomes, such as how long a patient remained in the hospital. The variables in the dataset are:

- `Days` - Days to death or hospital discharge
- `Age` - Age on day of hospital admission
- `Sex` - Female or male
- `Comorbidity` - Patient diagnosed with more than one chronic disease
- `EdYears` - Years of education
- `Education` - Education level; high or low
- `Income` - Income level; high or low
- `Charges` - Hospital charges, in dollars
- `Care` - Level of care required; high or low
- `Race` - Non-white or white
- `Pressure` - Blood pressure, in mmHg
- `Blood` - White blood cell count, in gm/dL
- `Rate` - Heart rate, in bpm

For this exercise, we will use `Age`, `Education`, `Income`, and `Sex` in an attempt to model `Blood`. Essentially, we are attempting to model white blood cell count using only demographic information.

(a) Load the data, and check its structure using `str()`. Verify that `Education`, `Income`, and `Sex` are factors; if not, coerce them to be factors. What are the levels of `Education`, `Income`, and `Sex`?

Solution:

```r
library(kableExtra)

df = read_csv('hospital.csv')
#str(df)

# Convert Education, Income and Sex to factors
df = df %>%
  mutate(Sex = as.factor(Sex),
         Education = as.factor(Education),
         Income = as.factor(Income))

#str(df)
```

Table 1: Factor Levels for Sex, Income and Education Variables

Variable	Factor Levels
Sex	female, male
Income	high, low
Education	high, low

```r
# Get factor levels for each variable
levels_sex = str_flatten(levels(df$Sex), collapse = ', ')
levels_income = str_flatten(levels(df$Income), collapse = ', ')
levels_education = str_flatten(levels(df$Education), collapse = ', ')

# Create table for displaying factor levels
df_levels = tibble(Variable = c('Sex', 'Income', 'Education'),
                   `Factor Levels`=  c(levels_sex, levels_income, levels_education))

# Table display
df_levels %>%
  kbl(caption = 'Factor Levels for Sex, Income and Education Variables') %>%
  kable_styling()
```

(b) Fit an additive multiple regression model with `Blood` as the response using `Age`, `Education`, `Income`, and `Sex` as predictors. What does `R` choose as the reference level for `Education`, `Income`, and `Sex`?

Solution:

```r
mod = lm(Blood ~ Age + Education + Income + Sex, data = df)
#summary(mod)
```

For the reference levels, `R` chose *low* for Education, *low* for Income and *male* for Sex.

(c) Fit a multiple regression model with `Blood` as the response. Use the main effects of `Age`, `Education`, `Income`, and `Sex`, as well as the interaction of `Sex` with `Age` and the interaction of `Sex` and `Income`. Use a statistical test to compare this model to the additive model using a significance level of $\alpha = 0.10$. Which do you prefer?

Solution:

```r
# Interaction model
mod2 = lm(Blood ~ Age + Education + Income + Sex + Sex:Age + Sex:Income, data = df)

# ANOVA test and p-value
anova_test = anova(mod, mod2)
pval = anova_test[2,6]
```

At a significance level of $\alpha = 0.10$, I would prefer the additive model since the p-value for the F-test is 0.1128 and is greater than the significance level of the test and indicates that there is likely not enough of a significant difference in the models at this level of α.

(d) Fit a model similar to that in **(c)**, but additionally add the interaction between `Income` and `Age` as well as a three-way interaction between `Age`, `Income`, and `Sex`. Use a statistical test to compare this model to the preferred model from **(c)** using a significance level of $\alpha = 0.10$. Which do you prefer?

Solution:

```r
# Bigger interaction model
mod3 = lm(Blood ~ Age + Education + Income + Sex + Sex:Age + Sex:Income + Income:Age + Age:Income:Sex,

anova_test2 = anova(mod, mod3)
pval2 = anova_test2[2,6]
```

At a significance level of $\alpha = 0.10$, I would prefer the larger interaction model over the additive model since the p-value of the F-test is 0.0744 and is lower than the test threshold of $\alpha = 0.10$ and there is likely a significant difference in the interaction model performance versus the additive model performance.

(e) Using the model in **(d)**, give an estimate of the change in average `Blood` for a one-unit increase in `Age` for a highly educated, low income, male patient.

Solution:

```r
#summary(mod3)

mod_change = coef(mod3)[2] + coef(mod3)[4] + coef(mod3)[5] +coef(mod3)[6] + coef(mod3)[7] + coef(mod3)[
```

The estimated average change in `Blood` for a one-unit increase in `Age` for a highly educated, low income and male patient is -2.9981.

Exercise 3 (Hospital SUPPORT Data, Stay Duration)

For this exercise, we will again use the data stored in `hospital.csv`. It contains a random sample of 580 seriously ill hospitalized patients from a famous study called "SUPPORT" (Study to Understand Prognoses Preferences Outcomes and Risks of Treatment). As the name suggests, the purpose of the study was to determine what factors affected or predicted outcomes, such as how long a patient remained in the hospital. The variables in the dataset are:

- `Days` - Days to death or hospital discharge
- `Age` - Age on day of hospital admission
- `Sex` - Female or male
- `Comorbidity` - Patient diagnosed with more than one chronic disease
- `EdYears` - Years of education
- `Education` - Education level; high or low
- `Income` - Income level; high or low
- `Charges` - Hospital charges, in dollars
- `Care` - Level of care required; high or low
- `Race` - Non-white or white
- `Pressure` - Blood pressure, in mmHg
- `Blood` - White blood cell count, in gm/dL
- `Rate` - Heart rate, in bpm

For this exercise, we will use `Blood`, `Pressure`, and `Rate` in an attempt to model `Days`. Essentially, we are attempting to model the time spent in the hospital using only health metrics measured at the hospital.

Consider the model

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_1 x_2 + \beta_5 x_1 x_3 + \beta_6 x_2 x_3 + \beta_7 x_1 x_2 x_3 + \epsilon,$$

where

- Y is `Days`
- x_1 is `Blood`
- x_2 is `Pressure`
- x_3 is `Rate`.

(a) Fit the model above. Also fit a smaller model using the provided `R` code.

```
hospital = read_csv('hospital.csv')

days_add = lm(Days ~ Pressure + Blood + Rate, data = hospital)
```

Use a statistical test to compare the two models. Report the following:

- The null and alternative hypotheses in terms of the model given in the exercise description
- The value of the test statistic
- The p-value of the test
- A statistical decision using a significance level of $\alpha = 0.10$
- Which model you prefer

Solution:

```
# Full interaction model
days_int = lm(Days ~ Pressure * Blood * Rate, data = hospital)

# Test full interaction model vs. additive model that is provided
anova_test = anova(days_add, days_int)
#anova_test

f_stat = anova_test[2,5]
p_val = anova_test[2,6]
```

$H_0 : \beta_4 = \beta_5 = \beta_6 = \beta_7 = 0$
H_1 : At least one of $\beta_4 ... \beta_7 \neq 0$

F-statistic: 2.0426

p-value: 0.087

Significance Level: $\alpha = 0.10$

Statistical Decision: Reject the *Null* hypothesis.

For an α of 0.10, I would prefer the full interaction model since the p-value for the F-test suggests that it is likely there is a significant difference in the performance of the full interacton model relative to the additive model.

(b) Give an expression based on the model in the exercise description for the true change in length of hospital stay in days for a 1 bpm increase in `Rate` for a patient with a `Pressure` of 139 mmHg and a `Blood` of 10 gm/dL. Your answer should be a linear function of the βs.

Solution:

```
#summary(days_int)
```

(-0.3339 * 139) + (-0.8111 * 10) + -0.1609 + (0.0125 * 139 * 10) + (0.0037 * 139) + (0.0071 * 10) + $(-9.2506 \times 10^{-5}$ * 139 * 10)

(c) Give an expression based on the additive model in part **(a)** for the true change in length of hospital stay in days for a 1 bpm increase in `Rate` for a patient with a `Pressure` of 139 mmHg and a `Blood` of 10 gm/dL. Your answer should be a linear function of the βs.

Solution:

```
#summary(days_add)
```

(0.0801 * 139) + (0.2096 * 10) + 0.1337

Exercise 4 (t-test Is a Linear Model)

In this exercise, we will try to convince ourselves that a two-sample t-test assuming equal variance is the same as a t-test for the coefficient in front of a single two-level factor variable (dummy variable) in a linear model.

First, we set up the data frame that we will use throughout.

```
n = 30

sim_data = data.frame(
  groups = c(rep("A", n / 2), rep("B", n / 2)),
  values = rep(0, n))
str(sim_data)
```

```
## 'data.frame':    30 obs. of  2 variables:
##  $ groups: chr  "A" "A" "A" "A" ...
##  $ values: num  0 0 0 0 0 0 0 0 0 0 ...
```

We will use a total sample size of `30`, `15` for each group. The `groups` variable splits the data into two groups, `A` and `B`, which will be the grouping variable for the t-test and a factor variable in a regression. The `values` variable will store simulated data.

We will repeat the following process a number of times.

```
set.seed(20)
sim_data$values = rnorm(n, mean = 42, sd = 3.5) # simulate response data
summary(lm(values ~ groups, data = sim_data))
```

```
##
## Call:
## lm(formula = values ~ groups, data = sim_data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -9.04  -1.11  -0.14   2.23   7.33
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   40.922      0.950   43.07   <2e-16 ***
## groupsB        0.029      1.344    0.02     0.98
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.68 on 28 degrees of freedom
## Multiple R-squared:  1.66e-05,   Adjusted R-squared:  -0.0357
## F-statistic: 0.000465 on 1 and 28 DF,  p-value: 0.983
```

```
t.test(values ~ groups, data = sim_data, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  values by groups
## t = -0.022, df = 28, p-value = 1
## alternative hypothesis: true difference in means between group A and group B is not equal to 0
## 95 percent confidence interval:
##  -2.781  2.723
## sample estimates:
## mean in group A mean in group B
##           40.92           40.95
```

We use `lm()` to test

$$H_0 : \beta_1 = 0$$

for the model

$$Y = \beta_0 + \beta_1 x_1 + \epsilon$$

where Y is the values of interest, and x_1 is a dummy variable that splits the data in two. We will let `R` take care of the dummy variable.

We use `t.test()` to test

$$H_0 : \mu_A = \mu_B$$

where μ_A is the mean for the `A` group, and μ_B is the mean for the `B` group.

The following code sets up some variables for storage.

```
num_sims = 300
lm_t = rep(0, num_sims)
lm_p = rep(0, num_sims)
tt_t = rep(0, num_sims)
tt_p = rep(0, num_sims)
```

- `lm_t` will store the test statistic for the test $H_0 : \beta_1 = 0$.
- `lm_p` will store the p-value for the test $H_0 : \beta_1 = 0$.
- `tt_t` will store the test statistic for the test $H_0 : \mu_A = \mu_B$.

- tt_p will store the p-value for the test $H_0 : \mu_A = \mu_B$.

The variable num_sims controls how many times we will repeat this process, which we have chosen to be 300.

(a) Set a seed equal to your birthday. (Month and day are sufficient without year.) Then write code that repeats the above process 300 times. Each time, store the appropriate values in lm_t, lm_p, tt_t, and tt_p. Specifically, each time you should use sim_data$values = rnorm(n, mean = 42, sd = 3.5) to update the data. The grouping will always stay the same.

Solution:

```
set.seed(0221)

n = 30
num_sims = 300
lm_t = rep(0, num_sims)
lm_p = rep(0, num_sims)
tt_t = rep(0, num_sims)
tt_p = rep(0, num_sims)


for(i in 1:num_sims){

  # Simulate data
  sim_data = data.frame(
  groups = c(rep("A", n / 2), rep("B", n / 2)),
  values =  rnorm(n, mean = 42, sd = 3.5))

  # Run regression model and t-test
  lm = summary(lm(values ~ groups, data = sim_data))
  tt = t.test(values ~ groups, data = sim_data, var.equal = TRUE)

  # Add results to storage vector
  lm_t[i] = lm$coefficients[2,3]
  lm_p[i] = lm$coefficients[2,4]

  tt_t[i] = tt$statistic
  tt_p[i] = tt$p.value
}
```

(b) Report the value obtained by running mean(lm_t == tt_t), which tells us what proportion of the test statistics is equal. The result may be extremely surprising!

Solution:

```
mean(lm_t == tt_t)
```

```
## [1] 0
```

(c) Report the value obtained by running mean(lm_p == tt_p), which tells us what proportion of the p-values is equal. The result may be extremely surprising!

Solution:

```
mean(lm_p == tt_p)
```

```
## [1] 0.03
```

(d) If you have done everything correctly so far, your answers to the last two parts won't indicate the equivalence we want to show! What the heck is going on here? The first issue is one of using a computer to do calculations. When a computer checks for equality, it demands **equality**; nothing can be different. However, when a computer performs calculations, it can only do so with a certain level of precision. So, if we calculate two quantities we know to be analytically equal, they can differ numerically. Instead of `mean(lm_p == tt_p)` run `all.equal(lm_p, tt_p)`. This will perform a similar calculation, but with a very small error tolerance for each equality. What is the result of running this code? What does it mean?

Solution:

```
all.equal(lm_p, tt_p)
```

```
## [1] TRUE
```

Using the `all.equal()` function shows the values in the vectors `lm_p` and `tt_p` are equal after accounting for very small errors in the precision of the values in each vector.

(e) Your answer in **(d)** should now make much more sense. Then what is going on with the test statistics? Look at the values stored in `lm_t` and `tt_t`. What do you notice? Is there a relationship between the two? Can you explain why this is happening?

Solution:

```
df_stat = tibble(lm_t, tt_t)

head(df_stat, n = 10)
```

```
## # A tibble: 10 x 2
##      lm_t   tt_t
##     <dbl>  <dbl>
##  1  1.21  -1.21
##  2  1.72  -1.72
##  3 -2.40   2.40
##  4  0.404 -0.404
##  5  0.431 -0.431
##  6  0.691 -0.691
##  7 -1.10   1.10
##  8 -1.32   1.32
##  9 -0.553  0.553
## 10  1.48  -1.48
```

The values in each vector have the same numeric value with the +/− signs reversed for each vector. I would assume this is happening because `lm()` and `t.test()` are using different approaches to setting the reference level for comparing groups A and B. The absolute value of the difference for each algorithm is the same, but setting either A or B as the reference level will cause the sign of the difference to change.