Session 4 - ggplot2

Alex Mounsey

07/10/2020

Exercise 1

Consider again the data collected by a farm management company from nine farms in Devon about the number of cattle and sheep. It turns out that other information is available, including the farm location and whether the opportunity to participate in a working farm holiday is offered:

$$Cattle = \begin{pmatrix} 348 \\ 407 \\ 1064 \\ 750 \\ 593 \\ 1867 \\ 471 \\ 935 \\ 1443 \end{pmatrix} \qquad Sheep = \begin{pmatrix} 110 \\ 179 \\ 303 \\ 173 \\ 182 \\ 458 \\ 151 \\ 140 \\ 222 \end{pmatrix} \qquad Location = \begin{pmatrix} North \\ South \\ South \\ North \\ South \\ North \\ North \\ North \\ South \end{pmatrix} \qquad Holidays = \begin{pmatrix} No \\ No \\ Yes \\ Yes \\ No \\ Yes \\ No \\ No \\ Yes \end{pmatrix}$$

```r
cattle <- c(348, 407, 1064, 750, 593, 1867, 471, 935, 1443)
sheep <- c(110, 179, 303, 173, 182, 458, 151, 140, 222)
location <- c(’North’, ’South’, ’South’, ’North’, ’South’, ’North’, ’North’,
              ’North’, ’South’)
holidays <- c(’No’, ’No’, ’Yes’, ’Yes’, ’No’, ’Yes’, ’No’, ’No’, ’Yes’)
```

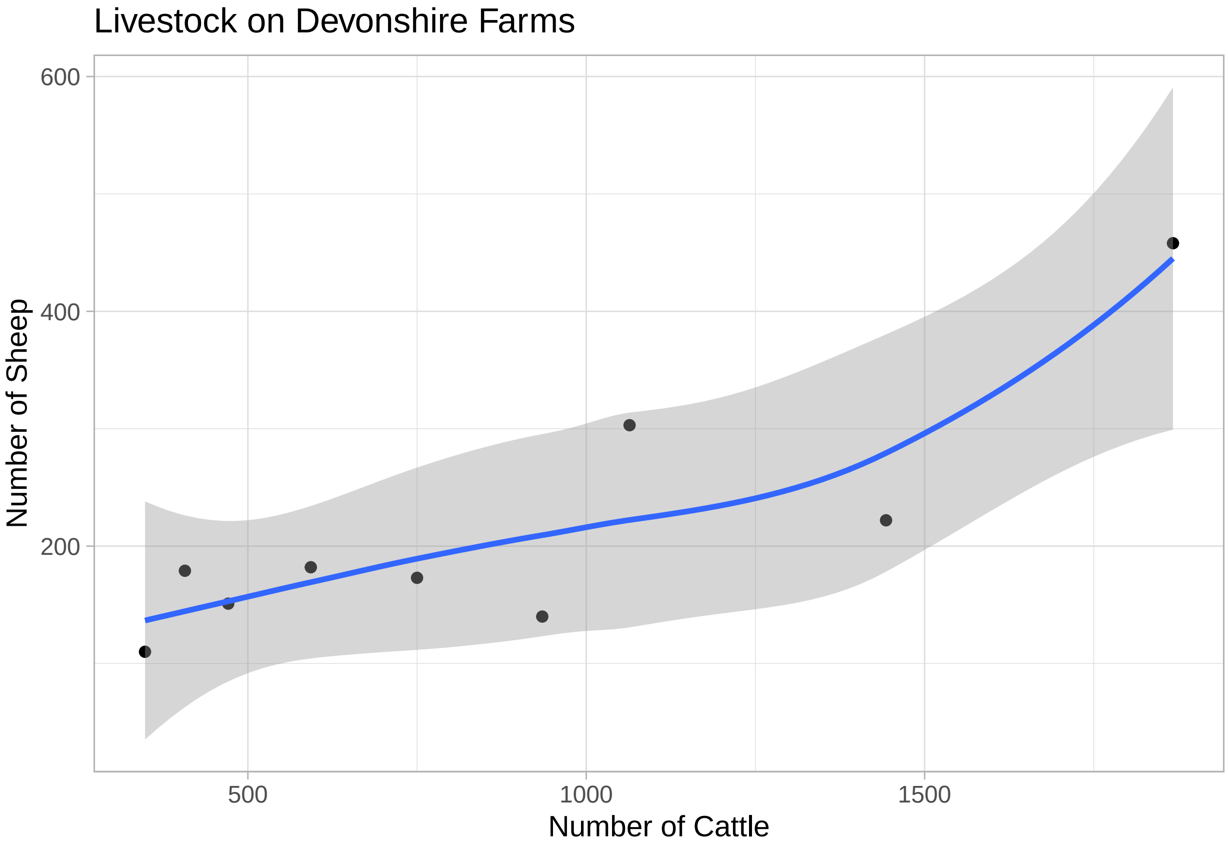
Define categorical variables `location` and `holidays` as factors and create a dataframe including these newly defined variables, as well as the `cattle` and `sheep` variables that you’ve worked with before:

```r
location_f <- factor(location, levels = c(’North’, ’East’, ’South’, ’West’))
holidays_f <- factor(holidays, levels = c(’Yes’, ’No’))

farms_df <- data.frame(cattle, sheep, location_f, holidays_f) %>%
  rename(location = location_f, holidays = holidays_f)

head(farms_df)
```

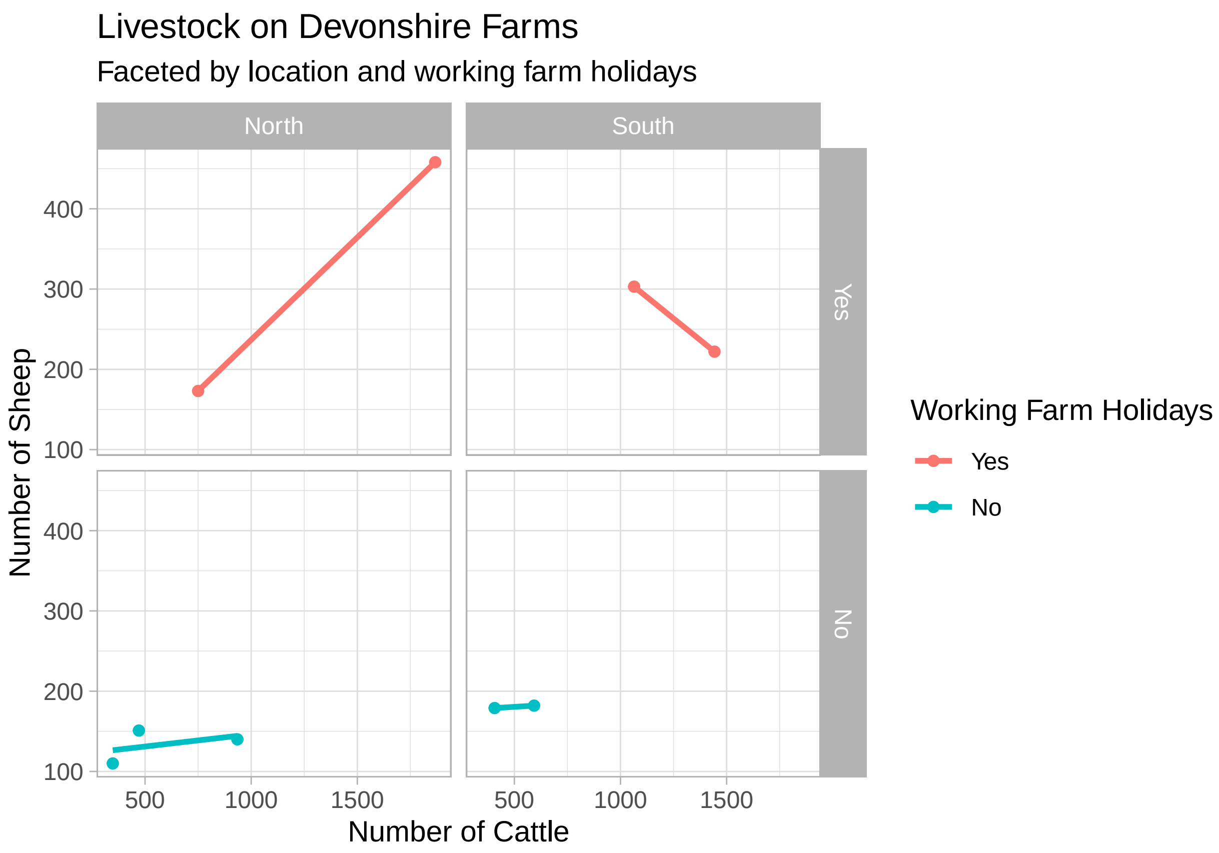
```
##   cattle sheep location holidays
## 1    348   110    North       No
## 2    407   179    South       No
## 3   1064   303    South      Yes
## 4    750   173    North      Yes
## 5    593   182    South       No
## 6   1867   458    North      Yes
```

Reproduce the plots shown in the tutorial file:

```
ggplot(farms_df, aes(x = cattle, y = sheep)) +
  theme_light() + geom_point() + geom_smooth(span = 1) +
  labs(x = "Number of Cattle", y = "Number of Sheep",
       title = "Livestock on Devonshire Farms")
```



```
ggplot(farms_df, aes(x = cattle, y = sheep, colour = holidays)) +
  theme_light() + geom_point() + geom_smooth(method = 'lm', se = F) +
  facet_grid(holidays ~ location) +
  labs(x = "Number of Cattle", y = "Number of Sheep",
       colour = "Working Farm Holidays",
       title = "Livestock on Devonshire Farms",
       subtitle = "Faceted by location and working farm holidays")
```



Exercise 2

Read the questionnaire data into a dataframe, `df`, in the usual way:

```
q_data <- read_csv('../data/MATH513_Questionnaire_Data.csv')
head(q_data)
```

```
## # A tibble: 6 x 19
##   Height   Age Sex   BirthPlace SiblingsNo EatMeat DrinkCoffee LikeBeer Sports
##    <dbl> <dbl> <chr> <chr>           <dbl> <chr>   <chr>       <chr>    <chr>
## 1    170  23   Fema~ essex               1 Yes     Yes         No       Yes
## 2    188  22.4 Male  London              1 Yes     Yes         No       No
## 3    180  30.1 Male  Athens              0 Yes     Yes         Yes      Yes
## 4    185  21   Male  China               0 Yes     Yes         Yes      Yes
## 5    170  22.1 Fema~ Plymouth            2 Yes     Yes         No       No
## 6    182  25   Male  Nigeria             4 Yes     No          No       Yes
## # ... with 10 more variables: Driver <chr>, LeftHanded <chr>, Abroad <chr>,
## #   Sleep <dbl>, Rent <dbl>, Happy_accommodation <chr>, Distance <dbl>,
## #   Travel_time <dbl>, Mode_of_transport <chr>, Safe <chr>
```

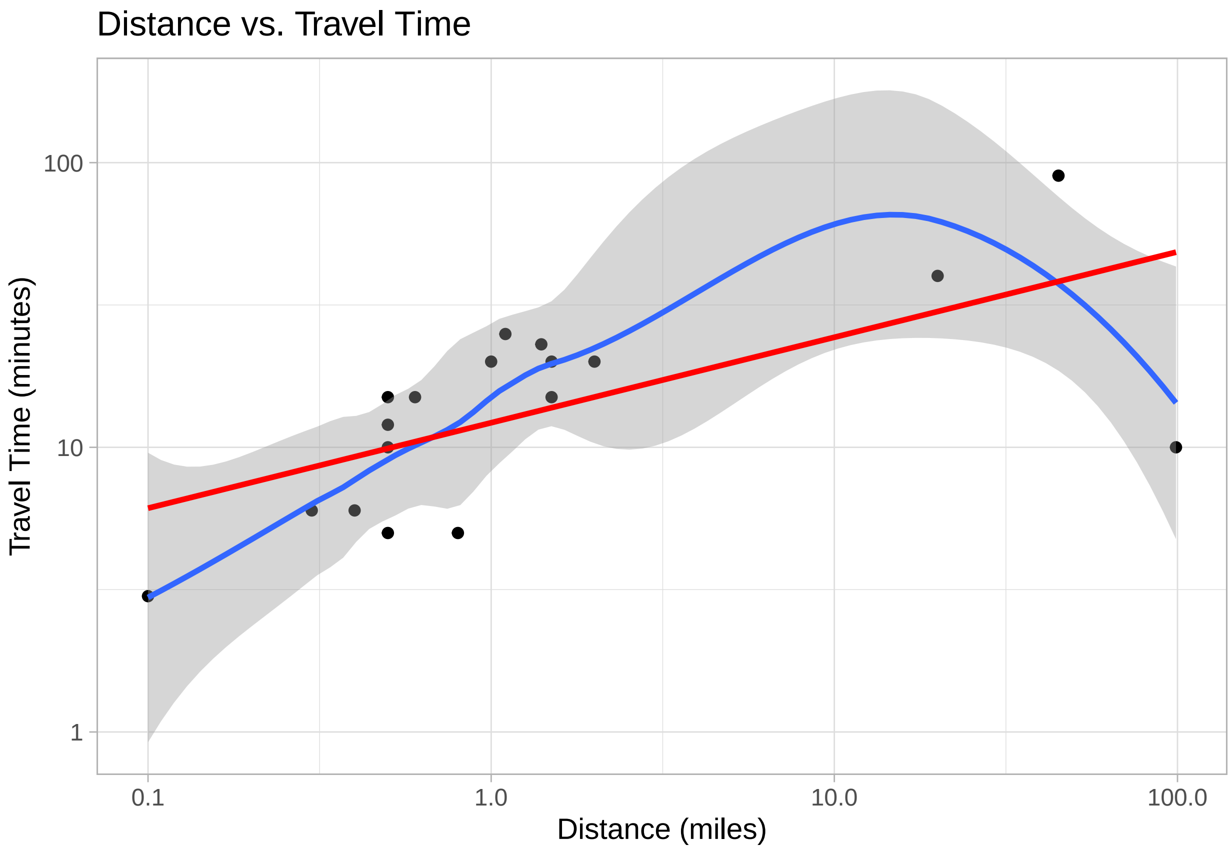
Using the `dplyr` package, select and display the columns *(variables)* `Travel_time` and `Distance`:

```r
q_data %>%
  select(Travel_time, Distance) %>%
  head()
```

```
## # A tibble: 6 x 2
##   Travel_time Distance
##         <dbl>    <dbl>
## 1          20      1
## 2           5      0.8
## 3           3      0.1
## 4          20      2
## 5          40     20
## 6          15      0.6
```

Your goal is to understand how `Travel_time` is dependent on `Distance`. Use `ggplot2` to plot `Travel_time` against `Distance`:

```r
ggplot(q_data, aes(x = Distance, y = Travel_time)) +
  theme_light() +
  geom_point() + geom_smooth() + geom_smooth(method = ’lm’, se = F, colour = ’red’) +
  scale_x_log10() + scale_y_log10() +
  labs(x = "Distance (miles)", y = "Travel Time (minutes)",
       title = "Distance vs. Travel Time")
```



Logarithmic Axis Scales

The logarithmic scale is used in situations where data points range between extremely large values and extremely small values, such that the detail of the smaller values is not lost.

Regression Line

The general mathematical equation of a straight line is:

$$y = \alpha + \beta x$$

where: α is known as the intercept and β is known as the slope.

When we're handling data, we can't expect the data points to lie perfectly on the line, so we allow an error. The simple linear regression model therefore takes the form:

$$y = \alpha + \beta x + error$$

Exercise 3

The file `companies.xlsx` contains information collected from 100 companies belonging to different sectors of the economy. The following variables have been recorded:

- `company`: The company's ID
- `net_income_2015`: The company's net income in 2015
- `net_income_2014`: The company's net income in 2014
- `oper_result`: The company's operational result in 2015
- `lab_cost`: The company's labor expenditure in 2015
- `n_empl`: The number of employees in the company
- `sector`: The economic sector the company operates within

```r
c_data <- read_excel('../data/companies.xlsx')
head(c_data)
```

```
## # A tibble: 6 x 7
##   company net_income_2015 net_income_2014 oper_result lab_cost n_empl sector
##     <dbl>           <dbl>           <dbl>       <dbl>    <dbl>  <dbl>  <dbl>
## 1       1        23109600        24324700     -277747  4859414 103000      4
## 2       2        20083509        17384211     2861590  8370166 109860      4
## 3       3        19453500        16665000     2747100  5288500  89475      4
## 4       4        14308944        14144552      256078   286478   4022      2
## 5       5        11243950         9556867     2025679   391221   6415      3
## 6       6         9350460         9061480    -4356350 10739080 179311      4
```

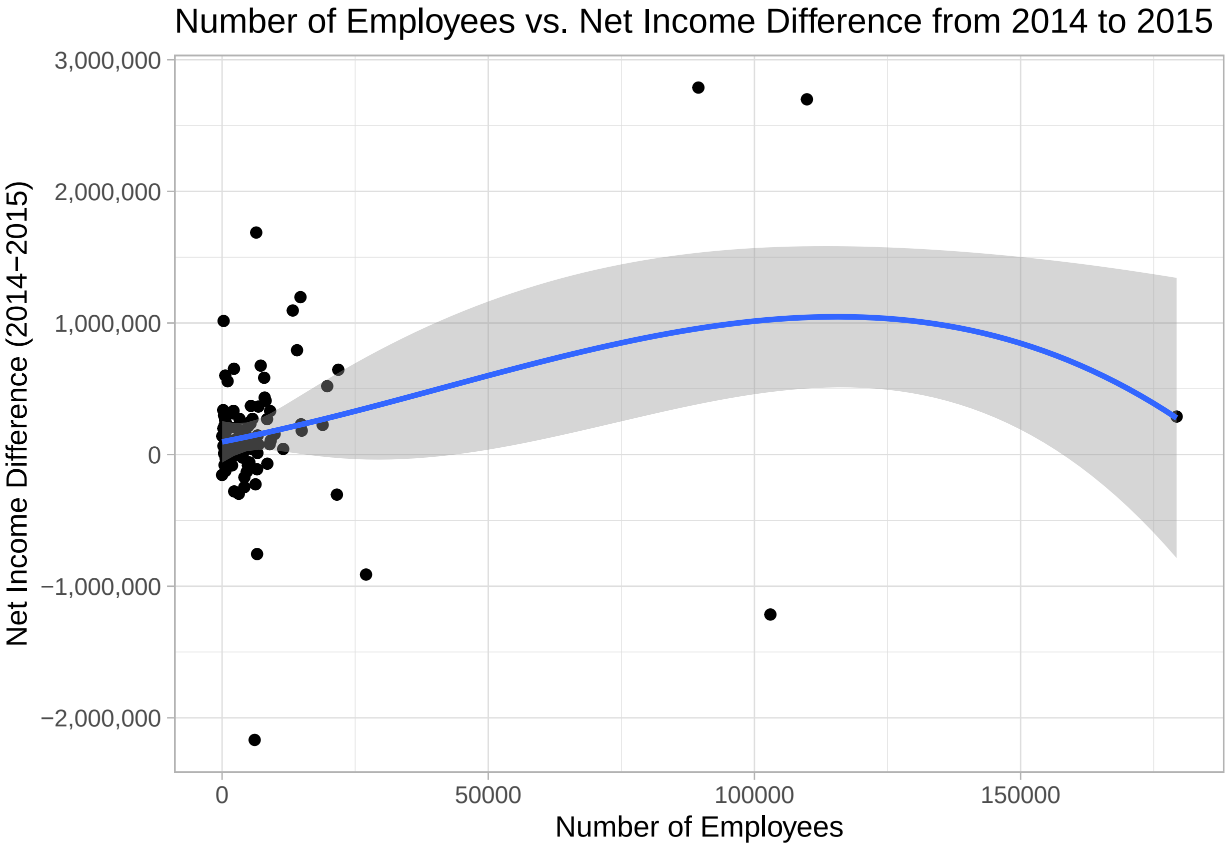
Create a new variable, `net_income_diff`, as the difference between the 2015 net income nad the 2014 net income:

```r
c_data <- c_data %>%
  mutate(net_income_diff = net_income_2015 - net_income_2014)

head(c_data$net_income_diff)
```

```
## [1] -1215100  2699298  2788500   164392  1687083   288980
```

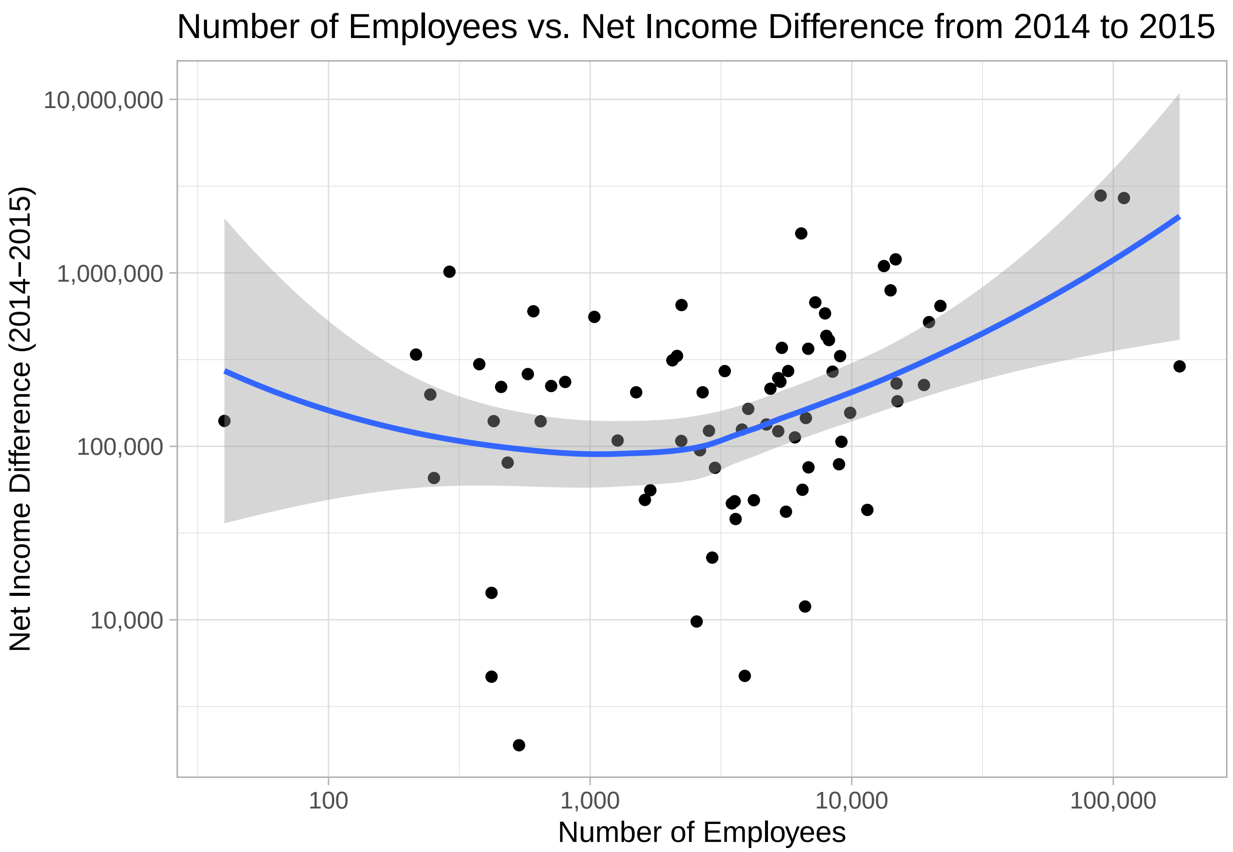
**Produce a scatterplot of the net income difference *(on the vertical axis)* against the number of employees *(on the horizontal axis)*. Add a smooth curve to your scatter plot.

```
ggplot(c_data, aes(x = n_empl, y = net_income_diff)) +
  theme_light() + geom_point() + geom_smooth(span = 1) +
  scale_y_continuous(label = comma) +
  labs(x = "Number of Employees", y = "Net Income Difference (2014-2015)",
       title = "Number of Employees vs. Net Income Difference from 2014 to 2015")
```



Using only the companies with a positive number of employees and a positive net income difference, modify the above plot to use a logarithmic scale on both axis:

```
filter(c_data, n_empl > 0 & net_income_diff > 0) %>%
  ggplot(aes(x = n_empl, y = net_income_diff)) +
    theme_light() + geom_point() + geom_smooth(span = 1) +
    scale_x_log10(label = comma) + scale_y_log10(label = comma) +
    labs(x = "Number of Employees", y = "Net Income Difference (2014-2015)",
         title = "Number of Employees vs. Net Income Difference from 2014 to 2015")
```



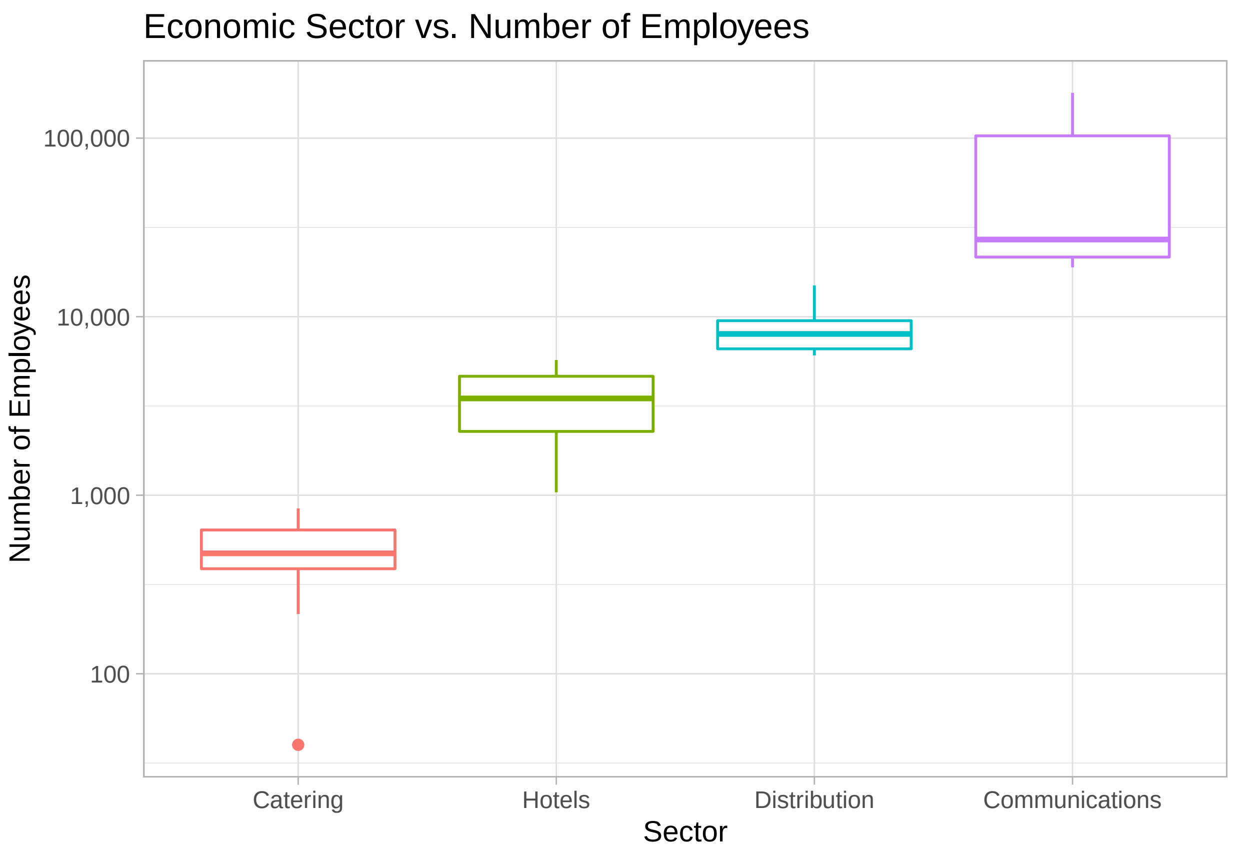
Using all of the companies, create a factor, `sector_new`, based on the variable `sector`, such that the levels of the factor are labelled `catering`, *(corresponding to 1)*, `hotels` *(corresponding to 2)*, `distribution` *(corresponding to 3)*, and `communications` *(corresponding to 4)*:

```
c_data <- c_data %>%
  mutate(sector_new = factor(sector, levels = 1:4,
                             labels = c(’Catering’, ’Hotels’, ’Distribution’,
                                        ’Communications’)))

head(c_data$sector_new)
```

```
## [1] Communications Communications Communications Hotels         Distribution
## [6] Communications
## Levels: Catering Hotels Distribution Communications
```

Using all of the companies, produce a boxplot of `n_empl` against `sector_new`:

```
ggplot(c_data, aes(x = sector_new, y = n_empl, colour = sector_new)) +
  theme_light() + geom_boxplot() +
  scale_y_log10(label = comma) +
  labs(x = "Sector", y = "Number of Employees",
       title = "Economic Sector vs. Number of Employees") +
  theme(legend.position = ’none’)
```



Using all of the companies, convert the continuous variable, `oper_result`, *(which has units of GBP)* into the continous variable `oper_result_millions` *(with units of millions of GBP)*:

```
c_data <- c_data %>%
  mutate(oper_result_millions = oper_result / 1000000)

head(c_data$oper_result_millions)
```

```
## [1] -0.277747  2.861590  2.747100  0.256078  2.025679 -4.356350
```

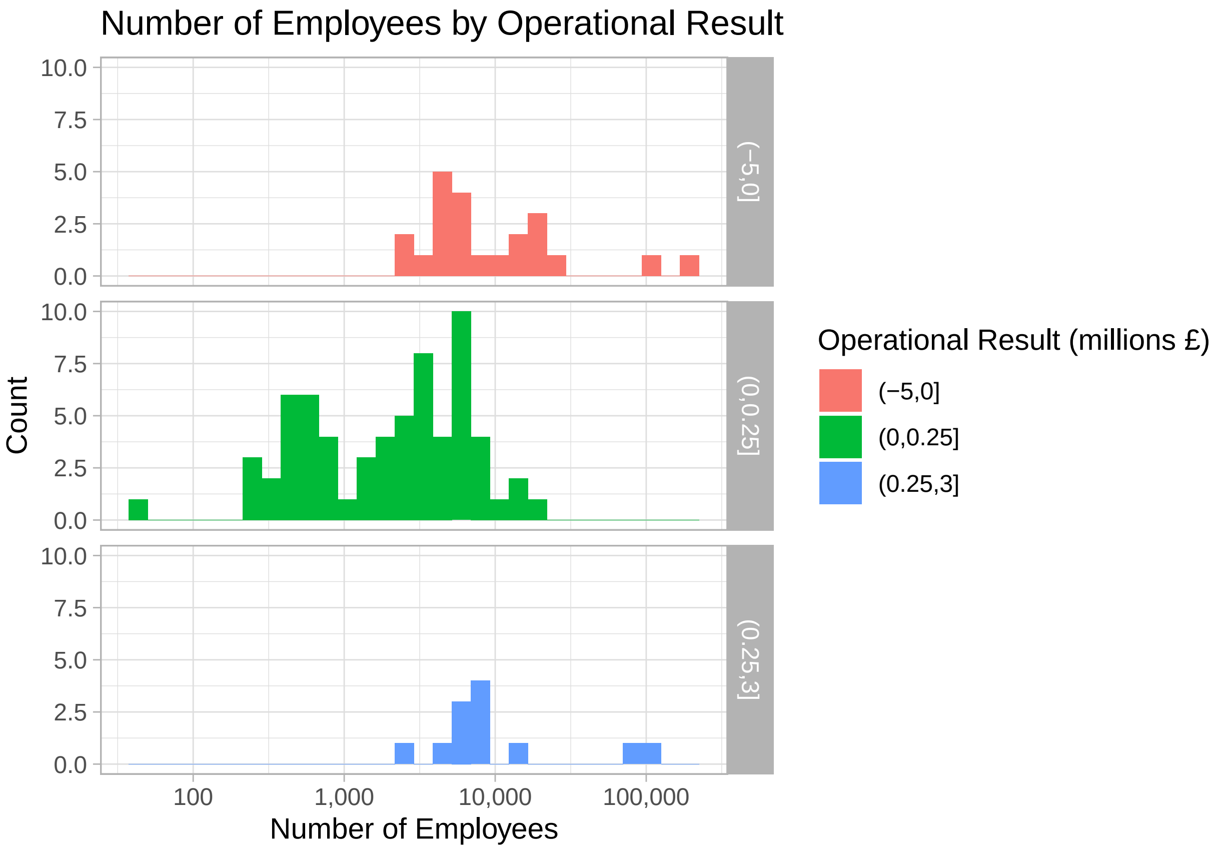
Using all of the companies, convert the continuous variable `oper_result_millions` into the factor `oper_result_new`, with breaks at -5, 0, 0.25, and 3, in such a way that the corresponding intervals are |-5, 0), |0, 0.25) and |0.25, 3):

```
c_data <- c_data %>%
  mutate(oper_result_new = cut(x = oper_result_millions, breaks = c(-5, 0, 0.25, 3)))

table(c_data$oper_result_new)
```

```
##
##   (-5,0] (0,0.25] (0.25,3]
##       23       65       12
```

Using all of the companies, produce histograms of `n_empl` split by `oper_result_new`, using a logarithmic scale for the number of employees:

```r
ggplot(c_data, aes(x = n_empl, fill = oper_result_new)) +
  theme_light() + geom_histogram() +
  scale_x_log10(label = comma) +
  facet_grid(oper_result_new ~ .) +
  labs(x = "Number of Employees", y = "Count", fill = "Operational Result (millions £)",
       title = "Number of Employees by Operational Result")
```



Exercise 4: Displaying Student Assessment Data

The file `Module_Marks_Invented_Example.csv` contains some student assessment data. It would not be ethically correct to release students' marks, so these marks are entirely fabricated, although they do share some of the properties of real marks.

Students take three modules: M1, M2, and M3. The coursework/examination/overall marks are indicated by the suffix of the column title *(.C/.E/.F respectively)*.

```r
m_data <- read_csv('../data/Module_Marks_Invented_Example.csv')
head(m_data)
```

```
## # A tibble: 6 x 9
##    M1.C  M1.E  M1.F  M2.C  M2.E  M2.F  M3.C  M3.E  M3.F
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 2    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 3    NA    NA    NA    66    74    72    72    49    56
## 4    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 5    NA    NA    NA    NA    NA    NA    NA    NA    NA
## 6    NA    NA    NA    NA    NA    NA    NA    NA    NA
```

Note that as not all students take all modules, there are a lot of missing values.

Use the `gather()` function from the `tidyr` package to place all of the marks into one column names `marks`, with another column named `source` indicating the source of the marks *(i.e. M1.C, M1.E, M1.F, M2.C, M2.E, M2.F, M3.C, M3.E, M3.F)*:

```
m_long <- gather(m_data, 'source', 'marks')
head(m_long)
```

```
## # A tibble: 6 x 2
##   source marks
##   <chr>  <dbl>
## 1 M1.C      NA
## 2 M1.C      NA
## 3 M1.C      NA
## 4 M1.C      NA
## 5 M1.C      NA
## 6 M1.C      NA
```

Use the `separate()` function from the `tidyr` package to separate the `source` column of `m_long` into two columns named `module` *(containing the module code M1, M2, or M3)* and `component` *(containing the assessment component C, E, or F)*. Do not change the `marks` column:

```
m_long_2 <- m_long %>%
  separate(col = source, into = c('module', 'component'), sep = '\\.')
head(m_long_2)
```

```
## # A tibble: 6 x 3
##   module component marks
##   <chr>  <chr>     <dbl>
## 1 M1     C            NA
## 2 M1     C            NA
## 3 M1     C            NA
## 4 M1     C            NA
## 5 M1     C            NA
## 6 M1     C            NA
```

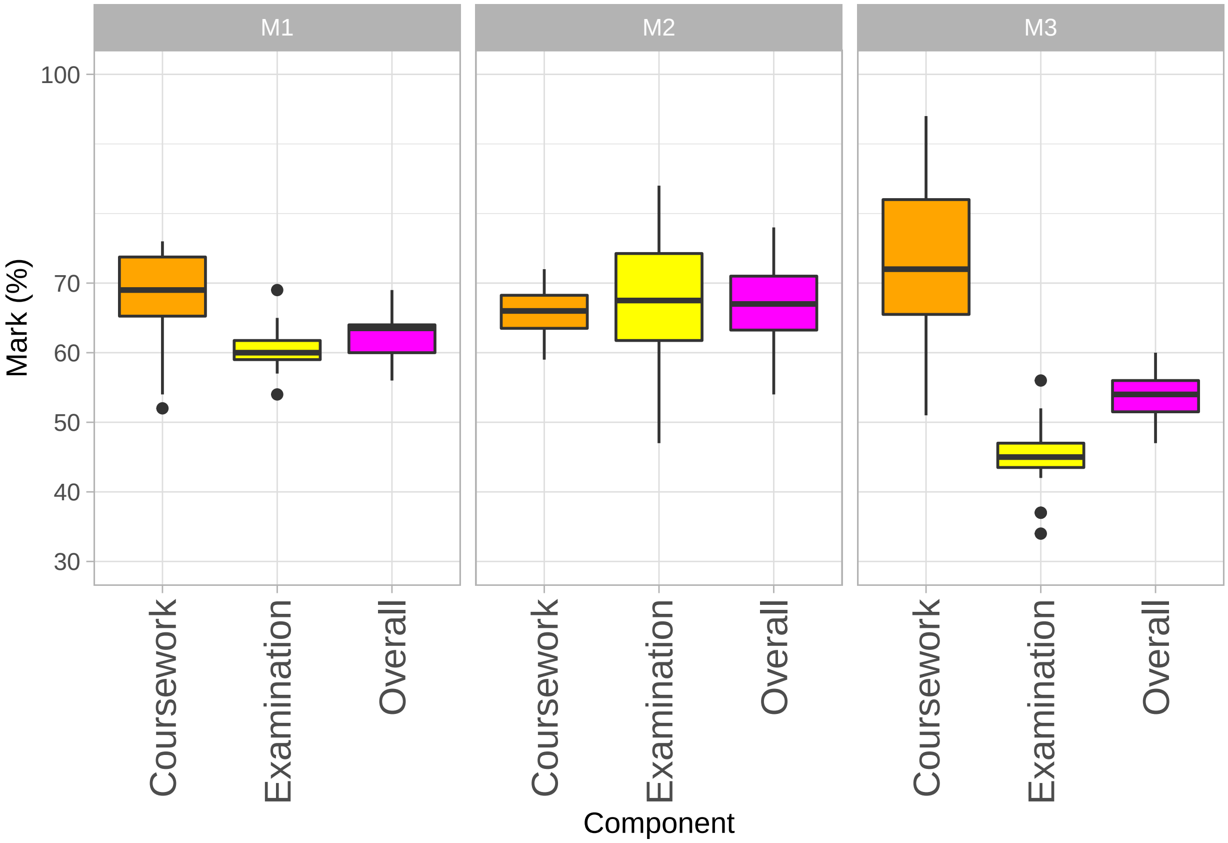
Use the `mutate()` function from `dplyr` to define `component` as a factor, named `component_f`, with levels: "C", "E", and "F" and labels: "Coursework", "Examination", and "Overall":

```
m_long_3 <- m_long_2 %>%
  mutate(component_f = factor(component, levels = c('C', 'E', 'F'),
                              labels = c('Coursework', 'Examination', 'Overall')))
head(m_long_3$component_f)
```

```
## [1] Coursework Coursework Coursework Coursework Coursework Coursework
## Levels: Coursework Examination Overall
```

Finally, produce a boxplot of this data using ggplot2:

```
ggplot(m_long_3, aes(x = component_f, y = marks, fill = component_f)) +
  theme_light() + geom_boxplot() + facet_grid(. ~ module) +
  scale_fill_manual(values = c('orange', 'yellow', 'magenta')) +
  scale_y_continuous(breaks = c(30, 40, 50, 60, 70, 100),
                     minor_breaks = c(30, 40, 50, 60, 70, 80, 90, 100),
                     limits = c(30, 100)) +
  labs(x = "Component", y = "Mark (%)") +
  theme(axis.text.x = element_text(size = 14, angle = 90, vjust = 0.5, hjust = 1),
        legend.position = 'none')
```


M1
M2
M3
Mark (%)
Coursework
Examination
Overall
Coursework
Examination
Overall
Coursework
Examination
Overall
Component