Data Analysis with R for Social Scientists

Jakob Tures & Jasper Tjaden

2023-08-29

Contents

Intro

This course offers an accessible and easy introduction to one of the fastest growing statistical packages used in social science and data science more generally.

Please download the data used in the course here. To find more about me, have a look at my website. Also, feel free to watch me as I walk you through each lesson here.

Overview over the Course :

- **Week 1: Introduction to Seminar**
- **Week 2: Exploratory Data Analysis-I**
- **Week 3: Exploratory Data Analysis-II**
- **Week 4: DAGs**
- **Week 5: Linear Regression Theory I: Simple Linear Regression**
- **Week 6: Linear Regression Theory II: Multiple Linear Regression**
- **Week 7: Linear Regression Theory III: Diagnostics**
- **Week 8: Linear Regression - Application**
- **Week 9: Logistic Regression - Exercises**
- **Week 10: Mediation**
- **Week 11: Prediction - Theory**
- **Week 12: Prediction - Application**
- **Week 13: Other Estimators**
- **Week 14: Course Paper - Discussion - Outlook**

Chapter 1

Introduction to Seminar

1.1 Introduction

Welcome to this course! In this course, you will learn how to analyse data using multiple regression in R. The course is aimed at undergraduate students who have completed the course "Intro to R for Social Scientists.

1.2 Why should I take this course?

- What is regression?
- What do you use it for?

1.3 Objectives

-
-

1.4 What is not covered

-
-

1.5 Prerequisites

- Basic knowledge of how to use R (data cleaning, management etc.) (include hyperlink to "Intro to R course")
- Descriptive statistics
- Data visualization using ggplot()

1.6 Structure

1. Intro to seminar

BLOCK I - Pre-processing/ EDA

2. Exploratory data analysis (EDA) - I (ggplot; gtsummary; x)
3. EDA - II -> in class exercise

–> add visualization/ reporting here.

–> knitting

BLOCK II - Modelling/ Linear regression

4. DAGS
5. Linear Regression - theory I: Simple Linear Regression
6. Linear Regression - theory II: Multiple Linear Regression
7. Linear Regression - theory III: Diagnostics

Block III - Application

8. Linear Regression – application
9. Linear Regression - exercise

Block IV - Advanced uses

10. Mediation (?)
11. Prediction - Theory
12. Prediction - Application
13. Other estimators: Logistic/ Poisson/ Multilevel/ FE
14. Course paper/Discussion/ Outlook

Chapter 2

Exloratory Data Analysis - I

2.1 Objectives

- Remember how to load and explore datasets in R
- Conduct basic descriptive data analysis
- Undestand and visualize distribution of and relationship between variables

2.2 R functions covered this week

- `load()`
- `read_excel()`
- `str()`
- `glimpse()`
- `table()`
- `summary()`
- `mutate()`
- `case_when()`
- `ggplot()`
- `corr ()`

2.3 Why is EDA so important?

- Every regression analysis is based on proper EDA; EDA often used for "hypothesis generation" (i.e. finding things that could be interesting to study).

- `EDA` helps understand the data and issues in the data. The better we understand the data, the better we can "fine-tune" our regression model later.
- `EDA` helps prepping data for regression (i.e. "cleaning"; "pre-processing"). Small errors in the data can lead to completely wrong conclusions or even prevent the model from working altogether.

2.4 Importing data into R

The first step of any data analysis is getting data into R. To get started, we first need to follow some preparatory steps

1. In this course, we will use data on NBA players (Basketball). Usually, you need to first download the data and documentation and save them in a folder on your own computer. We have already done this, and provide the data for you here.
2. Install R and Rstudio. If you don't already have R installed, here is a link to how it is done (hyperlink)
3. In the folder which you will use for this class, create a new R project. You will see that all files appear in the bottom right window in R studio.

Now, Let's get started.

2.5 Import data into R

First, we need to install some packages.

```
library("tidyverse")
library("readxl")
```

Now, let's import the data.You can see in the folder that we have 2 csv files. We can use `read_delim()` or `read_csv` function. Note that you need other function for Stata datasets (`read_data` from the haven package). To load Rdata files, you use the `load()` function.

Make sure you name the correct sub-directory in case you saved the data a sub-folder of your project folder (which I have done).

```
# import data
nba_salaries <- read_csv("../datasets/nba/salaries_1985to2018.csv", show_col_types = FA
nba_players <- read_csv("../datasets/nba/players.csv", show_col_types = FALSE)
```

Great, the two dataframes should appear in your environment in the upper right side in R studio.

Let's take a quick look at these trend dataframe using str() function The str() function shows the number of rows (observations) and columns (variables). It also provides information on the name of each column, its type and an example of some of the values in each column.

```
str(nba_players)
```

```
## spc_tbl_ [4,685 x 25] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ index      : num [1:4685] 0 1 2 3 4 5 6 7 8 9 ...
##  $ _id        : chr [1:4685] "abdelal01" "abdulza01" "abdulka01" "abdulma02" ...
##  $ birthDate  : chr [1:4685] "June 24, 1968" "April 7, 1946" "April 16, 1947" "March 9, 1969"
##  $ birthPlace : chr [1:4685] "Cairo, Egypt" "Brooklyn, New York" "New York, New York" "Gulfpor
##  $ career_AST : num [1:4685] 0.3 1.2 3.6 3.5 1.1 2.5 1.2 1 0.7 0.5 ...
##  $ career_FG% : chr [1:4685] "50.2" "42.8" "55.9" "44.2" ...
##  $ career_FG3%: chr [1:4685] "0.0" NA "5.6" "35.4" ...
##  $ career_FT% : chr [1:4685] "70.1" "72.8" "72.1" "90.5" ...
##  $ career_G   : num [1:4685] 256 505 1560 586 236 830 319 1 56 174 ...
##  $ career_PER : chr [1:4685] "13.0" "15.1" "24.6" "15.4" ...
##  $ career_PTS : num [1:4685] 5.7 9 24.6 14.6 7.8 18.1 5.6 0 9.5 5.3 ...
##  $ career_TRB : chr [1:4685] "3.3" "8.0" "11.2" "1.9" ...
##  $ career_WS  : num [1:4685] 4.8 17.5 273.4 25.2 3.5 ...
##  $ career_eFG%: chr [1:4685] "50.2" NA "55.9" "47.2" ...
##  $ college    : chr [1:4685] "Duke University" "Iowa State University" "University of Californ
##  $ draft_pick : chr [1:4685] "25th overall" "5th overall" "1st overall" "3rd overall" ...
##  $ draft_round: chr [1:4685] "1st round" "1st round" "1st round" "1st round" ...
##  $ draft_team : chr [1:4685] "Portland Trail Blazers" "Cincinnati Royals" "Milwaukee Bucks" "D
##  $ draft_year : chr [1:4685] "1990" "1968" "1969" "1990" ...
##  $ height     : chr [1:4685] "6-10" "6-9" "7-2" "6-1" ...
##  $ highSchool : chr [1:4685] "Bloomfield in Bloomfield, New Jersey" "John Jay in Brooklyn, New
##  $ name       : chr [1:4685] "Alaa Abdelnaby" "Zaid Abdul-Aziz" "Kareem Abdul-Jabbar" "Mahmoud
##  $ position   : chr [1:4685] "Power Forward" "Power Forward and Center" "Center" "Point Guard"
##  $ shoots     : chr [1:4685] "Right" "Right" "Right" "Right" ...
##  $ weight     : chr [1:4685] "240lb" "235lb" "225lb" "162lb" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   index = col_double(),
##   ..   `_id` = col_character(),
##   ..   birthDate = col_character(),
##   ..   birthPlace = col_character(),
##   ..   career_AST = col_double(),
##   ..   `career_FG%` = col_character(),
##   ..   `career_FG3%` = col_character(),
##   ..   `career_FT%` = col_character(),
```

```
##   ..   career_G = col_double(),
##   ..   career_PER = col_character(),
##   ..   career_PTS = col_double(),
##   ..   career_TRB = col_character(),
##   ..   career_WS = col_double(),
##   ..   `career_eFG%` = col_character(),
##   ..   college = col_character(),
##   ..   draft_pick = col_character(),
##   ..   draft_round = col_character(),
##   ..   draft_team = col_character(),
##   ..   draft_year = col_character(),
##   ..   height = col_character(),
##   ..   highSchool = col_character(),
##   ..   name = col_character(),
##   ..   position = col_character(),
##   ..   shoots = col_character(),
##   ..   weight = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(nba_salaries)
```

```
## spc_tbl_ [14,163 x 8] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ index       : num [1:14163] 0 1 2 3 4 5 6 7 8 9 ...
##  $ league      : chr [1:14163] "NBA" "NBA" "NBA" "NBA" ...
##  $ player_id   : chr [1:14163] "abdelal01" "abdelal01" "abdelal01" "abdelal01" ...
##  $ salary      : num [1:14163] 395000 494000 500000 805000 650000 1530000 2030000 2
##  $ season      : chr [1:14163] "1990-91" "1991-92" "1992-93" "1993-94" ...
##  $ season_end  : num [1:14163] 1991 1992 1993 1994 1995 ...
##  $ season_start: num [1:14163] 1990 1991 1992 1993 1994 ...
##  $ team        : chr [1:14163] "Portland Trail Blazers" "Portland Trail Blazers" "B
##  - attr(*, "spec")=
##   .. cols(
##   ..   index = col_double(),
##   ..   league = col_character(),
##   ..   player_id = col_character(),
##   ..   salary = col_double(),
##   ..   season = col_character(),
##   ..   season_end = col_double(),
##   ..   season_start = col_double(),
##   ..   team = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

We see that most variables/ columns, already are in the type which we want. R automatically picked up on the format. Text variables are “chr” for “character”;

Numeric variables are "num". It is very important that you understand the types of columns/ variables that R recognized. If you need a refresher, go here [hyperlink].

2.6 Merge datasets

We see that "nba_salaries" contains the salaries of players for various seasons. "players" contains many career statistics about players, for example, how many points they scored on average per game, across their whole career.

Now we want to link both datasets. We can do this using the `merge()` function. An alternative would be `join()`.

```
data_nba <- merge(nba_players, nba_salaries, by.x = c("_id"), by.y=c("player_id"))
rm(nba_players, nba_salaries)
```

2.7 Clean dataset

We can use the select() function to kick out columns and the filter() function to kick out rows. First, we need to look at the codebook and the questionnaire to understand the what each variable refers to (see .txt file "data_description").

When using the select function, we will also rename the variables to make them more intuitive.

First, let's filter out Let's filter all years between 2012-2022.

Afterwards, we use the save() function to store the data as a .Rdata file and the write_excel() function to store the reduced dataset. This way we can simply load that one next time and save time.

```
data_nba <- data_nba %>%
        select(everything(), -league, -highSchool) %>%
        filter(season_start>=1998)
save(data_nba, file ="../datasets/nba/data_nba.RData")
```

2.8 change variables

Let's do some data cleaning. First, we want to calculate each players' age at the beginning of each season. Currently, we only have the date of birth and the year for each season.

Second, we want recode the "position" variable. Some players played multiple position, so that info is messy. We want to create varies dummy variables for each position.

```
# let's calculate age for every season
class(data_nba$birthDate) # nice, it is already a date variable
```

```
## [1] "character"
```

```
library(lubridate)
data_nba <- data_nba %>%
  mutate(year_of_birth = year(mdy(birthDate)),
         age = season_start - year_of_birth)


# let's clean the position

table(data_nba$position)
```

```
##
##                                                                 Center
##                                                                   1204
##                                              Center and Power Forward
##                                                                    686
##                          Center and Power Forward and Small Forward
##                                                                      3
##                          Center and Small Forward and Power Forward
##                                                                     30
##                                                            Point Guard
##                                                                   1162
## Point Guard and Power Forward and Small Forward and Shooting Guard
##                                                                      5
##                                         Point Guard and Shooting Guard
##                                                                    573
##                     Point Guard and Shooting Guard and Small Forward
##                                                                     13
##                                       Point Guard and Small Forward
##                                                                      3
##                     Point Guard and Small Forward and Shooting Guard
##                                                                     21
##                                                       Power Forward
##                                                                    667
##                                              Power Forward and Center
##                                                                    884
```

```
##                         Power Forward and Center and Small Forward
##                                                                  51
##                                  Power Forward and Shooting Guard
##                                                                  14
##                Power Forward and Shooting Guard and Small Forward
##                                                                  31
##                                   Power Forward and Small Forward
##                                                                 418
##                         Power Forward and Small Forward and Center
##                                                                   3
##                Power Forward and Small Forward and Shooting Guard
##                                                                  11
##                                                       Shooting Guard
##                                                                 735
##                                    Shooting Guard and Point Guard
##                                                                 581
##                  Shooting Guard and Point Guard and Small Forward
##                                                                  17
##                Shooting Guard and Power Forward and Small Forward
##                                                                  25
##                                  Shooting Guard and Small Forward
##                                                                 592
##                  Shooting Guard and Small Forward and Point Guard
##                                                                  73
##                Shooting Guard and Small Forward and Power Forward
##                                                                  25
##                                                     Small Forward
##                                                                 620
##                                          Small Forward and Center
##                                                                   4
##                         Small Forward and Center and Power Forward
##                                                                  72
##                  Small Forward and Point Guard and Shooting Guard
##                                                                  19
##                                   Small Forward and Power Forward
##                                                                 425
##                         Small Forward and Power Forward and Center
##                                                                  49
##                Small Forward and Power Forward and Shooting Guard
##                                                                  33
##                                  Small Forward and Shooting Guard
##                                                                 588
##                  Small Forward and Shooting Guard and Point Guard
##                                                                  22
##                Small Forward and Shooting Guard and Power Forward
##                                                                  69
```

```
data_nba <- data_nba %>%
  mutate(
    position_center =
      case_when(position = str_detect(position,"Center") ~ 1,
                TRUE ~ 0),
    position_sf =
      case_when(position = str_detect(position,"Small Forward") ~ 1,
                TRUE ~ 0),
    position_pf =
      case_when(position = str_detect(position,"Power Forward") ~ 1,
                TRUE ~ 0),
    position_sg =
      case_when(position = str_detect(position,"Shooting Guard") ~ 1,
                TRUE ~ 0),
    position_pg =
      case_when(position = str_detect(position,"Point Guard") ~ 1,
                TRUE ~ 0))



data_nba <- data_nba %>%
  select("_id", name, age, weight, height, birthPlace, everything(), -position, -birthl
save(data_nba, file ="../datasets/nba/data_nba.RData")
```

Now, we want to create a variable that gives us the number of seasons (i.e.years) that each player played. Since the dataset is organized in seasons, each row is one season. Counting the rows per player gives us the years they played.

```
data_nba <- data_nba %>%
  group_by("_id") %>%
  mutate(seasons_played = n()) %>%
  ungroup()
```

Almost done. When we browse the dataset, we recognize that the height and weight variables are stored as characters. Let's convert them to numeric, so we can use them in operations.

```
str(data_nba$weight)
```

```
##  chr [1:9728] "162lb" "223lb" "223lb" "223lb" "223lb" "223lb" "223lb" ...
```

```
str(data_nba$height)
```

```
##  chr [1:9728] "6-1" "6-6" "6-6" "6-6" "6-6" "6-6" "6-6" "6-6" "6-6" "6-6" ...
```

```
data_nba <- data_nba %>%
  mutate(weight = str_replace(weight, "lb", ""),
         weight = as.numeric(weight),
         height = str_replace(height, "-", "."),
         height = as.numeric(height))

str(data_nba)
```

```
## tibble [9,728 x 36] (S3: tbl_df/tbl/data.frame)
##  $ _id            : chr [1:9728] "abdulma02" "abdulta01" "abdulta01" "abdulta01" ...
##  $ name           : chr [1:9728] "Mahmoud Abdul-Rauf" "Tariq Abdul-Wahad" "Tariq Abdul-Wahad"
##  $ age            : num [1:9728] 31 24 25 26 27 28 29 30 31 32 ...
##  $ weight         : num [1:9728] 162 223 223 223 223 223 223 223 223 223 ...
##  $ height         : num [1:9728] 6.1 6.6 6.6 6.6 6.6 6.6 6.6 6.6 6.6 6.6 ...
##  $ birthPlace     : chr [1:9728] "Gulfport, Mississippi" "Maisons Alfort, France" "Maisons Alf
##  $ index.x        : num [1:9728] 3 4 4 4 4 4 4 4 4 4 ...
##  $ career_AST     : num [1:9728] 3.5 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 ...
##  $ career_FG%     : chr [1:9728] "44.2" "41.7" "41.7" "41.7" ...
##  $ career_FG3%    : chr [1:9728] "35.4" "23.7" "23.7" "23.7" ...
##  $ career_FT%     : chr [1:9728] "90.5" "70.3" "70.3" "70.3" ...
##  $ career_G       : num [1:9728] 586 236 236 236 236 236 236 236 236 236 ...
##  $ career_PER     : chr [1:9728] "15.4" "11.4" "11.4" "11.4" ...
##  $ career_PTS     : num [1:9728] 14.6 7.8 7.8 7.8 7.8 7.8 7.8 7.8 7.8 7.8 ...
##  $ career_TRB     : chr [1:9728] "1.9" "3.3" "3.3" "3.3" ...
##  $ career_WS      : num [1:9728] 25.2 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5 3.5 ...
##  $ career_eFG%    : chr [1:9728] "47.2" "42.2" "42.2" "42.2" ...
##  $ college        : chr [1:9728] "Louisiana State University" "University of Michigan, San Jos
##  $ draft_pick     : chr [1:9728] "3rd overall" "11th overall" "11th overall" "11th overall" ..
##  $ draft_round    : chr [1:9728] "1st round" "1st round" "1st round" "1st round" ...
##  $ draft_team     : chr [1:9728] "Denver Nuggets" "Sacramento Kings" "Sacramento Kings" "Sacra
##  $ draft_year     : chr [1:9728] "1990" "1997" "1997" "1997" ...
##  $ shoots         : chr [1:9728] "Right" "Right" "Right" "Right" ...
##  $ index.y        : num [1:9728] 17 19 20 21 22 23 24 25 26 27 ...
##  $ salary         : num [1:9728] 798500 1411000 1594920 4500000 5062500 ...
##  $ season         : chr [1:9728] "2000-01" "1998-99" "1999-00" "2000-01" ...
##  $ season_end     : num [1:9728] 2001 1999 2000 2001 2002 ...
##  $ season_start   : num [1:9728] 2000 1998 1999 2000 2001 ...
##  $ team           : chr [1:9728] "Vancouver Grizzlies" "Sacramento Kings" "Denver Nuggets" "De
##  $ position_center: num [1:9728] 0 0 0 0 0 0 0 0 0 0 ...
##  $ position_sf    : num [1:9728] 0 0 0 0 0 0 0 0 0 0 ...
##  $ position_pf    : num [1:9728] 0 0 0 0 0 0 0 0 0 0 ...
##  $ position_sg    : num [1:9728] 0 1 1 1 1 1 1 1 1 1 ...
##  $ position_pg    : num [1:9728] 1 0 0 0 0 0 0 0 0 0 ...
##  $ "_id"          : chr [1:9728] "_id" "_id" "_id" "_id" ...
##  $ seasons_played : int [1:9728] 9728 9728 9728 9728 9728 9728 9728 9728 9728 9728 ...
```

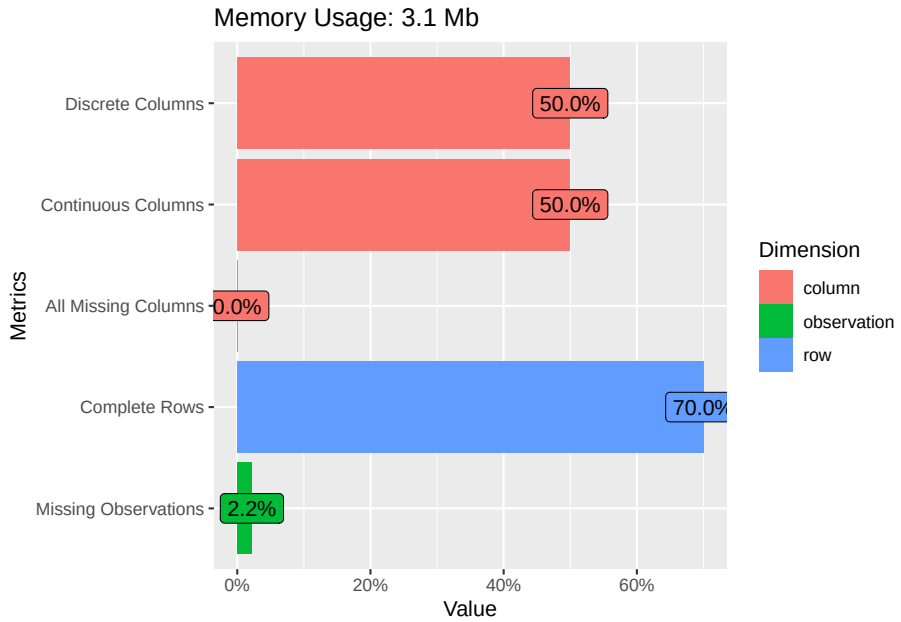
2.9 Explore the whole dataset

Now, let's explore some packages which help us to explore the dataframe as a whole. Let's start with the DataExplorer package. It is nice to get an overview of variables and the "missingness" of data.

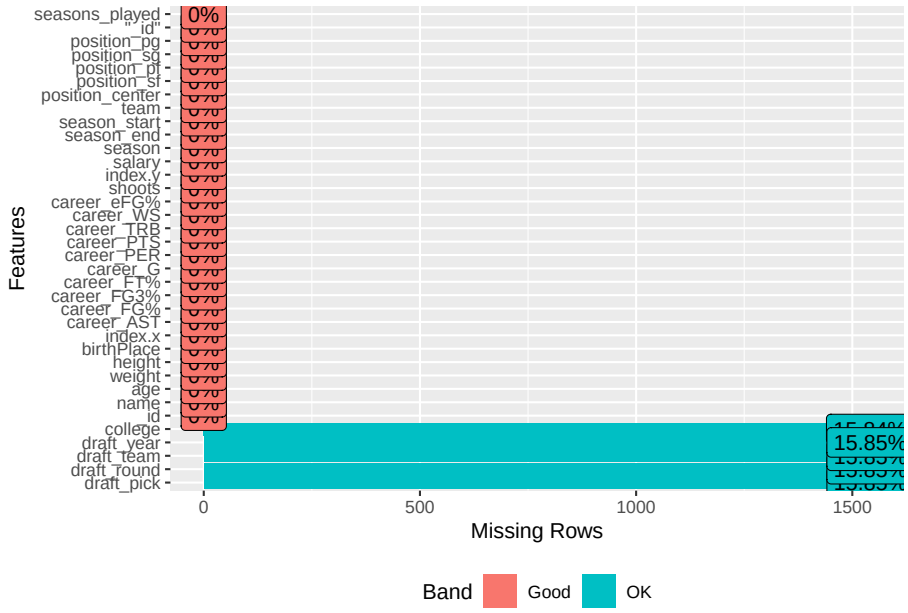
```
library(DataExplorer)

# overview of types of variables and missingness
introduce(data_nba)
```

```
## # A tibble: 1 x 9
##    rows columns discrete_columns continuous_columns all_missing_columns
##   <int>   <int>            <int>              <int>               <int>
## 1  9728      36               18                 18                   0
## # i 4 more variables: total_missing_values <int>, complete_rows <int>,
## #   total_observations <int>, memory_usage <dbl>
```

```
# plots the info from above
plot_intro(data_nba)
```

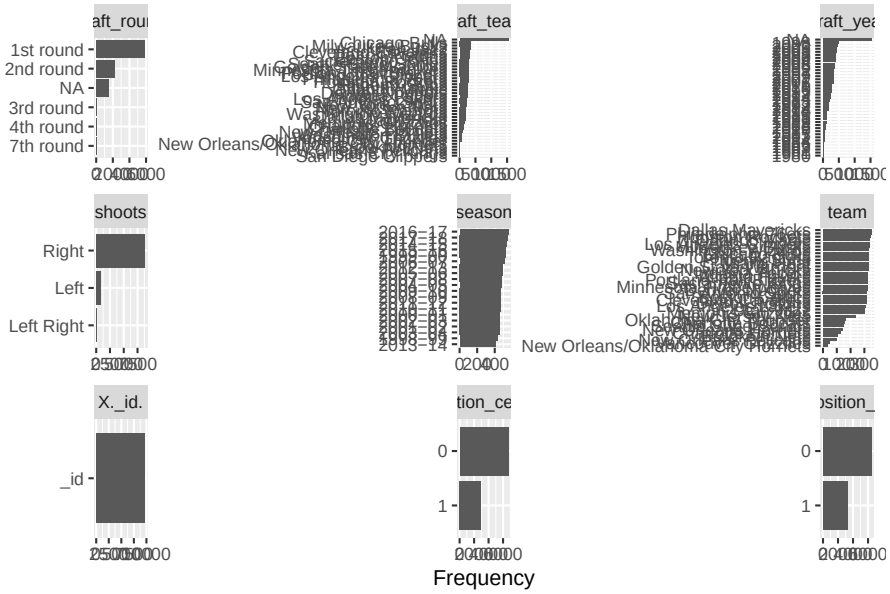


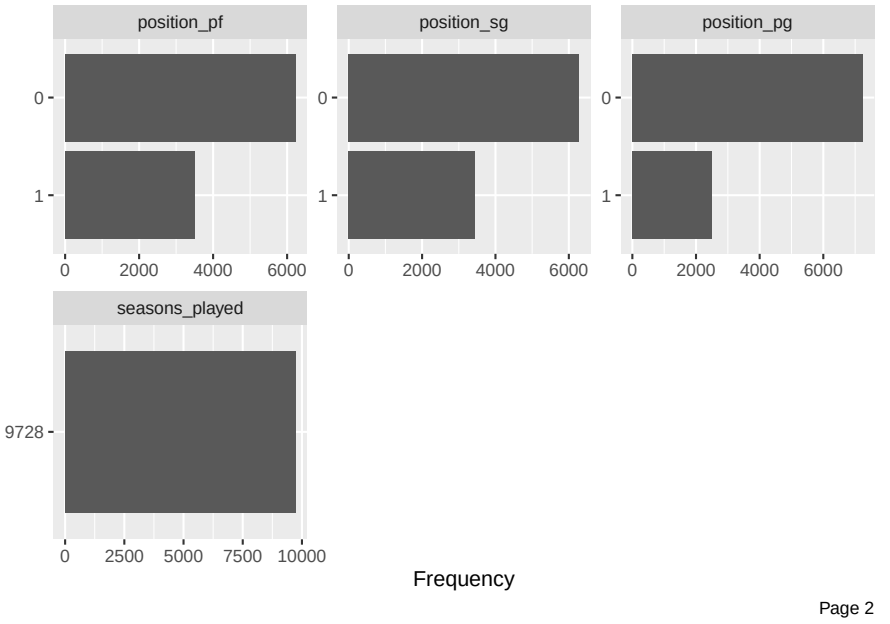

```
#plots percentages missing across variables
plot_missing(data_nba)
```




```
# plots frequencies across variables
plot_bar(data_nba)
```

```
## 11 columns ignored with more than 50 categories.
## X_id: 1794 categories
## name: 1790 categories
## birthPlace: 849 categories
## career_FG.: 333 categories
## career_FG3.: 297 categories
## career_FT.: 413 categories
## career_PER: 274 categories
## career_TRB: 113 categories
## career_eFG.: 310 categories
## college: 380 categories
## draft_pick: 67 categories
```





Now, let's try gtSummary for summary tables. A summary table is always a useful start once you have identified the type of variables you are interested in.

Let's assume we are interested in age, seasons played, career points, salary, position, and right or left handed (i.e. "shoots")

```
library(gtsummary)

data_nba %>%
  select(age, seasons_played, shoots, career_PTS, salary, contains("position")) %>%
         tbl_summary(
           statistic = all_continuous() ~ c("{mean} ({min}, {max})"))
```

```
## Table printed with `knitr::kable()`, not {gt}. Learn why at
## https://www.danieldsjoberg.com/gtsummary/articles/rmarkdown.html
## To suppress this message, include `message = FALSE` in code chunk header.
```

Characteristic	**N = 9,728**
age	27.0 (18.0, 42.0)
seasons_played	
9728	9,728 (100%)
shoots	
Left	856 (8.8%)
Left Right	7 (<0.1%)
Right	8,865 (91%)
career_PTS	8.9 (0.0, 30.1)
salary	4,072,633 (2,706, 34,682,550)
position_center	2,986 (31%)
position_sf	3,222 (33%)
position_pf	3,501 (36%)
position_sg	3,447 (35%)
position_pg	2,489 (26%)

Now, let's look at a correlation matrix between all numeric variables in the dataset.

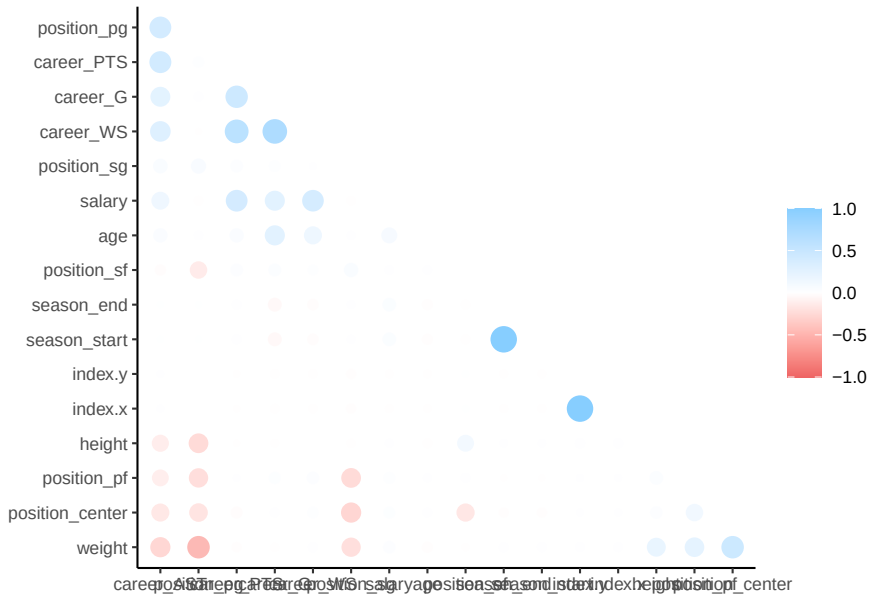
```
library("corrr")
library("corrplot")

data_nba_numeric <- data_nba %>%
  select(where(is.numeric)) %>%
  na.omit()

# Find constant columns
constant_columns <- sapply(data_nba_numeric, function(x) length(unique(x)) == 1)

# Remove constant columns
data_nba_numeric <- data_nba_numeric[, !constant_columns]
# as number
corrmatrix <- data_nba_numeric %>%
```

```
  correlate() %>%    # Create correlation data frame (cor_df)
  rearrange() %>%  # rearrange by correlations
  shave()

fashion(corrmatrix)
```

```
##              term career_AST position_pg career_PTS career_G career_WS
## 1       career_AST
## 2      position_pg        .61
## 3       career_PTS        .61         .08
## 4         career_G        .47         .04        .65
## 5        career_WS        .52        -.01        .76       .83
## 6      position_sg        .19         .22        .12       .09        .01
## 7           salary        .35        -.04        .60       .48        .59
## 8              age        .18         .03        .18       .49        .36
## 9      position_sf       -.08        -.33        .12       .13        .07
## 10      season_end        .01         .01        .04      -.17       -.09
## 11    season_start        .01         .01        .04      -.17       -.09
## 12         index.y        .01         .00       -.01      -.04       -.03
## 13         index.x        .01         .00       -.01      -.04       -.03
## 14          height       -.31        -.46       -.01      -.02       -.00
## 15     position_pf       -.29        -.44        .01       .11        .12
## 16 position_center       -.36        -.39       -.10       .04        .08
## 17          weight       -.49        -.66       -.06      -.03        .05
##    position_sg salary  age position_sf season_end season_start index.y index.x
## 1
## 2
## 3
## 4
## 5
## 6
## 7         -.03
## 8          .03    .25
## 9          .18    .03  .03
## 10         .02    .16 -.07        -.04
## 11         .02    .16 -.07        -.04       1.00
## 12        -.05   -.02 -.02         .01       -.03        -.03
## 13        -.06   -.02 -.02         .01       -.03        -.03    1.00
## 14        -.03    .03 -.04         .30        .02         .02     .05     .05
## 15        -.46    .09  .03         .04       -.00        -.00     .01     .01
## 16        -.49    .10  .04        -.37       -.06        -.06     .01     .02
## 17        -.43    .12 -.07        -.01        .04         .04     .04     .04
##    height position_pf position_center weight
## 1
## 2
```

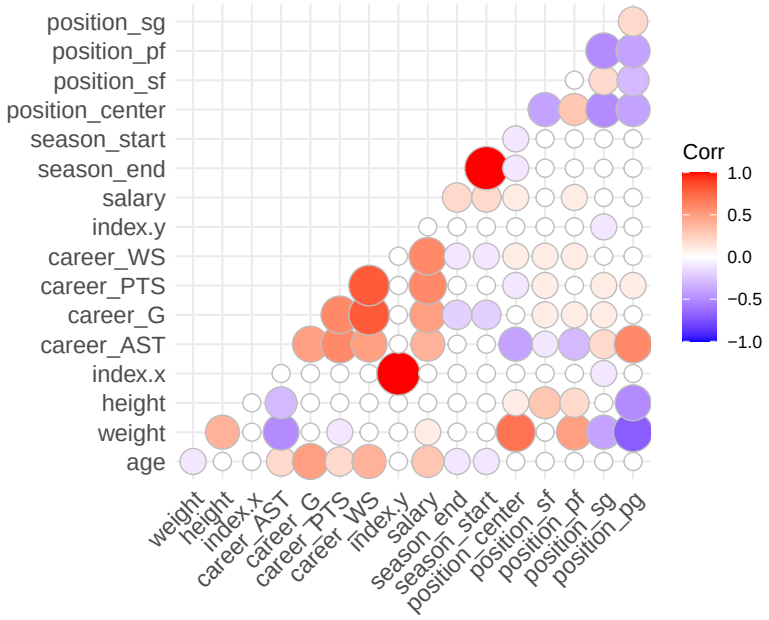
```
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15    .15
## 16    .12         .33
## 17    .41         .45             .66
```

```
# as plot
rplot(corrmatrix)
```




```
# or...ggplot approach
library("ggcorrplot")

corrmatrix <- round(cor(data_nba_numeric), 1)
```

```
ggcorrplot(corrmatrix,
           method = "circle",
           type="lower")
```



This is interesting for a first look. For example, it seems that the weight is strongly correlated with whatever position you play. Centers are heavy, point guards are light weights. We also see that most performance metrics ("career_…") are correlated with each other and also with salary. Good players seem to be good in many things, and good players seem to be paid more.

2.10 explore individual variables

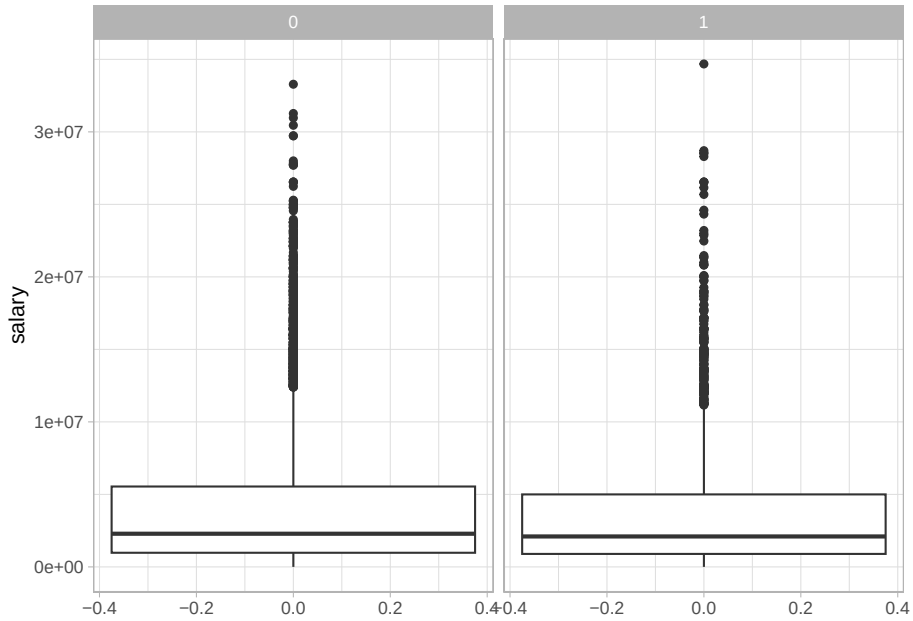
Now, that we have a feeling for the whole dataset, we want to explore individual variables. To keep it focused, we want to further explore the question of whether players that score more on average are also paid more.

Maybe roles are clearly divided on the team. Maybe really good passers are highly paid because they give great passes to people who then score. Or maybe teams don't care about passers and just pay more to people who score more.

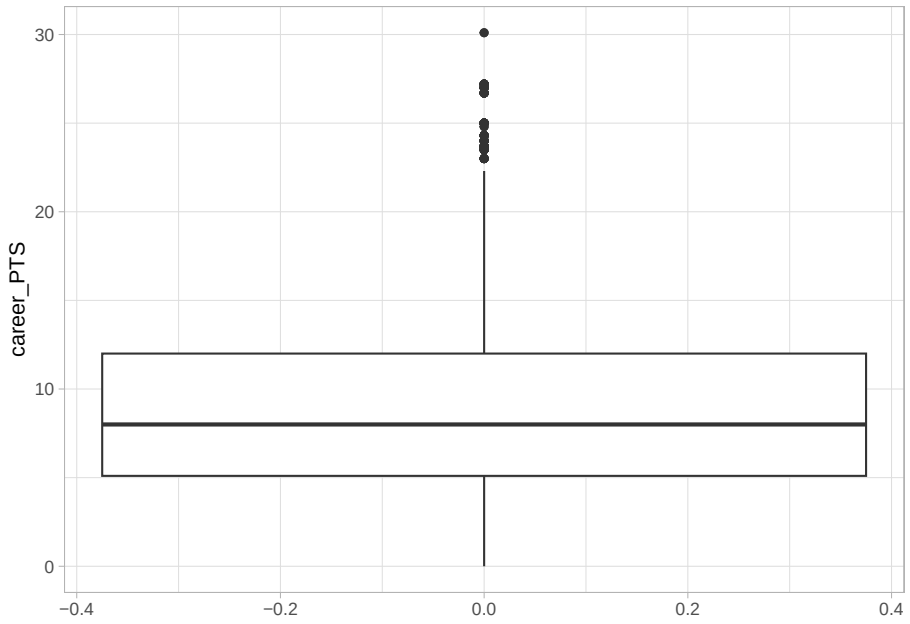
So, now, let's look at salary and average number of points scored by game. Also, we want to know whether point guards (short people who pass a lot) are paid less then other positions (who score more).

First, let's look how the two continuous variables are distributed:

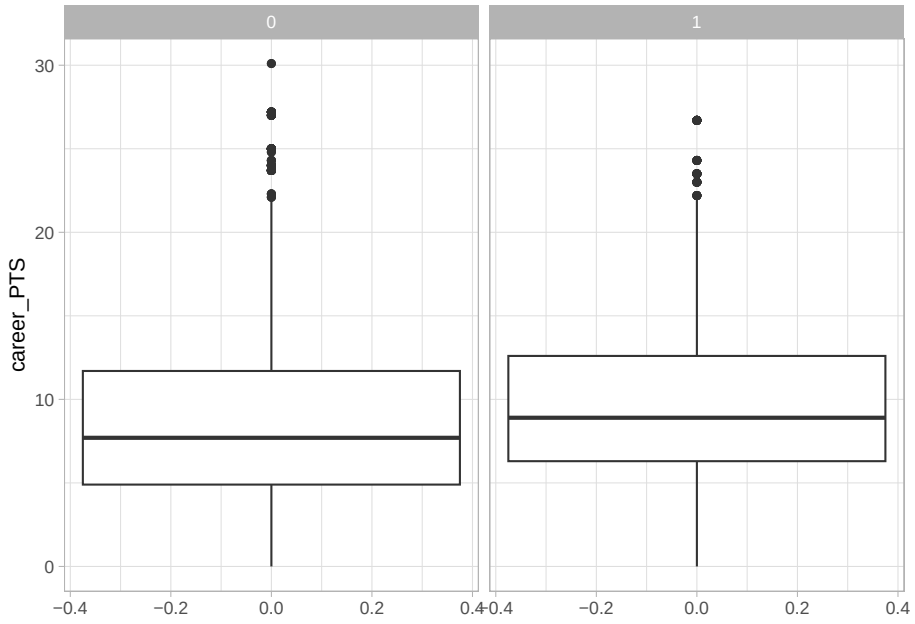
```
data_nba %>% ggplot() +
geom_boxplot(aes(y=salary)) +
  facet_wrap(~position_pg) +
  theme_light()
```




```
data_nba %>% ggplot() +
geom_boxplot(aes(y=career_PTS)) +
  theme_light()
```




```
data_nba %>% ggplot() +
geom_boxplot(aes(y=career_PTS)) +
  facet_wrap(~position_pg) +
  theme_light()
```



Let's look at the relationship between salary and position as well as the relationship between position and points.

```
str(data_nba$position_pg)
```

```
##  num [1:9728] 1 0 0 0 0 0 0 0 0 0 ...
```

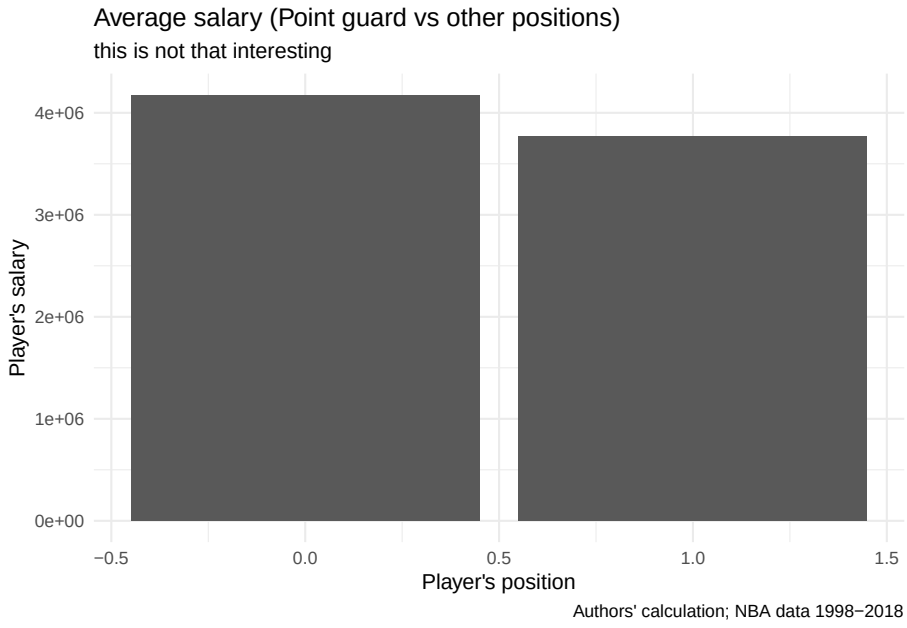
```
str(data_nba$salary)
```

```
##  num [1:9728] 798500 1411000 1594920 4500000 5062500 ...
```

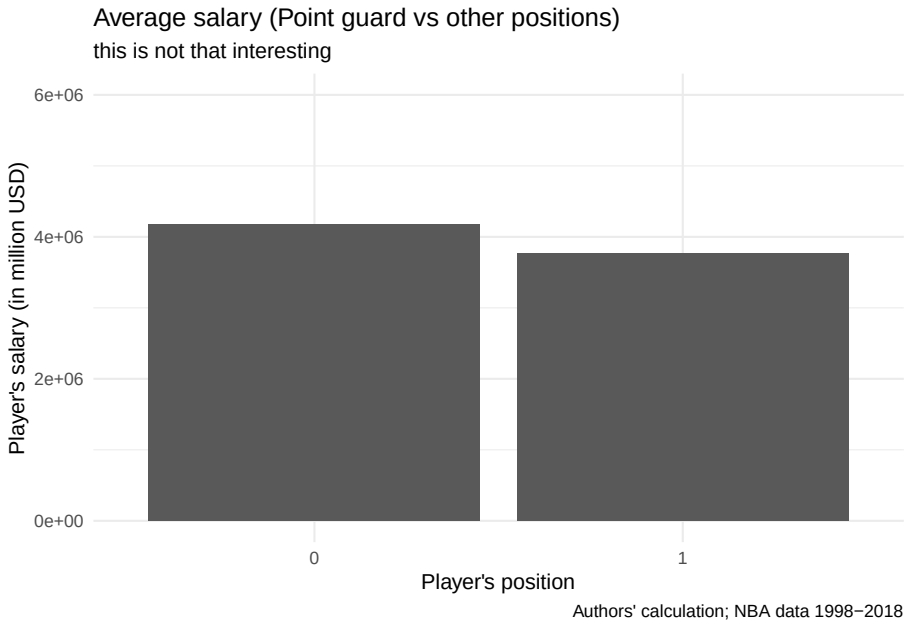
```
str(data_nba$career_pts)
```

```
##  NULL
```

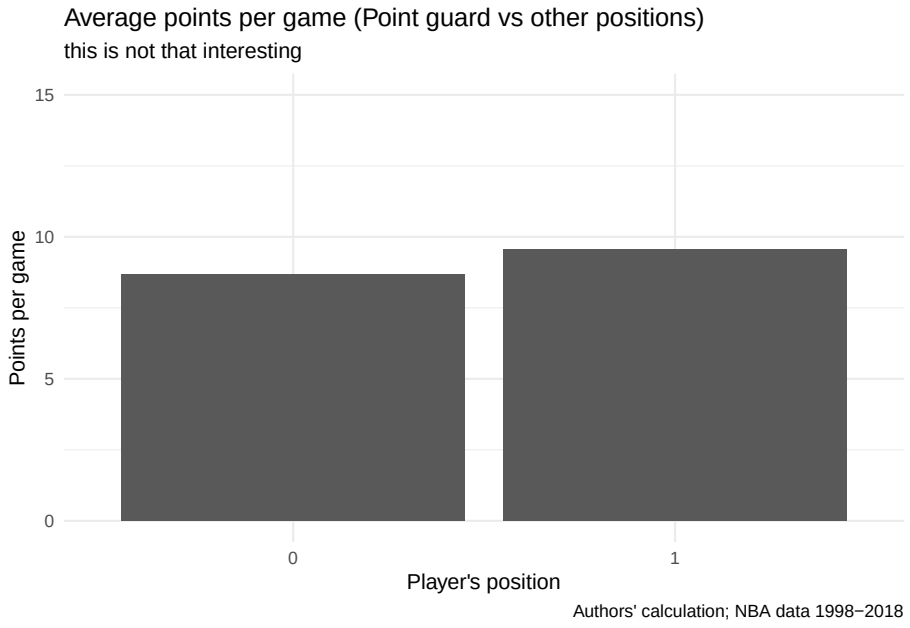
```
data_nba %>%
  ggplot() +
  geom_bar(aes(x = position_pg,
               y = salary),
           position = "dodge",
           stat = "summary",
           fun = "mean") +
  #xlim(0, 10000000) +
  labs(title = "Average salary (Point guard vs other positions)",
       subtitle = "this is not that interesting",
       caption = "Authors' calculation; NBA data 1998-2018") +
  xlab("Player's position") +
  ylab("Player's salary") +
  theme_minimal()
```




```
## alternatively:

data_nba %>%
  group_by(position_pg) %>%
  summarize(mean_salary = mean(salary, na.rm=T)) %>%
  ggplot() +
  geom_bar(aes(x = as.factor(position_pg),
               y = mean_salary),
           stat = "identity") +
  ylim(0, 6000000) +
  labs(title = "Average salary (Point guard vs other positions)",
       subtitle = "this is not that interesting",
       caption = "Authors' calculation; NBA data 1998-2018") +
  xlab("Player's position") +
  ylab("Player's salary (in million USD)") +
  theme_minimal()
```




```
# Now the relationship between position and points:

data_nba %>%
  group_by(position_pg) %>%
  summarize(mean_points = mean(career_PTS, na.rm=T)) %>%
  ggplot() +
  geom_bar(aes(x = as.factor(position_pg),
               y = mean_points),
           stat = "identity") +
  ylim(0,15) +
  labs(title = "Average points per game (Point guard vs other positions)",
       subtitle = "this is not that interesting",
       caption = "Authors' calculation; NBA data 1998-2018") +
  xlab("Player's position") +
  ylab("Points per game") +
  theme_minimal()
```

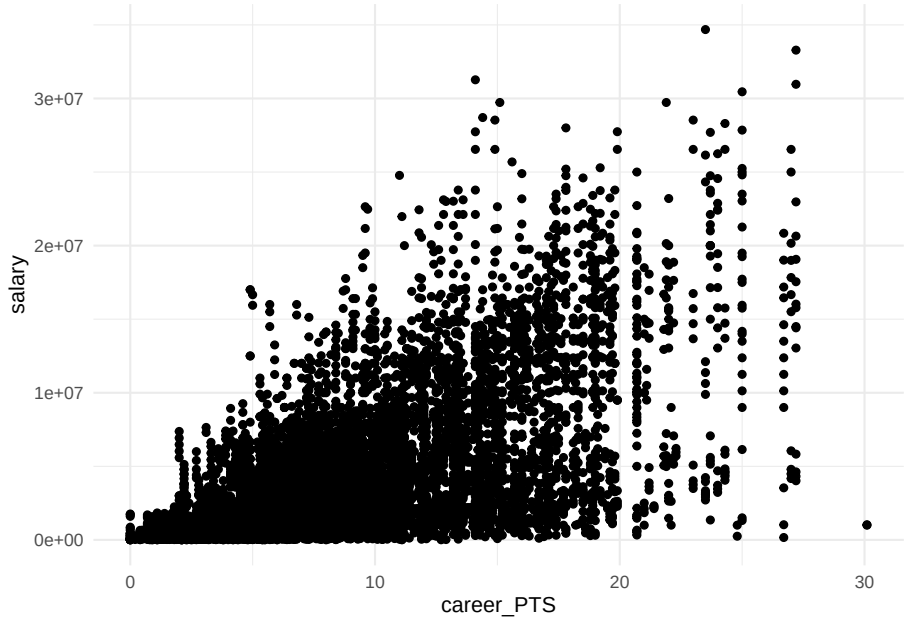


So, it seems that point gards are paid a little less even though they make a few more points on average. Interesting puzzle to explore.

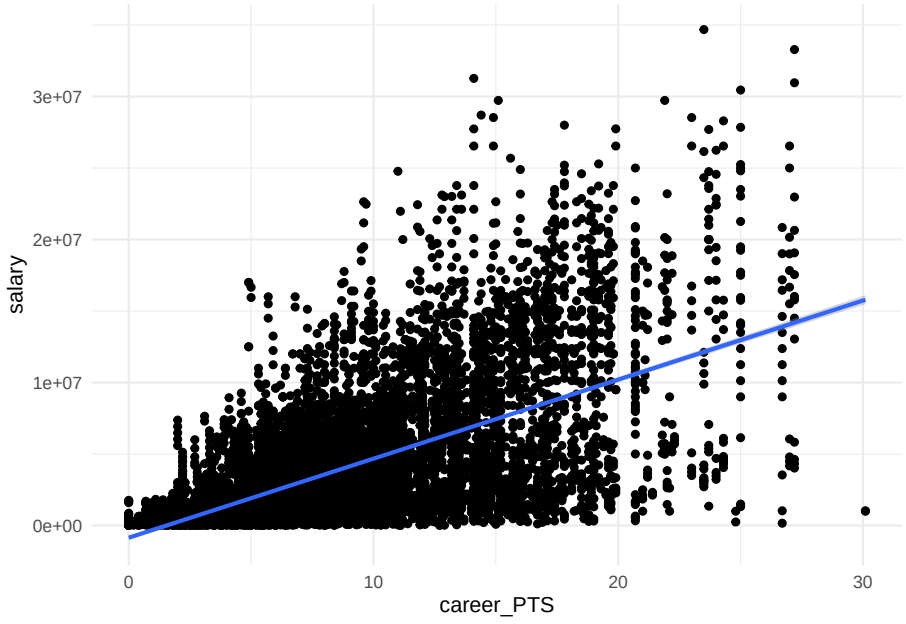
```
# find example for using histogramm()
# find example for first just creating cross tabs as perecntages tbl_cross(); summariz
```

Now, let's explore the relationship which is at the heart of our analysis from now on: salary and average points.

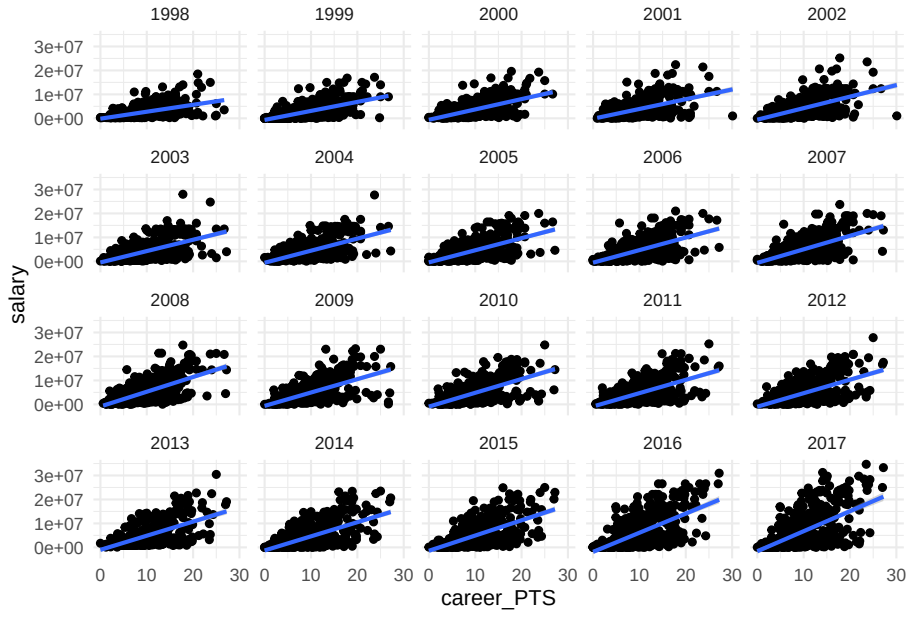
```
# simple scatterplot
data_nba %>% ggplot() +
geom_point(aes(y=salary, x=career_PTS)) +
  theme_minimal()
```



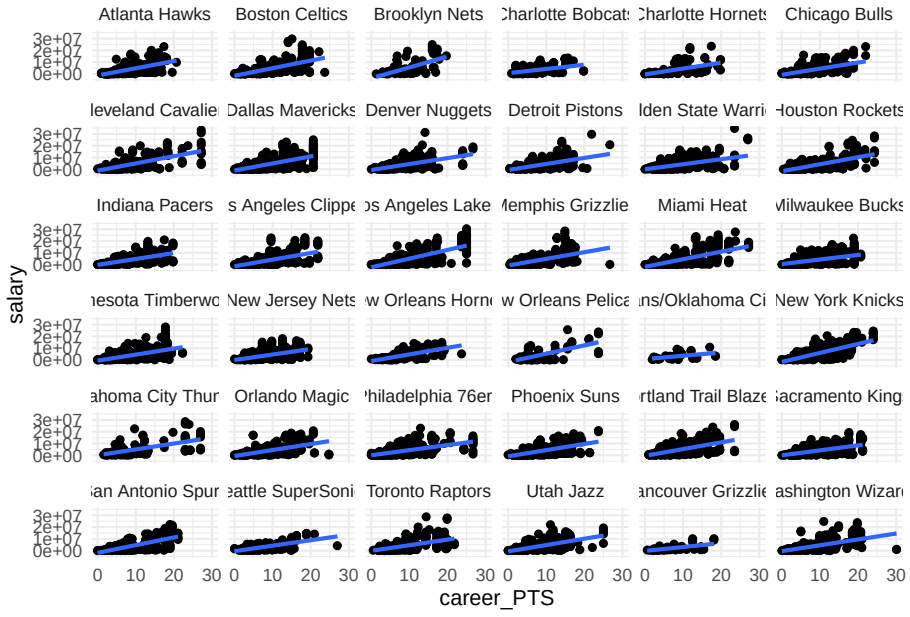

```
# Now, let's add a line
data_nba %>%
  ggplot(aes(y=salary, x=career_PTS)) +
    geom_point() +
     stat_smooth(method = "lm") +
       theme_minimal()
```




```
# let's look the relatiship separate for every year
data_nba %>%
  ggplot(aes(y=salary, x=career_PTS)) +
    geom_point() +
     stat_smooth(method = "lm") +
        facet_wrap(~season_start) +
       theme_minimal()
```

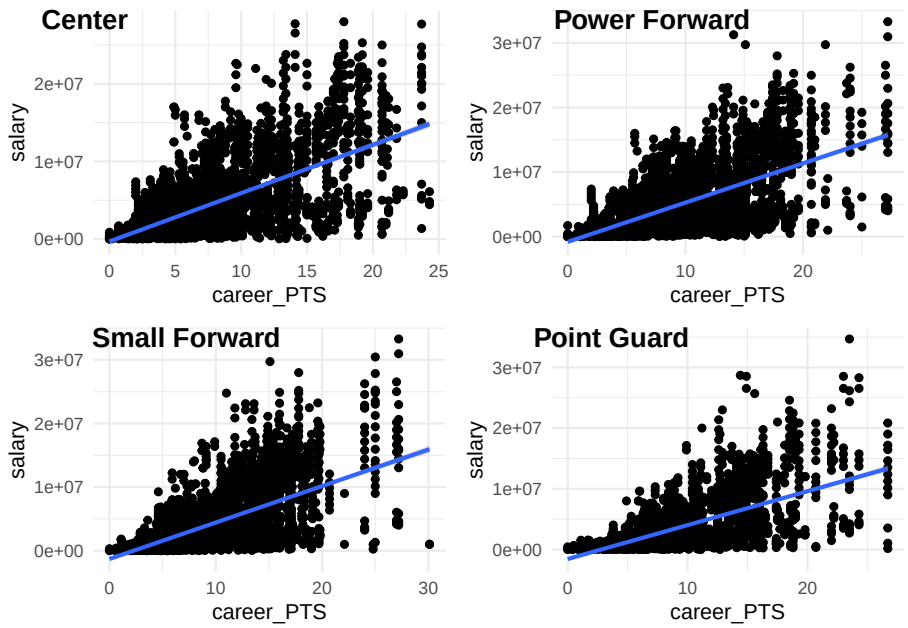



```
# let's look the relatiship for separate teams
data_nba %>%
  ggplot(aes(y=salary, x=career_PTS)) +
    geom_point() +
     stat_smooth(method = "lm") +
        facet_wrap(~team) +
       theme_minimal()
```

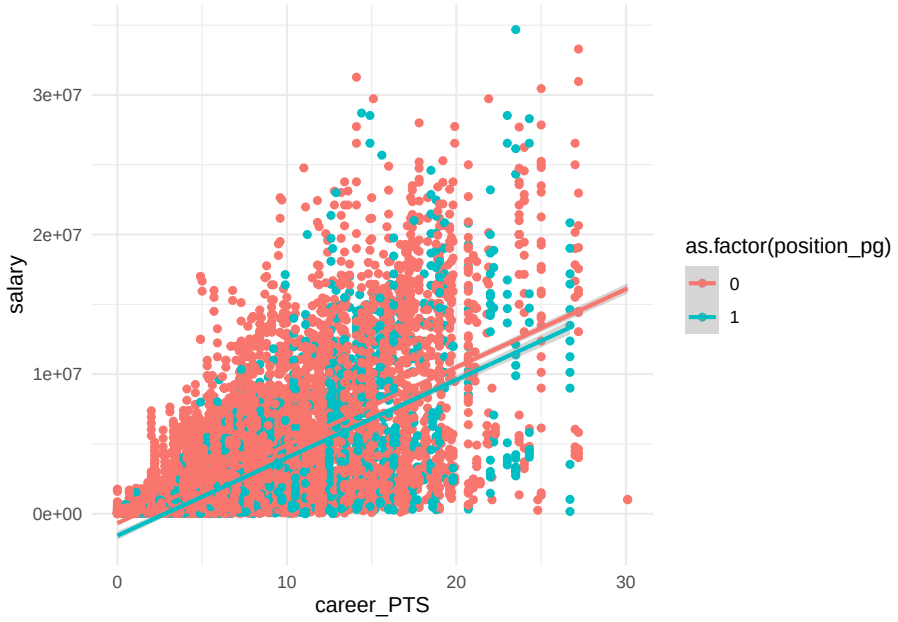



```
# let's look at it by position
scatter_pg <- data_nba %>% filter(position_pg ==1) %>%
  ggplot(aes(y=salary, x=career_PTS)) +
    geom_point() +
     stat_smooth(method = "lm") +
       theme_minimal()

scatter_center <- data_nba %>% filter(position_center ==1) %>%
  ggplot(aes(y=salary, x=career_PTS)) +
    geom_point() +
     stat_smooth(method = "lm") +
       theme_minimal()

scatter_pf <- data_nba %>% filter(position_pf ==1) %>%
  ggplot(aes(y=salary, x=career_PTS)) +
    geom_point() +
     stat_smooth(method = "lm") +
       theme_minimal()

scatter_sf <- data_nba %>% filter(position_sf ==1) %>%
  ggplot(aes(y=salary, x=career_PTS)) +
    geom_point() +
     stat_smooth(method = "lm") +
       theme_minimal()
```

```
library("ggpubr")
ggarrange(scatter_center, scatter_pf,
          scatter_sf, scatter_pg,
          ncol = 2, nrow = 2,
          labels = c("Center",
                     "Power Forward",
                     "Small Forward",
                     "Point Guard"))
```




```
# yet a better way to compare this could be:

data_nba %>%
  ggplot(aes(y=salary, x=career_PTS, color=as.factor(position_pg))) +
    geom_point() +
     stat_smooth(method = "lm",
                 aes(group=as.factor(position_pg))) +
       theme_minimal()
```




```
data_nba %>%
  ggplot(aes(y=salary, x=career_PTS, color=as.factor(position_center))) +
    geom_point() +
     stat_smooth(method = "lm",
                 aes(group=as.factor(position_center))) +
       theme_minimal()
```

Let's reflect a moment what we can learn from all this.

First, there seems to be a somewhat linear relationship between how many points a player scores and how much they are paid. This relationship seems pretty robust across years and teams. It also holds true for point guards just as much as for non-point guards. However, the average salary for point guards is lower in comparison. We also learn that the link between salary and points is stronger for centers. They seems to be paid more, the more they score.

We have set the stage now for linear regression (next week). Linear regression is all about further exploring the relationship between two variables, one outcome (often called "y") and one independent variable (often called "x"). Independent variables have many names. They are sometimes called "covariates"; "predictors"; "exposure", depending on the context.

There are a number of cool things that regression can do for us that simple EDA cannot:

1) It can model the relationship between two variables while considering simultaneously the potential influence of other factors. Image we are interested in the effect of points per game on salary regardless of position, season or team. With regression we can estimate how much more a player would ear every season if he scored 10 more points a game (regardless of the position he plays).

2) It can assess what explains the effect of one variable on another (i.e. mediation). Maybe we find that point guards earn less and we want to know

why. Is it because they score less? Is it because they play less time on average?

3) It can be used to predict salaries for players for whom we don't know the salary or even for hypothetical players. We could also look at the performance trend of players and predict whether they earn more next season or not.

Chapter 3

Exploratory Data Analysis - II

This week we will try to apply our last weeks knowledge into analysis.

3.1 Markdown Introduction

R Markdown is a powerful tool that allows you to create dynamic documents, presentations, and reports using R code. It combines the core syntax of markdown (an easy-to-write plain text format) with embedded R code chunks that are run when the document is rendered[1].

R Markdown documents are fully reproducible, meaning that anyone can re-run the code and generate the same results. This makes it easy to share your work with others and ensure that your results are accurate and reliable.

One of the great things about R Markdown is its flexibility. You can use it to create a wide variety of output formats, including HTML, PDF, and Microsoft Word documents. You can even create interactive documents with Shiny components[1].

To get started with R Markdown, you'll need to install the `rmarkdown` package from CRAN. This can be done by running the command `install.packages("rmarkdown")` in the R console[2]. Once you have the package installed, you can create a new R Markdown document in RStudio by going to `File > New File > R Markdown`.

An R Markdown document is made up of text written in markdown syntax and chunks of R code. When you render the document, the R code is executed and its output (such as plots or tables) is inserted into the final document[1].

When you render this document, the text and code will be combined to create an HTML file that includes both the markdown text and the output of the R code chunks.

You can easily add images to an R Markdown document using the standard markdown syntax for images. The basic syntax for adding an image is `![Alt text](image_url)`, where `Alt text` is the text that will be displayed if the image cannot be loaded, and `image_url` is the URL of the image you want to include.

I hope this helps you understand how to add images to an R Markdown document! Let me know if you have any further questions

R Markdown is a powerful tool that offers many advantages for data analysis and reporting. Some of the key benefits of using R Markdown include:

1. **Reproducibility**: R Markdown documents are fully reproducible, meaning that anyone can re-run the code and generate the same results. This makes it easy to share your work with others and ensure that your results are accurate and reliable.

2. **Flexibility**: R Markdown is incredibly flexible and can be used to create a wide variety of output formats, including HTML, PDF, and Microsoft Word documents. You can even create interactive documents with Shiny components.

3. **Ease of use**: R Markdown is easy to use, even for people with little or no programming experience. The core syntax of markdown is simple and intuitive, and the ability to embed R code directly into the document makes it easy to include dynamic content.

4. **Integration with R**: R Markdown is tightly integrated with R, making it easy to access and use the full power of the R language for data analysis and visualization.

5. **Collaboration**: R Markdown makes it easy to collaborate with others on data analysis projects. You can share your code and results with others, and they can easily reproduce your work and build on it.

Overall, R Markdown is a powerful tool that offers many advantages for data analysis and reporting. It's a great way to create dynamic, reproducible documents that are easy to share and collaborate on

I hope this introduction helps you understand what R Markdown is and how it can be used. If you want to learn more, there are many great resources available online, including the R Markdown website

3.2 Applying EDA(WVS/own data)

3.2.1 Exercise - 1

Boston Housing Dataset

Housing data contains 506 census tracts of Boston from the 1970 census. The dataframe BostonHousing contains the original data by Harrison and Rubinfeld (1979), the dataframe BostonHousing2 the corrected version with additional spatial information.

The dataset has two prototasks: `nox`, in which the nitrous oxide level is to be predicted; and `price`, in which the median value of a home is to be predicted.

Other Details: *Origin* -The origin of the boston housing data is Natural. *Usage* -This dataset may be used for Assessment. *Number of Cases* -The dataset contains a total of 506 cases. *Order* -The order of the cases is mysterious. *Variables* -There are 14 attributes in each case of the dataset. They are: 1. CRIM - per capita crime rate by town 2. ZN - proportion of residential land zoned for lots over 25,000 sq.ft. 3. INDUS - proportion of non-retail business acres per town. 4. CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise) 5. NOX - nitric oxides concentration (parts per 10 million) 6. RM - average number of rooms per dwelling 7. AGE - proportion of owner-occupied units built prior to 1940 8. DIS - weighted distances to five Boston employment centres 9. RAD - index of accessibility to radial highways 10. TAX - full-value property-tax rate per $10,000 11. PTRATIO - pupil-teacher ratio by town 12. B - 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town 13. LSTAT - % lower status of the population 14. MEDV - Median value of owner-occupied homes in $1000's

You can include this data by installing mlbench library:

```
#install.packages("mlbench") ## installing the library
library(mlbench) #adding the library
library(openxlsx)
library(dplyr)
data(BostonHousing2)
housing <- BostonHousing2
write.xlsx(housing, "../datasets/boston.xlsx")
```

1. **Read the dataset**: Read the Boston Housing Dataset from the Excel file.

```
library(readxl)
BostonHousing <- read_excel("../datasets/boston.xlsx")
```

2. **Inspect the dataset**: Use the proper functions to inspect the structure and contents of the dataset. How many Categorical variables are there? How many numerical variables are there? Is there any null values?

```
str(BostonHousing)
```

```
## tibble [506 x 19] (S3: tbl_df/tbl/data.frame)
##  $ town   : chr [1:506] "Nahant" "Swampscott" "Swampscott" "Marblehead" ...
##  $ tract  : num [1:506] 2011 2021 2022 2031 2032 ...
##  $ lon    : num [1:506] -71 -71 -70.9 -70.9 -70.9 ...
##  $ lat    : num [1:506] 42.3 42.3 42.3 42.3 42.3 ...
##  $ medv   : num [1:506] 24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
##  $ cmedv  : num [1:506] 24 21.6 34.7 33.4 36.2 28.7 22.9 22.1 16.5 18.9 ...
##  $ crim   : num [1:506] 0.00632 0.02731 0.02729 0.03237 0.06905 ...
##  $ zn     : num [1:506] 18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...
##  $ indus  : num [1:506] 2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
##  $ chas   : chr [1:506] "0" "0" "0" "0" ...
##  $ nox    : num [1:506] 0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524
##  $ rm     : num [1:506] 6.58 6.42 7.18 7 7.15 ...
##  $ age    : num [1:506] 65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
##  $ dis    : num [1:506] 4.09 4.97 4.97 6.06 6.06 ...
##  $ rad    : num [1:506] 1 2 2 3 3 3 5 5 5 5 ...
##  $ tax    : num [1:506] 296 242 242 222 222 222 311 311 311 311 ...
##  $ ptratio: num [1:506] 15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
##  $ b      : num [1:506] 397 397 393 395 397 ...
##  $ lstat  : num [1:506] 4.98 9.14 4.03 2.94 5.33 ...
```

```
glimpse(BostonHousing)
```

```
## Rows: 506
## Columns: 19
## $ town    <chr> "Nahant", "Swampscott", "Swampscott", "Marblehead", "Marblehea~
## $ tract   <dbl> 2011, 2021, 2022, 2031, 2032, 2033, 2041, 2042, 2043, 2044, 20~
## $ lon     <dbl> -70.9550, -70.9500, -70.9360, -70.9280, -70.9220, -70.9165, -7~
## $ lat     <dbl> 42.2550, 42.2875, 42.2830, 42.2930, 42.2980, 42.3040, 42.2970,~
## $ medv    <dbl> 24.0, 21.6, 34.7, 33.4, 36.2, 28.7, 22.9, 27.1, 16.5, 18.9, 15~
## $ cmedv   <dbl> 24.0, 21.6, 34.7, 33.4, 36.2, 28.7, 22.9, 22.1, 16.5, 18.9, 15~
## $ crim    <dbl> 0.00632, 0.02731, 0.02729, 0.03237, 0.06905, 0.02985, 0.08829,~
## $ zn      <dbl> 18.0, 0.0, 0.0, 0.0, 0.0, 0.0, 12.5, 12.5, 12.5, 12.5, 12.5, 1~
## $ indus   <dbl> 2.31, 7.07, 7.07, 2.18, 2.18, 2.18, 7.87, 7.87, 7.87, 7.87, 7.~
## $ chas    <chr> "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0~
## $ nox     <dbl> 0.538, 0.469, 0.469, 0.458, 0.458, 0.458, 0.524, 0.524, 0.524,~
## $ rm      <dbl> 6.575, 6.421, 7.185, 6.998, 7.147, 6.430, 6.012, 6.172, 5.631,~
## $ age     <dbl> 65.2, 78.9, 61.1, 45.8, 54.2, 58.7, 66.6, 96.1, 100.0, 85.9, 9~
```

```
## $ dis     <dbl> 4.0900, 4.9671, 4.9671, 6.0622, 6.0622, 6.0622, 5.5605, 5.9505~
## $ rad     <dbl> 1, 2, 2, 3, 3, 3, 5, 5, 5, 5, 5, 5, 5, 4, 4, 4, 4, 4, 4, 4, 4,~
## $ tax     <dbl> 296, 242, 242, 222, 222, 222, 311, 311, 311, 311, 311, 311, 31~
## $ ptratio <dbl> 15.3, 17.8, 17.8, 18.7, 18.7, 18.7, 15.2, 15.2, 15.2, 15.2, 15~
## $ b       <dbl> 396.90, 396.90, 392.83, 394.63, 396.90, 394.12, 395.60, 396.90~
## $ lstat   <dbl> 4.98, 9.14, 4.03, 2.94, 5.33, 5.21, 12.43, 19.15, 29.93, 17.10~
```

```r
# Check for missing values in all columns
colSums(is.na(BostonHousing))
```

```
##    town   tract     lon     lat    medv   cmedv    crim      zn   indus    chas
##       0       0       0       0       0       0       0       0       0       0
##     nox      rm     age     dis     rad     tax ptratio       b   lstat
##       0       0       0       0       0       0       0       0       0
```

3. **Summarize categorical variables**: Create frequency table for categorical variables CHAS (Charles River dummy variable).

```r
table(BostonHousing$chas)
```

```
##
##   0   1
## 471  35
```

4. **Summarize numerical variables**: Generate summary statistics for numerical variables CRIM (per capita crime rate by town) and RM (average number of rooms per dwelling).

```r
summary(BostonHousing$crim)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##   0.00632   0.08204   0.25651   3.61352   3.67708 88.97620
```

```r
summary(BostonHousing$rm)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.561   5.886   6.208   6.285   6.623   8.780
```

5. **Create new variables**: Create new variable indicating whether a house is located near the Charles River or not.

```
library(dplyr)
BostonHousing <- BostonHousing %>%
  mutate(near_river = case_when(chas == 1 ~ "Yes",
                                chas != 1 ~ "No"))
```

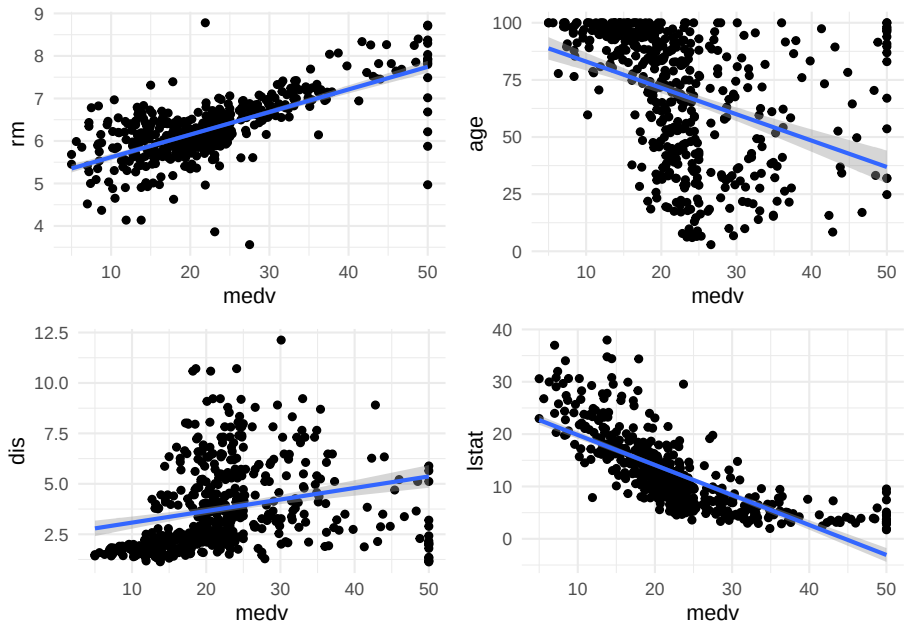
6. **Recoding variables**: Create a new variable that groups houses by their proximity to employment centers.

```
BostonHousing <- BostonHousing %>%
  mutate(distance_group = case_when(
    dis < 2 ~ "Near",
    dis >= 2 & dis < 5 ~ "Medium",
    dis >= 5 ~ "Far"
  ))
```

7. **Visualize relationships**: Create scatter plots to explore the relationship between housing prices and other numeric variables(`rm`, `age`, `dis`, `lstat`). Interpret the plots. Also show the distribution of ptratio by distance_group.

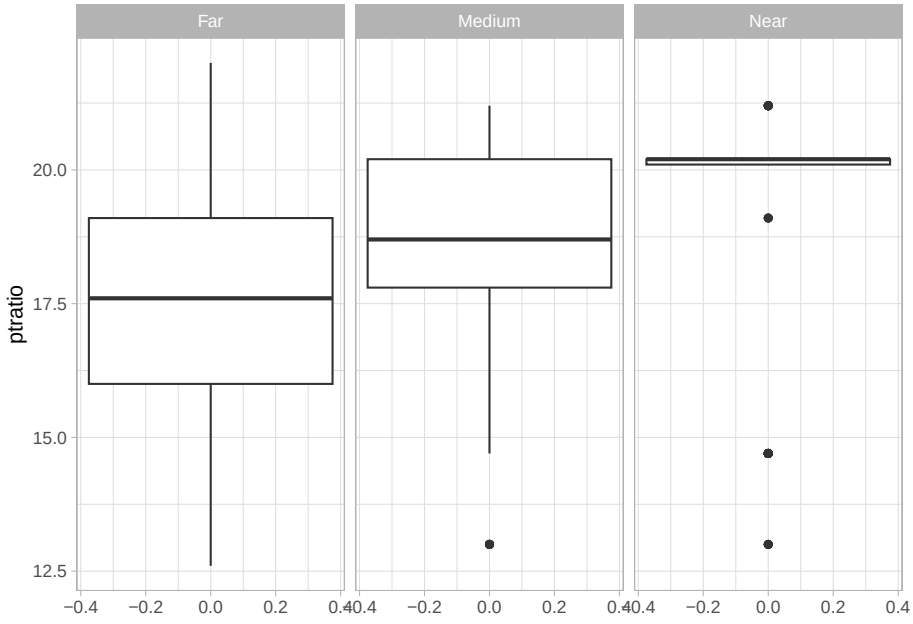
```
# Create a list of numeric variables to plot against MEDV
vars <- c("rm", "age", "dis", "lstat")

# Create a scatter plot for each variable
#rm
rm_sp <- BostonHousing %>%
  ggplot(aes(y=rm, x=medv)) +
    geom_point() +
     stat_smooth(method = "lm") +
       theme_minimal()
#age
age_sp <- BostonHousing %>%
  ggplot(aes(y=age, x=medv)) +
    geom_point() +
     stat_smooth(method = "lm") +
       theme_minimal()
#dis
dis_sp <- BostonHousing %>%
  ggplot(aes(y=dis, x=medv)) +
    geom_point() +
     stat_smooth(method = "lm") +
       theme_minimal()
#lstat
lstat_sp <- BostonHousing %>%
```

```
  ggplot(aes(y=lstat, x=medv)) +
    geom_point() +
     stat_smooth(method = "lm") +
       theme_minimal()
#adding all plots in one
library("ggpubr")
ggarrange(rm_sp, age_sp,
          dis_sp, lstat_sp,
          ncol = 2, nrow = 2)
```



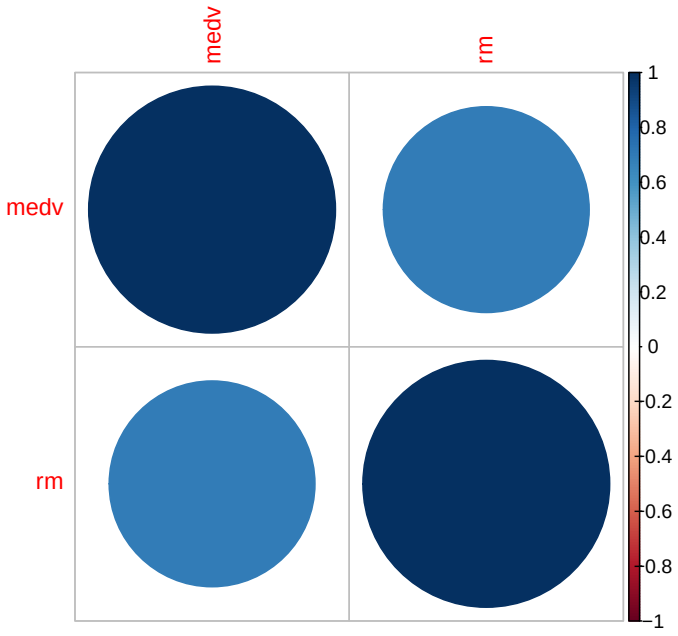
boxplot

```
# Create a boxplot of PTRATIO grouped by distance_group
BostonHousing %>% ggplot() +
geom_boxplot(aes(y=ptratio)) +
  facet_wrap(~distance_group) +
  theme_light()
```

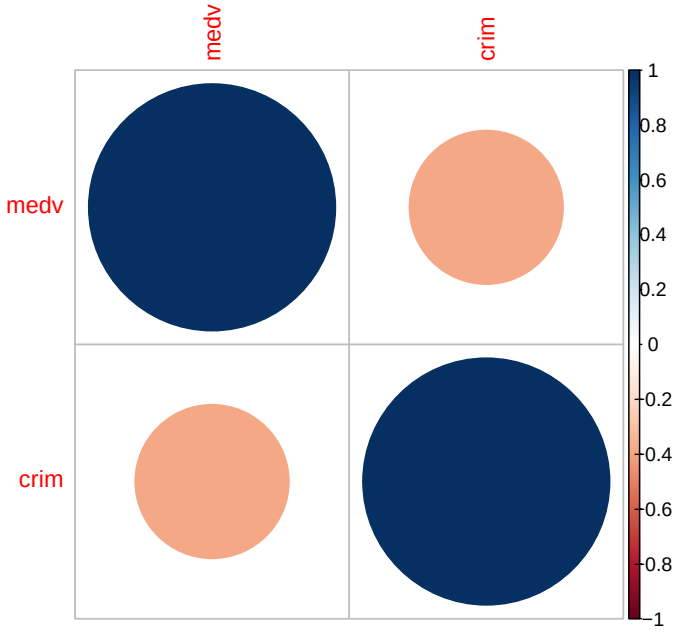


8. **Correlation analysis**: Calculate correlation coefficients between housing prices and average number of rooms per dwelling, also between housing prices and crime rate. Interpret the result.

```
library(corrplot)
corr_matrix0 <- cor(BostonHousing[, c("medv", "rm")])
corr_matrix1 <- cor(BostonHousing[, c("medv", "crim")])
corrplot(corr_matrix0)
```




```
corrplot(corr_matrix1)
```



Chapter 4

DAGs

4.1 Objectives

- Getting to know DAGs
- Understand its implications on achieving unbiased estimates for effects of interest
- Building a DAG for the NBA data to model the effect of scored points on salary (in session 8)

4.2 Modelling

This session kicks off the block on linear regression as a statistic modelling technique, but before we jump into the deep end and learn what linear regression is and how we can apply it, we first have to understand what modelling is and why we might do it. We also need a way to find out what we actually have to include in the model and what we might not want or even can not include to achieve robust results. This is what this session is all about.

4.2.1 What is modelling?

Before we approach this question, we should briefly think about what steps a typical data analysis project comprises. We usually start with an interesting problem and derive a research question from it. Based on this question we would go into the literature and read up on theories and already published research papers that are relevant to our question. We construct a theoretical framework for our particular problem and formulate some hypotheses, collect or identify appropriate data and conduct an exploratory data analysis. At this point we

should already have a firm understanding about what we actually want to find out and how our data is structured. The next step would be modelling.

So what is modelling? In general we have one *dependent variable*, typically denoted as y. This variable has some varying values. Our goal in modelling is to estimate how these values are generated. Generating here refers to the *data generating process* that we assume responsible for y having the values it has. One or multiple *independent variables*, x_1 x_2 ... x_k, influence how the values of y are generated; thus y *depends* on the values of our independent variable(s).

When we model, we do not know the data generating process, but based on theoretical considerations, careful thinking and exploratory data analysis, we can make assumptions on how we think the process operates. DAGs are a tool that can assist us in this step. We can use it to formally clarify our assumptions on the data generating process and to formulate a model based on its implications.

4.2.2 Estimating effects vs. prediction

There are two main reasons for modelling in the social sciences.

Our goal can be *prediction*, as in predicting our dependent variable y with the highest possible accuracy. This maybe is the less classical approach to modelling, but one that has come to the forefront in recent years, especially in the context of *machine learning*.

Take ChatGPT for instance. The underlying GPT model, a large language model (LLM), is used to predict what the next word in a sequence of words should be. Based on the context of the question and the prior words in the answer, which we can understand as independent variables for our example, it calculates what word has the highest probability of being the correct next one. It is all about prediction.

An example closer to home are annotations for text data. Imagine you have a lot of text, hundred thousands of social media posts, and you want to explore the sentiment expressed in those. Do they lean to the positive or to the negative? You can now go and take a "small" sample, let us say a few thousands and annotate them by hand. A lot of work, but based on those manually annotated posts you can train a machine learning model that learns from those posts and then, if everything goes well, is able to automatically annotate the remaining hundred thousands of posts for you. Again, this is all about prediction; here predicting the sentiment of a post, the dependent variable, based on the words it contains, the independent variables.

When prediction is our goal, we most often are not primarily interested in understanding what independent variables influence the dependent variables in which direction and with which magnitude. We are interested in the most accurate prediction for the dependent variable possible. These approaches are therefore also called *y-centered*.

When our interest is focused on one or multiple independent variables, our approach is *x-centered*. This is the more classical usage of modelling in statistics, at least for the social sciences. Here our goal is to estimate an effect of interest as accurately as possible. y is still our dependent variable but our focus lies on understanding which x variables influence y, in which direction this influence goes and what the magnitude is.

Let us say we are interested in why people cast their vote for a certain party. We may have some hypotheses that proposes that voters who find certain issues important have a higher probability of casting their ballot for this party. Our interest would not be predicting the vote accurately but explaining why someone votes the way they do. We can build a model from our assumptions, maybe there are other important factors that correlate with the issues and the vote, and test our hypotheses based on the results. Does holding certain issues important really increase the chances of voting for this party or is there no effect?

Over the last sessions we build an interest in the relationship between points scored and the salary received for NBA players. We could approach this as a prediction problem, i.e. trying to predict the salary as accurately as possible based on a model that incorporates the scored points as well as other factors that we assume of having an effect on the salary. We will return to this in session 11. We could also approach this as estimation problem, i.e. trying to estimate the effect of scored points on the salary. We will most probably have to include other variables that affect the relationship of score on salary as well, but the model used will not necessarily be the same. This approach is what we will tackle in this and the next 5 sessions.

Having settled on estimating the effect of score on salary, how can we find out which variables we have to include in the model? The first step, and we can never replace this with ever so fancy a statistical technique, is thinking about the problem. We should also have a theoretical understanding of our problem, know the current research on the topic and do some exploratory data analysis. Based on this we will already have developed some assumptions concerning our proposed underlying data generating process. Should we now throw everything into our model that we deem interesting or relevant for the relationship between score and salary? No, we should not. What we should do is use a tool that helps us formalise our assumptions and figure out which variables are actually relevant for measuring our effect of interest; and which variables we can not include in our model as they would potentially lead to incorrect estimations. This is where DAGs come in.

4.3 DAGs

4.3.1 Directed acyclical graphs

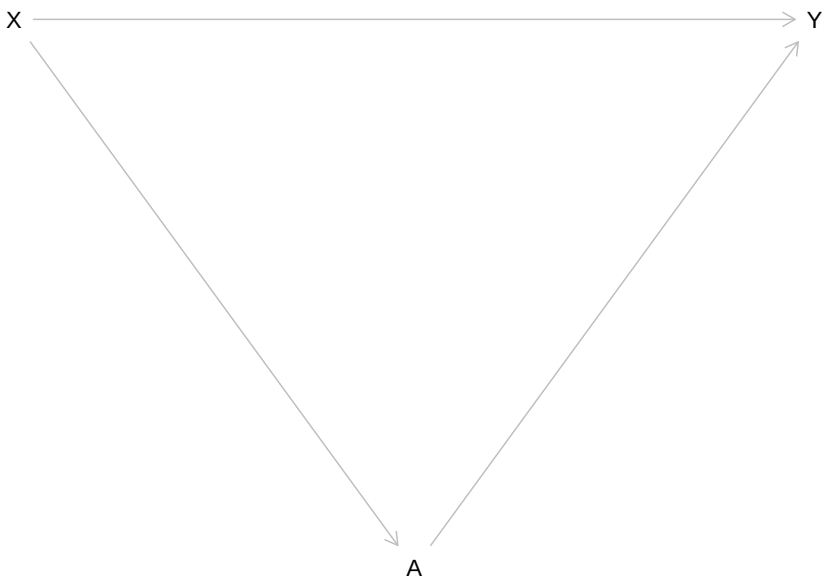
DAG is short for *directed acyclical graph*. DAGs are *graphs* that display the assumed relationship between variables as arrows, or missing arrows, between them. An arrow represents our assumption that one variable has an effect on the other, a missing arrow represents our assumption that one variable has no effect on the other. These arrows are *directed*. This represents our assumptions about the direction of the effect. We do not only assume that two variables are somehow associated, but we explicitly state which one influences the other.

A simple DAG could look like this:

What are the assumptions about the data generating process we have encoded here? We have one dependent variable X that we assume to have a direct effect on the independent variable Y. We know that our assumption was that X has an effect on Y and not the other way around, because the arrow is *directed* from X to Y.

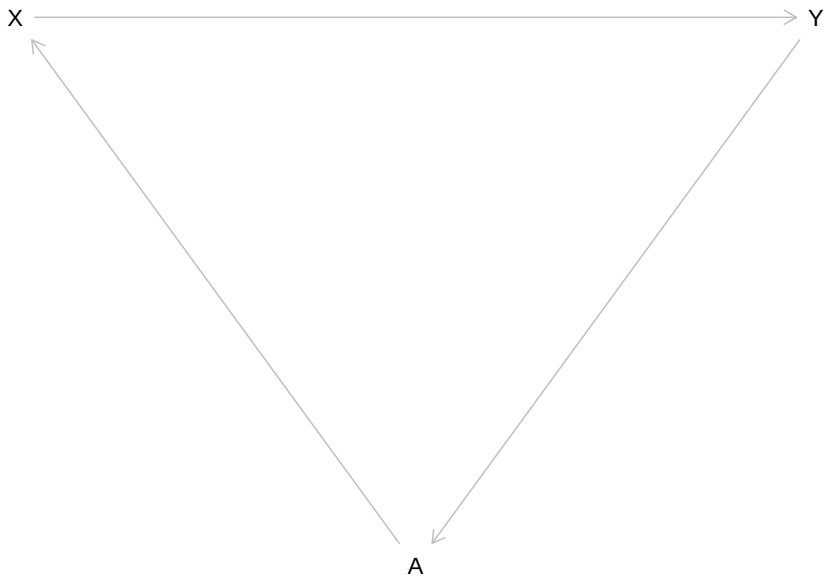
We call the sequence of one or many arrows that do not pass a single variable more than once a *path*. When trying to estimate an effect of interest we are foremost interested in the paths going from our independent variable of interest to the dependent variable. In the first example we only have the one path $X \rightarrow Y$, but in most "real" DAGs we will have multiple paths between X and Y.

Consider this DAG, were we introduce a second variable A:



Here there are two paths from X to Y, $X \rightarrow Y$ and $X \rightarrow A \rightarrow Y$. Our assumption here was that X directly influences Y, but that X also directly influences A which in turn directly influences Y. The latter is an indirect effect from X on Y through A.

DAGs are also *acyclical*, meaning that there can not be any cyclical relationships between variables. A cyclical relationship would be present if we start from one variable and follow a path that leads us back to this variable.



In this example there is a path $X \to Y \to A \to X$ that leads us from X back to X. This is not allowed in a DAG.

Now that we know the basics, what do we actually do with a DAG? A DAG is a way to graphically formalise our assumptions about the data generating process. But it is about more than drawing nice formalisations, it is also about figuring out which variables we have to include and which we are not allowed to include to get an unbiased estimate of our effect of interest. We do this by *blocking* all paths from X to Y that do not represent the relationship we want to estimate and at the same time opening up all paths that do. For this to make sense, we need to know the three patterns of relationships between a set of three variables and how we can open or close paths with them.

4.3.2 Patterns of relationships

4.3.2.1 Chains/Pipe

Three variables can be connected in a *chain* or *pipe* like this:

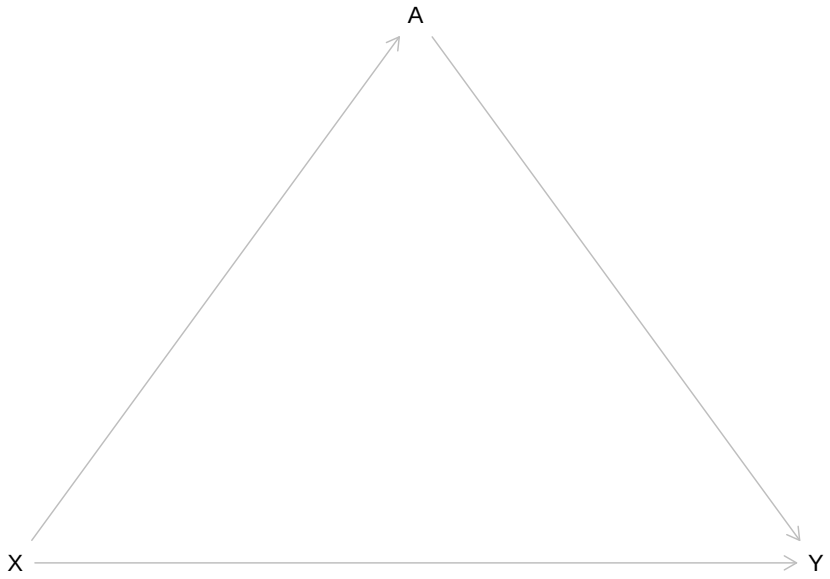
X has an effect on A which in turn affects the value of Y. This implies that X and Y are statistically correlated. When the value of X changes, the value of Y also changes, "transmitted" by the indirect effect X has on Y through A. Remember we are still interested in measuring the effect of X on Y, so we also want to measure this indirect effect.

The DAG tells us that there is a relationship between all three variables. We therefore could be tempted to include A in our model as well. But what would happen is that by including A we would *block* the path between X and Y. We would not be able to measure the association we are actually interested in. Including such a variable and thereby blocking a path of interest is called *overcontrol bias*.

In some cases overcontrolling can make the effect of interest unmeasurable. We would then conclude from our analysis that X has no effect on Y and that our hypotheses was wrong, while there actually could be an effect that we made "disappear" by blocking its path. Drawing a DAG based on our assumptions helps us to prevent this pitfall.

4.3.2.1.1 Mediation A special case of pipes is *mediation*. We will only take about this briefly here and return to the topic in session 10.

Consider this DAG:



We see a direct effect through the path $X \to Y$ as well as indirect effect through $X \to A \to Y$. Should we include A in our model? This depends on what we want to measure.

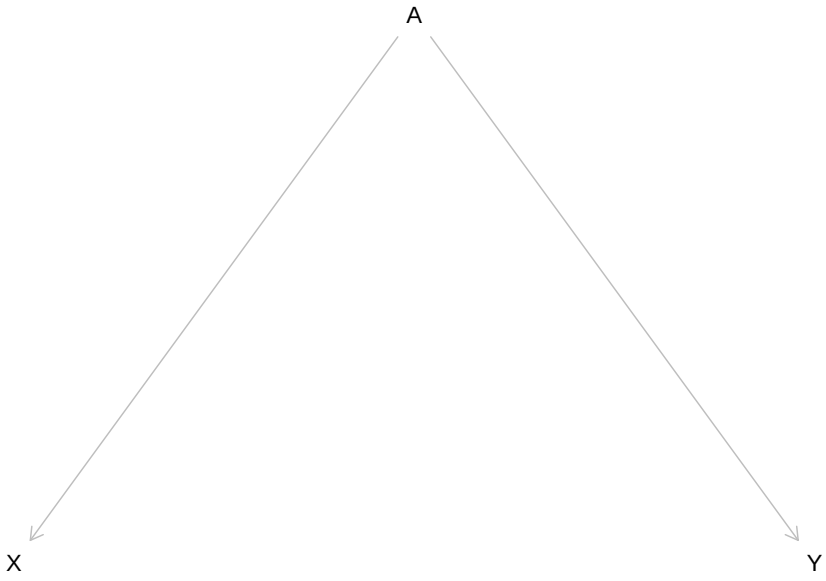
If we are interested in the *total effect* of X on Y, we should not. Both effects, the direct and the indirect paths, are of interest here, so we keep both paths open by not controlling for A.

Our interest could also lie exclusively in the *direct effect*. Here we would want to measure the effect of X stripped by all indirect effects. In this case we want the path $X \to Y$ to keep open, but we would close the path $X \to A \to Y$ by controlling for A.

We could also only be interested in the *indirect effect*, the effect of X on Y that goes through A. We can not directly model this, but we can compute the indirect effect as the difference between total and direct effect.

4.3.2.2 Confounders

The second pattern we may see is the *fork* or *confounder*.

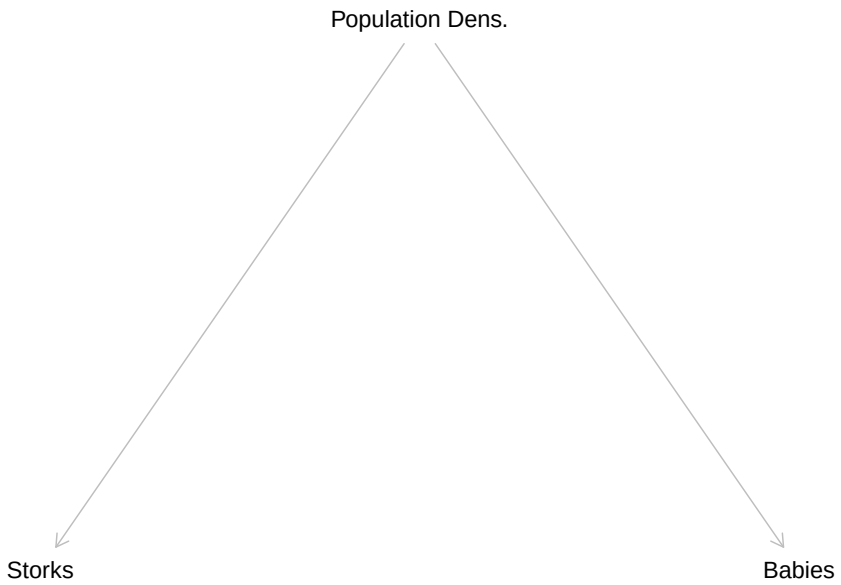


We see that there is no implied direct relationship between X and Y but that both variables are influenced by A. If we would measure the relationship between X and Y we *should* see no statistical correlation. The problem is, that we *would* see a correlation despite this. Why is that? The values of X and Y both depend on the value of A. If we for example assume that both effects are positive the value of X would rise with the value of A and the value of Y would rise with the value of A also. The opposite would be true if both effects were negative. Lower values for A would lead to lower values for X and Y. But even the effects would be opposite, they would never cancel each other out perfectly. X and Y vary together. If we just include X and Y in our model this would show up as an effect from X on Y.

In DAG terms, there is an open path between X and Y although we did not draw a direct path: $X \leftarrow A \rightarrow Y$. This may seem counterintuitive at first, as the arrows go in opposite directions, but for paths in a DAG to exist, the direction of the arrows does not matter. Every connection between variables is a path. So there is an open path, in this case a so called *backdoor path*, that leads to a statistical association between X and Y, but we can close it by controlling for A. If we do this, we would see no remaining association between X and Y and thus get an unbiased estimate for the effect of interest, i.e. no effect.

If this still seems unintuitive consider the following example, which you can also find in many statistical textbooks. Do storks bring babies? We could tackle this analytically by taking a measure of babies born in a region as our dependent variable and the number of storksightings in the same region as our dependent variable. Let statistics come to the rescue and help us discard the notion of

the baby bringing stork once and for all. But alas, our model will tell us that there is a positive effect from storksightings on the number of newborns. Should we conclude that everything we learned from our parents and teachers was one big lie to hide away the magical truth about reproduction? Before we do that, let us return to rationality. Maybe we have missed something important. It turns out that there is a confounder we did not include in our model. More rural areas have higher birthrates and also have a higher rate of storksightings while both variables have lower values in more urban regions. Our dependent and independent variables both vary by the value of the confounder and thus it seems that there is a correlation where there actually is none. If we now include a measure for the type of region, let us say population density, this spurious association disappears and we can return to normalitty. Storks do not bring babies after all.



4.3.2.3 Colliders

The last pattern we have to consider are *colliders*. A collider is a variable on a path that has two arrows pointing into it:

Here A is influenced by the value of X as well as Y. Again, there should be no effect of X on Y and in this case there is none if just include X and Y in our model. If we also include A we introduce an association between X and Y even if there should be none. This implies that we should not control for colliders because we would open up a path that creates a spurious association between two variables that are not related.

4.3.3 Adjustment set

Now we have all the building blocks for identifying which variables we have to include in our model and which we are not allowed to include. If we do not follow these rules we may statistically find relationships where there are none or miss relationships that actually exist. We then would draw the wrong conclusions for our hypotheses and research question. We would produce bad science.

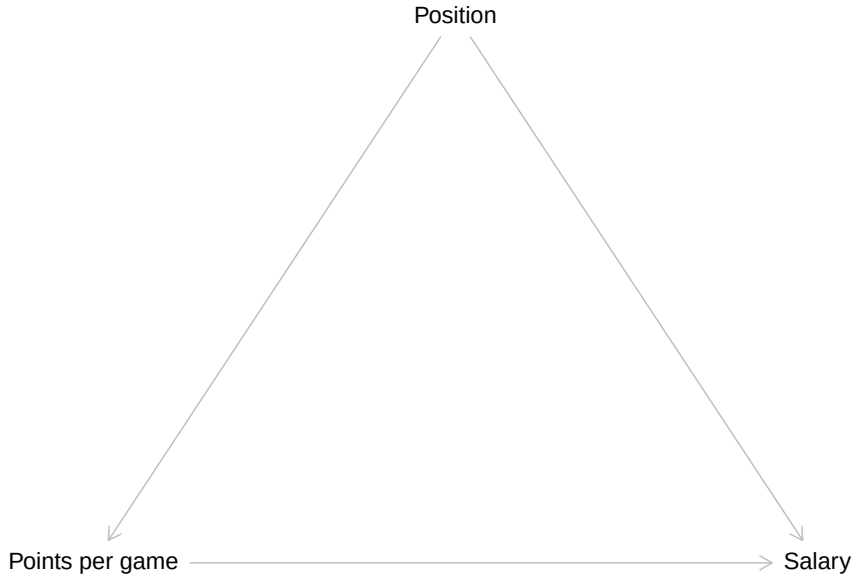
If we draw out our DAG and use its implications to identify the correct *adjustment set* Z of control variables, we do not fall into this trap. We only control what we have to, and nothing that we should not. We thus create the best model to get an unbiased estimate for our effect of interest; but there is always a caveat and this is a big one. The model is only correct if our DAG is also correct and we can never know for certain if it is. We could make wrong assumptions, forget important relationships, and make all matters of mistakes while building our DAG. While DAGs are a great tool for identifying the adjustment set, the technique alone can never replace careful thinking.

4.4 NBA DAG

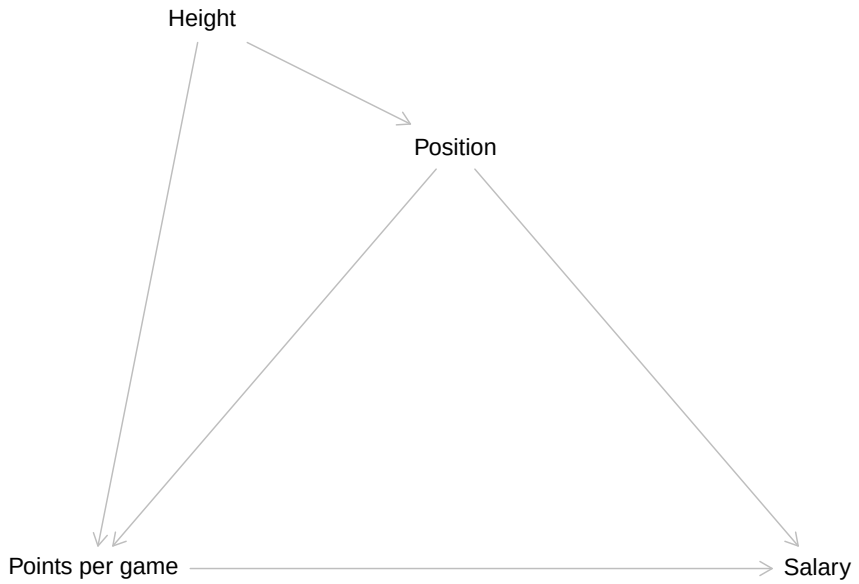
We will now pick up where we left off in session 2 and return to the NBA data. Equipped with our new tool we can now draw a DAG with the goal of building a model to estimate the effect of points scored on the salary a player receives. The assumption is, that the higher the point average, the higher the salary. This makes intuitive sense as a high scoring player is more valuable to the team and thus may receive a higher monetary compensation.

Let us start building a DAG with the information we already have.

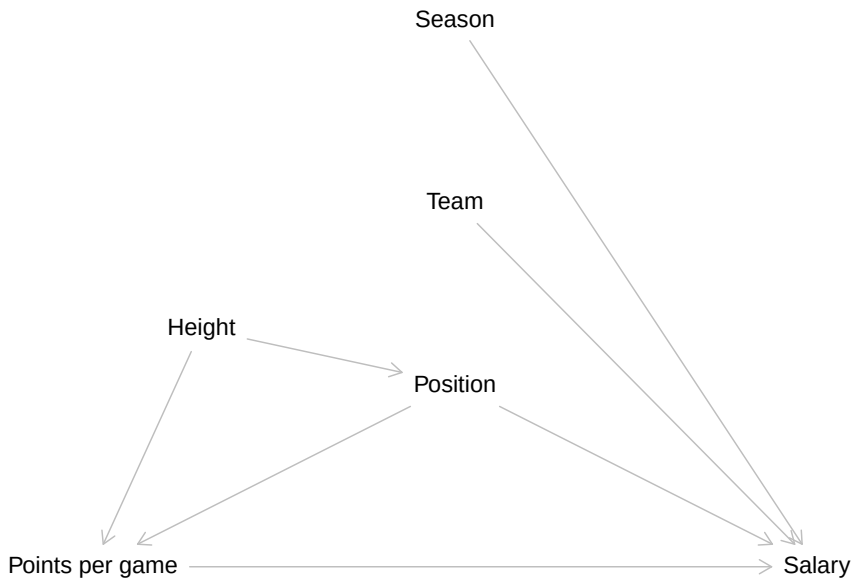
Now we have to think about other factors that could influence the relationship between points and salary. One variable we already identified as having an effect on both was the position a player occupies. The position influences how many points per game a player can score and we also already saw that centers make more money compared to point guards. Right now we have no reason to believe that other positions do not also have an effect on the received salary. Following this reasoning, position is a confounder for points and salary.



It is also reasonable that body height influences which position a player can occupy. Centers have to be big while smaller players tend to play on other positions. At the same time, height is an advantage if you want to score in a basketball game. Thus height is a confounder for points and position.



Another factor that will have an effect on the salary is the team a player plays for. More successful teams will be able to pay higher salaries. The season an observation was recorded in should also influence the paid salary as we can expect a general inflation of salaries over time.



This is our final DAG. Does it correctly depict the underlying data generating process? We do not know for sure, but the DAG correctly reflects our assumptions at this point. Those may be wrong. Maybe the data generating process actually works slightly or even completely different; but to the best of our current knowledge, this is how the scored points affect the salary a player receives.

We can now inspect what implications the DAG has for our model. To do this, let us list the paths from our independent to the dependent variable.

$$A : Points \rightarrow Salary$$

$$B : Points \leftarrow Position \rightarrow Salary$$

$$C : Points \leftarrow Height \rightarrow Position \rightarrow Salary$$

Path A is a direct path from our independent variable to our dependent variable. We will have to include scored points and salary into our model, but that is a given.

Paths B & C are both backdoor paths - these are easily spotable by an arrow pointing into the independent variable - that we have to address somehow. Let

us consider B first. Position is a confounder for points and salary. Above we already learned how to deal with this, we control for it. This removes the spurious part of the association between points and salary introduced by the confounder. Path C also includes a confounder, namely the body height of a player. Do we also control for this variable to close path C? No, we do not. If we further examine path C we will see that it also includes position as a pipe. When we control for a variable in a pipe, the path gets closed. As we have to include position to close path B, path C is also already closed.

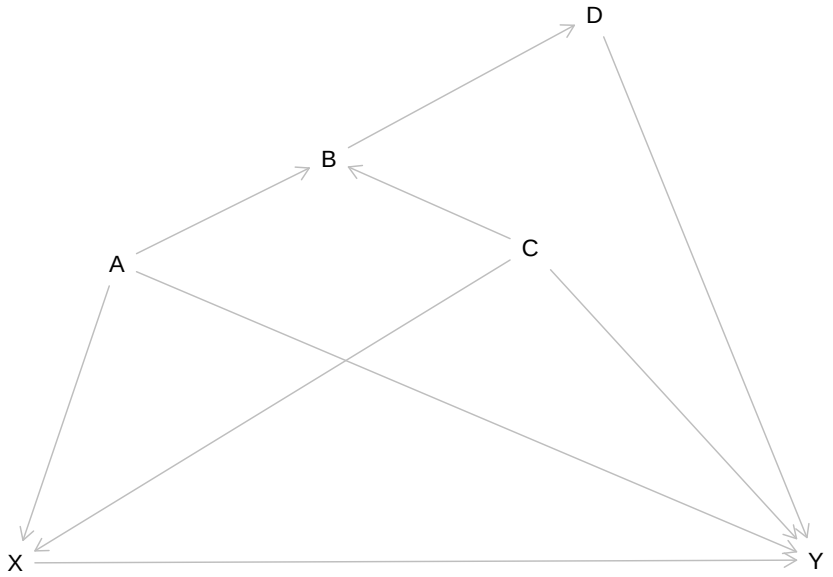
The two remaining variables, team and season, have direct effects on the salary but do not lie on a path from points to salary. This implies, that we do **not** have to control for them if our goal is estimating the effect of points on salary.

Above we briefly talked about the two possible goals of modelling. Here and over the next sessions our goal is estimating an effect of interest without bias. We used the DAG to identify an adjustment set of variables we have to control for to reach this goal. If our goal was predicting the salary as accurately as possible we would made other conclusions. The team a player is employed by and the season where an observation was measured are both relevant predictors for the received salary. For prediction we would have included both variables because this should increase the accuracy of the prediction. We will return to this in more detail in sessions 11 and 12.

4.5 Resources

4.5.1 dagitty.net

While the underlying rules of DAGs are relatively straightforward, identifying the adjustment set can get harder the more complex a DAG gets. Consider this for instance:



What variables do we have to include in our adjustment set to get unbiased measure for the effect from X on Y? We would suggest you take out pen and paper and try to figure it out. You can approach this as above. List all paths from X to Y that do not include a variable twice and try to find the adjustment set that closes all paths besides the one(s) needed to measure the effect from X on Y. Spoiler: In this case the only path we want to keep open is the direct effect $X \rightarrow Y$.

While this is doable, you may be unsure if you found the correct set. Luckily there is a convenient way to find out. https://dagitty.net/ provides a browser tool where you can draw DAGs - as well as export them for your papers - and check for the correct adjustment set. After launching the tool you should click on "Model" and then "New model". Now you can start drawing. New variables are added by clicking on the canvas and naming them, arrows are added by first clicking on the variable where the arrow should start and then on the variable where it should end. For everything to work you should also declare the independent variable of interest the "Exposure" and the dependent variable the "Outcome". Both can be set on the left under "Variable" When the DAG is drawn correctly, the top right panel "Causal effect identification" should show which variables need to be part of the adjustment set to estimate the effect of interest.

Note that "Causal effect identification" is set to display the adjustment set for the total effect. This also is what we are interested in here. If we are interested in mediation, we can also set the panel to display the set for the direct effect.

4.5.2 More on DAGs

If you are interested in diving deeper into DAGs, we can recommend this resources, which were also used for writing this session.

Richard McElreath provides a great introduction into the topic with many clear examples. While the book chapter on DAGs may require some knowledge of advanced statistical topics, the corresponding lecture on YouTube is more approachable:

McElreath, Richard (2020). Statistical Rethinking: A Bayesian Course with Examples in R and STAN. Second Edition. Boca Raton & Oxon: CRC Press. Chapter 5: The Many Variables & The Spurious Waffles, 123-160.

Corresponding YouTube lecture: https://www.youtube.com/watch?v=mBEA7PKDmiY

Felix Elwert wrote a concise paper on DAGs with a perspective that is more focused on causality than we have presented here:

Elwert, Felix (2013). Graphical Causal Models. In S. L. Morgan (Hrsg.), Handbook of Causal Analysis for Social Research, 245–273. Dordrecht [u.a.]: Springer.

If you really want to get into it, the work of Judea Pearl was central for establishing DAGs. An approachable starting point would be the "Book of Why":

Pearl, Judea & Dana Mackenzie (2018). The book of why : the new science of cause and effect. New York: Basic Books.

Chapter 5

Linear Regression Theory I: Simple Linear Regression

The next three sessions will be an introduction for linear regressions. We will look at the theoretical underpinnings, the interpretation of results and the underlying assumptions of these models. For this introduction we will keep the NBA data aside and use some simulated data that plays nice with us. We will return to the NBA data in session 8 applying everything we learned to assess the effect of scored points on salary.

5.1 Objectives

- Understand simple linear regression
- Understand the regression formula
- Interpret the results

5.2 What is Linear Regression

As we eluded to last session, there are two main approaches to using statistical modelling in the social sciences. The more classical approach is to use modelling for estimating the effect that one or several independent variables have on one dependent variable. Maybe we are interested in knowing if a higher income has an effect on life satisfaction and if yes, what the direction and magnitude of this effect is. Does more money actually make you happier?

The other and more recent approach is to use modelling for making predictions with high accuracy. Based on the relationships between many independent

variables and one dependent variable. We try to predict the latter for actual or hypothetical cases based on their values for the independent variables. This approach lies at the heart of *machine learning* and drives many of the technologies we use on a daily basis from E-Mail spam filters to ChatGPT. Returning to the example above, we are not interested in measuring the effect of money on life satisfaction, but in predicting the value for life satisfaction based on money and a host of other variables as accurately as possible.

Linear regression is one of the many available modelling techniques and it can serve both approaches. Over the next sessions we will focus on using linear regression for estimating an effect of interest but we will return to prediction in session 11 & 12.

How do we know if we should choose linear regression for a specific task? This is not easy to answer as there are many alternatives and even variations of linear regression which may be better suited for a specific empirical problem. As this is an introduction to modelling and time is of the essence we opted to solely focus on linear regression. This technique is suited for many problems and is comparably easy to understand and use. Also, after learning the ins and outs of linear regression, we are in a good position to build upon that knowledge and learn all of those more complex and specific models that we will encounter in textbooks and scientific papers.

With the pool of options trimmed down to one, the central question remains unanswered. Should I use linear regression for my task? As we have no alternatives to chose from, we can change the question to: Can I use linear regression for my task? The answer no mostly depends on what type of dependent variable we want to use. If it is metric, we can use linear regression. In our cases, the simulated data and our NBA data, our dependent variables both are metric. If we had other types of dependent variables, e.g. binary or categorical, we would have to use different models. We will give you some pointers for these at a later point.

5.3 Examplary research question & data

For this introduction, et us imagine that we are interested in a research question that asks what makes a good grade in a seminar paper. In particular we are interested in the effect that the hours a student invests in working on it has on the grade. Based on some theoretical considerations, and maybe some idealistic views, we derive our main hypotheses that putting in more hours will result in a better grade.

Now we also - hypothetically - held a small survey and asked 200 imaginary students some questions on how they approached writing a seminar paper. In particular we asked them how much time they spent working on the paper, if they have attended (almost) all seminar sessions, how closely they worked with

their lecturers in preparing the paper and what the mean grade for previous papers was. As these imaginary students have already turned in their papers, we also know the grades they achieved.

Please note, that this is data on **imaginary** students, meaning we have simulated the data making some assumptions on how to achieve a good (or bad) grade in a paper. The assumptions we made do not necessarily reflect the way *you* write a good paper, while still being based in our experience on what it takes to achieve a good grade. But remember, no real students were harmed in making up this data.

Let us have a first look on the data: XXX ALIGN WITH EDA XXX

```
## Warning: package 'skimr' was built under R version 4.2.3
```

```
##
## Attaching package: 'skimr'
```

```
## The following object is masked from 'package:corrr':
##
##     focus
```

```
## Warning: package 'knitr' was built under R version 4.2.3
```

skim_type	skim_variable	n_missing	complete_rate	factor.ordered	factor.n_unique	factor
factor	contact	0	1	FALSE	3	No :
logical	attendance	0	1	NA	NA	NA
numeric	grade	0	1	NA	NA	NA
numeric	hours	0	1	NA	NA	NA
numeric	previous_grades	0	1	NA	NA	NA
numeric	previous_grades_centered	0	1	NA	NA	NA
numeric	hours_centered	0	1	NA	NA	NA

Right now, the observations are ordered by the grade of the seminar paper which run from 1.0 to 5.0 in increments of 0.1. While this is somewhat unrealistic - the German grading system actually only uses the increments .0, .3 and .7 - simulating the data in this way will make the demonstrations on linear regression easier and more straightforward. The variable `previous_grades` is set up in the same way and represents the mean of the grades the student received up to this point. `hours` represents the time a student spent on writing the paper, ranging from $23 - 57$ hours, with a mean of about 40. Besides these metric variables, the data set also contains two categorical measures. `attendance` is a *binary* or *dummy variable*, meaning it can only have the values 1 or 0 or `TRUE` and `FALSE` in this case, as it is saved as a logical variable. `TRUE` represents that a student attended almost all seminar sessions before writing the paper - which about 77 did -, `FALSE` states that they did not. `contact` is a factor variable with three

categories and shows the answers to the imaginary question on how much contact the student had to the lecturer before starting the writing process. Besides `No contact` the students could have had `E-Mail` contact to state their research question and get some short written feedback or meet the lecturer `In Person` to achieve a deeper discussion of the question and laid out plan for writing the paper. The two additional variables are versions of `previous_grades` and `hours` that are centered on their respective means. They will come into play at a later point in this session.

Let's have a look at some observations.

```
## # A tibble: 10 x 7
##    grade hours previous_grades attendance contact   previous_grades_centered
##    <dbl> <int>           <dbl> <lgl>      <fct>                       <dbl>
##  1     1    50             1.4 TRUE       E-Mail                     -1.54
##  2     1    46             1   TRUE       E-Mail                     -1.94
##  3     1    42             1   TRUE       In Person                  -1.94
##  4     1    49             1   FALSE      In Person                  -1.94
##  5     1    42             1.2 TRUE       In Person                  -1.74
##  6     1    46             1.8 TRUE       In Person                  -1.14
##  7     1    44             1.4 FALSE      In Person                  -1.54
##  8     1    45             2   TRUE       In Person                  -0.935
##  9     1    48             1   TRUE       In Person                  -1.94
## 10     1    45             2   TRUE       In Person                  -0.935
## # i 1 more variable: hours_centered <dbl>
```

From this first 10 rows, we can see that the students with the best grades spent more than 40 hours on writing, have already achieved good grades in their papers up to this point and at least had some contact to the lecturers. Most also regularly attended the seminar but two did not and still achieved a 1.0 in their grade.

So what makes a bad grade?

```
## # A tibble: 10 x 7
##    grade hours previous_grades attendance contact    previous_grades_centered
##    <dbl> <int>           <dbl> <lgl>      <fct>                         <dbl>
##  1   4.8    37             4.2 TRUE       No contact                    1.27
##  2   4.8    38             4.3 TRUE       E-Mail                        1.36
##  3   4.8    35             4.4 TRUE       E-Mail                        1.47
##  4   4.9    40             4.2 TRUE       E-Mail                        1.27
##  5   5      35             3.9 FALSE      No contact                    0.965
##  6   5      41             4.9 TRUE       No contact                    1.97
##  7   5      24             4.7 TRUE       E-Mail                        1.76
##  8   5      33             5   TRUE       E-Mail                        2.06
##  9   5      29             4.1 FALSE      E-Mail                        1.16
```

```
## 10   5      50             4.6 FALSE      E-Mail                      1.66
## # i 1 more variable: hours_centered <dbl>
```

Here the picture seems less clear. While most students did not put in as many hours, some did and still failed to pass. Half of the students that received a 5.0 regularly attended and most at least had E-Mail contact before writing their paper. What seems to be more consistent though is that the mean of the previous grades is rather low.

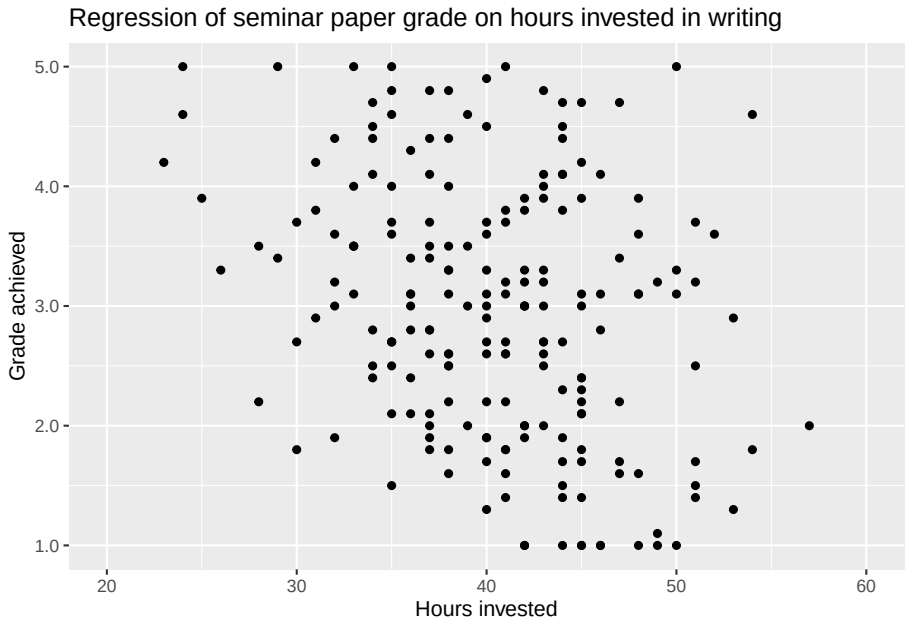
So what do we know now? Does a good or bad track record in grades predict all future grades? This seems not only unrealistic but is also a kind of sad take home message. To get a better understanding on which of the potential influential variables has an effect on the final grade and what the magnitude and direction of these effects is, we now turn to linear regression.

5.4 Simple Linear Regression

In a *simple linear regression*, the model is used to describe the relationship between *one* dependent and *one* independent or explanatory variable. The question this model can answer for us is, by how much does the dependent variable increase or decrease, when the explanatory variable increases by 1?

Returning to our exemplary research question on what makes a good grade in a seminar paper, an intuitive hypotheses would be that the grade gets better the more hours a student invests in writing the paper. In this case we assume a linear relationship between the independent variable `hours` and the dependent variable `grade`. As German grades are better the lower their value, we thus would assume a negative effect from `hours` on `grade`.

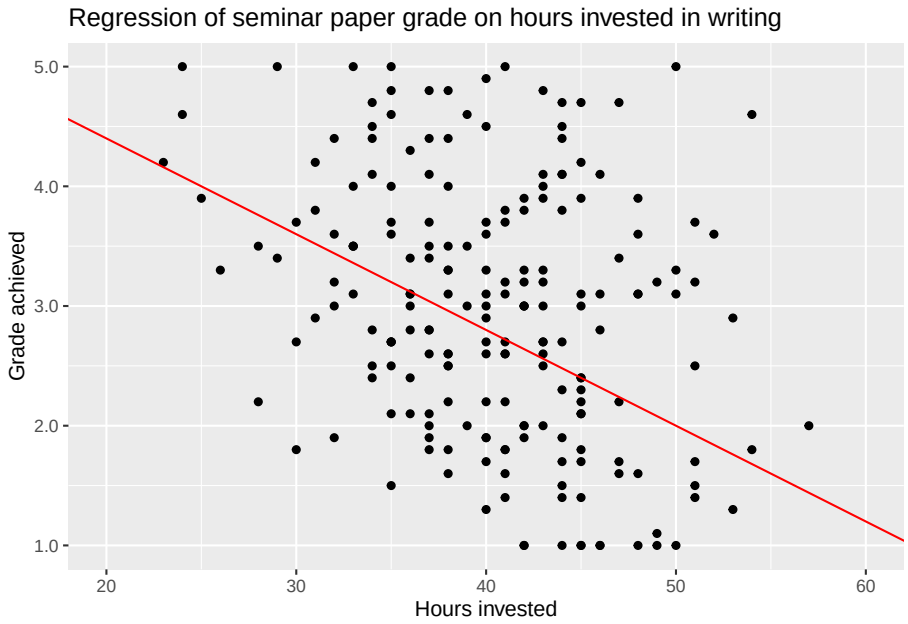
Before turning to the formalities and practical application of a simple linear regression model, let us first have a look on this relationship by plotting the variables against each other.



When we are talking about dependent and independent variables, there is the convention to plot former on the x-axis and the latter on the y-axis. So the *y-variable* is to be explained and the *x-variable* is used to explain it. This convention will also be used in all formulas in this seminar.

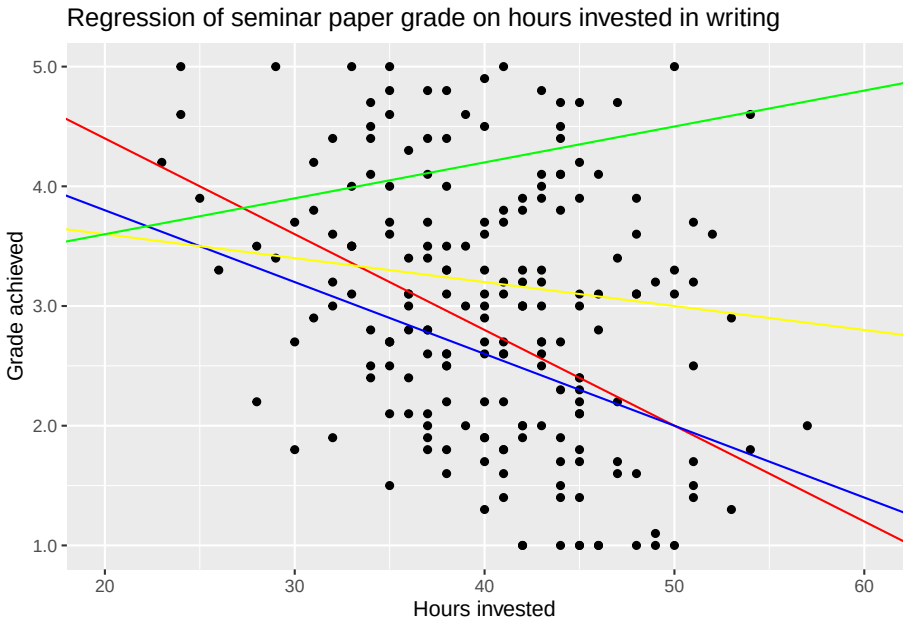
Looking at the plot we first see a cloud of dots, representing all combinations of `hours` and `grade` in all our 200 observations. It may be hard to pick out any pattern, but looking closely we can observe that overall the dots seem to follow a downward slope from the upper left - indicating few hours worked and a worse grade - towards the lower right - indicating more invested hours and a better grade. This would be the relationship stated in our hypotheses. The more hours a student works on a seminar paper the better the final grade will be.

We can try to describe this pattern by adding a line from the upper left to the lower right.



This describes the realtionship between the two variables as linear. Each hour invested decreases the grade by a certain amount, for this proposed line by exactly 0.08 points. Remember that decreasing the value of the grade actually means getting a better grade.

But is this the only possible line or even the *correct* one? Most certainly not, as the values used to draw the line were only a wild guess. We could imagine several other lines that also look more or less reasonable - as well as some that look unreasonable - and add them to the plot.



While we have some intuition that the green line completely misses the mark, we can't really decide between the others just by looking at the plot. The data points are way to dispersed to see the relationship clearly.

The goal of using a simple linear regression model is to identify the *one* line that describes the relationship the best. Here *best* means, with as little error as possible.

5.4.1 Regression Formula

To understand how these lines in the above plot were conceived and how to find the line with the best *fit*, i.e. the lowest error, we have to understand the formula for linear regression. While formulas may always be kind of daunting, we are in luck as this particular one is actually quite easy to understand, especially when paired with a graphical representation.

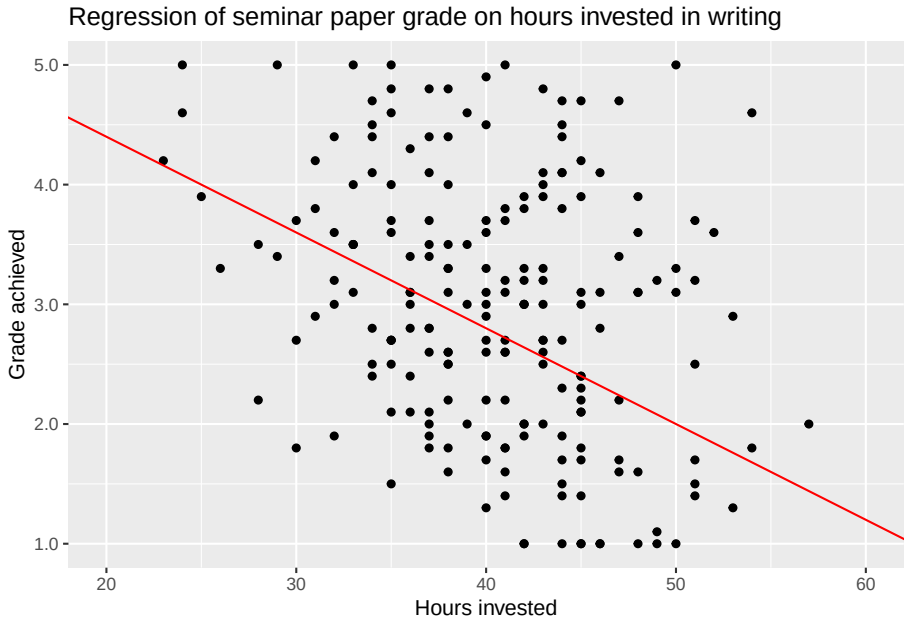
$$y = \beta_0 + \beta_1 * x_1 + \epsilon$$

Let us first look at the parts we already know. y is the dependent variable, in our case the grade achieved. So one thing is for sure, the whole right part of the equation has to be used to calculate the value of y from the data, i.e. from the dependent variable x. Here we have three terms. Let us skip the first one for now and focus on the second one $\beta_1 * x_1$.

x_1 is the dependent variable, in our case `hours`. β_1 is the *regression coefficient* for x_1. This value gives us the *slope* of the regression line. Based on this, we can start rewriting the general formula and tailor it to our specific use case.

$$y_{grade} = \beta_0 + \beta_{hours} * x_{hours} + \epsilon$$

Let us return to the first wild guess we made above.

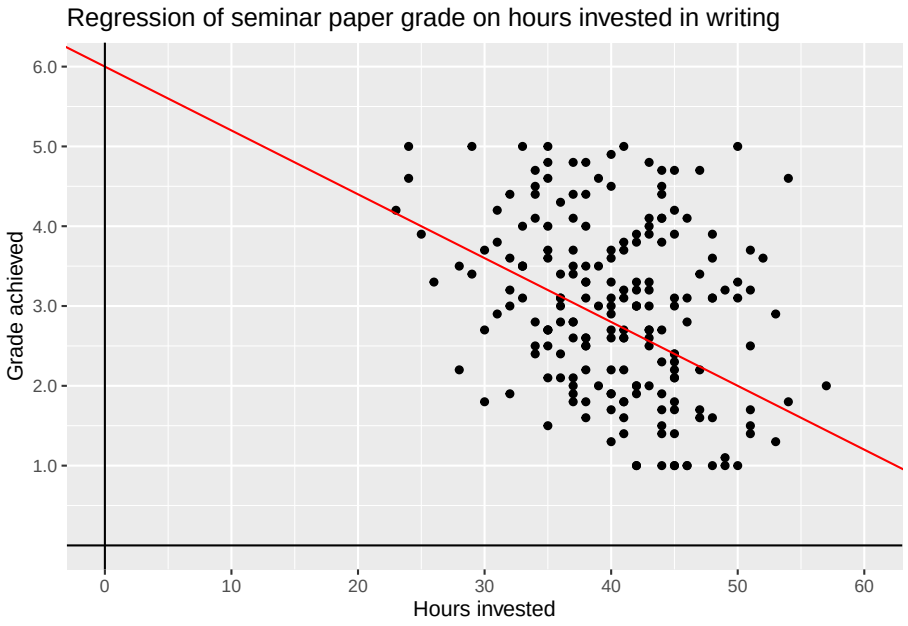


Here we guessed that an increase in invested time of one hour decreases the value of `grade` by 0.08. This is the slope of the red line and thus also the coefficient in the regression formula that is used in computing said line. So, $\beta_{hours} = -0.08$. We can insert this value into our formula.

$$y_{grade} = \beta_0 - 0.08 * x_{hours} + \epsilon$$

In this way the value of x_{hours} is multiplied by -0.08. Let us assume a student worked 40 hours on their paper. $-0.08 * 40$ being -3.2, we assume that working 40 hours on a paper *on average* - more on that later - leads to a 3.2 lower grade value. But 3.2 lower than what?

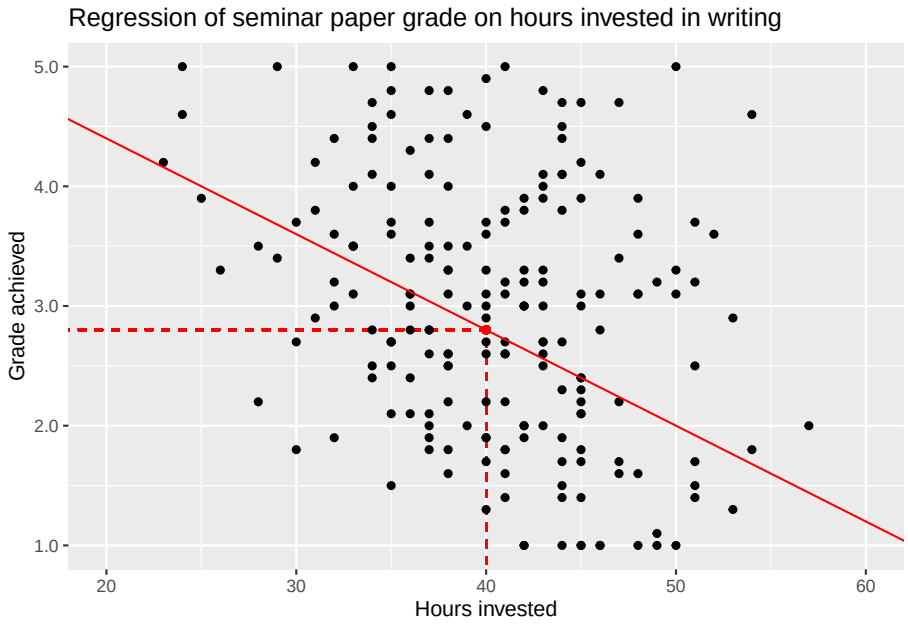
Looking at the formula again, we see that we subtract this value from β_0. This is the *intercept*, the value at which the line intersects with the y-axis. Let us zoom out on our plot to see what happens.



We can now see the point where the red line intersects with the y-axis. This is the intercept of this line, i.e. $\beta_0 = 6$.

$$y_{grade} = 6 - 0.08 * x_{hours} + \epsilon$$

If we now again assume a time investment of 40 hours, we can compute $6 - 0.08 * 40 = 2.8$. So our red regression line - which is still only a wild guess - assumes, that working 40 hours on a seminar paper will result in a grade of 2.8, on average. We can mark these values in our plot:



The red dot is the intersection of the values `hours = 40` and `grade = 2.8`. As this is the value for y our regression line assumes a student with a time investment of 40 hours achieves, the red dot also lies exactly on the red line.

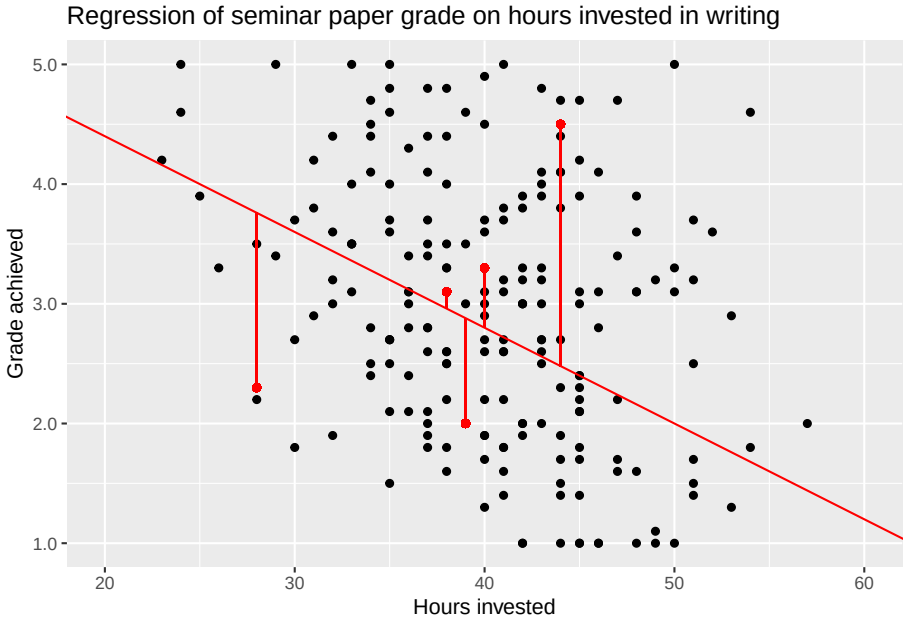
But if we look at the plot once again, we can see that most actual observations for students that invested 40 hours do not actually lie on the regression line but are scattered above and below the line. Some of these students achieve much worse or much better grades than 2.8 investing the same amount of time in their work. This leads us to the last part of the formula, ϵ.

This is the *error term*. Having data that is dispersed like this - and any real world data will always be - our linear line will never be able to pass exactly through every data point. Some points may lie exactly on the line, but many or most will not.

We can visualize this. To keep the plot readable, we only do this for some random observations but in reality the distance of every data point from the regression line is taken into account.

```
## # A tibble: 5 x 7
##   grade hours previous_grades attendance contact    previous_grades_centered
##   <dbl> <int>           <dbl> <lgl>      <fct>                         <dbl>
## 1   2.3    45             2.6 TRUE       No contact                   -0.335
## 2   1.9    44             2.8 TRUE       In Person                    -0.135
## 3   3.1    38             3.7 TRUE       In Person                     0.765
## 4   3.3    40             3.4 TRUE       E-Mail                        0.465
## 5   4.6    54             4.5 TRUE       No contact                    1.56
```

```
## # i 1 more variable: hours_centered <dbl>
```



The distance of these or rather all points from the line, the *residuals*, are represented in the error term ϵ. It is a measure for how wrong our line is in describing the data in its entirety. So why is it wrong? We can not say for sure, but there are two common main reasons.

For one, there may be other variables that also influence the relationship between invested hours and achieved grade, something that we will return to in the next session, when we expand the idea of linear regression to multiple independent variables.

But there is also random variation present in every bit of real world data. While our data is simulated we also added random variation on purpose. Because this is what real world data is, it is messy and it is noisy.

Not every seminar paper that had the same time investment, e.g. 40 hours, will have the same quality in results. There may be other influential variables, e.g. the student's general skill level or if they sought assistance by their lecturer in preparing the paper, influencing the final grade. But even if the quality of the paper after working 40 hours would be the same for each student, measurement error, i.e. noise, will be introduced because not every lecturer will grade exactly the same or maybe because papers were submitted at different time points and grading standards may have changed. If we can not measure these variables we have to accept these unobservable sources of noise and hope, where *hope* actually means thorough theoretical and methodical thinking, that we can still measure

our effect of interest. This also means, that measuring and modelling **always** includes uncertainty. We never know for certain if and to what extent our results are influenced by unobservable variables and random variation. Still, there are ways to assess this uncertainty, which we will regularly return to during the course. This should not stop quantitative social scientists from making strong or even bold arguments based in thorough theoretical thinking and responsible data analysis, but we always have to acknowledge the uncertainty included in every step and make it a part of our interpretations and conclusions.

The error term ϵ is the final piece of the puzzle in actually computing a linear regression model. Without jumping into the mathematics of it all, the technique that is used to estimate the coefficients β_0 and β_1 is called *OLS* - Ordinary Least Squares. What it basically does, is to take the squares of all residuals, i.e. the distances of the data points from the regression line, sum them up and minimise this value. All this substantially means is, that OLS searches for the regression line with the lowest amount of error, i.e. the lowest overall distance from the actual data points.

OLS gives us estimates for the regression coefficients in this formula:

$$\hat{y} = b_0 + b_1 * x_1$$

We can see two differences to the formula we started with. First, we write $\hat{y}$ - pronounced as "y hat" - instead of y. At the same time, we exclude the error term ϵ. This means that we are no longer computing the actual value of y, as in the point on the regression line for a certain value of x_1 + the error, but the estimate $\hat{y}$, as in the point on the regression line that is predicted for a certain value of x_1. Second, we write b instead of β. This also alludes to the fact that we are now computing an estimate for the coefficients based on the data available and not the real but unknown value of β.

This implies that we now estimate the same grade for every student who invested the same amount of time, the $\hat{y}$ that lies exactly on our regression line at a certain value of x_1. For all students who invested 40 hours in writing, we would estimate exactly the same grade. As we have seen above, these students received different grades in reality, or more accurately our simulated reality. The value of $\hat{y}$ is still the best guess our model can make. That is what we mean when we say "on average". On average a student is estimated to receive the grade $\hat{y}$ after investing x_1 hours in writing the paper. We have to keep in mind, that this will not be true for many students; there is always an error involved in our estimates.

5.4.2 Regressing `grade` on `hours`

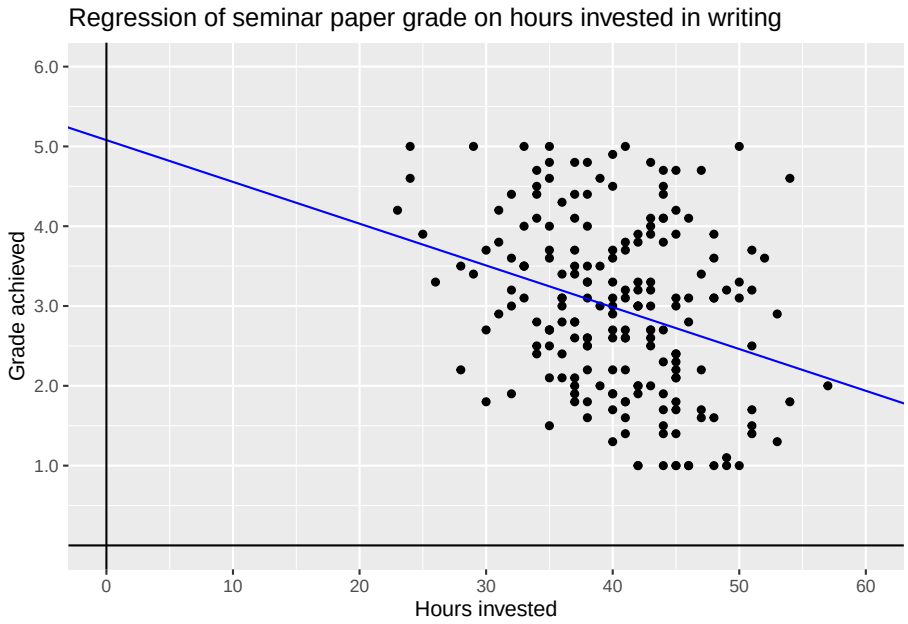
Now that we have a firmer understanding on what linear regression actually is and does, we can finally get to the fun part and use the technique for estimating the effect of `hours` on `grade` or in other words, regress `grade` on `hours`.

```
##
## Call:
## lm(formula = grade ~ hours, data = grades)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.88006 -0.83961 -0.08006  0.77006  2.53881
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.07912    0.47306  10.737  < 2e-16 ***
## hours       -0.05236    0.01159  -4.517 1.07e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.028 on 198 degrees of freedom
## Multiple R-squared:  0.09344,    Adjusted R-squared:  0.08886
## F-statistic: 20.41 on 1 and 198 DF,  p-value: 1.075e-05
```

This is the output from a simple linear regression for `grade` on `hours` R returns to us. How we can do this in practice and what the first two lines mean will be the topic of session 8. For now we will focus on the estimates in the coefficient block and introduce the additional elements of the output on by one over the next sessions.

The column `Estimate` gives us the values for β_0 and β_1 discussed above. The estimated coefficient for `hours` tells us that out intuition was right, the more hours a student invests in writing a paper, the better the grade will be. In this case every additional hour spent on working is estimated to decrease the value of the grade by -0.05236 points. In keeping with the example of a 40 hour workload this leads to a decrease of $-0.05236 * 40 = -2.0944$ points. Adding the intercept from the same column, the estimated grade after working 40 hours is $5.07912 - 0.05236 * 40 = 2.98472$. So on average a student from our simulated data set will pass after 40 hours of work but will not get a great grade. Remember, this is the expected average value. This does not mean that some students will not get better or worse grades, or even fail to pass with this amount of time investment.

Now that we know the coefficients for the regression line with the best fit, i.e. the lowest error, we can again visualise the result.



What grade can a student expect, on average, if they invest exactly 0 hours, i.e. do nothing and hand in a blank paper. We can look at the graph or, to achieve a more precise result, calculate it.

$$5.07912 - 0.05236 * 0 = 5.07912$$

For this theoretical example of $x_{hours} = 0$, the estimated value $\hat{y}$ or y_{grade} is the same as the intercept β_0. This is what the intercept represents in general, the estimated value $\hat{y}$ when the dependent variable equals 0.

Investing zero hours in a seminar paper is not only not advisable, it is also not a value we observed in our data. If the data would include observations with zero hours of time invested, the grade would be a firm 5.0 and the same would be true for low single digits, i.e. turning in a two-pager as a seminar paper. The takeaway is, that the model is highly dependent on the data that it is trained on. If the data would have included such cases we could expect a higher intercept and a steeper slope, i.e. stronger negative coefficient.

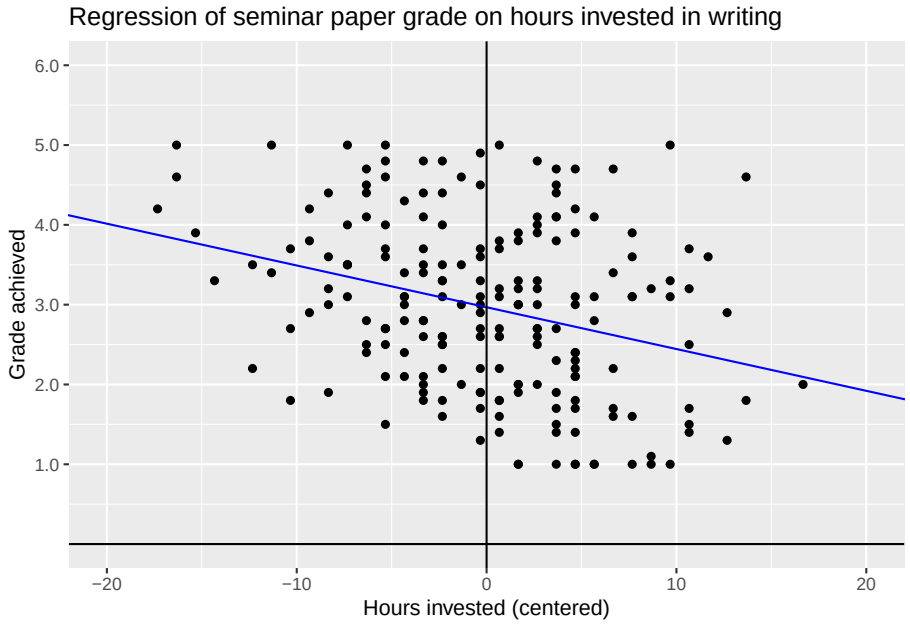
Luckily all our simulated students have put in at least some hours. But as we do not have data for zero to 22 hours, we can not really make reliable estimates in this range. Because of this, it does not really make sense to enter `hours` into the regression model as ranging from 0 to 57. One solution that is often used for metric variables is to center them on their mean. This can be achieved by simply subtracting the mean of x from each individual value:

$$x_i - \bar{x}$$

We can now rerun the regression.

```
##
## Call:
## lm(formula = grade ~ hours_centered, data = grades)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.88006 -0.83961 -0.08006  0.77006  2.53881
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     2.96750    0.07267  40.835  < 2e-16 ***
## hours_centered -0.05236    0.01159  -4.517 1.07e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.028 on 198 degrees of freedom
## Multiple R-squared:  0.09344,    Adjusted R-squared:  0.08886
## F-statistic: 20.41 on 1 and 198 DF,  p-value: 1.075e-05
```

Comparing the results to the first model shows us, that the coefficient for $b_{hours\_centered}$ is exactly the same as for b_{hours}. So the effect of working more hours has not changed. What has changed is the value of the intercept. This will make more sense if we again plot the regression line.



By centering the x-Variable on its mean we have changed its interpretation. A value of `hours = 0` now stands for investing as much time as the mean of `hours` in the whole data set, which in this case is 40.33 hours. Positive values indicate that a student worked x hours more, negatives indicate $-x$ hours less compared to the mean. In this way, we also moved the y-axis and thus changed the interpretation of the intercept. Its new value of 2.9675 now indicates the estimate for a student who invests the mean value of `hours` in their work, i.e. 40.33.

5.5 Moving on

Based on our simple linear regression model we achieved an estimate for our effect of interest. Working more hours results in receiving a better grade. But there could be other variables that influence this relationship. In the next session we learn how we can include these in our model, when we move from simple to multiple linear regression.

Chapter 6

Linear Regression Theory II: Multiple Linear Regression

Maybe explaining the grade a student receives solely based on the hours of invested time, does not paint the whole picture. As we have alluded to, there may be other variables that could affect the relationship between `hours` and `grade`. If we fail to include these in our model, we may not get an unbiased estimate for our effect of interest. Maybe the actual effect for `hours` is even stronger, maybe it is weaker, or maybe there is no effect at all. To assess this, we have to move from simple to multiple linear regression.

6.1 Objectives

- Expand the idea to multiple linear regression
- How can we interpret different types of independent variables?
- Understand measures of uncertainty

6.2 Multiple Linear Regression

A *simple linear regression* only allows for one independent variable. This is why we need *multiple linear regression* if we want to start introducing additional variables into the model. Luckily this is easy to understand as we already know the formula for a simple linear regression:

$$y = \beta_0 + \beta_1 * x_1 + \epsilon$$

To change a simple into a multiple linear regression, we just start adding the additional variables and their coefficients additively to the formula.

$$y = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + ... + \beta_k * x_k + \epsilon$$

So to add a second variable and its coefficient we add the term $+\beta_2 * x_2$ and so on until we added all independent variables of interest k to the model. Everything else works exactly as described above for the simple model.

6.2.1 Adding additional metric variables

We already expected that the mean of the previous grades could be a strong predictor for future grades. We could understand these as a *proxy* variable for the general skill level of a student. The higher the skill level, the higher previous grades will have been.

How we can add additional variables in R code will again be a topic for the next session, but let us look at the results of a regression of `grade` on `hours_centered` and `previous_grades_centered`, the latter being centered on the mean previous grade of 2.935.

```
##
## Call:
## lm(formula = grade ~ hours_centered + previous_grades_centered,
##     data = grades)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.44462 -0.30556  0.00622  0.32878  1.31002
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                2.967500   0.038316  77.449   <2e-16 ***
## hours_centered            -0.056543   0.006114  -9.248   <2e-16 ***
## previous_grades_centered   0.904079   0.039830  22.699   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5419 on 197 degrees of freedom
## Multiple R-squared:  0.7492, Adjusted R-squared:  0.7467
## F-statistic: 294.3 on 2 and 197 DF,  p-value: < 2.2e-16
```

As we added a new variable, we now see three coefficients. The intercept has not changed. It now indicates the estimated grade for a student who invests the mean amount of hours, 40.33, and whose previous grades are exactly 2.935, the mean of the variable.

The coefficient for `hours_centered` got mildly more negative, still telling us that the value of `grade` gets lower, the more hours are invested in writing the paper. This coefficient now gives us the effect while *controlling* for the effect of `previous_grades_centered`. This is what multiple linear regression does, giving us the coefficients for our variables of interest while keeping all other independent variables at specific values. As we have centered the variable for previous grades, the coefficient for `hours centered` gives us the effect when the previous grades were exactly at the mean of $2,935$.

In the same way, the coefficient for `previous_grades_centered` gives us the effect of previous grades when the invested hours are controlled for, in this case when the invested hours were exactly 40.33. The coefficient is rather high and positive. This indicates that a student with a previous grade value that is 1 above the mean, is estimated to receive a new grade that is 0.9 points above the intercept. This means, that the previous grade is a very strong predictor for the new grade.

While plotting in more than two dimensions gets really hard, we can still calculate $\hat{y}$ for certain values of both independent variables. We already know the predicted grade for a student with mean values on both independent variables, as this is the intercept. To make sure that we correct, we can calculate it again.

$$b_0 + b_{hours\_centered} * 0 + b_{previous\_grades\_centered} * 0 = 2.9675$$

For this case we can see, that the previous grade actually is a strong predictor, as the previous and new grades are substantially the same.

What if a student whose previous grades were 1 above the mean, so just below 4.0 but who decides to invest 10 hours more than the mean for the new paper?

$$2.9675 - 0.056543 * 10 + 0.904079 * 1 = 3.306149$$

So the good message is, while previous grades are a strong predictor, putting in more hours still leads to better grades.

What if a really good student decides to rely on their skill and to work less this time?

$$2.9675 - 0.056543 * -10 + 0.904079 * -2 = 1.724772$$

While 1.7 is still a very good grade, working 10 less hours than the mean of students leads to a substantially worse estimate compared to the about 1.0 received in previous grades.

6.2.2 Adding dummy variables

Another variable that could be of interest in explaining the received grade, is if a student attended most of the seminar sessions. `attendance` holds this information in the form of a dummy variable. Dummies can only have two states. "Yes" or "No", "1" or "0" or in this case "`TRUE`" or "`FALSE`".

Let us add the variable to our model.

```
##
## Call:
## lm(formula = grade ~ hours_centered + previous_grades_centered +
##     attendance, data = grades)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.41059 -0.30910  0.01667  0.35607  1.29849
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                3.157411   0.078658  40.141  < 2e-16 ***
## hours_centered            -0.053942   0.006088  -8.860 4.85e-16 ***
## previous_grades_centered   0.911802   0.039282  23.212  < 2e-16 ***
## attendanceTRUE            -0.248250   0.090246  -2.751   0.0065 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5331 on 196 degrees of freedom
## Multiple R-squared:  0.7586, Adjusted R-squared:  0.7549
## F-statistic: 205.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

This gives us a new line in the R Output holding an estimate for `attendanceTRUE`. What is meant by this? In contrast to the metric variables we have uses in our model up to this point, a dummy variable - or binary variable - can only have two states. As we are using a logical variable here, it can only have the value `TRUE` - here indicating regular attendance - or `FALSE`. So what the output shows us, is the effect of attendance being `TRUE` compared to being `FALSE`. If a student did regularly attend the seminar, the estimated grade is -0.248250 lower compared to when they did not.

We can observe what happens in the formula:

$$\hat{y} = b_0 + b_{hours\_centered} * x_{hours\_centered} + b_{previous\_grades\_centeterd} * x_{previous\_grades\_centeterd} + b_{attendance} * x$$

If you calculate with `TRUE` and `FALSE` in R, the values 1 and 0 are used respectively. So $x_{attendance}$ can either have the value 1 for regular attendance or 0 for not so regular attendance.

If a student did regularly attend, the coefficient $b_{attendance} becomes a part of the estimate $\hat{y}$:

$$\hat{y} = b_0 + b_{hours\_centered} * x_{hours\_centered} + b_{previous\_grades\_centeterd} * x_{previous\_grades\_centeterd} + b_{attendance} * 1$$

If student did not regularly attended, this happens:

$$\hat{y} = b_0 + b_{hours\_centered} * x_{hours\_centered} + b_{previous\_grades\_centeterd} * x_{previous\_grades\_centeterd} + b_{attendance} * 0$$

$$\hat{y} = b_0 + b_{hours\_centered} * x_{hours\_centered} + b_{previous\_grades\_centeterd} * x_{previous\_grades\_centeterd}$$

The coefficient is no longer a part of the estimate. One can basically say, the coefficient gets switched on or off by the value of the dummy variable.

So while the estimate for a student with mean values for invested hours and previous grades who did not attend is equal to the intercept of 3.157411, for a similar student who attended we can calculate the estimate as:

$$3.157411 - 0.053942 * 0 + 0.911802 * 0 - 0.248250 * 1 = 3.157411 - 0.248250 = 2.909161$$

It seems attending class is an easy way to raise one's grades.

6.2.3 Adding categorical variables

We have one further variable in our simulated data set that could be of interest in explaining, what makes a good grade in a seminar paper.`contact` is a categorical variable, or a factor variable in R terms. It can take three different categories. `No contact` indicates that the student did not contact the lecturer to discuss a research question or the laid out plan for the paper. `E-Mail` means that there was some written contact and at least the basics for the paper were discussed before writing. Lastly, `In Person` stands for an in depth discussion with the lecturer, clearing up problems beforehand and thus potentially having a more stringent vision for the paper before writing the first word.

Let us add the variable to our model.

```
##
## Call:
```

```
## lm(formula = grade ~ hours_centered + previous_grades_centered +
##     attendance + contact, data = grades)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3835 -0.2525  0.0167  0.2678  0.9347
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                3.617949   0.068077  53.145  < 2e-16 ***
## hours_centered            -0.050830   0.004433 -11.466  < 2e-16 ***
## previous_grades_centered   0.874123   0.028657  30.503  < 2e-16 ***
## attendanceTRUE            -0.324653   0.065781  -4.935 1.72e-06 ***
## contactE-Mail             -0.413808   0.069817  -5.927 1.39e-08 ***
## contactIn Person          -0.853252   0.063964 -13.340  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3869 on 194 degrees of freedom
## Multiple R-squared:  0.8741, Adjusted R-squared:  0.8709
## F-statistic: 269.4 on 5 and 194 DF,  p-value: < 2.2e-16
```

Wait, we entered three categories into the model and got estimates for two of them. What happened? What R does is to create two dummy variables on the fly. The first discerns between having E-Mail contact and no contact at all. The second one between having contact in person and no contact at all. So for categorical variables in regression models we always compare being in one of the category to being in the *base category*. In this case the base category is `No contact`, but we could also change the base category. It depends on what we are interested in comparing to. For our example comparing the effects of having more in depth contact to having none makes sense.

Let us look at our formula again:

$$\hat{y} = b_0 + b_{hours\_centered} * x_{hours\_centered} + b_{previous\_grades\_centeterd} * x_{previous\_grades\_centeterd} + b_{attendance} * a$$

Now there are three possibilities. A student can have no contact at all. In this case both dummy variables equal 0. To make our formula easier to read, we have abbreviated the middle part for now:

$$\hat{y} = b_0 + ... + b_{E-Mail} * 0 + b_{InPerson} * 0$$

So in this case controlling all other independent variables at their default values, the mean for the metric variables and `FALSE` for `attendance`, the intercept gives

us the estimate for the grade as both dummy variables that were created for `contact` are "switched off".

The two other possibilities are that a student either had E-Mail contact or an in person discussion:

$$\hat{y} = b_0 + ... + b_{E-Mail} * 1 + b_{InPerson} * 0$$

$$\hat{y} = b_0 + ... + b_{E-Mail} * 0 + b_{InPerson} * 1$$

In both cases the relevant dummy variable is "switched on" while the other does not factor into the equation.

Looking at the estimates we can see that having contact to the lecturer before writing has strong negative effects, resulting in better grades. Having E-Mail contact reduces the value of `grade` by -0.413808 points, having an in person discussion by -0.853252.

So what grade can a student whose previous grades were at the mean of 2.935, but who decided to put in 20 hours more compared to their peers, regularly attend the seminar and have an in-depth personal discussion before writing their paper expect on average as their new grade?

$$3.617949 - 0.050830 * 20 + 0.874123 * 0 - 0.324653 * 1 - 0.413808 * 0 - 0.853252 * 1 = 1.423444$$

Putting in the hours, attending and working with your lecturer seems to pay off, at least in our simulated data set.

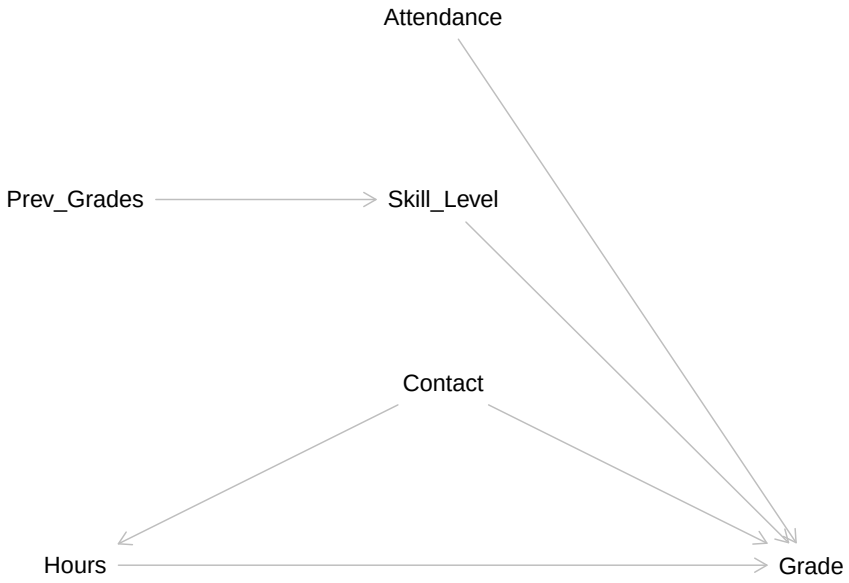
6.3 Returning to our research question

Our exemplary research question concerned itself with what makes a good grade in a seminar paper. In particular we were interested in the effect of the invested hours, as our main hypothesis was that more hours lead to better grades. What do we know now?

All analysis point towards a clear effect from `hours` on `grade`. This effect was consistently visible in all of our models. But did we correctly identify and estimate the effect of interest? Maybe. The problem is, we actually did not approach the analysis correctly. In a real analysis we should **absolutely** refrain from adding variables to our model that *could be* relevant until we are satisfied or even until all available variables are bunched into one huge model. It was fine to do this in this introduction to linear regression to learn how different types of variables can be used in a regression model. But in a real project, we have to

invest time to think about which variables to add because we assume that they have a relevant effect based on theoretical assumptions about the processes we are interested in.

So let us do this now and vow do make this our first step in all future endeavors. While we do not have a clear theoretical basis, we can make clear assumptions on the data generating process and draw these in a DAG.



Our central assumption, and the effect we want to identify and estimate, is the direct effect from `hours` on `grade` in the bottom line. The more hours a student invests, the better the grade should be.

The assumed effect of `contact` is more complex. For one we assume that a more in-depth contact with the lecturer will increase the grade directly. The research question will be more focused, the student will know what is important to a certain lecturer, common mistakes can be avoided if they are cleared up beforehand and so on. But we will also assume that `contact` will have an effect on `hours` in the sense that the hours invested can be used more efficiently if an in-depth discussion has taken place. Instead of wasting time running into problems that could have been avoided most of the invested time can actually go into constructive work. This makes `contact` a confounder for `grade` and `hours`.

A student's skill level will also have a direct effect on `grade`. As we do not have a direct measure of skill in our data, we use `previous_grades` as a proxy for skill level. `attendance` is also assumed to have a direct effect on `grade` as students who were present in the seminar will not only have learned the

seminar's contents, but will also have a better understanding of what is expected in their seminar papers.

Tapping into the knowledge from session 4, we can now make implications for our model from the DAG. Let us list all paths from `hours` to `grade`:

$$A : Hours \rightarrow Grade$$

$$B : Hours \leftarrow Contact \rightarrow Grade$$

Path A represents our effect of interest. On path B, `contact` is a confounder for `hours` and `grade`. To close this path, we have to control for `contact`. As neither skill level - or `previous_grades` - nor `attendance` lie on a path from our independent variable of interest to our dependent variable, we should not control for them in our model. That leaves us with `hours` and `contact` to be included in our linear regression, if our goal is to get an unbiased estimate for the effect of invested time on the final grade. So let us do this:

```
##
## Call:
## lm(formula = grade ~ hours_centered + contact, data = grades)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.85595 -0.74624 -0.02106  0.66648  2.50161
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       3.44352    0.10404  33.098  < 2e-16 ***
## hours_centered   -0.04967    0.01052  -4.723 4.43e-06 ***
## contactE-Mail    -0.46482    0.16785  -2.769  0.00616 **
## contactIn Person -1.02804    0.15240  -6.746 1.67e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9305 on 196 degrees of freedom
## Multiple R-squared:  0.2643, Adjusted R-squared:  0.253
## F-statistic: 23.47 on 3 and 196 DF,  p-value: 5.072e-13
```

This is our estimate. Each hour invested beyond the mean of 40.33 hours changes the grade by about -0.05 points. This supports our hypotheses and we can conclude, that investing more hours into writing a seminar paper actually is a worthwhile investment.

But remember: This is correct as long as our DAG is drawn correctly. This is always debatable. Maybe we should assume an effect from skill level on `hours`. The higher the skill level the more efficiently the available time can be used. For this example we know the DAG is correct, because we have simulated the data exactly in this way. For real world applications we never know if our DAG is correct. All we can - and have to - do is base it on thorough thought, theoretical work, exploratory data analysis and sound arguments.

This all is true *if* our goal is to estimate an effect of interest as precisely as possible. But as we have alluded to in the introduction to this session we could also use modelling with a different goal, i.e. predicting a grade as accurately as possible. For this task, the model which only includes `hours` and `contact` will not do the best job. From our DAG we know that `attendance` and `previous_grade` should have an effect on `grade`, as we have also seen in our models. For this task the full model including all these variables will produce better estimates. We will return to this in a later session, but for now we should remember that we have to know our task because the task dictates which is the best model to use.

6.4 Adressing the uncertainty

Looking at the coefficient block from the output, we see more than just our estimates. The `Std. Error` - *standard error* - is a measure for the uncertainty of our estimates. It basically tells us, how far away the actual values of the observations used to compute the model are from our estimate *on average*. The smaller the *standard error*, the more accurate our estimate. The standard error is presented in the units of the estimate and we can thus compare them. A large standard error for a large estimate is far less problematic compared to a large standard error for a small estimate.

The estimate and it's standard error are the basis for *hypothesis testing*. What we are testing is the *alternative hypotheses* H_a that there actually is an effect of our independent variable on the dependent variable against the *null hypothesis* H_0 that there is no effect. To reject the null hypothesis and be confident that we are observing an actual effect, versus an effect that is just based on random variation in our sample, the estimate has to be far away enough from 0 and be accurate enough, i.e. have a small standard error. This relationship is computed in the *t-statistic*, `t value` in our output. From this the *p-value* can be computed, `Pr(>|t|)` in the output. The *p-value* tells us the probability to observe an association between the independent and the dependent variable as large or larger than our estimate suggests, if the true association would actually be 0. If the p-value is small enough, we can reject H_0 and conclude that we observed an actual effect. There are certain agreed upon cutoffs in statistics while values that meet this cutoffs are considered *statistically significant*. The most common cutoff in social sciences is 0.05 indicated by one `*` in the output. Other common

cutoffs are indicated by more asterisks.

Interpreting p-values correctly and not falling into the common pitfalls is a topic on its own. We do not have the time to dive into this here, so for now we can agree that p-values below 0.05 indicate that we can reject H_0 and thus conclude that we have actually observed an effect. Still, our interpretation of regression results should not focus solely on p-values or lead us to disregard any effects that did not meet the cutoff. For example, we can have very small p-values for effects that are so small that they are substantially irrelevant. One way to address this is to inspect the actual magnitudes of the effects. On the other hand, we can have p-values larger than 0.05 for effects that are still relevant. Maybe the problem is not that there is no effect but that we were not able to measure the variable in question precisely enough or that we just did not have enough observations. We can not go any deeper than this here, but we should remember that the practice of declaring every effect with stars a win and disregarding everything without them may still be common but is not the way to go forward.

In our model, we can see that the effect of interest is statistically highly significant. We can thus conclude, that we have observed an actual effect from `hours` on `grade`. Our estimate is large enough and our standard error small enough to reach this conclusion.

6.5 Moving on

We have attained an estimate for our effect of interest which supports our hypotheses that investing more hours into writing a paper leads to better grades. So can we wrap a bow on the question and move on to finally figuring out what is going on in our NBA data? Almost, but not yet. We still do not know, if our model actually works as intended. Linear regression, as well as every modelling technique, has some underlying assumptions that we have to meet for the model to accurately estimate an effect. In the next session we will get to know this assumptions and how we can test for them.

Chapter 7

Linear Regression Theory III: Diagnostics

In this session we will get to know the central underlying assumptions for linear regression models. To find out if our model actually works as intended and thus gives us a reliable estimate for the effect of `hours` on `grade`, we have to check if we have met this assumptions, and if we did not, we have to correct our model accordingly. Before we do this, we should briefly consider another part of the regression output, the model fit.

7.1 Objectives

- Learn about model fit and its limits
- Understand the statistical assumptions underlying linear regression
- Test for violated assumptions and learn how to correct for those

7.2 Model fit

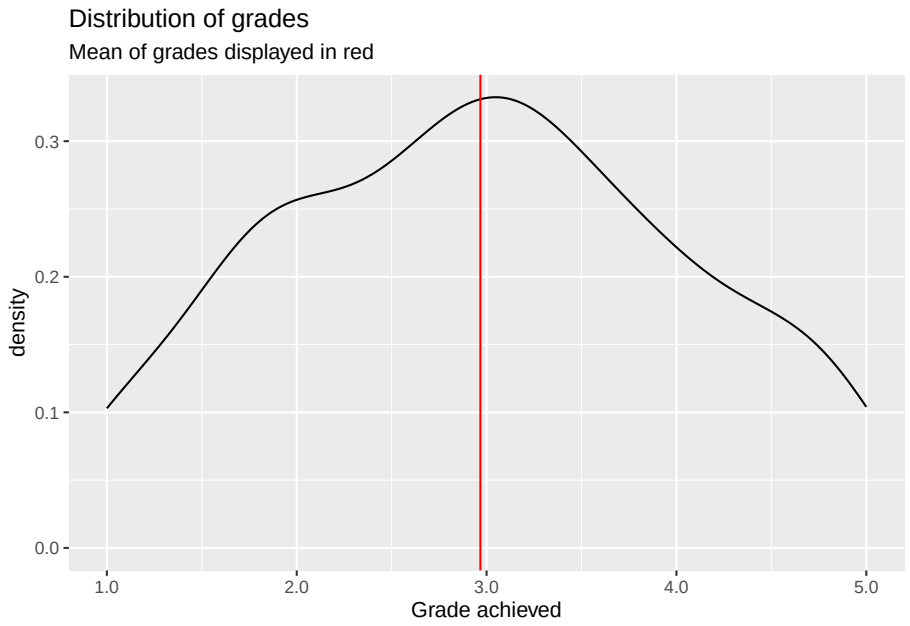
Let us again inspect the output from the simplest model we computed, regressing the grade solely on the invested hours:

```
##
## Call:
## lm(formula = grade ~ hours_centered, data = grades)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -1.88006 -0.83961 -0.08006  0.77006  2.53881
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     2.96750    0.07267  40.835  < 2e-16 ***
## hours_centered -0.05236    0.01159  -4.517 1.07e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.028 on 198 degrees of freedom
## Multiple R-squared:  0.09344,    Adjusted R-squared:  0.08886
## F-statistic: 20.41 on 1 and 198 DF,  p-value: 1.075e-05
```

Up to now, we exclusively talked about the coefficient block. We will return to the "Call" next session and to the "Residuals" later in this session. For now let us focus on the bottom block in the output.

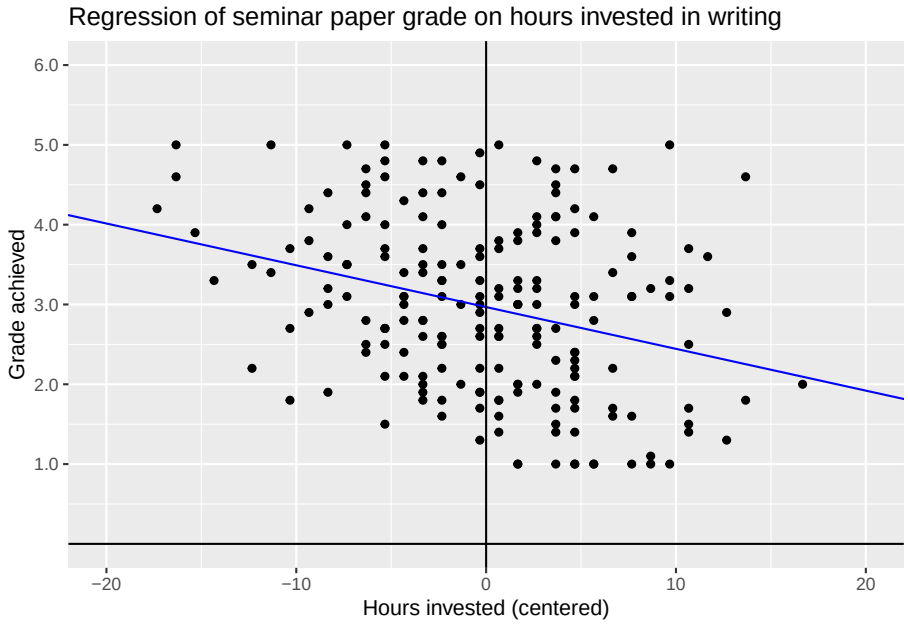
R^2 or *R-squared* is a measure for the amount of variance in the data that is "explained" by the model. Real world data will always have variance. Not every value will neatly fall onto the mean value of a variable. Rather the data is dispersed around it. The same is true for our dependent variable:



This is a density plot and shows the distribution of a metric variable as a smoothed line. We do not see every individual actual value but the general shape of the data. The red line represent the mean of `grade`, which is about 2.97. Most actual values are not exactly at the mean but are rather dispersed

around it, ranging between 1.0 and 5.0. This is the variance in our outcome variable.

Let us now plot `grade` against `hours_centered` and add the regression line from our model above:



Without our regression line, all we would have is a cloud of points without much order to it. What linear regression does, is trying to bring order into this by fitting a line that best explains the variance of the dependent variable, `grade` in our case by its relationship to one or multiple dependent variables, here `hours_centered`. But this linear line can never explain the variance completely. For this it had to pass through every data point. Our line does not. Actually most data points do not lie on the regression line but at some distance to it. You will remember that OLS computes *the* regression line for which the squared distances are smallest. This is the line that explains most of the variance of y by its relationship to x, but not all variance is explained. An unexplained part remains. These are the residuals, the distance that points fall from the regression line. R^2 tells us the relative amount of how much we reduced the initial variance by fitting the line and thus explaining a part of said variance.

A R^2 of 0 would mean that no variance is explained, a value of 1 that all variance is explained. Two highly unlikely outcomes. We will almost always explain something and never explain everything.

In our model R^2 equals 0.09344. This means we explained about 9.3% of the variance of `grade` by its relationship with `hours_centered`. That's nice, but this also means that over 90% are still unexplained. We will not explain all of

the variance, i.e. $R^2 = 1$, but in general a higher R^2 is desirable.

So what can we do? We can try to add additional variables to the model that help to explain the variance in the outcome variable. Last session we concluded that the best model to measure the effect of invested hours on the achieved grade would also have to include `contact`:

```
##
## Call:
## lm(formula = grade ~ hours_centered + contact, data = grades)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.85595 -0.74624 -0.02106  0.66648  2.50161
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       3.44352    0.10404  33.098  < 2e-16 ***
## hours_centered   -0.04967    0.01052  -4.723 4.43e-06 ***
## contactE-Mail    -0.46482    0.16785  -2.769  0.00616 **
## contactIn Person -1.02804    0.15240  -6.746 1.67e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9305 on 196 degrees of freedom
## Multiple R-squared:  0.2643, Adjusted R-squared:  0.253
## F-statistic: 23.47 on 3 and 196 DF,  p-value: 5.072e-13
```

We can use R^2 to compare the *model fit* of multiple models. Here the larger model achieved a considerably higher value of $R^2 = 0.2643$. The model fit improved as we can now explain a higher ratio of the variance in `grade`.

After R^2 we see another value, *Adjusted R-squared*. This becomes relevant if we add additional variables to our model. R^2 almost always increases, and never decreases, when adding additional variables to the model, especially if we have few observations. Because of this R^2 can get less reliable when we have many variables and few observations. *Adjusted R-squared* corrects for this by including both factors in the calculation. When we have many observations the differences are negligible. This is true for our case. We have relatively many observations and few variables in our model, so the values of both measures are rather close. But in cases where this relationship is not as favorable, adjusted R-squared should be used in place of the regular R^2.

The block in our output also gives us the *Residual standard error*. As we have seen above, most actual data point do not lie on the regression line but some distance away from it. These are the residuals. Thus their standard error basically tells us how much we miss the spot on average. As it is given in units

of the dependent variable, we can say that the estimates for `grade` based on our second model are on average 0.93 off. A considerable amount, as this is almost one whole grading step. This is still an improvement from the 1.028 in the first model but nevertheless a substantial error.

The last line in the output gives us two connected measures. The *F-statistic* is the test statistic for R^2 and is used the compute the corresponding *p-value*. In this case we are testing if the R^2 our model returned based on our sample is possible, when the actual population value of R^2 is 0. In other words, could we have achieved this R^2 by chance if the independent variables in our model actually do not explain part of the variance in the population? Both of our models have very small p-values, so it is highly unlikely that we have just explained some variance by chance. This gives further credibility to our model specification.

We can conclude that the second model was an improvement over the first. But can we do more? Sure! We can add additional explanatory variables:

```
##
## Call:
## lm(formula = grade ~ hours_centered + previous_grades_centered +
##     attendance + contact, data = grades)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3835 -0.2525  0.0167  0.2678  0.9347
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                3.617949   0.068077  53.145  < 2e-16 ***
## hours_centered            -0.050830   0.004433 -11.466  < 2e-16 ***
## previous_grades_centered   0.874123   0.028657  30.503  < 2e-16 ***
## attendanceTRUE            -0.324653   0.065781  -4.935 1.72e-06 ***
## contactE-Mail             -0.413808   0.069817  -5.927 1.39e-08 ***
## contactIn Person          -0.853252   0.063964 -13.340  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3869 on 194 degrees of freedom
## Multiple R-squared:  0.8741, Adjusted R-squared:  0.8709
## F-statistic: 269.4 on 5 and 194 DF,  p-value: < 2.2e-16
```

The p-value is even lower, and the F-statistic even higher, compared to our second model, but this was never an issue. What is more interesting is that we have substantially increased R^2 and decreased the residual standard error. As we have concluded last week, this larger model is better at predicting the actual

values of `grade`. Thus the explained variance has to increase and the average error in estimating y has to decrease. But is this the better model? The values on the model fit would suggest so. And this also is true, if our aim is predicting `grade` to the best of our abilities. But if our aim is still measuring the effect of `hours` on `grade` we know from our DAG that we do not have to or even should not control for the additional variables to get an unbiased estimator for the effect of interest.

What can we take away from this? While the model fit measures are an important tool for comparing multiple possible models and better values are desirable in general, it should not be our goal to just max out all measures and declare this model the "winner". It is never that easy in statistics. One thing we can never replace is thorough theoretical work before even computing our first model. Based on our DAG, if it is correct, we know that we do not have to control for previous grades and attendance. Including them may give us a larger R^2, but is still not the correct way to build our model.

Based on this our best model is still the second one:

```
summary(m2)
```

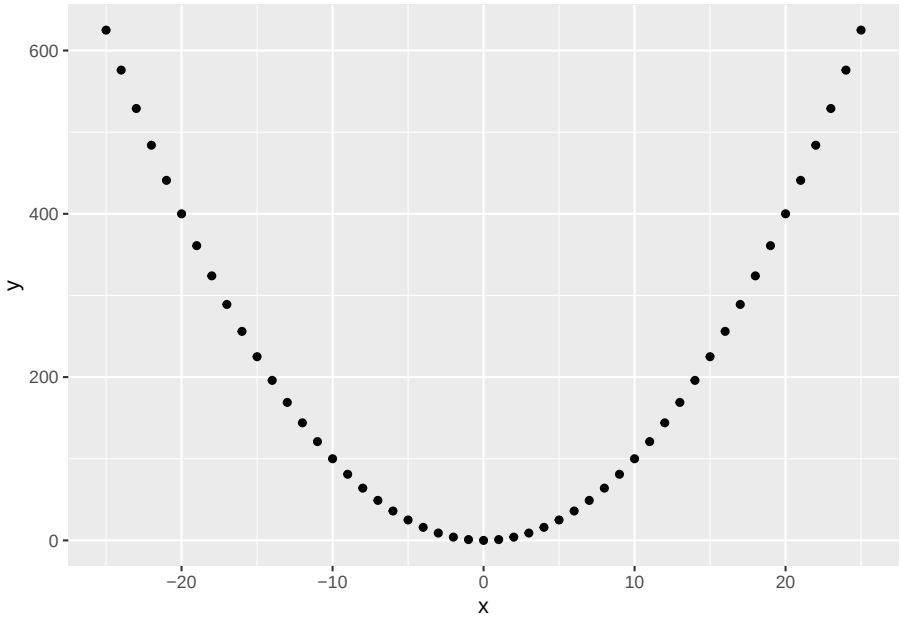
```
##
## Call:
## lm(formula = grade ~ hours_centered + contact, data = grades)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.85595 -0.74624 -0.02106  0.66648  2.50161
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       3.44352    0.10404  33.098  < 2e-16 ***
## hours_centered   -0.04967    0.01052  -4.723 4.43e-06 ***
## contactE-Mail    -0.46482    0.16785  -2.769  0.00616 **
## contactIn Person -1.02804    0.15240  -6.746 1.67e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9305 on 196 degrees of freedom
## Multiple R-squared:  0.2643, Adjusted R-squared:  0.253
## F-statistic: 23.47 on 3 and 196 DF,  p-value: 5.072e-13
```

7.3 Regression diagnostics

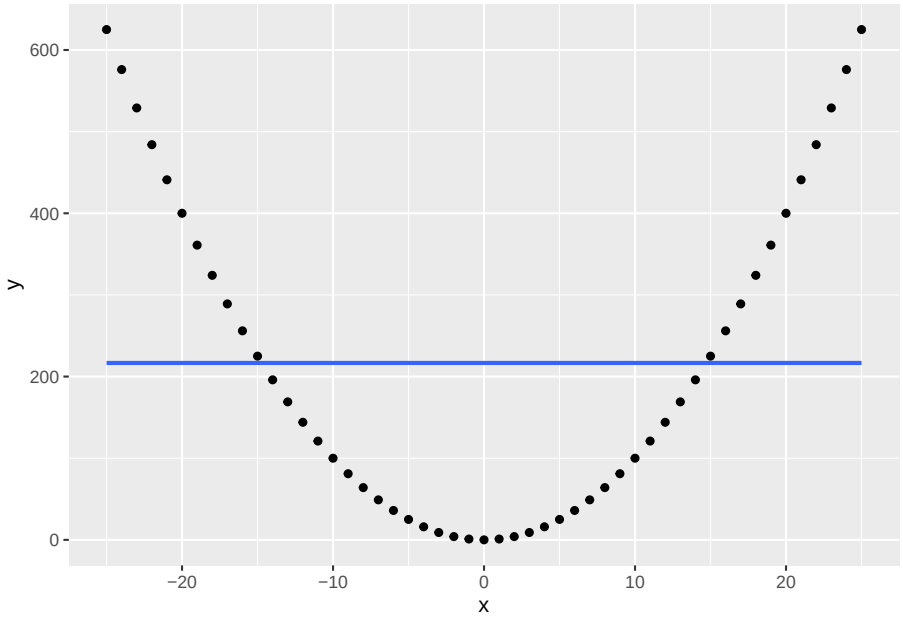
As linear regression is a statistical technique, there are certain statistical assumptions we have to meet. If we violate those, the best laid plans may falter and our results may be not as robust as we hoped. Let us go through these assumptions and the tests to check for them one by one.

7.3.1 Linearity

The name already gives it away, a *linear* regression is used to estimate **linear** relationships between variables. For this to work, the relationships actually have to be linear. But a relationship between two variables can have other functional forms. Consider this for example:

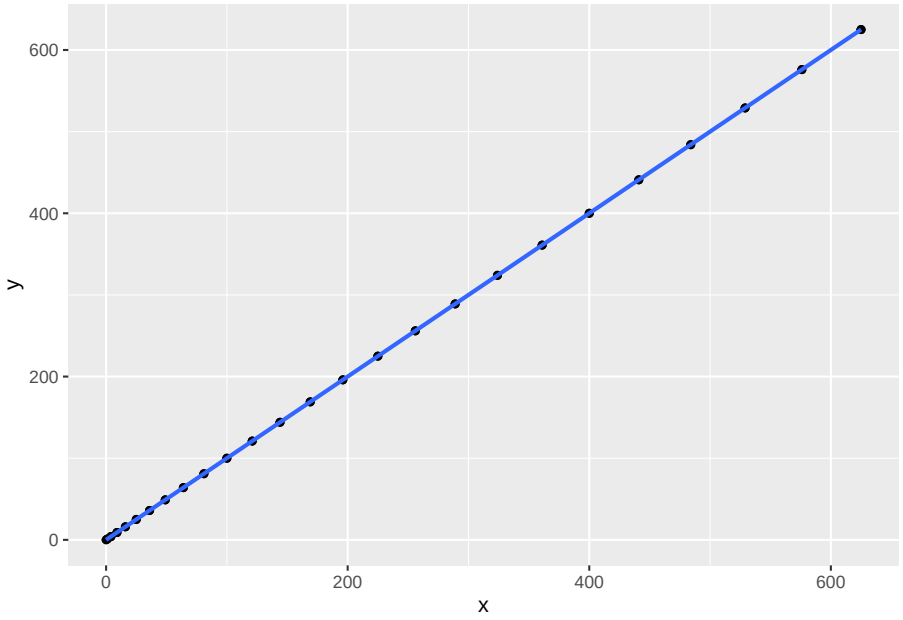


The relationship is clearly not linear. But we can still fit a regression and get a result:



The regression line shows us that `x` and `y` are completely uncorrelated. This is clearly not true, but as our linear regression assumes linearity, it tries to model the relationship in linear terms.

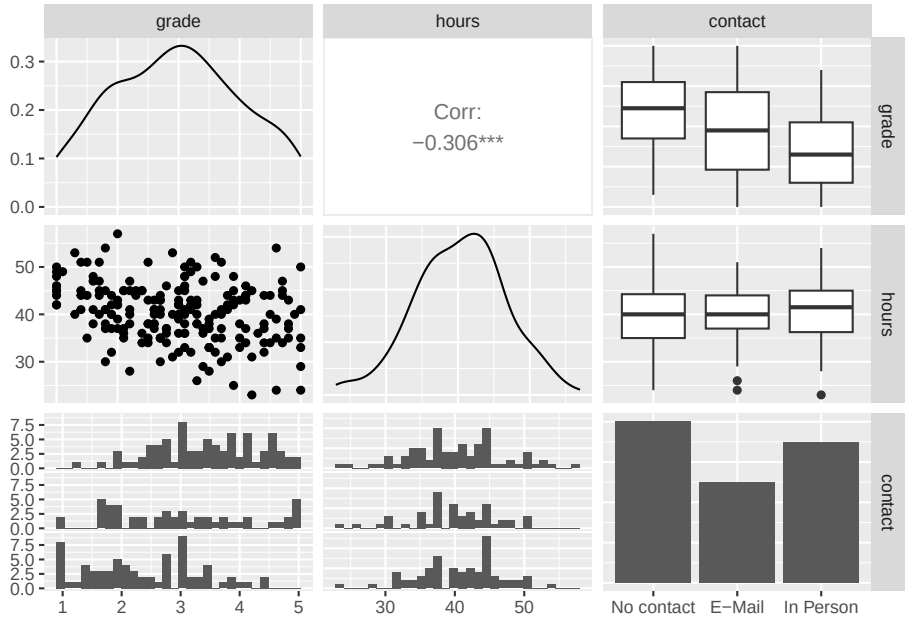
What we can do in such cases is to transform the variable in question in a non-linear way. Here the quadratic relationship is easy to spot, so if we transform x to x^2, this happens:



The non-linear relationship between `x` and `y` has been transformed into a linear one.

For real world data, the non-linearity most often is not as straightforward to spot as in this example. A first step to approach this, is inspecting a scatterplot matrix. This is usually done before starting to model to identify relationships between the variables used.

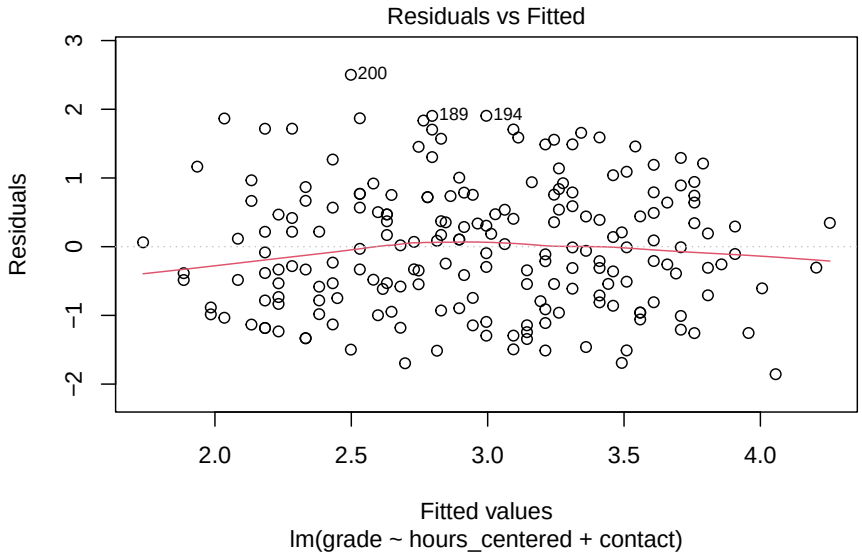
```
## Warning: package 'GGally' was built under R version 4.2.3
```



The diagonal displays the distribution of all variables included. Here metric variables are displayed as density plots and categorical variables as bar plots. Below and above the diagonal the relationship between two variables is shown. The scatterplot on the left of the second row is the one between `hours` and `grade` we have already seen several times. There is no indication of non-linearity here. What we have not inspected yet, is the relationship between `contact` and the two other variables.

The bottom row contains histograms of the two metric variables by the category of contact, the right column boxplots for the same combination. Without going into too much detail on both types of plots, both show us how the distribution for both metric variables changes by category. The more personal the contact with the lecturer, the lower the distribution of final grades is. This makes sense, as we have already seen this correlation in the results of our model. Between `hours` and `contact` there seems to be no correlation. The amount of hours a student invests in writing the paper, does not lower the hours invested in a systematic way.

But this does not clear the model of suspicions of non-linearity just yet. Even when all pairwise relationships are linear, controlling for multiple variables at the same time can introduce non-linearity for this specific combination. One way to approach this is to inspect the *Residuals vs. Fitted* plot. As the name suggests, this plots the fitted values, i.e. the estimates for our dependent variables based on the model, against the residuals of the dependent variable. When the relationship is linear, we should see a more or less straight line along the x-axis, where $y = 0$.



For many use cases, the line is straight enough, indicating no clear and strong patterns of non-linearity. Still the residuals seem to be slightly off for very good and very bad estimated grades.

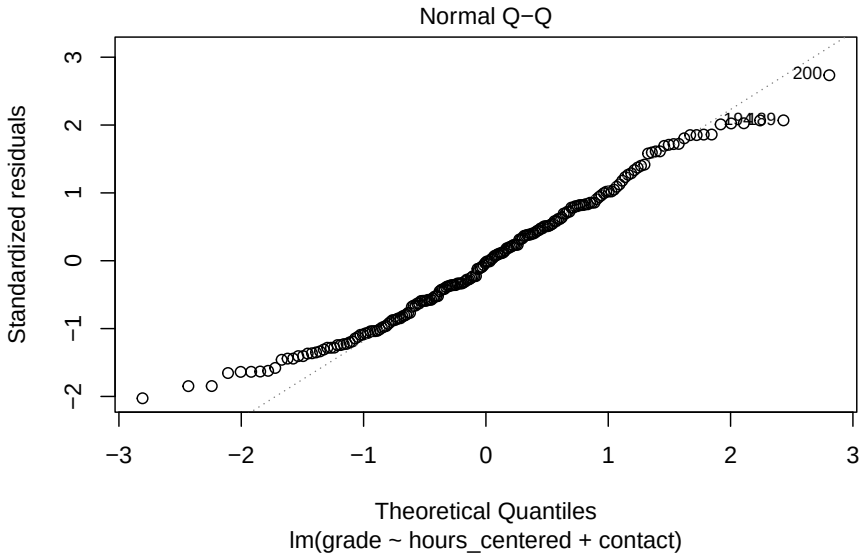
Besides violating the assumption of linearity, patterns in the residuals vs. fitted plot can also indicate that there is some important explanatory variable missing from the model. We will return to this after we have applied the further tests and discussed the remaining assumptions.

7.3.2 Normally distributed residuals

Another assumptions in linear regression is that the residuals are normally distributed. This is especially relevant for sample with a small n as the test statistics tend to get unreliable in these cases if the residuals are not normally distributed. For larger samples, as in our case, this is not as problematic. Still, systematic deviations from normality can indicate that the model is not *parsimonious*. This means that either not all relevant variables are included or that variables are included that are not necessary for the model.

We get a first idea of the distribution from the model summary. This shows us the median as well as the 25- & 75-percentiles and the minimum and maximum values. While strong and clear violations against the normality assumption could already be visible here, these measures are not enough to actually test for normality. A more informative and accurate approach is using a *Q-Q plot*. This plots the standardizes residuals, the residuals divided by their standard

error, against a theoretical normal distribution. If the residuals are perfectly normally distributed, each data point lies on the diagonal, if they are not they move away from the line. Small deviations, especially in the tails, should not be over emphasized. What we are looking for are clear and systematic deviations.



In the case of our model, most data points lie on the diagonal, while there are some small deviations in the tails. As our n is large enough, this should not be problematic. It may indicate that the model is not parsimonious but there is no clear cause for concern here.

7.3.3 Homoscedasticity

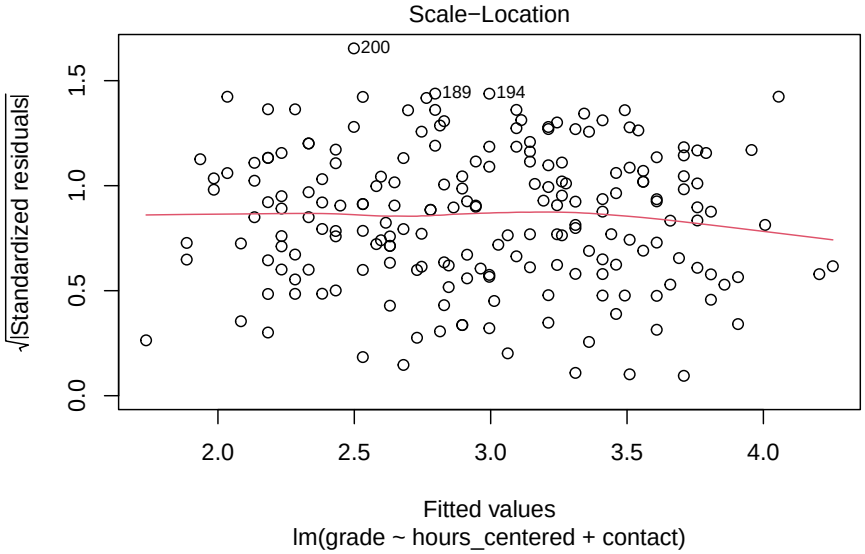
The *homoscedasticity* assumption states that the residuals are expected to have a constant variance over the whole range of the dependent variable. Let us assume that the variance of our residuals would be lower for very good grades and higher for very bad ones. This would indicate that we can make more accurate estimates for better grades than for worse ones as a small variance would indicate smaller residuals and thus a smaller error. For the assumption to hold we must be able to make about the same quality, be it high or low, for all values of `grade`.

The problem is that the computation of the standard errors, test statistics and p-values depends on this assumption. If the assumption is violated, if we have *heteroscedasticity*, these measures are not reliable anymore.

The problem often occurs when the dependent variable is not symmetric. In

the scatterplot matrix above, we already saw that `grade` is fairly symmetrically distributed, so we would not expect problems here. If our dependent variable was unsymmetrical, transforming it to be more symmetrical, e.g. by using the logarithm or a square root, could help.

To check for problems with heteroscedasticity, we can use the *Scale-Location* plot. This plots the fitted values against the square root of the standardized residuals. For homoscedasticity to hold, we should see our data points as a horizontal band with more or less constant width running from the left to the right. The same goes for the plotted line.



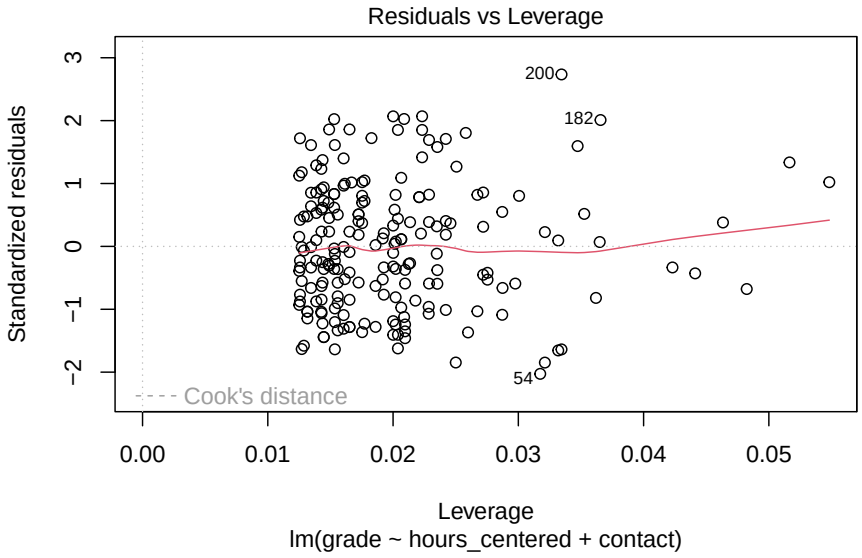
In our case the homoscedasticity assumption holds. Slight variations are not problematic and overall the variance is constant.

7.3.4 No overly influential data points

Observations can get highly influential if they have unusual values. Sometimes these are extremely low or high values on some variable. But even "normal" values on two or more variables can get unusual in their combination. Imagine a student with 60 invested hours. A high value but not overly extreme. Now the same student had in person contact with their lecturer but still received a 5.0. This could potentially be an influential data point as this combination is unusual in terms of what the model expects. In case of our model this observation would most probably not be overly influential. But imagine the same observation with 300 invested hours. Such extreme cases can influence the fit by figuratively

"puling" the regression line in their direction.

We can divide influential data points into unusual values on the dependent variable, *outliers*, and unusual values on independent variables, *high leverage points*. The latter have high leverage because they "pull" on the regression lines and thus change the slope. As a rule of thumb, we can consider values with standardized residuals over 3 or under -3 outliers. Concerning the dependent variables we can compute the *leverage statistic*. Here values that exceed $2 * (p + 1)/n$, where p is the number of predictors, are considered as having high leverage. We can inspect both at the same time in the *Residuals vs. Leverage* plot.



We can see that there are no clear outliers. To assess points with high leverage we first have to compute the threshold as: $2 * (3 + 1)/200 = 0.04$. Note that while we have two independent variables in our model, we actually have three predictors due to our categorical variable. Thus we have to compute with $p = 3$ instead of $p = 2$. We can see that there are a number of points that exceed this value. The question is, why do these values exist? Sometimes these are measurement errors, extreme values or unusual combinations that come down to the researcher recording the wrong values into the data set. In these cases we can try to fix the errors or remove the observations from the data. As we have several values with high leverage, this seems highly unlikely. But if we had not simulated the data ourselves and knew that there is no error, we should at least check. What seems more probable though, and is often the actual root of high leverage, is that there are variables missing from the model that could explain the high values. In this case the values should be lower after we include

the missing variables into the model. We will return to this later.

We also may not have a serious problem here. Leverage on its own does not have to be problematic. Some data points will always be more influential than others and remember that the cutoff is always just a rule of thumb. As said above, it is the combination of unusual values that tends to get problematic. We can check for observations that are outliers and have high leverage visually in our plot. Problematic observations tend to gather in the upper and lower right corners. As neither are populated for our model, we can not really conclude that we have overly influential data points.

XXX IS LEVERAGE CORRECTLY COMPUTED? XXX

7.3.5 No (multi)collinearity

The final assumption we will discuss here, is the absence of (high) collinearity between the predictor variables. Collinearity is present, if two independent variables are highly correlated with each other. This can become a problem as it gets harder to individually estimate the effects for both variables on the outcome as the collinear variables vary together at the same time.

Often collinearity can already be spotted in the correlation matrix. Considering our matrix above we saw no clear indication that `hours` and `contact` are correlated. But the problem can get more complicated if we include three or more independent variables in our model. While none of the pairs of variables may be highly correlated, correlation may exist for a set of three or more of those variables. In these cases we speak of *multicollinearity*. We can not spot this in a correlation matrix, but there is an easy to use measure available.

The *variance inflation factor* (VIF) can be used to inspect (multi)collinearity between two or more independent variables accurately. A VIF of 1 would indicate no collinearity. For real world data this is almost never true as some amount of collinearity always exists. But in general we can say that the VIF should be near 1 and should not exceed a value of 5.

Let us compute the measure for our model with `hours_centered` and `contact`:

```
## Warning: package 'car' was built under R version 4.2.3
```

```
## hours_centered        contact
##       1.004352       1.004352
```

Both values are very close to 1 so we can conclude that we did not violate the assumption. But what could we do, if we did? One approach is to just delete one of the highly correlated independent variables from the model. As they vary together, it may be save to exclude one of them without losing to much information. Another approach would be to combine both variables into a new

measure. Let us imagine that besides `contact` we would have another variable in our model, measuring how well a student feels supported by their lecturer in writing the paper. We could also imagine both variables being strongly correlated as they measure comparable concepts. We could then either drop one of the variables, maybe losing some information in the process, or we could combine both into a new variable which measures the form and the feeling of support at the same time, maybe leading to a more accurate estimate while at the same time eliminating the problem of collinearity. Which one is the right solution depends on the specific case.

7.4 Returning to our research question

When we tested for linearity above, we saw a mild pattern in the data which is not explainable by a violation of the assumption of linearity and thus could be an indication of a missing relevant explanatory variable in our model. Some of the other tests also supported this notion. The Q-Q plot showed us that the residuals have some slight deviations from normality in the tails. While these deviations are small enough to not cause concern on their own, taken together with the residuals vs. fitted plot this gives more weight to the suspicion that some important variable is missing. We also identified some of observations with high leverage. While we can rule out errors in our data, the high leverage could also be explainable by a missing variable.

But which variable could be missing from the model? If our DAG is correct, we can rule out `attendance` and `previous_grades`. We did assume that `contact` is a confounder for `hours` and `grade` and thus included it in our model. Of course we could also miss a variable that is not in our data at all, or even one that is not measureable. As we did simulate the data, we know this is not true, but with real world data this is always a possibility.

Let us think about the `contact`variable once more. We did assume, that the more personal the contact, the more efficiently the time working on the paper can be used. And here may lie the problem. The way we included `contact` in the model is not the way we reasoned in our DAG. It would be correct if we assumed that the more personal the contact, the less time has to be invested. But we already saw in the scatterplot matrix that there is no such relationship between the variables. To specify the effect of `contact` in the model correctly, reflecting the idea of a more efficient use of time the closer the contact was, we have to include it as an interaction with `hours`.

7.4.1 Interactions

In an *interaction*, we assume that the effect of one variable differs based on the value of another variable. Let us return to the formula for a multiple regression with two variables:

$$y = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \epsilon$$

Here we assume that the value of y varies with the value of x_1 and x_2 as indicated by the coefficients β_1 and β_2.

But we could also follow the notion that the value of x_1 influences y differently based on the value of x_2. For example the effect of x_1 on y could be higher when x_2 also has a high value. This is an interaction and is reflected in the formula by adding an additional multiplicative term between the two dependent variables with an additional associated coefficient:

$$y = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \beta_3 * x_1 * x_2 + \epsilon$$

To get a better understanding of this, let us return to our model and add an interaction between `hours_centered` and `contact`.

```
##
## Call:
## lm(formula = grade ~ hours_centered + contact + hours_centered *
##     contact, data = grades)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.77816 -0.72882 -0.08719  0.56140  2.53271
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      3.44466    0.10345  33.298  < 2e-16 ***
## hours_centered                  -0.02893    0.01535  -1.885  0.06098 .
## contactE-Mail                   -0.46775    0.16729  -2.796  0.00569 **
## contactIn Person                -1.01493    0.15171  -6.690 2.33e-10 ***
## hours_centered:contactE-Mail    -0.02377    0.02657  -0.895  0.37204
## hours_centered:contactIn Person -0.05017    0.02442  -2.055  0.04125 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9253 on 194 degrees of freedom
## Multiple R-squared:   0.28,  Adjusted R-squared:  0.2615
## F-statistic: 15.09 on 5 and 194 DF,  p-value: 1.63e-12
```

How can we interpret these results? While the estimates for the intercept and for having e-mail or personal contact in comparison to having no contact at all have barely changed, the coefficient for the amount of hours invested substantially shrunk to almost half its former value. Untll now, we assumed that the effect

of `hours` would be the same for each student. Now that we have included an interaction we assume that the effect of `hours` differs, based on the form of contact a student had.

Let us rewrite our formula for $\hat{y}$ including the interaction. As we are interacting with a categorical variable with three categories, we have to add two interaction terms. The first for the effect of invested hours when e-mail contact was made and the second for the effect of hours when contact was made in person, in both cases compared to having had no contact.

$$\hat{y} = b_0 + b_{hours\_centered} * x_{hours\_centered} + b_{E-Mail} * x_{E-Mail} + b_{InPerson} * x_{InPerson} + b_{hours\_E-Mail} * x_{hours\_c}$$

Let us now also add the coefficients from the model:

$$\hat{y} = 3.44466 - 0.02893 * x_{hours\_centered} - 0.46775 * x_{E-Mail} - 1.01493 * x_{InPerson} - 0.02377 * x_{hours\_centered} * x_{I}$$

We can now consider the three possible forms of contact one by one.

What happens, when a student had no contact? To explore this, we return to the regression formula and equal x_{E-Mail} and $x_{InPerson}$ to 0, which means that no contact was made beforehand. Note that for now we do not care about the actual value of `hours_centered`.

$$\hat{y} = 3.44466 - 0.02893 * x_{hours\_centered} - 0.46775 * 0 - 1.01493 * 0 - 0.02377 * x_{hours\_centered} * 0 - 0.05017 * x_{hours}$$

This shortens to:

$$\hat{y} = 3.44466 - 0.02893 * x_{hours\_centered}$$

For a student who did not make contact, we would estimate the final grade as the intercept minus 0.02893 per hour invested more than the mean of `hours_centered`. Having equaled x_{E-Mail} and $x_{InPerson}$ to 0 not only "switched off" the effects of `contact` but also removed the interaction effects from the equation, the estimated effect for `hours_centered` is only its coefficient of -0.02893.

What happens, when a student had e-mail contact?

$$\hat{y} = 3.44466 - 0.02893 * x_{hours\_centered} - 0.46775 * 1 - 1.01493 * 0 - 0.02377 * x_{hours\_centered} * 1 - 0.05017 * x_{hours}$$

This shortens to:

$$\hat{y} = 3.44466 - 0.02893 * x_{hours\_centered} - 0.46775 - 0.02377 * x_{hours\_centered}$$

and further to:

$$\hat{y} = 2.97691 - 0.0527 * x_{hours\_centered}$$

The intercept is reduced by the coefficient of having e-mail contact, but what is actually of interest here is the effect that `hours_centerd` has. For a student who had e-mail contact, each hour invested above the mean reduces the estimated grade by -0.0527.

We can compute the same for a student with personal contact:

$$\hat{y} = 3.44466 - 0.02893 * x_{hours\_centered} - 0.46775 * 0 - 1.01493 * 1 - 0.02377 * x_{hours\_centered} * 0 - 0.05017 * x_{hours\_centered} * 1$$

$$\hat{y} = 3.44466 - 0.02893 * x_{hours\_centered} - 1.01493 - 0.05017 * x_{hours\_centered}$$

$$\hat{y} = 2.42973 - 0.0791 * x_{hours\_centered}$$

For a student who had in person contact, each hour invested above the mean reduces the estimated grade by -0.0791.

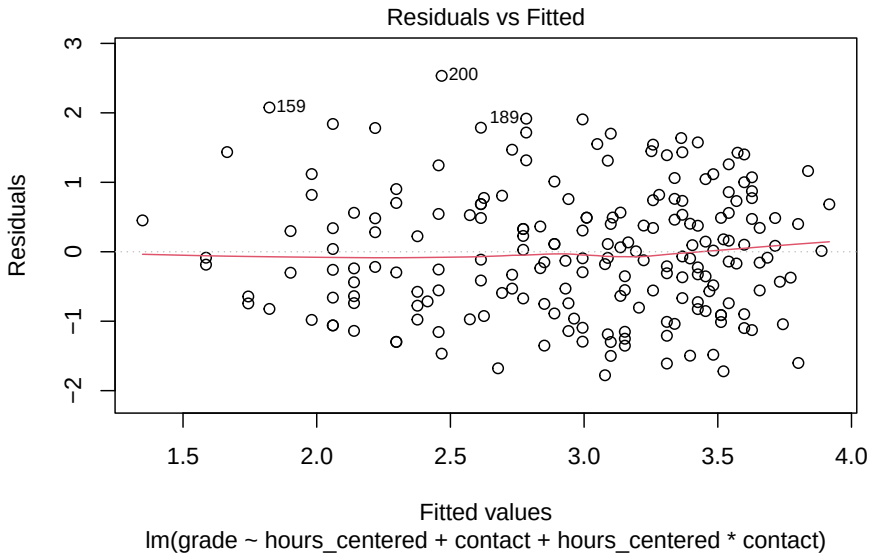
In practice we would have reached these conclusions more quickly by inspecting the output from our model and just subtracting the corresponding interaction effect from the effect for `hours_centered`, but it is important to understand what happens in the formula to get a full grasp on linear regression models.

In the model without the added interaction we concluded that on average each hour invested above the mean decreases the final grade by about -0.5. Now we see that the effect of hours depends on the form of contact had. This reflects the theoretical assumption from our DAG that time can be used more efficiently the more personal the form of contact was.

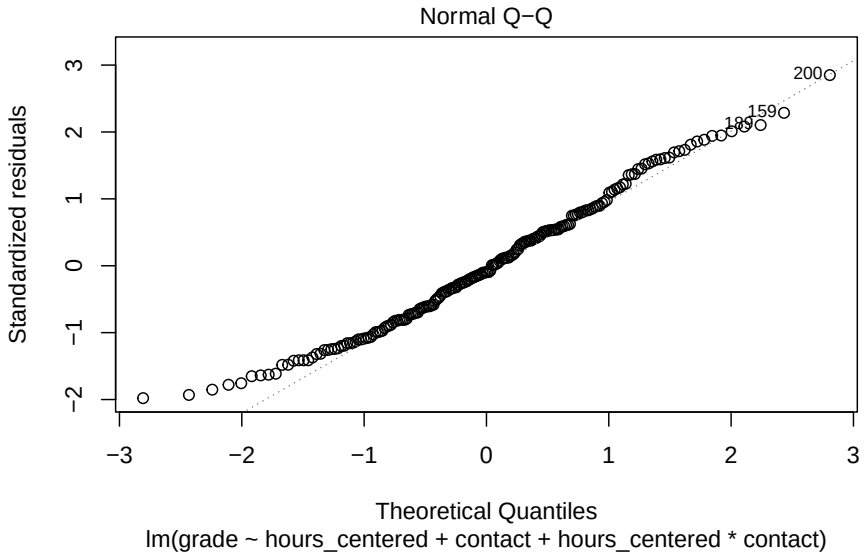
The same DAG that informed our best model from last week now lead us to including the interaction. This underlines the importance of thorough theoretical thinking before starting to model. It was not even the case that the DAG was wrong, but our conclusions we draw from it were at least not completely right. If we had invested more time, we could have build the correct model directly. In our case we first needed the regression diagnostics to tell us that something might be off before we figured out our error.

7.4.2 Regression diagnostics (revisited)

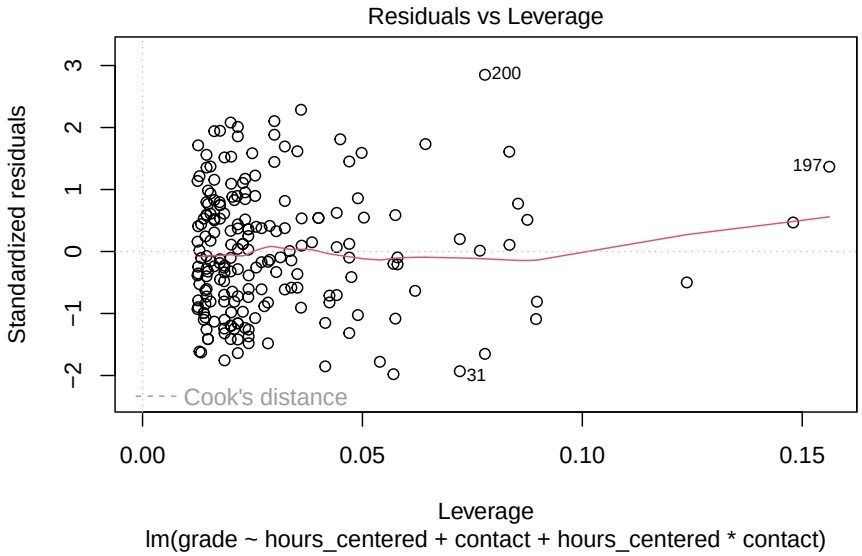
We now have a theoretically sound model, but did we also solve the problems indicated in the regression diagnostics?



The residuals vs. fitted plot now shows an almost straight horizontal line with no clear visible patterns. This indicates that the problem we saw with our former model actually came down to a missing variable, or to be more precise a missing term in our case.



The Q-Q plot now also shows more normally distributed residuals. While there are still some small deviations at the lower tail these do not indicate a remaining severe problem. The deviations are smaller than before and also not drastic in absolute terms. Also, as stated above, violating the normality assumption is less problematic with a high n and few variables in the model, which is still true for our case.



Adding the interaction term actually increased the leverage of the more influential observations. When we recompute the threshold as $2 * (5 + 1)/200 = 0.06$, we see that there still some observations with values higher than this. What has not changed, is that there are no clusters of observations in the lower or upper right corners. Overall we can conclude, that this problem is present but negligible. When there are observations that are problematic on both values at the same time, these also get marked by a red dashed line in the plot. This is also not present.

7.5 Conclusion

Over the last three sessions we have learned what a linear regression is, how its formula worked, how to interpret the results for different kinds of variables as well as how to check and correct violations of its underlying assumptions.

At the same time we built a model, which in its final version is able to accurately estimate our effect of interest. But we had one immeasurable advantage: We simulated the data ourselves and thus knew where the journey was going to end up from the start. We new our DAG was correct because the data was simulated in this way and we also knew that there was going to be a interaction effect to solve the remaining diagnostic problems. Sneaky, right? But in real world data, we do not have these advantages. Our DAGs can be incorrect and we may or may not find the missing part of the puzzle that elevates an OK model to a great one. All we can do is think, explore our data, think again, run

diagnostics, think again and maybe most importantly do not give up along the way.

In the next session we will return to our NBA data and try to apply everything that we have learned over the last sessions.

Chapter 8

Linear Regression - Application

After we have learned the ins and outs of linear regression we will now return to our NBA data. We already saw, that there was an interesting relationship between the points a player makes per game and the salary he receives in session 2. In session 4 we also built a DAG that reflects our assumptions about the data generating process. Based on the DAGs implications we can now build a linear regression model and try to estimate the effect of interest as accurately as possible.

8.1 Objectives

- Estimate the effect of interest, scored point on salary, using a linear regression
- Applying diagnostics to the model and correct mistakes
- Interpret the final model

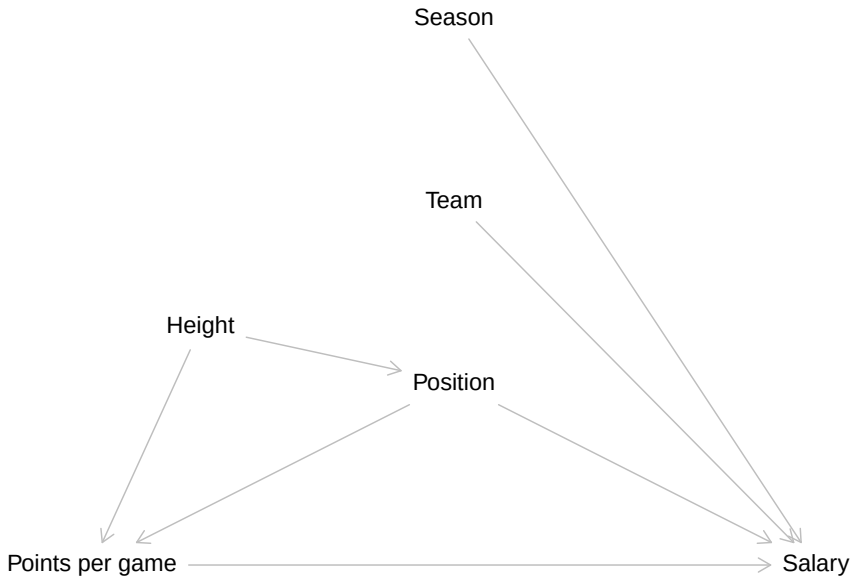
8.2 R functions covered this week

- `lm()` XXX FILL IN XXX

8.3 Research question

Picking up from session 2 & 4, our research question is still to get an unbiased estimate for the effect from points scored on the salary an NBA player receives.

We already constructed a DAG that reflects our assumptions for the underlying data generating process. Let us revist this briefly:



The implications of our DAG were that we only have to control for the position a player occupies to get an unbiased estimate for our effect of interest. The path that passes the body height is already closed by controlling for position and the team a player plays for as well as the season an observation was recorded in do not lie on an paths from our main independent to our dependent variable. Based on this we construct our model.

Now let us get to it and load the NBA data we prepared in week 2.

```
library(tidyverse)
load("../datasets/nba/data_nba.RData")
```

8.4 Simple linear regression in R

To conduct a multiple linear regression in R, we can use the built-in *base R* function `lm()`, short for *linear model*. The function is straightforward to use. As the first argument we write the regression formula in R's *formula syntax*.

We start building the formula by writing the name of our `dependent_variable` followed by a *tilde* `~`. You can read this as an = or as "regress the dependent variable on". After the tilde we add our first `indepedent variable` by again writing out its name. If we have multiple independent variables in our model -

when we are running a *multiple linear regression* - we can add those by writing a + followed by the name of the variable to be added.

As a second argument, the function needs the name of the object that holds our data.

The goal of our research question is to estimate the effect of the points per game on the received salary. So to regress `salary` on `career_PTS`, we just write:

```
lm(salary ~ career_PTS, data = data_nba)
```

```
##
## Call:
## lm(formula = salary ~ career_PTS, data = data_nba)
##
## Coefficients:
## (Intercept)   career_PTS
##     -851914       552843
```

This gives us a short output. The first line just echoes our code used to run the regression. We have seen this in the last session already, but now we know what the meaning was. After this we have a short block with the estimated coefficients. As we have run a simple linear regression, we only get the intercept and the coefficient for the sole independent variable used in the model. If we would have run a multiple linear regression, the result would basically look the same, only with more coefficients to display.

Before we dive into the results, we should talk about how to receive a more verbose output that does not hide all the other vital information that is associated with the model.

The easiest way is to use the base R function `summary()`. This is a generic R function that returns different summaries, depending on the object it is used on. We can for example use it on a data frame or tibble to get some descriptive statistics for the included variables. For example, we can information on the distribution of points per game by writing:

```
summary(data_nba$career_PTS)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   5.100   8.000   8.908  12.000  30.100
```

When we use `summary()` on a model object, like the one created by `lm()`, we get a different output. Before we apply this we should save our model in an object. This is good practice in most cases as we can now apply all additional analysis of the model on this object and we do not have to rerun the model every time.

```
m1 <- lm(salary ~ career_PTS, data = data_nba)
```

We can now apply `summary()` on the object `m1`, short for "model 1":

```
summary(m1)
```

```
##
## Call:
## lm(formula = salary ~ career_PTS, data = data_nba)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -14788659  -2023969   -434599   1311807  24326060
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -851915      76417  -11.15   <2e-16 ***
## career_PTS    552843       7453   74.17   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3732000 on 9726 degrees of freedom
## Multiple R-squared:  0.3613, Adjusted R-squared:  0.3612
## F-statistic:  5502 on 1 and 9726 DF,  p-value: < 2.2e-16
```

This is the output we saw over the last weeks and it includes extended and better readable coefficient block as well as the information on the residuals and the model fit.

An alternative method of displaying the coefficients in a regular tibble format, is to use `tidy()` from the `broom` package.

```
library(broom)
```

```
## Warning: package 'broom' was built under R version 4.2.3
```

```
tidy(m1)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept) -851914.    76417.     -11.1 1.09e-28
## 2 career_PTS   552843.     7453.      74.2 0
```

8.4.1 Interpretation

While we know our model is not complete yet, let us still inspect the results. For each point a player scores per game, his salary rises by about 552,000$. We see a clear positive and substantial effect. Let us also inspect the intercept. This tells us that a player who makes no points per game has to pay the team about 850,000. Wait, this does not make sense… To make the intercept more readily interpretable we should again center our metric dependent variable `career_PTS` on its mean.

```
mean(data_nba$career_PTS)
```

```
## [1] 8.907679
```

```
data_nba <- data_nba %>%
  mutate(PTS_centered = career_PTS - mean(career_PTS))
```

As we have now centered the independent variable of interest on its mean of 8.9 we can rerun the model.

```
m1 <- lm(salary ~ PTS_centered, data = data_nba)

summary(m1)
```

```
##
## Call:
## lm(formula = salary ~ PTS_centered, data = data_nba)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -14788659  -2023969   -434599   1311807  24326060
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4072633      37840  107.63   <2e-16 ***
## PTS_centered   552843       7453   74.17   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3732000 on 9726 degrees of freedom
## Multiple R-squared:  0.3613, Adjusted R-squared:  0.3612
## F-statistic:  5502 on 1 and 9726 DF,  p-value: < 2.2e-16
```

The coefficient for points per game has not changed but its interpretation has. For each point per game over the mean of 8.9 points per game, the salary is

estimated to increase by about $552,000$\$. At the same time, for each point below the mean the salary is estimated to decrease by the same amount. The intercept now shows us the estimated salary of a player who scores 8.9 points per game which is slightly upwards of $4,000,000$\$ This makes way more sense.

This model model already achieved a considerable R^2 of 0.36. About 36% of the variance in salaries is explained by the points per game a player scores.

8.5 Multiple linear regression in R

The DAG we have constructed above based on our research question indicated that we also have to include the position a player occupies in our model. We can add additional independent variables to the formula used in `lm()` with a `+` and the name of the additional variable(s). This works the same way for all types of variables, i.e. metric, dummies or categorical variables. So let us do this now by adding the 5 dummies we constructed for the positions:

```
m2 <- lm(salary ~ PTS_centered + position_center + position_sf +  position_pf + positi

summary(m2)
```

```
##
## Call:
## lm(formula = salary ~ PTS_centered + position_center + position_sf +
##     position_pf + position_sg + position_pg, data = data_nba)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -14511723  -1950255   -372906   1358768  24433660
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3679728     114300  32.193  < 2e-16 ***
## PTS_centered      568019       7474  75.994  < 2e-16 ***
## position_center  1380246     114539  12.050  < 2e-16 ***
## position_sf       125384     102174   1.227 0.219790
## position_pf       206505      94882   2.176 0.029547 *
## position_sg      -331033      96841  -3.418 0.000633 ***
## position_pg      -114552     117486  -0.975 0.329572
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3652000 on 9721 degrees of freedom
## Multiple R-squared:  0.3888, Adjusted R-squared:  0.3884
## F-statistic:  1031 on 6 and 9721 DF,  p-value: < 2.2e-16
```

8.5.1 Interpretation

We still see a clear positive effect of points per game on the received salary after controlling for the position a player occupies. Among those centers are by far the top earners, making about $1,400,00$ more than players on other positions. Most other positions show relatively small effects on the earnings. Power and small forwards earn somewhat more than other positions on average while point and especially shooting guards earn less.

We can now compare two fictive cases of a center and a point guard who each make about 20 points per game. What is the estimated salary for them?

As we have extensively worked with the formulas over the last sessions, we can now keep it short and calculate the estimate directly. Remember that we centered the points per game on the mean of about 8.9, so making 20 per game would mean making about 11.1 than the average player. We will keep it simple here and calculate with 11.

$$\hat{y_{center\_20}} = 3679728 + 568019 * 11 + 1380246 = 11,308,183$$

$$\hat{y_{pg\_20}} = 3679728 + 568019 * 11 - 114552 = 9,813,385$$

Despite making the same amount of points per game for their team, the model estimates that a point guard earns about $1,500,000$ less compared to a center.

8.5.2 Sidenote: Adding interactions

We will not use interactions in this session but we briefly want to state how we could add them in the formula syntax.

Remember that interactions are multiplicative terms in our regression formula. Adding them to the R formula syntax works the same way. We add the new term with a `+` and use a `*` between the two variables that we want to interact.

Here is a non running toy example where we interact two x-variables:

```
lm(y ~ x1 + x2 + x1 * x2, data = some_data)
```

8.6 Regression Diagnostics

So how does our model perform? Did we meet all the regression assumptions that were introduced last week?

To access the visual tests we used last session, we can just use the base R function `plot()`, applied to the model object. If we just write `plot(m2)`, the

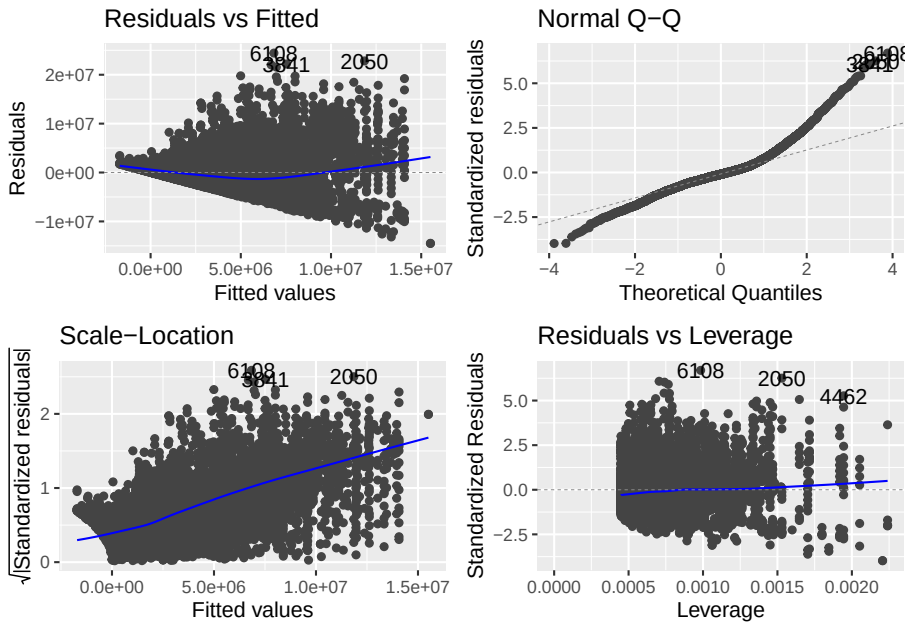
console asks us to press `ENTER` to go through each plot one by one. We can also add a number as a second argument, specifying which plot we want to see. For example, `plot(m2, 1)` gives us the residuals vs. fitted plot.

But there is an easier way to see all four plots at once. The package `ggfortify` expands the functionalities of `ggplot2` so that we can use it's `autoplot()` function to automatically plot all four visual tests of interest. An added benefit, depending on your taste, is that the plots are rendered in the style of `ggplot2`.

```
library(ggfortify)
```

```
## Warning: package 'ggfortify' was built under R version 4.2.3
```

```
autoplot(m2)
```



The residuals vs. fitted does not show us a more or less straight line but starts mildly positive, then dips below 0 and rises again for higher estimated salaries. This could indicate at least two thing. Either we have missed an important independent variable, like in the last session, or we are actually dealing with some amount on non-linearity. If it is non-linearity, it is still *mild* non-linearity, but maybe we should still inspect this.

The Q-Q plot this time shows that the actual residuals are far from being distributed normally. While we can never expect a perfectly normal distribution, here the deviations are striking, especially for high residuals.

The scale-location plot is used to address the assumption of homoscedasticity. What we want to see, is a straight line with data points equally distributed around it. This clearly is not the case here. As it is, the plot indicates that we may be able to estimate small salaries reasonably well but that the higher the estimate, the more unreliable our model gets.

The residuals vs. leverage plot also indicates some problems. There are some observations that have larger or smaller standardized residuals compared to the thresholds of 3 and -3. The threshold for leverage is computed as $2 * (6 + 1)/9728 = 0.001439145$. We also see some observations with higher values. While both are rules of thumb and may not necessarily point to severe problems by themselves, things can get problematic when there are observation that do not meet the thresholds for both measures at the same time. This is indicated by clusters in the lower or upper right corners. This time we can observe this in the lower right.

We should also test for multicollinearity. We can compute the VIF measures using a function from the package `car`.

```
library(car)

vif(m2)
```

```
##    PTS_centered position_center     position_sf     position_pf     position_sg
##        1.050400        2.035722        1.686736        1.512765        1.565004
##     position_pg
##        1.916933
```

The values for any variable should not exceed 5 and should be closer to 1. Our value for points shows no signs of multicollinearity. The values for the position dummies have somewhat higher values, which makes sense. While there are some players that play multiple positions, for most a value of 1 on one position predicts the other positions as having a value of 0. But as we are still far away from the threshold of 5, there is no need for concern here.

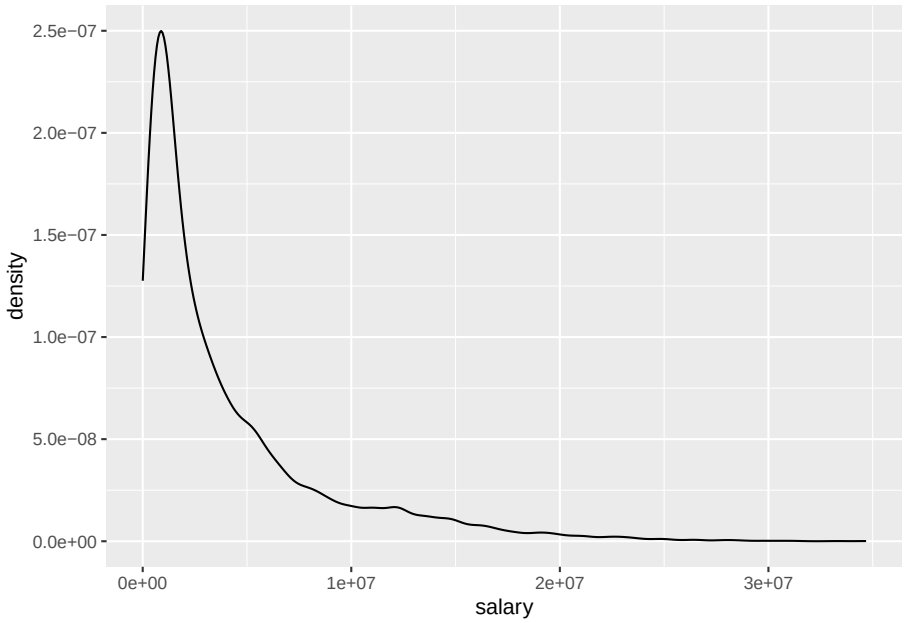
Overall, we have problems! While we do not see signs of problematic multicollinearity, all other tests indicated clear and in parts severe problems. We have to put in some more work before we can be confident that our model accurately predicts the effect of points per game on the received salary.

Before we start addressing the problems, we should not that the four plots are highly interactive. It is entirely possible that solving one of the problems also solves the others or, for added fun, even generates new ones. This means that we should refrain from turning too many dials at once and rather change the model one step at a time, see if it improves things and then address remaining problems in the same way.

8.6.1 Skewed outcome variable

The deviation from normality and the clearly present heteroscedasticity could both point to the same problem, namely a skewed dependent variable. Let us examine its distribution first.

```
data_nba %>%
  ggplot(aes(x = salary)) +
  geom_density()
```



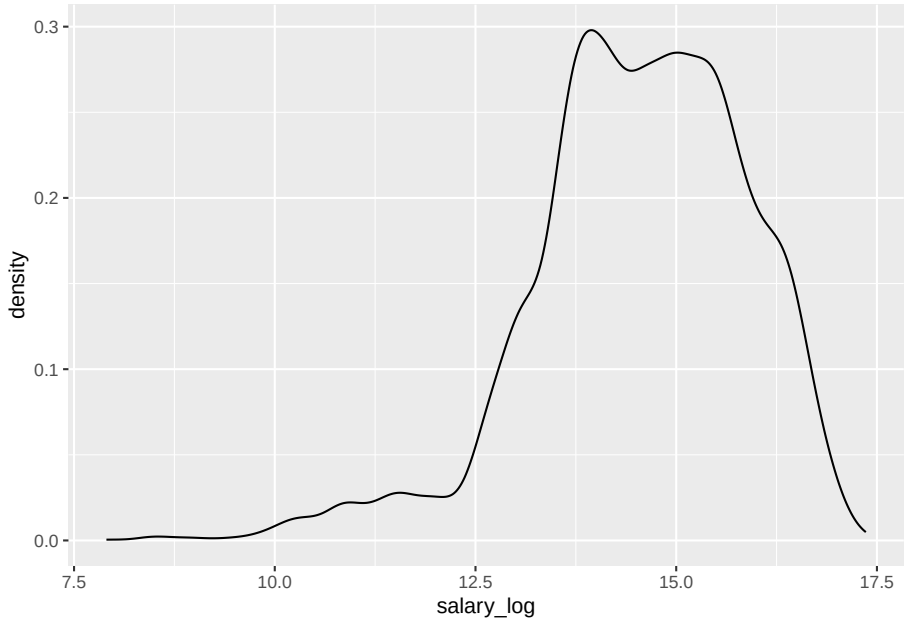
Our outcome variable is not only skewed, it is **highly skewed**. While there are many salaries in the "lower" six to seven digits regions, we also see some extremely high wages up to about $35,000,000$\$. The higher the salary the fewer observations we have. That is why we see such a long flat tail to the right.

This distribution actually is relatively common for income data. In most surveys of the general population we have many people receiving relatively low incomes while fewer individuals receive higher or extremely high incomes. It is still interesting that this also holds true for a population of high earners such as NBA players. Inequality is relative. Compared to the general population almost all our players would be somewhere in the long tail to the right. Compared to their own population we still see highly substantial differences in outcomes.

We can transform the dependent variable to a different scale to get a less skewed distribution. A common transformation for income data is to take the *logarithmus naturalis* of the actual value and then use this as our dependent variable.

To achieve the transformation we can simply use the base R function `log()` which as its default computes the ln.

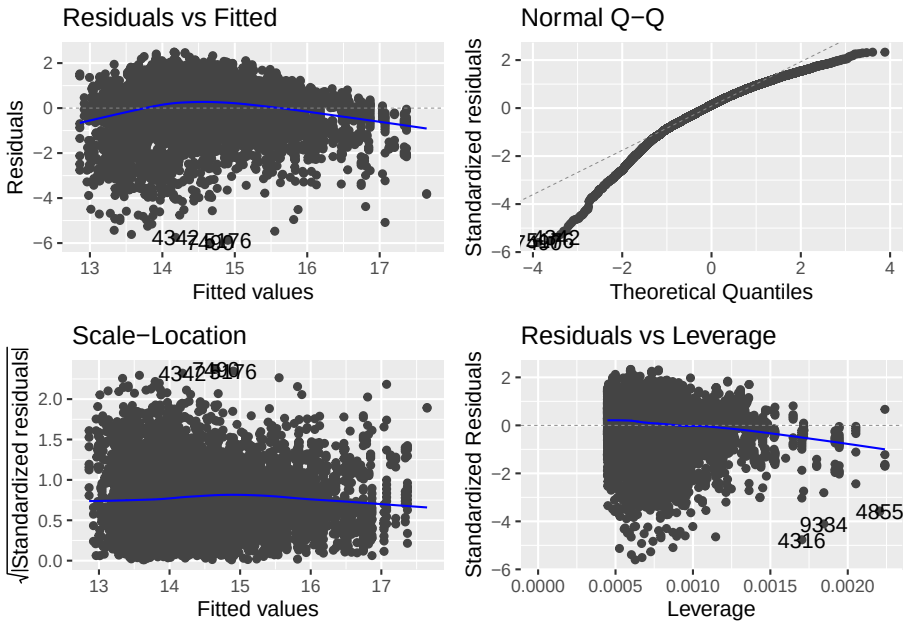
```
data_nba <- data_nba %>%
  mutate(salary_log = log(salary))

data_nba %>%
  ggplot(aes(x = salary_log)) +
  geom_density()
```



While the distribution of the transformed variable also is somewhat skewed, now to the left, overall it is much more evenly distributed.

We should use the new variable as our outcome an check the tests again.

```
m3 <- lm(salary_log ~ PTS_centered + position_center + position_sf +  position_pf + position_sg +

autoplot(m3)
```



Looking at the scale-location plot first, we can now see a straight line with our residuals fairly evenly distributed around it. Thus we now longer see any signs of heteroscedasticity. The Q-Q plot now also indicates a somewhat more normal distribution of our residuals but there are substantial deviations still. While high residuals now appear to more or less follow the normal distribution, small residuals now deviate stronger than they have before. This reflects the transformation and its distribution, which now has long tail on the left and not on the right anymore. Turning to the residuals vs. leverage plot we still see some observations that do not meet the respective thresholds. At the same time, there appear to be less that simultaneously have high absolute standardized residuals and high leverage. The residuals vs. fitted plot now also shows a more even distribution while the signs on non-linearity remain. We do not have to recompute the VIF measure as we did not change any independent variables in the model.

8.6.2 Non-linearity

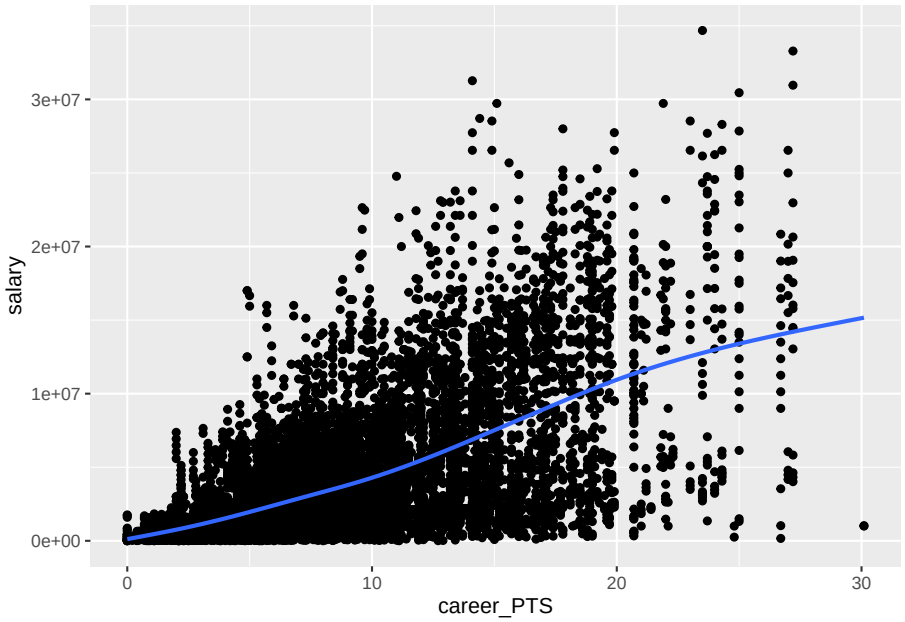
Let us now address the non-linearity that is still indicated in the first plot. We can approach non-linear relationships in our inherently linear model by adding non-linear transformations of a dependent variable to the model. But before we start squaring random variables, we should think about what could be non-linear in our case. We can rule out our dummy variables for position. This leaves the points scored. The model already tells us that our suspicion that salary rises with the points scored could be true. But maybe this relationship is

not linear over its whole range. If you already are among high scorers, scoring one or two points more than your peers may not be such a substantial difference and thus may not have the same strong effect on salary.

We should first inspect the relationship between both variables again. This time we add a LOWESS curve to the plot. This is often helpful in detecting non-linearity as the curve can change its slope over the range of the dependent variable. This is also the default for `geom_smooth()`.

```
data_nba %>%
  ggplot(aes(x = career_PTS, y = salary)) +
  geom_point() +
  geom_smooth(se = FALSE)
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```

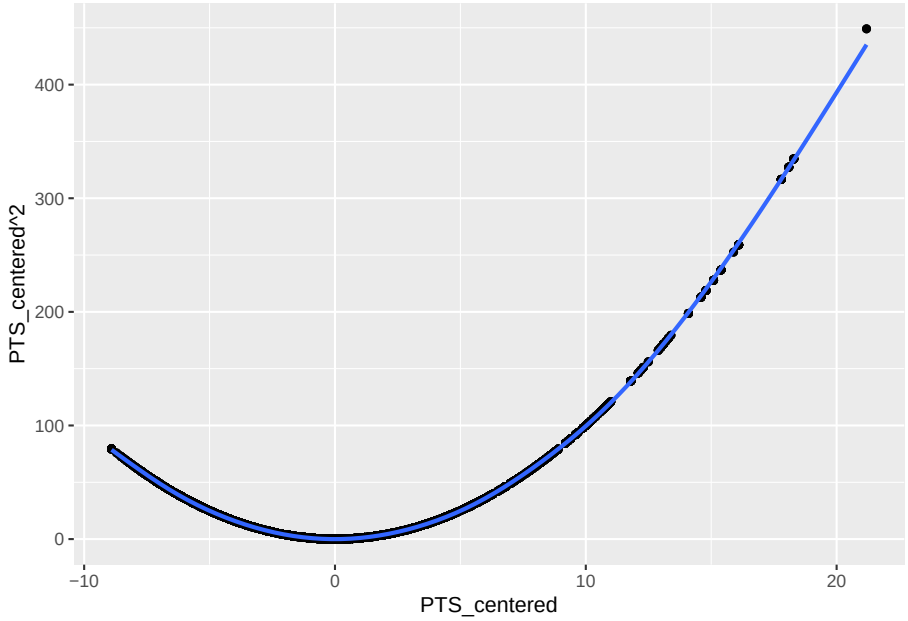


While this is not as clear as we hoped, the line may still indicate some mild non-linearity as it flattens somewhat for really high point values. Also we have to keep in mind that the non-linearity may be stronger when we control for additional variables, as our position dummies.

One common way to address the non-linearity is taking the square of the dependent variable in question. We should not square our centered points variable though. Let us inspect what would happen if we squared it.

```
data_nba %>%
  ggplot(aes(x = PTS_centered, y = PTS_centered ^ 2)) +
  geom_point() +
  geom_smooth(se = FALSE)
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```

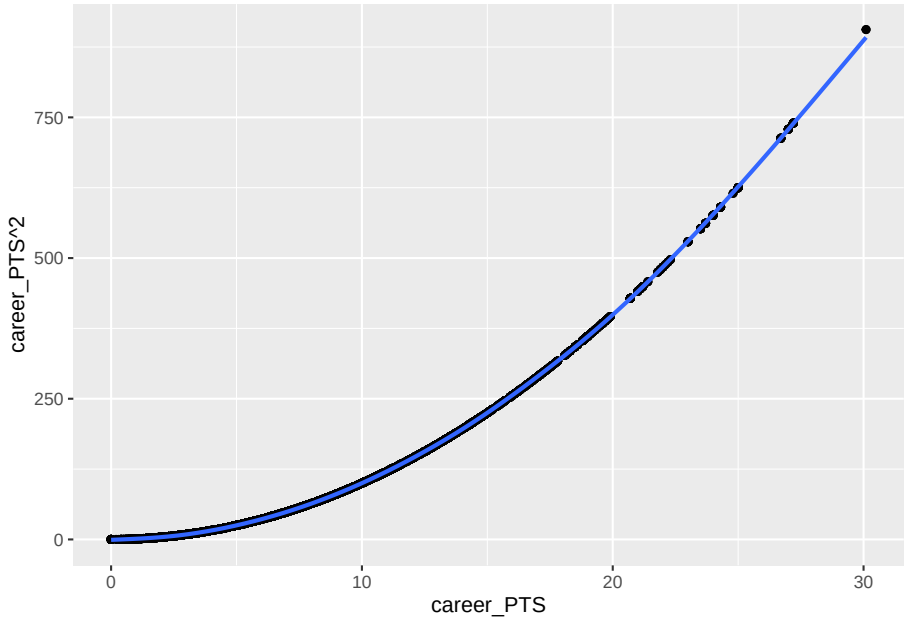


As the square of negative values is positive we would basically introduce the assumption into the model that there is non-linearity for low and high scorers and that the effect will be in the same direction. While our assumption is that there are diminishing returns between being a high scorer and a **really** high scorer, we do not assume that making more points if you are among the lower scorers should have the same effect. If at all, in these regions additional points could have an even larger effect.

Because of this we should return to the uncentred version of our variable. What happens if we square this?

```
data_nba %>%
  ggplot(aes(x = career_PTS, y = career_PTS ^ 2)) +
  geom_point() +
  geom_smooth(se = FALSE)
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



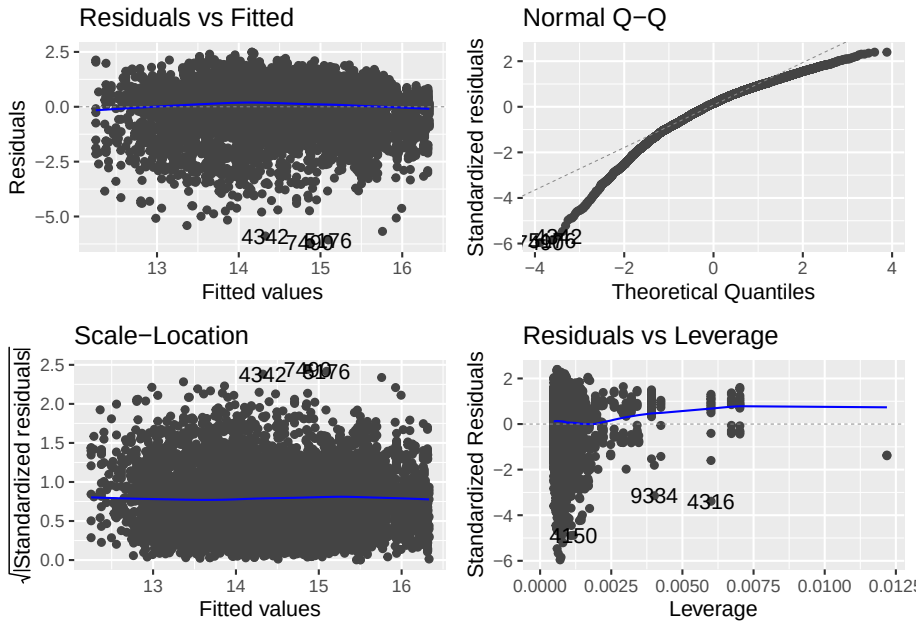
This is what we wanted, a transformation that expects a stronger difference the higher the score value is. For the final model we will thus work with the uncentred variable. We included the centered version because it is more straightforward to interpret. This is not really a concern anymore because dreams of easy interpretability are long gone after transforming two variables.

We could again transform the variable in our data and thus add a second version, but we can also do so directly in the formula syntax. When we use the function `I()` we tell R to interpret anything within the parentheses as a mathematical expression. This is what we will do below. Note that we add the point variable as its untransformed and transformed versions. The first represents the linear parts and the second the non-linear parts of the effect.

```
m4 <- lm(salary_log ~ career_PTS + I(career_PTS^2) + position_center + position_sf + position_pf
```

We can now reassess the tests for the last model.

```
autoplot(m4)
```



The line in the residuals vs. fitted plot got more straight. It seems that we actually captured the mild non-linearity that was present before by adding the squared points value to our model. The scale-location plot also still indicates no more problems of heteroscedasticity. Contrary, the data points now are even more evenly distributed compared to `m3`. The Q-Q plot also has not substantially changed, still showing non-normally distributed residuals. We can not really fix this now, but we also learned that this test is less consequential if we have a large n. Turning to the residuals vs. leverage plot, we still see several points that do not meet the thresholds but at the same time we do not see any points with high values for both. Overall there seem to be no overly influential points which we had to address. Let us also reexamine the VIF.

```
vif(m4)
```

```
##      career_PTS I(career_PTS^2) position_center     position_sf     position_pf
##       12.135096       11.883255        2.040563        1.709312        1.520284
##     position_sg     position_pg
##        1.569463        1.949078
```

We now see high VIF values for both versions of our point variable, which is the only substantial change. Did we introduce a new problem? If we take the measure at face value, yes. But if we think about it, no. All this means is that both versions of our variable are highly correlated. Of course they are. One is computed from the other. We can perfectly predict the value of `career_PTS^2`

from career_PTS. There is collinearity by design. If we want to assess multicollinearity we should apply the function to m3. If would have used interactions, the situation would be similar. This is just a small reminder that all our tests do not work without thinking about what we are actually doing.

8.7 Returning to our research question

As we now settled on m4 as our best model, it is time to discuss what we actually found out about the effect of scored points on the received salary.

```
summary(m4)
```

```
##
## Call:
## lm(formula = salary_log ~ career_PTS + I(career_PTS^2) + position_center +
##     position_sf + position_pf + position_sg + position_pg, data = data_nba)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.1886 -0.5744  0.1583  0.7318  2.4876
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     12.2544680  0.0428953 285.684  < 2e-16 ***
## career_PTS       0.3124511  0.0072287  43.224  < 2e-16 ***
## I(career_PTS^2) -0.0071879  0.0003113 -23.092  < 2e-16 ***
## position_center  0.5482072  0.0326289  16.801  < 2e-16 ***
## position_sf      0.1067444  0.0292656   3.647 0.000266 ***
## position_pf      0.1214253  0.0270641   4.487 7.32e-06 ***
## position_sg     -0.0025577  0.0275935  -0.093 0.926150
## position_pg      0.0312286  0.0337077   0.926 0.354234
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.039 on 9720 degrees of freedom
## Multiple R-squared:  0.3978, Adjusted R-squared:  0.3973
## F-statistic: 917.1 on 7 and 9720 DF,  p-value: < 2.2e-16
```

The more points a player scores, the higher the salary is estimated. At the same time we have identified a non-linear aspect to this relationship. The non-linear effect is small but negative. This indicates diminishing returns for high scorers. The higher the score, the less positive the effect of additional points is.

The problem after two transformations of involved variables is, that interpretation has lost all its intuitiveness. The effects now describe the change in

logarithmised salaries. While we can still easily assess if an effect is positive or negative we do not really know what the magnitude of an effect is. For this we have to reverse the transformation.

Let us revisit our example of a center and a point guard making 20 points per game from above. Note that we also have to take the square of points for the non-linear term.

$$\widehat{y_{center\_20}} = 12.2544680 + 0.3124511 * 20 - 0.0071879 * 20^2 + 0.5482072 = 16.17654$$

$$\widehat{y_{pg\_20}} = 12.2544680 + 0.3124511 * 20 - 0.0071879 * 20^2 + 0.0312286 = 15.659565$$

To find out what this means in hard dollars, we have to reverse the logarithmus naturalis. We can do this by calculating $e^{\hat{y}}$. R gives us the function `exp()` to do just that.

```
exp(16.17654)
```

```
## [1] 10601860
```

```
exp(15.659565)
```

```
## [1] 6322119
```

For our center who scored 20 points we thus estimate a salary of $10,601,860$\$, for a point guard with the same amounts of points $6,322,119$\$. These estimates are not only lower than the ones derived above, the difference between the positions is also more pronounced. For a point guard, scoring additional hoops does not have the same payoff compared to a center. While the latter receive a higher pay in general they also receive more per additional point scored.

As the name suggests and as we have seen in our exploratory data analysis, point guards make a lot of points. At the same time they earn considerable less. Above we see an estimated difference of over $4,000,000$\$ between high scoring centers and point guards. The question remains why this is the case. Maybe there are some additional variables we had to consider to fully unravel this. For example it is reasonable to expect some effects on salary that are not connected to the performance in terms of scoring. Both positions fill different roles in basketball game. Centers, besides scoring, have to get rebounds and facilitate turnarounds. Point guards on the other hand have the role to build opportunities for their team and pass to other players. Both measures of performance were not considered in our model but are highly valuable to any team.

Also there is something we can call “starfactor” or “flashiness”. A center is much more visible in the game, takes big jumps and dunks. Maybe a center is not only more valuable in terms of performance but also in being an attracting force for fans. This could change the relationship between points and salary for players with a high “starpower”. Such a variable would be hard to measure, but could maybe solve the parts of the puzzle that still remain.

8.8 Moving on

But there is another possibility. We built the best model based on our DAG, but maybe the DAG is not entirely correct. Maybe we made some faulty assumptions or maybe we missed a variable with an important role in the data generating process. In the next session we will see, how we can come up with a different DAG that incorporates the same variables but makes some different assumptions.

Chapter 9

Linear Regression - Exercises

9.1 Exercises of Linear Regression

In this exercise, you will use the Boston Housing Dataset to explore the relationship between housing prices and various features of the houses and their surroundings. The dataset contains 506 observations and 15 columns. The last column, MEDV, is the median value of owner-occupied homes in $1000's. This is the target variable that you will try to predict using linear regression models. The other 14 columns represent different features of the houses and their surroundings, such as crime rate, nitric oxides concentration, pupil-teacher ratio, etc.

DRAFT

1. **Simple linear regression**: Use simple linear regression to predict the median value of owner-occupied homes (`MEDV`) based on a single predictor variable which is the average number of rooms per dwelling (`RM`). Fit the model.

```
# loading dataset
library(readxl)
BostonHousing <- read_excel("../datasets/boston.xlsx")
# applying linear regression
model_l <- lm(medv ~ rm, data = BostonHousing)
```

2. **Multiple linear regression**: Use multiple linear regression to predict `MEDV` based on multiple predictor variables, such as `RM`, `CRIM` (per capita

crime rate by town), and `LSTAT` (% lower status of the population). Fit the model, generate predictions, and calculate the R-squared value to assess the model's performance.

```
# applying linear regression
model_m <- lm(medv ~ rm + crim + lstat, data = BostonHousing)
```

3. **Model summary**: Generate summary of a both regression model, including information about the coefficients, standard errors, t-values, and p-values. Interpret the results. Calculate the R-squared value to assess both model's performance. Which model has better goodness of fit (R-Squared) ?

```
summary(model_l)
```

```
##
## Call:
## lm(formula = medv ~ rm, data = BostonHousing)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -23.346  -2.547   0.090   2.986  39.433
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -34.671      2.650  -13.08   <2e-16 ***
## rm             9.102      0.419   21.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.616 on 504 degrees of freedom
## Multiple R-squared:  0.4835, Adjusted R-squared:  0.4825
## F-statistic: 471.8 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
summary(model_m)
```

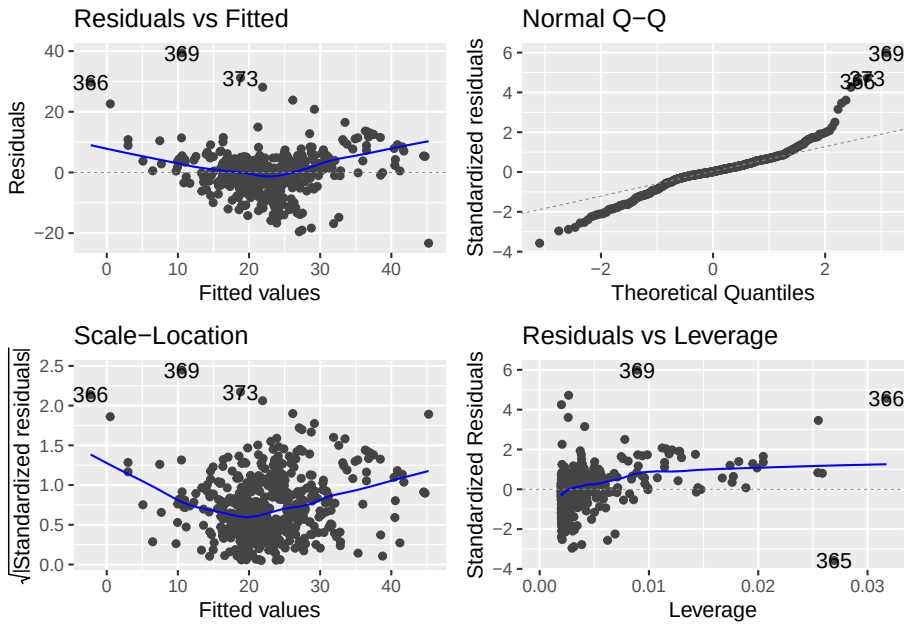
```
##
## Call:
## lm(formula = medv ~ rm + crim + lstat, data = BostonHousing)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.925  -3.566  -1.157   1.906  29.024
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.56225    3.16602  -0.809  0.41873
## rm           5.21695    0.44203  11.802  < 2e-16 ***
## crim        -0.10294    0.03202  -3.215  0.00139 **
## lstat       -0.57849    0.04767 -12.135  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.49 on 502 degrees of freedom
## Multiple R-squared:  0.6459, Adjusted R-squared:  0.6437
## F-statistic: 305.2 on 3 and 502 DF,  p-value: < 2.2e-16
```

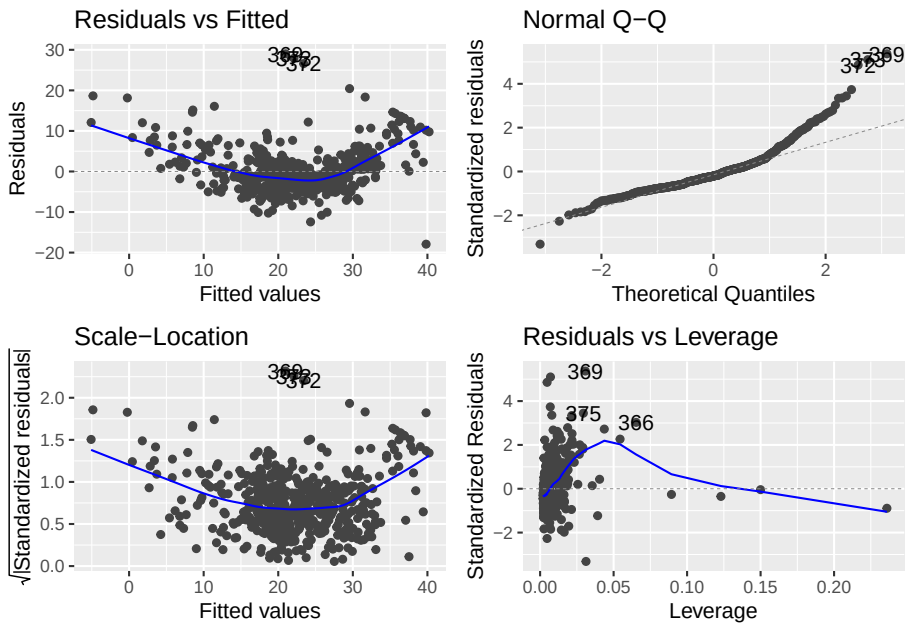
XXX–INTERPRETATION HERE–XXX

4. **Model diagnostics**: Create diagnostic plots to assess the assumptions of the models. Check whether your regression models from Tasks 1 and 2 satisfy the assumptions of linearity, multicollinearity, normality and homoscedasticity.Use numerical method variance inflation factor (VIF) to detect any multicollinearity issues among your predictor variables.

```
library(ggfortify)
library(car)
autoplot(model_1)
```



```
autoplot(model_m)
```




```
vif(model_m)
```

```
##       rm     crim    lstat
## 1.616468 1.271372 1.941883
```

XXX—INTERPRETATION HERE—XXX

Chapter 10

Mediation

In this week, we will introduce to the concept of mediation analysis.

Mediation is a way of using linear regression with a different interest in mind. So far, we have been interested in estimating the effect of an x variable on a y variable. In our example using NBA data, we were interested in the effect of 'points scored' by a player on player salary. We learned how to "adjust" or "condition" the effect by adding covariates to the model (i.e. control variables). Going back to out week on DAGs, we saw that we want to control for so-called confounders, i.e. those variables which have an effect on both x and y. The "adjusted" effect is the "total causal effect" of x on y. Mediation now is all about trying to explain "how" x affect y. Mediation analysis is about identifying the mechanism between x and y. In other words, what share of the total effect can be explained by something else (i.e. the mediator). When considering mediation, we can distinguish 3 types of effects:

- total effect (Effect of x on y (conditional on z), without mediator)
- direct effect (Effect of x on y (conditional on z), left after also adjusting for the mediator))
- indirect effect (The effect of x on y (conditional on z) which only goes through the mediator.

Go back to week X for a refresher.

Let's turn to our NBA data to make this more tangible. Let's assume now that we are interested in how a player's position on the team affects his salary. Maybe guards feel discriminated because they earn less.

First, we identify the year of the season as a confounder which we want to adjust. Tactics over the years changed, so the year when games took place had an influence on whether a player would be assigned to the guard or forward position, for example. Also, the NBA become more popular, more fans means more

money, and higher salaries for players. As a result, we will include "season_year" as a control variable.

Second, we have a hypothesis that the points scored on average affect the salaries. In the end, fans come to see spectacular dunks and shots, not rebounds and passes. Certain positions play further away from the basket (e.g. guards) and thus have a harder time scoring. They also are supposed to pass the ball to the big guys (usually centers). So maybe, the fact guards and forwards simply score less than centers explains their salary gap. We can test this using mediation.

To visualize this mini-theory and identification strategy, let's have a look at this DAG.

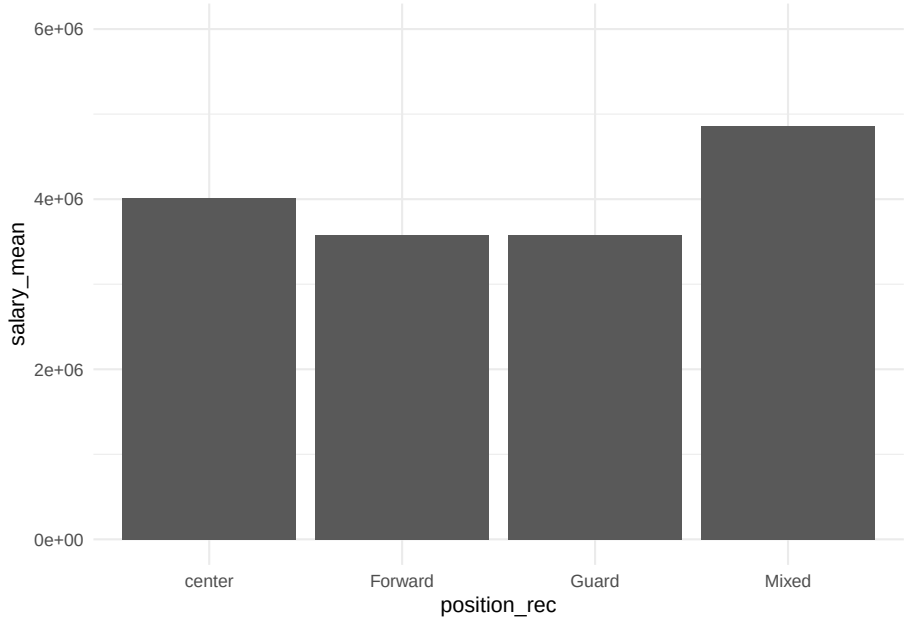
–> Jasper: visual here with the DAG from my powerpoint

Now let's test this out using R.

- First, we load the data.
- Second, we reduce the position variable to 4 categories: Center, Guards, Forwards, and Mixed. As we will see, it is actually not common that players take on several roles even within the same season. This likely affects their salary as well. Players that can play multiple positions may get more playing time and as a result, score more. We are interested, however, in the difference between "pure" guards, forwards and centers, so we will run the analysis excluding "mixed" position players.
- Third, we check whether there are any differences in average salaries by position using a graph and a table
- Fourth, we run a linear regression, regressing position (as factor variable) on salary while adjusting for year of season.

```
# load packages

load(file = "../datasets/nba/data_nba.RData")



# recode "position" variables"
data_nba <- data_nba %>%
  mutate(position_num = rowSums(across(c("position_center", "position_pf", "position_si
                                         "position_pg", "position_sg"))),
         position_rec =
           case_when(
             position_center==1 & position_num==1 ~ "center",
             position_pf==1 & position_num==1 ~ "Forward",
             position_sf==1 & position_num==1 ~ "Forward",
             position_pg==1 & position_num==1 ~ "Guard",
```

```
            position_sg==1 & position_num==1 ~ "Guard",
            position_sg==1 & position_pg==1 & position_num==2 ~ "Guard",
            position_pf==1 & position_sf==1 & position_num==2 ~ "Forward",
            TRUE ~ "Mixed"))

# pot mean salary by position
data_nba %>% group_by(position_rec) %>%
  summarize(salary_mean = mean(salary, rm.na =T)) %>%
  ggplot() +
  geom_bar(aes(x=position_rec, y=salary_mean), stat="identity") +
  ylim(0,6000000) +
  theme_minimal()
```




```
# linear regression
summary(lm(salary ~ as.factor(position_rec), data=data_nba, subset=(position_rec!="Mixed")))
```

```
##
## Call:
## lm(formula = salary ~ as.factor(position_rec), data = data_nba,
##     subset = (position_rec != "Mixed"))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -4011485 -2826365 -1708014  1102456 31105006
```

```
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     4016014     126850  31.659  < 2e-16 ***
## as.factor(position_rec)Forward  -439249     158703  -2.768  0.00566 **
## as.factor(position_rec)Guard    -438470     149803  -2.927  0.00343 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4402000 on 6382 degrees of freedom
## Multiple R-squared:  0.001519,   Adjusted R-squared:  0.001206
## F-statistic: 4.855 on 2 and 6382 DF,  p-value: 0.007821
```

```
# adjust for confounder
summary(lm(salary ~ as.factor(position_rec) + as.factor(season_start), data=data_nba, s
```

```
##
## Call:
## lm(formula = salary ~ as.factor(position_rec) + as.factor(season_start),
##     data = data_nba, subset = (position_rec != "Mixed"))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -5983717 -2733737 -1557867  1229567 29209309
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     2614310     272704   9.587  < 2e-16 ***
## as.factor(position_rec)Forward  -510295     156308  -3.265 0.001102 **
## as.factor(position_rec)Guard    -523632     147538  -3.549 0.000389 ***
## as.factor(season_start)1999       22191     334992   0.066 0.947186
## as.factor(season_start)2000      705147     348782   2.022 0.043245 *
## as.factor(season_start)2001      818596     352465   2.322 0.020238 *
## as.factor(season_start)2002     1035391     354870   2.918 0.003539 **
## as.factor(season_start)2003     1069793     359704   2.974 0.002950 **
## as.factor(season_start)2004     1117189     358397   3.117 0.001834 **
## as.factor(season_start)2005     1090880     354570   3.077 0.002102 **
## as.factor(season_start)2006      958637     352167   2.722 0.006504 **
## as.factor(season_start)2007     1654377     357706   4.625 3.82e-06 ***
## as.factor(season_start)2008     1944946     357392   5.442 5.46e-08 ***
## as.factor(season_start)2009     1832838     359404   5.100 3.50e-07 ***
## as.factor(season_start)2010     1748651     360028   4.857 1.22e-06 ***
## as.factor(season_start)2011     1582423     353355   4.478 7.66e-06 ***
## as.factor(season_start)2012     1651497     346695   4.764 1.94e-06 ***
## as.factor(season_start)2013     2121273     361746   5.864 4.75e-09 ***
```

```
## as.factor(season_start)2014     1243733     331916   3.747 0.000180 ***
## as.factor(season_start)2015     1804493     332700   5.424 6.05e-08 ***
## as.factor(season_start)2016     2645083     326043   8.113 5.89e-16 ***
## as.factor(season_start)2017     3382563     324615  10.420  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4327000 on 6363 degrees of freedom
## Multiple R-squared:  0.03774,    Adjusted R-squared:  0.03456
## F-statistic: 11.88 on 21 and 6363 DF,  p-value: < 2.2e-16
```

Looking at the regression ouput, we see that, indeed, centers make more than guards and forwards. Actually, a lot less, approx. 500.000 USD less on average.

Let's ignore for the moment that this is a terrible model. The r-squared is only 0.03 which means that position and season only explain ~ 3% of variation in salaries.

Now, we want to know whether scoring explain this salary penalty. Here is a simple mediation.

```
# let'S compare models side by side in a regression table

#install.packages("modelsummary")
library(modelsummary)
```

```
## Warning: package 'modelsummary' was built under R version 4.2.3
```

```
models <- list(
  m1 = lm(salary ~ as.factor(position_rec) + as.factor(season_start), data=data_nba, subset=(posi
  m2 <- lm(salary ~ as.factor(position_rec) + career_PTS + as.factor(season_start), data=data_nba
)

modelsummary(models)
```

At first look, our hypothesis actually does not seem to hold. When we adjust for points scored, the gap between centers, forwards and guards. This means that guards don't make less because they make less points. Guards and forwards with average points, compared to centers with average points, make even less than centers, compared to when we do not account for points at all. This could mean that teams know that guards are important despite the fact that they score less.

Let's use a new package for "causal mediation analysis".

	m1	
(Intercept)	2 614 309.605	−939 410.632
	(272 704.206)	(221 569.259)
as.factor(position_rec)Forward	−510 295.017	−1 413 516.300
	(156 308.018)	(123 630.475)
as.factor(position_rec)Guard	−523 632.304	−2 000 142.285
	(147 537.586)	(118 260.027)
as.factor(season_start)1999	22 190.937	302 958.632
	(334 991.928)	(263 200.021)
as.factor(season_start)2000	705 146.970	685 025.470
	(348 782.404)	(273 995.766)
as.factor(season_start)2001	818 595.739	860 051.794
	(352 464.794)	(276 889.168)
as.factor(season_start)2002	1 035 390.857	1 185 765.987
	(354 869.752)	(278 787.933)
as.factor(season_start)2003	1 069 793.041	1 154 734.396
	(359 704.108)	(282 578.656)
as.factor(season_start)2004	1 117 189.300	1 335 647.564
	(358 397.141)	(281 570.162)
as.factor(season_start)2005	1 090 880.186	1 408 951.168
	(354 570.248)	(278 588.367)
as.factor(season_start)2006	958 636.479	1 404 509.355
	(352 166.838)	(276 745.293)
as.factor(season_start)2007	1 654 377.014	1 740 042.609
	(357 705.703)	(281 008.828)
as.factor(season_start)2008	1 944 945.963	1 844 901.044
	(357 392.398)	(280 763.910)
as.factor(season_start)2009	1 832 838.212	1 469 738.620
	(359 403.707)	(282 398.558)
as.factor(season_start)2010	1 748 650.500	1 381 348.241
	(360 027.993)	(282 890.255)
as.factor(season_start)2011	1 582 423.228	1 261 385.859
	(353 354.876)	(277 634.621)
as.factor(season_start)2012	1 651 497.174	1 339 040.290
	(346 694.638)	(272 400.865)
as.factor(season_start)2013	2 121 273.316	1 360 644.125
	(361 745.546)	(284 436.812)
as.factor(season_start)2014	1 243 732.606	1 141 626.786
	(331 915.955)	(260 750.740)
as.factor(season_start)2015	1 804 493.065	1 543 464.545
	(332 699.559)	(261 394.268)
as.factor(season_start)2016	2 645 083.101	2 527 969.077
	(326 042.835)	(256 138.668)
as.factor(season_start)2017	3 382 563.347	3 172 576.179
	(324 614.892)	(255 032.022)
career_PTS		545 303.023
		(8677.934)
Num.Obs.	6385	6385
R2	0.038	0.406
R2 Adj.	0.035	0.404
AIC	213 275.6	210 194.7
BIC	213 431.1	210 357.0
Log.Lik.	−106 614.789	−105 073.355
F	11.882	197.861
RMSE	4 319 963.19	3 393 398.83

```r
# now let's try a dedidcated package for mediation
library(mediation)

data_nba <- data_nba %>% mutate(position_rec = as.factor(position_rec),
                                season_start = as.factor(season_start),
                                salary = salary/10)

mean(data_nba$salary)
```

```
## [1] 407263.3
```

```r
# first path from x to mediator
path_a <- lm(career_PTS ~ position_rec + season_start, data=data_nba, subset=(position_rec!="Mixe

# Now full model, x to mediator to y
path_b <- lm(salary ~ position_rec + career_PTS + season_start, data=data_nba, subset=(position_r

# centers vs. forwards
results_mediation_forwards <- mediate(path_a, path_b,
                              treat = "position_rec", # x or main independent variable
                              mediator = "career_PTS",
                              treat.value = 1)

# centers vs. guards
results_mediation_guards <- mediate(path_a, path_b,
                              treat = "position_rec", # x or main independent variable
                              mediator = "career_PTS",
                              treat.value = 2)

summary(results_mediation_forwards)
```

```
##
## Causal Mediation Analysis
##
## Quasi-Bayesian Confidence Intervals
##
##                 Estimate 95% CI Lower 95% CI Upper p-value
## ACME             9.03e+04     7.11e+04     109770.6  <2e-16 ***
## ADE             -1.42e+05    -1.65e+05    -117872.5  <2e-16 ***
## Total Effect    -5.13e+04    -8.29e+04     -21008.8  <2e-16 ***
## Prop. Mediated -1.82e+00    -4.92e+00         -0.9  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Sample Size Used: 6385
##
##
## Simulations: 1000
```

```
summary(results_mediation_guards)
```

```
##
## Causal Mediation Analysis
##
## Quasi-Bayesian Confidence Intervals
##
##                 Estimate 95% CI Lower 95% CI Upper p-value
## ACME            9.06e+04     7.07e+04     1.09e+05  <2e-16 ***
## ADE            -1.42e+05    -1.65e+05    -1.17e+05  <2e-16 ***
## Total Effect   -5.11e+04    -8.12e+04    -2.22e+04  <2e-16 ***
## Prop. Mediated -1.79e+00    -4.36e+00    -9.10e-01  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Sample Size Used: 6385
##
##
## Simulations: 1000
```

```
# note: this dies not yet work. FOrwads and guards are the same. CHeck how to
# take different groups of the treatment variable. Problem is that the treatment
# variable is categorical.
```

The output looks different. Let's go through it step by step:

- ACME: Average Causal Mediation Effect - "indirect effect", this is the effect of position through points on salary.
- ADE: Average Direct Effect - This is the direct effect of position on salary. + Total Effect: ADE and ACME together are the "total effect"
- Prop. Mediated. This is the proportion of the total effect that is "explained" by the indirect effect.

Let's interpret this for our model:

- ACME: If only points could explain salaries, Guards would make, on average 900.000 USD more than centers (because they actually score more points)

- ADE: Adjusting for points, guards would make 1.4 Million USD less than centers
- Total Effect: Overall, guards actually make 500.000 USD less.
- Prop. Mediated. This is actually a negative value which makes no sense in this case. If points scored really "mediated" part of the total effect and had some explanatory power here, then we would see a positive percentage.

When you take a look at our first approach, where we just estimated both regression models side by side. The coefficient for position is the total effect when we don't include the mediator in the model. The coefficient is the direct effect when we do include the mediator. The indirect effect is the total effect minus the direct effect (-1.400.000 - (-500.000) = -900.000).

In sum, Guards are disadvantaged. They receive less money than center, although, if the points were considered, they should make even more. There must be other reasons why guards earn less. We need to go back to theory and come up with new hypothesis.

10.1 Find out more

- provide resources here
- references etc.

Chapter 11

Prediction - Theory

In prior weeks, you learned how to build a linear regression model. The main interest we pursued so far was to arrive at a good estimate of one independent variable (points scored in NBA basketball league) on an outcome (salary of NBA players).

With the help of DAGs, we identified relevant "counfounders" we should adjust for to "isolate' the effect of points as much as possible and reduce bias. COnfounders enter as covariates in in the model.

With the help of mediation analysis, we were then interested what "explains" or "mediates" the effect of x on y. We use additional variables which we assume operate as mechanisms of the causal effect of x on y and we test how much of the effect of x and y can be attributed to this mediator. In our example, we found that points scored do not really explain why guards earn less money in the NBA compared to centers.

All what we have done so far can be considered part of causal inference, i.e. understanding why an outcome varies. A different perspective is the perspective of PREDICTION.Prediction is at the heart of appraoches in "data science" and "machine learning".

In this scenario, we build regression models (or other models) simply to predict and outcome. The main interest is not to learn more about how the outcome can be explained but to predict something with it which we want to know. Machine learning then takes it a step further and simply iteratively select the best models among hundrets of options to arrive at the best possible prediction (more on that at the end of the class). First, we will learn how to predict values based on a linear regression model.

11.1 How prediction works

For a linear regression model, prediction is very straight forward. Linear regression is all about finding a straight line through a cloud of points that lie on as many dimensions as there are variables in model. The model provides you with an intercept (i.e. where the line touches the Y-Axis, and a slope; the increase in y given one unit increase in x. The unit is whatever the scale of the x variable is). The formula is y= b + ßx + €. Any prediction is the on the line. You know the intercept, you plug in a value for x and you get your predicted y.

-> visual here.

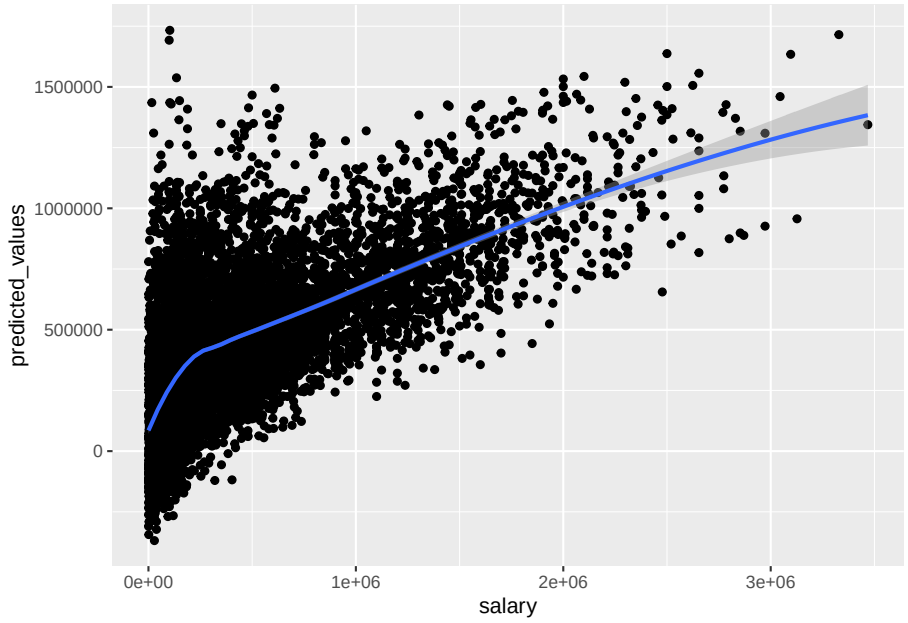
Let's apply this logic to our NBA data. Remember in prior weeks, we built a linear model to estimate the effect of points scored on average per game and salaries of players. Let's assume we now want to predict salaries of players and don't care too much about the scores. We can consider a range of variables which we think explain variation in salaries. The better we can capture variation in salaries between players, the more precise our prediction will be.

Now, you may rightfully ask: "Why do we want to predict salaries if we already know the actual salaries!?" Fair point. Prediction is commonly used to predict values which we don't have. Imagine there are some players that don't report their salaries. We could predict their salaries based on what we know from players who are similar to them in many other observable characteristics. Or imagine we want to predict the salary of a hypothetical player that does not exist. Imagine an average players would like to know how much he could earn more if player more like other players. We can predict that. Last example, imagine we want to forecast how much a player makes next year, depending on his past performance.

Machine learning is basically predicting outcomes that are no known based on very large datasets. You provide R with a million photos of animals, you build a model to explain which animal is a cat. The you apply the model to new data and the model predicts whether there is a cat in the photo. Of course, machine learning gets much more complicated quickly, however, the basic logic is the logic of prediction.

```
model1 <- lm(salary ~ career_PTS + position_rec + season_start +
             age, data = data_nba)

# base R way to get predicted values
data_nba$predicted_values <- model1$fitted.values
data_nba <- data_nba %>% dplyr::select('_id', name, salary, predicted_values, everythi

data_nba %>%
  ggplot(aes(x=salary, y=predicted_values)) +
  geom_point() +
  geom_smooth(method = "loess")
```

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




```
# new tidyverse way to get predicted values
library(broom)

# broom package has nice features to work with models

# tidy() converts the model output into a dataframe, makes it easy to process further, e.g. make
tidy(model1)
```

```
## # A tibble: 25 x 5
##    term                estimate std.error statistic  p.value
##    <chr>                  <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)         -568370.    29308.    -19.4  3.15e-82
##  2 career_PTS            53791.      731.     73.6  0
##  3 position_recForward -140645.    12779.    -11.0  5.24e-28
##  4 position_recGuard   -208234.    12166.    -17.1  9.89e-65
##  5 position_recMixed   -103661.    12104.     -8.56 1.26e-17
##  6 season_start1999      29357.    22884.      1.28 2.00e- 1
##  7 season_start2000      77209.    23559.      3.28 1.05e- 3
##  8 season_start2001     106436.    23625.      4.51 6.71e- 6
##  9 season_start2002     130348.    23627.      5.52 3.54e- 8
```

```
## 10 season_start2003     135221.    23648.      5.72 1.11e- 8
## # i 15 more rows
```

```
# glance provides meta-level info like r-squared etc.
glance(model1)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared   sigma statistic p.value    df   logLik    AIC    BIC
##       <dbl>         <dbl>   <dbl>     <dbl>   <dbl> <dbl>    <dbl>  <dbl>  <dbl>
## 1     0.432         0.431 352361.      308.       0    24 -138041. 2.76e5 2.76e5
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
# augment creates a dataframe with the predicted values for everyobservation in the da

nba_salalary_predicted <- augment(model1)
```

The output above, get you the predicted values for the observations in the dataset. This is mostly used to evaluate the model itself. The bigger the difference between the predicted values and the actual values (so-called residuals), the worse the model.

Now, let's predict the salary of hypothetical players:

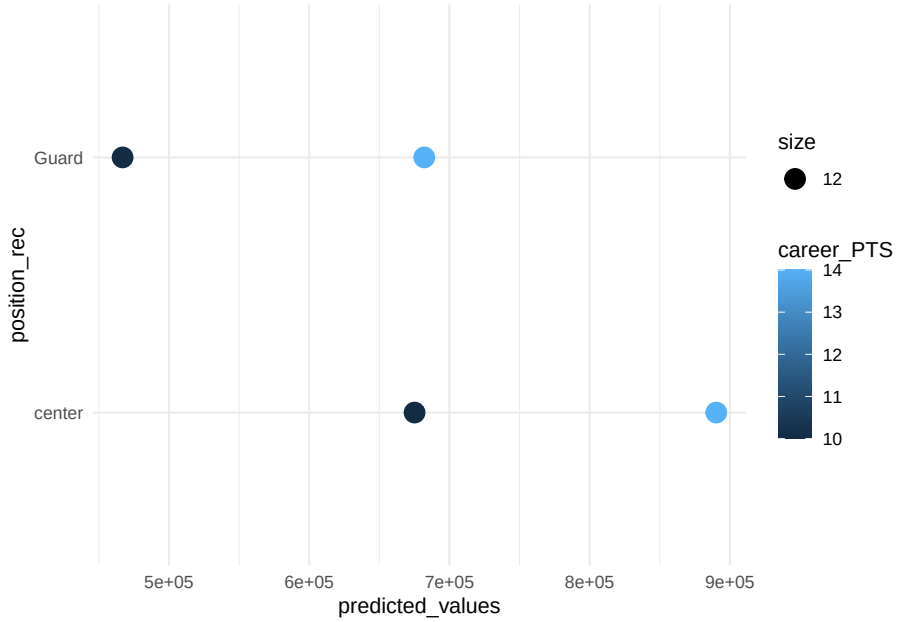
```
table(data_nba$position_rec)
```

```
##
##  center Forward   Guard   Mixed
##    1204    2130    3051    3343
```

```
# create all combination of the control variables which you want to predict
prediction.data <- tibble(
  position_rec = c("center", "center", "Guard", "Guard"),
  career_PTS = c(10, 14, 10, 14),
  age = c(20,20,20,20),
  season_start = c("2017", "2017", "2017", "2017")
)

# apply the model to the "new" dataset
predict(model1,
        prediction.data)
```

```
##        1        2        3        4
## 675129.3 890294.2 466895.1 682059.9
```

```
prediction.data$predicted_values <- predict(model1,
        prediction.data)

prediction.data %>% ggplot() +
  geom_point(aes(position_rec, predicted_values,
                 color=career_PTS,
                 size=12)) +
  coord_flip() +
  theme_minimal()
```



We can see that there is a huge salary increase predicted for just making two more baskets (i.e. 4 points) on average each game, for both guards and centers. We also see that centers make more money generally.

There is another way to get predicted values using the `margins()` function.

```
library(margins)
# This gets you the average change in predicted value for a unit-increase in the all model variab
margins(model1)
```

```
##  career_PTS   age position_recForward position_recGuard position_recMixed
##       53791 16809             -140645           -208234           -103661
##  season_start1999 season_start2000 season_start2001 season_start2002
##             29357            77209           106436           130348
```

```
##  season_start2003 season_start2004 season_start2005 season_start2006
##            135221           150576           174625           183653
##  season_start2007 season_start2008 season_start2009 season_start2010
##            206434           231505           204116           192220
##  season_start2011 season_start2012 season_start2013 season_start2014
##            183026           182509           199072           174359
##  season_start2015 season_start2016 season_start2017
##            206791           305228           369398
```

```
summary(margins(model1, at= list(position_rec= c("Guard", "center"),
                         career_PTS= c(10, 14))))
```

```
##              factor position_rec career_PTS         AME         SE       z
##                 age       1.0000    10.0000  16809.4290   822.1461 20.4458
##                 age       1.0000    14.0000  16809.4290   821.2659 20.4677
##                 age       2.0000    10.0000  16809.4290   821.3697 20.4651
##                 age       2.0000    14.0000  16809.4290   810.1705 20.7480
##          career_PTS       1.0000    10.0000  53791.2088   673.5739 79.8594
##          career_PTS       1.0000    14.0000  53791.2088   777.6596 69.1706
##          career_PTS       2.0000    10.0000  53791.2088   672.1461 80.0290
##          career_PTS       2.0000    14.0000  53791.2088   869.2152 61.8848
##  position_reccenter       1.0000    10.0000 208234.2670 12165.6828 17.1165
##  position_reccenter       1.0000    14.0000 208234.2670 12165.6828 17.1165
##  position_reccenter       2.0000    10.0000 208234.2670 12165.6828 17.1165
##  position_reccenter       2.0000    14.0000 208234.2670 12165.6828 17.1165
##    season_start1999       1.0000    10.0000  29356.8409 22884.2755  1.2828
##    season_start1999       1.0000    14.0000  29356.8409 22884.2755  1.2828
##    season_start1999       2.0000    10.0000  29356.8409 22884.2755  1.2828
##    season_start1999       2.0000    14.0000  29356.8409 22884.3468  1.2828
##    season_start2000       1.0000    10.0000  77209.1717 23559.4567  3.2772
##    season_start2000       1.0000    14.0000  77209.1717 23559.4567  3.2772
##    season_start2000       2.0000    10.0000  77209.1717 23559.4567  3.2772
##    season_start2000       2.0000    14.0000  77209.1717 23559.4567  3.2772
##    season_start2001       1.0000    10.0000 106435.5787 23624.8634  4.5052
##    season_start2001       1.0000    14.0000 106435.5787 23624.8634  4.5052
##    season_start2001       2.0000    10.0000 106435.5787 23624.8634  4.5052
##    season_start2001       2.0000    14.0000 106435.5787 23624.8621  4.5052
##    season_start2002       1.0000    10.0000 130348.2150 23627.2203  5.5169
##    season_start2002       1.0000    14.0000 130348.2150 23627.2203  5.5169
##    season_start2002       2.0000    10.0000 130348.2150 23627.2203  5.5169
##    season_start2002       2.0000    14.0000 130348.2150 23627.2203  5.5169
##    season_start2003       1.0000    10.0000 135220.5712 23648.3086  5.7180
##    season_start2003       1.0000    14.0000 135220.5712 23648.3086  5.7180
##    season_start2003       2.0000    10.0000 135220.5712 23648.3086  5.7180
##    season_start2003       2.0000    14.0000 135220.5712 23648.3047  5.7180
```

```
##    season_start2004       1.0000    10.0000 150575.6148 23423.7814  6.4283
##    season_start2004       1.0000    14.0000 150575.6148 23423.7814  6.4283
##    season_start2004       2.0000    10.0000 150575.6148 23423.7814  6.4283
##    season_start2004       2.0000    14.0000 150575.6148 23416.9662  6.4302
##    season_start2005       1.0000    10.0000 174625.3789 23247.7154  7.5115
##    season_start2005       1.0000    14.0000 174625.3789 23247.7154  7.5115
##    season_start2005       2.0000    10.0000 174625.3789 23247.7154  7.5115
##    season_start2005       2.0000    14.0000 174625.3789 23240.9439  7.5137
##    season_start2006       1.0000    10.0000 183653.2187 23057.8278  7.9649
##    season_start2006       1.0000    14.0000 183653.2187 23071.2667  7.9603
##    season_start2006       2.0000    10.0000 183653.2187 23071.2569  7.9603
##    season_start2006       2.0000    14.0000 183653.2187 23071.2471  7.9603
##    season_start2007       1.0000    10.0000 206434.1344 23336.8298  8.8459
##    season_start2007       1.0000    14.0000 206434.1344 23336.8298  8.8459
##    season_start2007       2.0000    10.0000 206434.1344 23336.8298  8.8459
##    season_start2007       2.0000    14.0000 206434.1344 23330.0326  8.8484
##    season_start2008       1.0000    10.0000 231505.0635 23507.2919  9.8482
##    season_start2008       1.0000    14.0000 231505.0635 23507.2919  9.8482
##    season_start2008       2.0000    10.0000 231505.0635 23507.2919  9.8482
##    season_start2008       2.0000    14.0000 231505.0635 23500.4440  9.8511
##    season_start2009       1.0000    10.0000 204115.6638 23475.4934  8.6948
##    season_start2009       1.0000    14.0000 204115.6638 23489.1731  8.6898
##    season_start2009       2.0000    10.0000 204115.6638 23489.1731  8.6898
##    season_start2009       2.0000    14.0000 204115.6638 23489.1586  8.6898
##    season_start2010       1.0000    10.0000 192219.8545 23548.5054  8.1627
##    season_start2010       1.0000    14.0000 192219.8545 23548.5054  8.1627
##    season_start2010       2.0000    10.0000 192219.8545 23548.5054  8.1627
##    season_start2010       2.0000    14.0000 192219.8545 23555.3511  8.1603
##    season_start2011       1.0000    10.0000 183025.8381 23534.7356  7.7768
##    season_start2011       1.0000    14.0000 183025.8381 23534.7356  7.7768
##    season_start2011       2.0000    10.0000 183025.8381 23534.7356  7.7768
##    season_start2011       2.0000    14.0000 183025.8381 23534.7356  7.7768
##    season_start2012       1.0000    10.0000 182509.4584 23219.2518  7.8603
##    season_start2012       1.0000    14.0000 182509.4584 23219.2518  7.8603
##    season_start2012       2.0000    10.0000 182509.4584 23219.2518  7.8603
##    season_start2012       2.0000    14.0000 182509.4584 23212.4961  7.8626
##    season_start2013       1.0000    10.0000 199072.4713 24314.5882  8.1874
##    season_start2013       1.0000    14.0000 199072.4713 24314.5882  8.1874
##    season_start2013       2.0000    10.0000 199072.4713 24314.5882  8.1874
##    season_start2013       2.0000    14.0000 199072.4713 24321.6667  8.1850
##    season_start2014       1.0000    10.0000 174358.9208 22746.2703  7.6654
##    season_start2014       1.0000    14.0000 174358.9208 22733.0380  7.6698
##    season_start2014       2.0000    10.0000 174358.9208 22733.0380  7.6698
##    season_start2014       2.0000    14.0000 174358.9208 22733.0380  7.6698
##    season_start2015       1.0000    10.0000 206791.4964 22900.6232  9.0300
##    season_start2015       1.0000    14.0000 206791.4964 22887.2931  9.0352
```

```
##    season_start2015       2.0000    10.0000 206791.4964 22887.2931  9.0352
##    season_start2015       2.0000    14.0000 206791.4964 22887.3012  9.0352
##    season_start2016       1.0000    10.0000 305227.6577 22617.9191 13.4949
##    season_start2016       1.0000    14.0000 305227.6577 22617.9191 13.4949
##    season_start2016       2.0000    10.0000 305227.6577 22617.9191 13.4949
##    season_start2016       2.0000    14.0000 305227.6577 22617.9191 13.4949
##    season_start2017       1.0000    10.0000 369398.3511 22665.9479 16.2975
##    season_start2017       1.0000    14.0000 369398.3511 22665.9479 16.2975
##    season_start2017       2.0000    10.0000 369398.3511 22665.9479 16.2975
##    season_start2017       2.0000    14.0000 369398.3511 22655.0335 16.3054
##       p       lower       upper
##  0.0000  15198.0523  18420.8057
##  0.0000  15199.7774  18419.0806
##  0.0000  15199.5740  18419.2841
##  0.0000  15221.5240  18397.3341
##  0.0000  52471.0281  55111.3895
##  0.0000  52267.0240  55315.3936
##  0.0000  52473.8267  55108.5909
##  0.0000  52087.5784  55494.8392
##  0.0000 184389.9668 232078.5671
##  0.0000 184389.9668 232078.5671
##  0.0000 184389.9668 232078.5671
##  0.0000 184389.9668 232078.5671
##  0.1995 -15495.5149  74209.1967
##  0.1995 -15495.5149  74209.1967
##  0.1995 -15495.5149  74209.1967
##  0.1995 -15495.6546  74209.3364
##  0.0010  31033.4850 123384.8584
##  0.0010  31033.4850 123384.8584
##  0.0010  31033.4850 123384.8584
##  0.0010  31033.4850 123384.8584
##  0.0000  60131.6973 152739.4600
##  0.0000  60131.6973 152739.4600
##  0.0000  60131.6973 152739.4600
##  0.0000  60131.6998 152739.4575
##  0.0000  84039.7141 176656.7158
##  0.0000  84039.7141 176656.7158
##  0.0000  84039.7141 176656.7158
##  0.0000  84039.7141 176656.7158
##  0.0000  88870.7382 181570.4043
##  0.0000  88870.7382 181570.4043
##  0.0000  88870.7382 181570.4043
##  0.0000  88870.7457 181570.3968
##  0.0000 104665.8469 196485.3827
##  0.0000 104665.8469 196485.3827
##  0.0000 104665.8469 196485.3827
```

```
##  0.0000 104679.2044 196472.0252
##  0.0000 129060.6941 220190.0637
##  0.0000 129060.6941 220190.0637
##  0.0000 129060.6941 220190.0637
##  0.0000 129073.9659 220176.7919
##  0.0000 138460.7067 228845.7307
##  0.0000 138434.3670 228872.0704
##  0.0000 138434.3861 228872.0513
##  0.0000 138434.4052 228872.0322
##  0.0000 160694.7885 252173.4804
##  0.0000 160694.7885 252173.4804
##  0.0000 160694.7885 252173.4804
##  0.0000 160708.1107 252160.1581
##  0.0000 185431.6180 277578.5090
##  0.0000 185431.6180 277578.5090
##  0.0000 185431.6180 277578.5090
##  0.0000 185445.0396 277565.0874
##  0.0000 158104.5422 250126.7855
##  0.0000 158077.7306 250153.5971
##  0.0000 158077.7306 250153.5971
##  0.0000 158077.7590 250153.5687
##  0.0000 146065.6320 238374.0769
##  0.0000 146065.6320 238374.0769
##  0.0000 146065.6320 238374.0769
##  0.0000 146052.2146 238387.4943
##  0.0000 136898.6039 229153.0723
##  0.0000 136898.6039 229153.0723
##  0.0000 136898.6039 229153.0723
##  0.0000 136898.6039 229153.0723
##  0.0000 137000.5611 228018.3557
##  0.0000 137000.5611 228018.3557
##  0.0000 137000.5611 228018.3557
##  0.0000 137013.8020 228005.1148
##  0.0000 151416.7540 246728.1885
##  0.0000 151416.7540 246728.1885
##  0.0000 151416.7540 246728.1885
##  0.0000 151402.8805 246742.0621
##  0.0000 129777.0502 218940.7913
##  0.0000 129802.9850 218914.8566
##  0.0000 129802.9850 218914.8566
##  0.0000 129802.9850 218914.8566
##  0.0000 161907.0996 251675.8931
##  0.0000 161933.2262 251649.7666
##  0.0000 161933.2262 251649.7666
##  0.0000 161933.2103 251649.7824
##  0.0000 260897.3510 349557.9645
```

```
##  0.0000 260897.3510 349557.9645
##  0.0000 260897.3510 349557.9645
##  0.0000 260897.3510 349557.9645
##  0.0000 324973.9095 413822.7927
##  0.0000 324973.9095 413822.7927
##  0.0000 324973.9095 413822.7927
##  0.0000 324995.3013 413801.4009
```

The margins function is handy to calculate predictions for different groups. It automatically can hold other control variables at their mean or at their observed value.

Now, let's also calculate confidence and prediction intervals. For background, watch these short videos to understand what they are [hyperlink] [hyperlink]. Confidence intervals basically tell how the following: "If we repeated our study on a different sample of people with the same sample size, then the estimate which we have (for example, a mean) would we within the confidence interval 95% of the time. This means in 5% of cases, our study would arrive at a lower or higher mean. The formula for confidence is not very intuitive:

-> formulate here: Margin of error = z * (standard deviation/ square-root of sample size)

Let's not worry about why this formula works, but let's focus on its ingredients: Sample size (N) is the number of people in our data; Standard Deviation is a measure for how much individual people deviate from the mean on average, in other words, how much the data spreads around the mean. and z is 1.96 and is derived from probability theory (i.e. in a normal distribution, there is a certain known likelihood that means fall within a range when re-sampling populations). In other words, the confidence intervals tells us how "confident" we can be that our estimate is within the range 95% of times.

Prediction intervals are very similar but only apply to predictions for specfific values. It gives us a measure for "confident" we can be that our prediction would be within the prediction interval (95% of times).

The 95% is an arbitrarily set value which is a standard in research. However, we can also set it at 99% or 90%.

Let's apply this to our data.

```
# Just for illustration, let's take a suimple model
model2 <- lm(salary ~ career_PTS, data = data_nba)
predict(model2)
```

```
##           1           2           3           4           5           6
##  721959.298  346026.074  346026.074  346026.074  346026.074  346026.074
##           7           8           9          10          11          12
```

```
##  346026.074  346026.074  346026.074  346026.074  915454.340  915454.340
##          13          14          15          16          17          18
##  915454.340  915454.340  915454.340  915454.340  915454.340  915454.340
##          19          20          21          22          23          24
##  915454.340  915454.340  207815.330  207815.330   64076.156   64076.156
##          25          26          27          28          29          30
##   64076.156  185701.611  185701.611  185701.611  185701.611  185701.611
##          31          32          33          34          35          36
##  185701.611  185701.611   53019.296   53019.296   91718.305   91718.305
##          37          38          39          40          41          42
##   91718.305  451066.239  451066.239  451066.239  451066.239  451066.239
##          43          44          45          46          47          48
##  357082.933  169116.321  169116.321  169116.321  169116.321  169116.321
##          49          50          51          52          53          54
##  511878.967  511878.967  511878.967  511878.967  511878.967  511878.967
##          55          56          57          58          59          60
##  511878.967  511878.967  511878.967  511878.967  511878.967  511878.967
##          61          62          63          64          65          66
##   30905.577   30905.577   30905.577   30905.577  169116.321  207815.330
##          67          68          69          70          71          72
##  207815.330  207815.330  207815.330  207815.330  207815.330  207815.330
##          73          74          75          76          77          78
##    8791.858   -2265.002   -2265.002   41962.437   41962.437   86189.875
##          79          80          81          82          83          84
##   86189.875   86189.875   86189.875   86189.875   86189.875   86189.875
##          85          86          87          88          89          90
##  998380.787  998380.787  998380.787  998380.787  998380.787  998380.787
##          91          92          93          94          95          96
##  998380.787  998380.787  998380.787  998380.787  998380.787  998380.787
##          97          98          99         100         101         102
##  -13321.861  -13321.861  218872.189  218872.189  218872.189  218872.189
##         103         104         105         106         107         108
##  218872.189  412367.231  412367.231  412367.231  412367.231  412367.231
##         109         110         111         112         113         114
##  147002.602  147002.602  147002.602  406838.801  445537.810   75133.015
##         115         116         117         118         119         120
##  180173.181  180173.181  180173.181  180173.181  180173.181  180173.181
##         121         122         123         124         125         126
##  185701.611  185701.611  185701.611  185701.611  185701.611  185701.611
##         127         128         129         130         131         132
##  185701.611  185701.611  185701.611  185701.611  959681.778  959681.778
##         133         134         135         136         137         138
##  959681.778  959681.778  959681.778  959681.778  959681.778  959681.778
##         139         140         141         142         143         144
##  959681.778  959681.778  959681.778  959681.778  959681.778  959681.778
##         145         146         147         148         149         150
```

```
##  959681.778  959681.778  362611.363  362611.363  362611.363  362611.363
##         151         152         153         154         155         156
##  362611.363  362611.363  362611.363  362611.363  362611.363  362611.363
##         157         158         159         160         161         162
##  362611.363  362611.363  362611.363  362611.363   86189.875   86189.875
##         163         164         165         166         167         168
##  473179.959  473179.959  473179.959  473179.959  473179.959  473179.959
##         169         170         171         172         173         174
##  473179.959  473179.959  473179.959  473179.959  473179.959  473179.959
##         175         176         177         178         179         180
##  257571.198  257571.198  257571.198  257571.198  257571.198  257571.198
##         181         182         183         184         185         186
##  340497.644  340497.644  340497.644  340497.644  340497.644  340497.644
##         187         188         189         190         191         192
##  340497.644  340497.644  119360.453  119360.453  119360.453  119360.453
##         193         194         195         196         197         198
##  119360.453  119360.453  119360.453  119360.453  119360.453  119360.453
##         199         200         201         202         203         204
##  119360.453  119360.453  318383.925  318383.925  318383.925  318383.925
##         205         206         207         208         209         210
##  318383.925  318383.925  318383.925  213343.759  213343.759  213343.759
##         211         212         213         214         215         216
##  213343.759  213343.759  213343.759  213343.759  213343.759  213343.759
##         217         218         219         220         221         222
##  213343.759  213343.759  213343.759  213343.759  213343.759  213343.759
##         223         224         225         226         227         228
##  185701.611  185701.611  578220.124  578220.124  578220.124  578220.124
##         229         230         231         232         233         234
##  578220.124  578220.124  578220.124  578220.124  578220.124  578220.124
##         235         236         237         238         239         240
##  578220.124  578220.124  578220.124  235457.478  235457.478  235457.478
##         241         242         243         244         245         246
##  235457.478  235457.478  235457.478  207815.330  207815.330  207815.330
##         247         248         249         250         251         252
##  611390.703  611390.703  611390.703  611390.703  611390.703  611390.703
##         253         254         255         256         257         258
##  611390.703  207815.330  207815.330  207815.330  207815.330  710902.439
##         259         260         261         262         263         264
##  710902.439  710902.439  710902.439  710902.439  600333.843  600333.843
##         265         266         267         268         269         270
##  600333.843  600333.843  600333.843  600333.843  600333.843  600333.843
##         271         272         273         274         275         276
##  600333.843  600333.843  323912.355  323912.355  323912.355  323912.355
##         277         278         279         280         281         282
##  323912.355  323912.355  323912.355  323912.355  323912.355  323912.355
##         283         284         285         286         287         288
```

```
##  323912.355  589276.984  -85191.448  -85191.448  -35435.580  202286.900
##         289         290         291         292         293         294
##  202286.900  954153.348  954153.348  954153.348  954153.348  954153.348
##         295         296         297         298         299         300
##   80661.445 1241631.696 1241631.696 1241631.696 1241631.696 1241631.696
##         301         302         303         304         305         306
## 1241631.696 1241631.696 1241631.696 1241631.696 1241631.696 1241631.696
##         307         308         309         310         311         312
## 1241631.696 1241631.696 1241631.696 1241631.696  318383.925  318383.925
##         313         314         315         316         317         318
##  318383.925  318383.925  318383.925   36434.007   36434.007   36434.007
##         319         320         321         322         323         324
##   36434.007   36434.007   36434.007   36434.007   36434.007   36434.007
##         325         326         327         328         329         330
##   36434.007  263099.627  268628.057  -18850.291   69604.585   69604.585
##         331         332         333         334         335         336
##   69604.585  -18850.291  -18850.291 1059193.514 1059193.514 1059193.514
##         337         338         339         340         341         342
## 1059193.514 1059193.514 1059193.514 1059193.514 1059193.514 1059193.514
##         343         344         345         346         347         348
## 1059193.514 1059193.514 1059193.514  506350.537  506350.537  506350.537
##         349         350         351         352         353         354
##  506350.537  506350.537  506350.537  506350.537  506350.537  506350.537
##         355         356         357         358         359         360
##  506350.537  506350.537  506350.537  506350.537  506350.537  456594.669
##         361         362         363         364         365         366
##  456594.669  456594.669   36434.007   36434.007   36434.007  423424.091
##         367         368         369         370         371         372
##  423424.091  423424.091  423424.091  423424.091  423424.091  423424.091
##         373         374         375         376         377         378
##  423424.091  423424.091  423424.091   80661.445   80661.445   80661.445
##         379         380         381         382         383         384
##   80661.445   80661.445  279684.917  279684.917  279684.917  279684.917
##         385         386         387         388         389         390
##  279684.917  279684.917  279684.917  279684.917  279684.917  644561.281
##         391         392         393         394         395         396
##  644561.281  644561.281  644561.281  644561.281  644561.281  644561.281
##         397         398         399         400         401         402
##  644561.281  644561.281  644561.281  644561.281  644561.281  644561.281
##         403         404         405         406         407         408
##  644561.281  644561.281  644561.281  644561.281  274156.487  274156.487
##         409         410         411         412         413         414
##  274156.487  274156.487  274156.487  274156.487  274156.487  274156.487
##         415         416         417         418         419         420
##  274156.487  274156.487  207815.330  207815.330  207815.330  207815.330
##         421         422         423         424         425         426
```

```
##  207815.330  207815.330  207815.330  207815.330  124888.883  124888.883
##         427         428         429         430         431         432
##  462123.099  462123.099  462123.099  462123.099  462123.099  462123.099
##         433         434         435         436         437         438
##  462123.099  462123.099  462123.099  462123.099  462123.099  462123.099
##         439         440         441         442         443         444
##  357082.933  357082.933  357082.933  357082.933  357082.933  357082.933
##         445         446         447         448         449         450
##  357082.933  357082.933  357082.933  456594.669  456594.669  456594.669
##         451         452         453         454         455         456
##  456594.669  456594.669  456594.669  456594.669  456594.669  456594.669
##         457         458         459         460         461         462
##  456594.669    3263.428    3263.428  334969.214  334969.214  334969.214
##         463         464         465         466         467         468
##  334969.214  224400.619   64076.156   64076.156   64076.156  174644.751
##         469         470         471         472         473         474
##  174644.751  174644.751  495293.678  495293.678  495293.678  495293.678
##         475         476         477         478         479         480
##  495293.678  495293.678  495293.678    3263.428  180173.181  180173.181
##         481         482         483         484         485         486
##  180173.181  180173.181  180173.181  180173.181  180173.181  196758.470
##         487         488         489         490         491         492
##   58547.726   58547.726   58547.726   58547.726  622447.562  622447.562
##         493         494         495         496         497         498
##   97246.734   97246.734  -18850.291  -18850.291   86189.875   86189.875
##         499         500         501         502         503         504
##  744073.017  744073.017  744073.017  744073.017  744073.017  744073.017
##         505         506         507         508         509         510
##  744073.017  744073.017  744073.017  744073.017  744073.017  744073.017
##         511         512         513         514         515         516
##  135945.743  135945.743  135945.743  135945.743  135945.743  135945.743
##         517         518         519         520         521         522
##  135945.743  467651.529  240985.908  240985.908  240985.908  240985.908
##         523         524         525         526         527         528
##  240985.908  240985.908  240985.908  240985.908  500822.107  500822.107
##         529         530         531         532         533         534
##  500822.107  500822.107  500822.107  500822.107  500822.107  500822.107
##         535         536         537         538         539         540
##  500822.107  500822.107  500822.107  500822.107  500822.107  500822.107
##         541         542         543         544         545         546
##  412367.231  412367.231  412367.231  412367.231  412367.231  412367.231
##         547         548         549         550         551         552
##  412367.231  412367.231  412367.231  412367.231  412367.231  412367.231
##         553         554         555         556         557         558
##  705374.009  705374.009  705374.009  705374.009  705374.009  705374.009
##         559         560         561         562         563         564
```

```
##  705374.009  705374.009  705374.009  705374.009  705374.009 1136591.531
##         565         566         567         568         569         570
## 1136591.531   75133.015   75133.015   75133.015  666675.000  666675.000
##         571         572         573         574         575         576
##  666675.000  666675.000  666675.000  666675.000  368139.793  368139.793
##         577         578         579         580         581         582
##  368139.793  368139.793  368139.793  368139.793  368139.793  368139.793
##         583         584         585         586         587         588
##  368139.793  368139.793  368139.793  368139.793  368139.793  368139.793
##         589         590         591         592         593         594
##  368139.793  368139.793   97246.734   97246.734  495293.678  495293.678
##         595         596         597         598         599         600
##  495293.678  495293.678  169116.321  169116.321  169116.321  169116.321
##         601         602         603         604         605         606
##  169116.321  169116.321  169116.321  428952.520  428952.520  428952.520
##         607         608         609         610         611         612
##  428952.520  428952.520  428952.520  428952.520  428952.520  428952.520
##         613         614         615         616         617         618
##  428952.520  428952.520  428952.520   36434.007   36434.007  229929.049
##         619         620         621         622         623         624
##  229929.049  229929.049  229929.049  229929.049  229929.049  229929.049
##         625         626         627         628         629         630
##  229929.049  495293.678  495293.678  495293.678  495293.678  495293.678
##         631         632         633         634         635         636
##  495293.678   30905.577  395781.942  395781.942  395781.942  395781.942
##         637         638         639         640         641         642
##  395781.942  395781.942  395781.942  395781.942  395781.942  395781.942
##         643         644         645         646         647         648
##  395781.942  395781.942   64076.156   64076.156   64076.156   64076.156
##         649         650         651         652         653         654
##   64076.156  102775.164  102775.164    8791.858    8791.858  268628.057
##         655         656         657         658         659         660
##  390253.512  390253.512  390253.512  390253.512  390253.512  390253.512
##         661         662         663         664         665         666
##  390253.512  390253.512  390253.512  390253.512  390253.512  390253.512
##         667         668         669         670         671         672
##  390253.512  252042.768  252042.768  252042.768  252042.768  252042.768
##         673         674         675         676         677         678
##  252042.768  252042.768  252042.768  252042.768  252042.768  252042.768
##         679         680         681         682         683         684
##  252042.768  252042.768  252042.768  567163.265  567163.265  567163.265
##         685         686         687         688         689         690
##  567163.265  567163.265  567163.265  567163.265  567163.265  567163.265
##         691         692         693         694         695         696
##  567163.265  130417.313  130417.313  130417.313  130417.313  379196.652
##         697         698         699         700         701         702
```

```
##  379196.652  379196.652  379196.652  379196.652  379196.652  379196.652
##         703         704         705         706         707         708
##  379196.652  379196.652  379196.652  213343.759  213343.759  213343.759
##         709         710         711         712         713         714
##  213343.759  213343.759  213343.759  395781.942  395781.942  395781.942
##         715         716         717         718         719         720
##  395781.942  395781.942  395781.942 1009437.646 1009437.646 1009437.646
##         721         722         723         724         725         726
## 1009437.646 1009437.646 1009437.646  -29907.150  -29907.150  312855.495
##         727         728         729         730         731         732
##  312855.495  600333.843  600333.843  600333.843  600333.843  600333.843
##         733         734         735         736         737         738
##  600333.843  600333.843  600333.843  600333.843  600333.843  600333.843
##         739         740         741         742         743         744
##  600333.843  600333.843  307327.065  307327.065  307327.065  307327.065
##         745         746         747         748         749         750
##   64076.156  467651.529  467651.529  467651.529  467651.529  467651.529
##         751         752         753         754         755         756
##  467651.529  467651.529  467651.529  467651.529  467651.529  467651.529
##         757         758         759         760         761         762
##  467651.529  384725.082  384725.082  384725.082  384725.082  384725.082
##         763         764         765         766         767         768
##  384725.082  384725.082  384725.082  130417.313  462123.099  462123.099
##         769         770         771         772         773         774
##  462123.099  462123.099  462123.099  462123.099  462123.099  462123.099
##         775         776         777         778         779         780
##  462123.099  462123.099  462123.099  462123.099   14320.288   14320.288
##         781         782         783         784         785         786
##   14320.288  263099.627  263099.627  207815.330  207815.330  218872.189
##         787         788         789         790         791         792
##  218872.189  218872.189  218872.189  218872.189  218872.189  218872.189
##         793         794         795         796         797         798
##  218872.189  -85191.448  -85191.448  545049.546  218872.189  218872.189
##         799         800         801         802         803         804
##  218872.189  158059.462  158059.462  158059.462  158059.462  158059.462
##         805         806         807         808         809         810
##  301798.636  -85191.448  -85191.448  -85191.448  -85191.448  257571.198
##         811         812         813         814         815         816
##  257571.198  334969.214  334969.214  334969.214  334969.214  334969.214
##         817         818         819         820         821         822
##  334969.214  334969.214  412367.231  412367.231  412367.231  412367.231
##         823         824         825         826         827         828
##  412367.231  412367.231  -46492.440  727487.728  727487.728  727487.728
##         829         830         831         832         833         834
##  727487.728  727487.728  727487.728  727487.728  727487.728  727487.728
##         835         836         837         838         839         840
```

```
##  727487.728  727487.728  727487.728  727487.728  727487.728  263099.627
##         841         842         843         844         845         846
##  263099.627  263099.627  263099.627  263099.627  263099.627  263099.627
##         847         848         849         850         851         852
##  263099.627  263099.627  263099.627  755129.877  755129.877  755129.877
##         853         854         855         856         857         858
##  755129.877  755129.877  755129.877  755129.877  755129.877  755129.877
##         859         860         861         862         863         864
##  755129.877  755129.877  755129.877  755129.877  755129.877  755129.877
##         865         866         867         868         869         870
##  755129.877  755129.877  163587.892  163587.892  185701.611  185701.611
##         871         872         873         874         875         876
##  185701.611  185701.611  185701.611  185701.611  185701.611  307327.065
##         877         878         879         880         881         882
##  307327.065  307327.065  185701.611  185701.611  185701.611  185701.611
##         883         884         885         886         887         888
##  290741.776  290741.776  290741.776  290741.776  290741.776  290741.776
##         889         890         891         892         893         894
##  290741.776  274156.487  274156.487  274156.487  274156.487  274156.487
##         895         896         897         898         899         900
##  274156.487  274156.487  274156.487  274156.487  274156.487  274156.487
##         901         902         903         904         905         906
##  274156.487  274156.487   14320.288  473179.959  473179.959  473179.959
##         907         908         909         910         911         912
##  473179.959  473179.959  473179.959  473179.959  473179.959  473179.959
##         913         914         915         916         917         918
##  473179.959  473179.959  661146.571  661146.571  661146.571  661146.571
##         919         920         921         922         923         924
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##         925         926         927         928         929         930
##  694317.149  694317.149  113832.024  113832.024  113832.024  113832.024
##         931         932         933         934         935         936
##  113832.024  113832.024  113832.024  368139.793  368139.793  368139.793
##         937         938         939         940         941         942
##  368139.793  368139.793  368139.793  368139.793  368139.793  368139.793
##         943         944         945         946         947         948
##  368139.793   53019.296  263099.627  263099.627  263099.627  263099.627
##         949         950         951         952         953         954
##  263099.627  263099.627  263099.627  263099.627  263099.627  263099.627
##         955         956         957         958         959         960
##  263099.627  627975.992  650089.711  650089.711  650089.711  650089.711
##         961         962         963         964         965         966
##  340497.644  340497.644  340497.644  340497.644  340497.644  340497.644
##         967         968         969         970         971         972
##  445537.810  445537.810  445537.810  445537.810  445537.810  445537.810
##         973         974         975         976         977         978
```

```
##  445537.810  445537.810  445537.810  445537.810  445537.810  445537.810
##         979         980         981         982         983         984
##  445537.810  445537.810   -7793.431  235457.478  235457.478  235457.478
##         985         986         987         988         989         990
##  235457.478  235457.478  235457.478  235457.478  235457.478  235457.478
##         991         992         993         994         995         996
##  235457.478  235457.478  235457.478 1097892.522 1097892.522 1097892.522
##         997         998         999        1000        1001        1002
##  296270.206  296270.206  296270.206  296270.206  296270.206  296270.206
##        1003        1004        1005        1006        1007        1008
##  296270.206  296270.206  296270.206  202286.900  202286.900  202286.900
##        1009        1010        1011        1012        1013        1014
##  202286.900   97246.734   97246.734   97246.734   97246.734   97246.734
##        1015        1016        1017        1018        1019        1020
##   97246.734   97246.734   97246.734   97246.734   97246.734   97246.734
##        1021        1022        1023        1024        1025        1026
##   97246.734   75133.015   75133.015  810414.174  810414.174  810414.174
##        1027        1028        1029        1030        1031        1032
##  810414.174  810414.174  810414.174  810414.174  810414.174  810414.174
##        1033        1034        1035        1036        1037        1038
##  810414.174  810414.174  810414.174  810414.174  810414.174   86189.875
##        1039        1040        1041        1042        1043        1044
##   86189.875   86189.875   86189.875  113832.024  976267.068  976267.068
##        1045        1046        1047        1048        1049        1050
##  976267.068  976267.068  976267.068  976267.068  976267.068  976267.068
##        1051        1052        1053        1054        1055        1056
##  976267.068  976267.068  976267.068  976267.068  976267.068  976267.068
##        1057        1058        1059        1060        1061        1062
##  976267.068  -29907.150  -29907.150  -29907.150   80661.445   80661.445
##        1063        1064        1065        1066        1067        1068
##   80661.445  252042.768  252042.768  252042.768  252042.768  252042.768
##        1069        1070        1071        1072        1073        1074
##  252042.768  252042.768  252042.768  252042.768  252042.768  252042.768
##        1075        1076        1077        1078        1079        1080
##  252042.768   58547.726   58547.726   58547.726   58547.726   58547.726
##        1081        1082        1083        1084        1085        1086
##   58547.726   58547.726   58547.726   58547.726   58547.726   58547.726
##        1087        1088        1089        1090        1091        1092
##  -35435.580  -35435.580  406838.801  406838.801  406838.801  406838.801
##        1093        1094        1095        1096        1097        1098
##  406838.801  406838.801  406838.801  406838.801  406838.801  406838.801
##        1099        1100        1101        1102        1103        1104
##  406838.801  406838.801  406838.801  -24378.721   14320.288   14320.288
##        1105        1106        1107        1108        1109        1110
##  578220.124  578220.124  578220.124  578220.124  578220.124  578220.124
##        1111        1112        1113        1114        1115        1116
```

```
##  578220.124  578220.124   69604.585   69604.585   69604.585   69604.585
##        1117        1118        1119        1120        1121        1122
##   69604.585  362611.363  362611.363  362611.363  362611.363  362611.363
##        1123        1124        1125        1126        1127        1128
##  362611.363  362611.363  362611.363  362611.363   30905.577   30905.577
##        1129        1130        1131        1132        1133        1134
##   30905.577  -29907.150  793828.885  793828.885  793828.885  793828.885
##        1135        1136        1137        1138        1139        1140
##  793828.885  793828.885  793828.885  793828.885  793828.885  793828.885
##        1141        1142        1143        1144        1145        1146
##  793828.885  793828.885  793828.885  793828.885  793828.885  793828.885
##        1147        1148        1149        1150        1151        1152
##  793828.885  793828.885  793828.885  677731.860  677731.860  677731.860
##        1153        1154        1155        1156        1157        1158
##  677731.860  677731.860  677731.860  274156.487  274156.487  395781.942
##        1159        1160        1161        1162        1163        1164
##  395781.942  395781.942  395781.942  395781.942  395781.942  395781.942
##        1165        1166        1167        1168        1169        1170
##  395781.942  395781.942  395781.942  395781.942  395781.942    3263.428
##        1171        1172        1173        1174        1175        1176
##    3263.428    3263.428    3263.428  346026.074  346026.074  346026.074
##        1177        1178        1179        1180        1181        1182
##  346026.074  346026.074  346026.074  346026.074  346026.074  312855.495
##        1183        1184        1185        1186        1187        1188
##  312855.495  312855.495  312855.495  312855.495  312855.495  312855.495
##        1189        1190        1191        1192        1193        1194
##  312855.495   30905.577   30905.577   30905.577   30905.577  622447.562
##        1195        1196        1197        1198        1199        1200
##  622447.562  451066.239  451066.239  451066.239  451066.239  451066.239
##        1201        1202        1203        1204        1205        1206
##  451066.239  451066.239  451066.239  451066.239  451066.239  451066.239
##        1207        1208        1209        1210        1211        1212
##  484236.818  357082.933  357082.933  357082.933  357082.933   58547.726
##        1213        1214        1215        1216        1217        1218
##   58547.726   58547.726  158059.462  158059.462  158059.462  158059.462
##        1219        1220        1221        1222        1223        1224
##  240985.908  240985.908   69604.585   69604.585  528464.256  528464.256
##        1225        1226        1227        1228        1229        1230
##  528464.256  312855.495  312855.495  312855.495  312855.495  312855.495
##        1231        1232        1233        1234        1235        1236
##  312855.495  312855.495  312855.495  312855.495   30905.577   30905.577
##        1237        1238        1239        1240        1241        1242
##   30905.577  202286.900  202286.900  202286.900  -29907.150   47490.866
##        1243        1244        1245        1246        1247        1248
##   47490.866  572691.694  533992.686  533992.686  113832.024  113832.024
##        1249        1250        1251        1252        1253        1254
```

```
##  113832.024  113832.024  279684.917  279684.917  279684.917  279684.917
##        1255        1256        1257        1258        1259        1260
##  279684.917  279684.917  279684.917  279684.917  279684.917  279684.917
##        1261        1262        1263        1264        1265        1266
##  279684.917  279684.917  279684.917  102775.164  102775.164  191230.040
##        1267        1268        1269        1270        1271        1272
##  417895.661  417895.661  417895.661  417895.661  417895.661  417895.661
##        1273        1274        1275        1276        1277        1278
##  417895.661  417895.661  417895.661  180173.181  180173.181  180173.181
##        1279        1280        1281        1282        1283        1284
##  180173.181  180173.181  334969.214  334969.214  334969.214  334969.214
##        1285        1286        1287        1288        1289        1290
##  334969.214  334969.214  334969.214  334969.214  207815.330  130417.313
##        1291        1292        1293        1294        1295        1296
##  130417.313   91718.305   91718.305   91718.305   91718.305   91718.305
##        1297        1298        1299        1300        1301        1302
##   91718.305   91718.305   91718.305   91718.305   58547.726   58547.726
##        1303        1304        1305        1306        1307        1308
## 1296915.994 1296915.994 1296915.994 1296915.994 1296915.994 1296915.994
##        1309        1310        1311        1312        1313        1314
## 1296915.994 1296915.994 1296915.994 1296915.994 1296915.994 1296915.994
##        1315        1316        1317        1318        1319        1320
## 1296915.994 1296915.994 1296915.994 1296915.994 1296915.994 1296915.994
##        1321        1322        1323        1324        1325        1326
##  213343.759  213343.759  213343.759  213343.759  213343.759  412367.231
##        1327        1328        1329        1330        1331        1332
##  191230.040  191230.040  191230.040  191230.040  191230.040  191230.040
##        1333        1334        1335        1336        1337        1338
##  191230.040  191230.040  191230.040  191230.040  191230.040  191230.040
##        1339        1340        1341        1342        1343        1344
##  191230.040  191230.040  191230.040  351554.504  351554.504  351554.504
##        1345        1346        1347        1348        1349        1350
##  351554.504  351554.504  351554.504  351554.504  351554.504   97246.734
##        1351        1352        1353        1354        1355        1356
##   97246.734   97246.734   97246.734   97246.734  268628.057  268628.057
##        1357        1358        1359        1360        1361        1362
##  268628.057  268628.057  268628.057  207815.330  207815.330  207815.330
##        1363        1364        1365        1366        1367        1368
##  207815.330  290741.776  290741.776  290741.776  290741.776  290741.776
##        1369        1370        1371        1372        1373        1374
##  119360.453  119360.453  119360.453  517407.397  517407.397  517407.397
##        1375        1376        1377        1378        1379        1380
##  517407.397  517407.397  440009.380  440009.380  440009.380  440009.380
##        1381        1382        1383        1384        1385        1386
##  440009.380  440009.380  440009.380   41962.437   41962.437   14320.288
##        1387        1388        1389        1390        1391        1392
```

```
##   14320.288  296270.206  296270.206  694317.149  694317.149  694317.149
##        1393        1394        1395        1396        1397        1398
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        1399        1400        1401        1402        1403        1404
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        1405        1406        1407        1408        1409        1410
##  694317.149  191230.040  191230.040  191230.040  191230.040  838056.323
##        1411        1412        1413        1414        1415        1416
##  838056.323  838056.323  838056.323  838056.323  838056.323  838056.323
##        1417        1418        1419        1420        1421        1422
##  202286.900  202286.900  202286.900  202286.900  329440.785  329440.785
##        1423        1424        1425        1426        1427        1428
##  329440.785  329440.785  329440.785  329440.785  329440.785  329440.785
##        1429        1430        1431        1432        1433        1434
##  329440.785  329440.785  329440.785  329440.785  263099.627  263099.627
##        1435        1436        1437        1438        1439        1440
##  550577.975  550577.975  550577.975  550577.975  550577.975  550577.975
##        1441        1442        1443        1444        1445        1446
##  550577.975  550577.975  550577.975  362611.363  362611.363  362611.363
##        1447        1448        1449        1450        1451        1452
##  362611.363  362611.363  362611.363  362611.363  362611.363  152531.032
##        1453        1454        1455        1456        1457        1458
##  152531.032  152531.032  152531.032  185701.611  185701.611  185701.611
##        1459        1460        1461        1462        1463        1464
##  185701.611  318383.925  318383.925  318383.925  318383.925  318383.925
##        1465        1466        1467        1468        1469        1470
##  318383.925  318383.925  318383.925  318383.925  169116.321  406838.801
##        1471        1472        1473        1474        1475        1476
##  406838.801  406838.801  406838.801  406838.801  406838.801  406838.801
##        1477        1478        1479        1480        1481        1482
##  406838.801  406838.801  406838.801  406838.801  406838.801  406838.801
##        1483        1484        1485        1486        1487        1488
##  406838.801   25377.147  572691.694  572691.694  572691.694  572691.694
##        1489        1490        1491        1492        1493        1494
##  572691.694  440009.380  440009.380  440009.380  440009.380  440009.380
##        1495        1496        1497        1498        1499        1500
##  440009.380  440009.380  440009.380  440009.380  440009.380  440009.380
##        1501        1502        1503        1504        1505        1506
##  440009.380  440009.380  440009.380  440009.380  440009.380  440009.380
##        1507        1508        1509        1510        1511        1512
##  440009.380  484236.818  484236.818  484236.818  484236.818  484236.818
##        1513        1514        1515        1516        1517        1518
##  484236.818  484236.818  362611.363  362611.363  362611.363  362611.363
##        1519        1520        1521        1522        1523        1524
##  362611.363  362611.363  578220.124  578220.124  578220.124  578220.124
##        1525        1526        1527        1528        1529        1530
```

```
##   25377.147   25377.147  169116.321  169116.321  169116.321  169116.321
##        1531        1532        1533        1534        1535        1536
##  169116.321  169116.321  169116.321  169116.321  169116.321  169116.321
##        1537        1538        1539        1540        1541        1542
##  169116.321  169116.321  -13321.861  240985.908  240985.908  240985.908
##        1543        1544        1545        1546        1547        1548
##  240985.908  240985.908  240985.908  428952.520  285213.346  285213.346
##        1549        1550        1551        1552        1553        1554
##  285213.346  285213.346  141474.172  417895.661  417895.661  417895.661
##        1555        1556        1557        1558        1559        1560
##  417895.661  417895.661  417895.661  417895.661  417895.661  417895.661
##        1561        1562        1563        1564        1565        1566
##  279684.917  279684.917  279684.917  279684.917  279684.917  279684.917
##        1567        1568        1569        1570        1571        1572
##  279684.917  279684.917  279684.917  180173.181  180173.181  180173.181
##        1573        1574        1575        1576        1577        1578
##  180173.181  180173.181  180173.181  180173.181  180173.181  180173.181
##        1579        1580        1581        1582        1583        1584
##  180173.181  180173.181  180173.181  517407.397  517407.397  517407.397
##        1585        1586        1587        1588        1589        1590
##  517407.397  517407.397  865698.472  865698.472  865698.472  865698.472
##        1591        1592        1593        1594        1595        1596
##  865698.472  865698.472  865698.472  865698.472  865698.472  865698.472
##        1597        1598        1599        1600        1601        1602
##  865698.472  865698.472  865698.472  865698.472  865698.472  865698.472
##        1603        1604        1605        1606        1607        1608
##  865698.472  865698.472  865698.472  865698.472  865698.472  782772.026
##        1609        1610        1611        1612        1613        1614
##  782772.026  782772.026  782772.026  782772.026  782772.026  782772.026
##        1615        1616        1617        1618        1619        1620
##  782772.026  782772.026  782772.026  782772.026  351554.504  351554.504
##        1621        1622        1623        1624        1625        1626
##  351554.504  351554.504  351554.504  351554.504  351554.504  351554.504
##        1627        1628        1629        1630        1631        1632
##  351554.504  351554.504  218872.189  218872.189  218872.189  218872.189
##        1633        1634        1635        1636        1637        1638
##  218872.189  218872.189  218872.189  218872.189  218872.189  473179.959
##        1639        1640        1641        1642        1643        1644
##  473179.959  473179.959  185701.611  185701.611  185701.611  124888.883
##        1645        1646        1647        1648        1649        1650
##  705374.009  705374.009  705374.009   41962.437   86189.875   86189.875
##        1651        1652        1653        1654        1655        1656
##  406838.801  406838.801  406838.801  406838.801  406838.801  406838.801
##        1657        1658        1659        1660        1661        1662
##  406838.801  406838.801  406838.801  379196.652  379196.652  379196.652
##        1663        1664        1665        1666        1667        1668
```

```
##  379196.652  379196.652  379196.652  379196.652  379196.652  379196.652
##        1669        1670        1671        1672        1673        1674
##  379196.652  379196.652  379196.652  379196.652  379196.652  379196.652
##        1675        1676        1677        1678        1679        1680
##  379196.652  379196.652  627975.992  627975.992  627975.992  627975.992
##        1681        1682        1683        1684        1685        1686
##  627975.992  627975.992  627975.992  627975.992  627975.992  627975.992
##        1687        1688        1689        1690        1691        1692
##  721959.298  721959.298  721959.298  721959.298  440009.380  440009.380
##        1693        1694        1695        1696        1697        1698
##  440009.380  440009.380  440009.380  440009.380  440009.380  440009.380
##        1699        1700        1701        1702        1703        1704
##   19848.718  152531.032  152531.032  296270.206  296270.206  296270.206
##        1705        1706        1707        1708        1709        1710
##  296270.206  296270.206  296270.206  417895.661  417895.661  417895.661
##        1711        1712        1713        1714        1715        1716
##  417895.661  417895.661  417895.661  417895.661  417895.661  417895.661
##        1717        1718        1719        1720        1721        1722
##   47490.866   41962.437  533992.686  533992.686  533992.686  533992.686
##        1723        1724        1725        1726        1727        1728
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        1729        1730        1731        1732        1733        1734
##  334969.214  334969.214   30905.577   30905.577   30905.577   75133.015
##        1735        1736        1737        1738        1739        1740
##   25377.147  158059.462  158059.462  158059.462  158059.462  158059.462
##        1741        1742        1743        1744        1745        1746
##  158059.462  224400.619  224400.619  224400.619  224400.619  727487.728
##        1747        1748        1749        1750        1751        1752
##  727487.728  727487.728  727487.728  368139.793  368139.793  368139.793
##        1753        1754        1755        1756        1757        1758
##  368139.793  368139.793  368139.793   91718.305   91718.305   91718.305
##        1759        1760        1761        1762        1763        1764
##  428952.520  428952.520  428952.520  428952.520  428952.520  428952.520
##        1765        1766        1767        1768        1769        1770
##  428952.520  428952.520  428952.520  428952.520  113832.024  113832.024
##        1771        1772        1773        1774        1775        1776
##  113832.024  113832.024  113832.024  130417.313  130417.313  130417.313
##        1777        1778        1779        1780        1781        1782
##    3263.428    3263.428  826999.464  826999.464  826999.464  826999.464
##        1783        1784        1785        1786        1787        1788
##  826999.464  826999.464  826999.464  826999.464  826999.464  301798.636
##        1789        1790        1791        1792        1793        1794
##  301798.636  301798.636  301798.636  301798.636  301798.636  346026.074
##        1795        1796        1797        1798        1799        1800
##  346026.074  346026.074  346026.074  346026.074  346026.074  346026.074
##        1801        1802        1803        1804        1805        1806
```

```
##  605862.273  605862.273  605862.273  605862.273  605862.273  605862.273
##        1807        1808        1809        1810        1811        1812
##  605862.273  605862.273  605862.273  224400.619  224400.619  224400.619
##        1813        1814        1815        1816        1817        1818
##  224400.619  224400.619  130417.313  130417.313  130417.313  130417.313
##        1819        1820        1821        1822        1823        1824
##  130417.313  130417.313  130417.313  130417.313  130417.313  130417.313
##        1825        1826        1827        1828        1829        1830
##  113832.024  113832.024  113832.024  113832.024  113832.024  113832.024
##        1831        1832        1833        1834        1835        1836
##  113832.024  113832.024  113832.024  113832.024  113832.024  113832.024
##        1837        1838        1839        1840        1841        1842
##  721959.298  130417.313  130417.313  130417.313  130417.313  240985.908
##        1843        1844        1845        1846        1847        1848
##  240985.908  240985.908  240985.908  240985.908  240985.908  240985.908
##        1849        1850        1851        1852        1853        1854
##  240985.908  240985.908  240985.908  240985.908  240985.908  240985.908
##        1855        1856        1857        1858        1859        1860
##  240985.908  240985.908  -35435.580  224400.619  738544.587  738544.587
##        1861        1862        1863        1864        1865        1866
##  738544.587  738544.587  738544.587  738544.587  738544.587  738544.587
##        1867        1868        1869        1870        1871        1872
##  738544.587  738544.587  738544.587  274156.487  274156.487  274156.487
##        1873        1874        1875        1876        1877        1878
##  169116.321  169116.321  169116.321  -63077.729  -63077.729  -63077.729
##        1879        1880        1881        1882        1883        1884
##  -63077.729  218872.189  218872.189  218872.189  218872.189  218872.189
##        1885        1886        1887        1888        1889        1890
##  218872.189  218872.189  218872.189  218872.189  268628.057  268628.057
##        1891        1892        1893        1894        1895        1896
##  268628.057  268628.057  268628.057  268628.057  268628.057    8791.858
##        1897        1898        1899        1900        1901        1902
##  235457.478  235457.478  235457.478  235457.478  423424.091  423424.091
##        1903        1904        1905        1906        1907        1908
##  423424.091  124888.883  124888.883  124888.883  124888.883   58547.726
##        1909        1910        1911        1912        1913        1914
## 1086835.663 1086835.663 1086835.663 1086835.663 1086835.663 1086835.663
##        1915        1916        1917        1918        1919        1920
## 1086835.663 1086835.663  616919.133  616919.133  616919.133  616919.133
##        1921        1922        1923        1924        1925        1926
##  616919.133  440009.380  440009.380  440009.380  440009.380  279684.917
##        1927        1928        1929        1930        1931        1932
##  279684.917  279684.917  279684.917  279684.917  279684.917  279684.917
##        1933        1934        1935        1936        1937        1938
##  721959.298  721959.298  721959.298  721959.298  721959.298  721959.298
##        1939        1940        1941        1942        1943        1944
```

```
##  721959.298  721959.298  721959.298  721959.298  721959.298  721959.298
##        1945        1946        1947        1948        1949        1950
##  721959.298  721959.298  721959.298  721959.298  721959.298  721959.298
##        1951        1952        1953        1954        1955        1956
##  721959.298  589276.984  589276.984  589276.984  589276.984  589276.984
##        1957        1958        1959        1960        1961        1962
##  589276.984  124888.883  207815.330  207815.330  207815.330  290741.776
##        1963        1964        1965        1966        1967        1968
##  290741.776  290741.776  290741.776  290741.776  290741.776  290741.776
##        1969        1970        1971        1972        1973        1974
##  290741.776  290741.776  290741.776  290741.776  290741.776  135945.743
##        1975        1976        1977        1978        1979        1980
##  135945.743  135945.743  135945.743  135945.743  135945.743  135945.743
##        1981        1982        1983        1984        1985        1986
##  440009.380  440009.380  440009.380  440009.380  440009.380  440009.380
##        1987        1988        1989        1990        1991        1992
##  821471.034  821471.034  821471.034  296270.206  296270.206  296270.206
##        1993        1994        1995        1996        1997        1998
##  235457.478  235457.478  235457.478  235457.478  235457.478  235457.478
##        1999        2000        2001        2002        2003        2004
##  235457.478  235457.478  235457.478   41962.437   41962.437   41962.437
##        2005        2006        2007        2008        2009        2010
##   41962.437  -52020.869  -52020.869   64076.156   64076.156   64076.156
##        2011        2012        2013        2014        2015        2016
##   64076.156  561634.835  561634.835  561634.835  561634.835  627975.992
##        2017        2018        2019        2020        2021        2022
##  627975.992  627975.992  627975.992  627975.992  627975.992  627975.992
##        2023        2024        2025        2026        2027        2028
##  627975.992  627975.992  627975.992  627975.992  627975.992  -85191.448
##        2029        2030        2031        2032        2033        2034
##  -85191.448  163587.892  163587.892  163587.892  163587.892  163587.892
##        2035        2036        2037        2038        2039        2040
##  163587.892  163587.892  428952.520  428952.520  428952.520  428952.520
##        2041        2042        2043        2044        2045        2046
##  428952.520 1213989.548 1213989.548 1213989.548 1213989.548 1213989.548
##        2047        2048        2049        2050        2051        2052
## 1213989.548 1213989.548 1213989.548 1213989.548  340497.644  340497.644
##        2053        2054        2055        2056        2057        2058
##  340497.644  340497.644  340497.644  340497.644  340497.644  340497.644
##        2059        2060        2061        2062        2063        2064
##  340497.644  340497.644  340497.644  340497.644  340497.644  340497.644
##        2065        2066        2067        2068        2069        2070
##  340497.644  323912.355  323912.355  323912.355  323912.355  323912.355
##        2071        2072        2073        2074        2075        2076
##  323912.355  323912.355  323912.355  323912.355  323912.355  323912.355
##        2077        2078        2079        2080        2081        2082
```

```
##  323912.355  323912.355  323912.355  334969.214  334969.214  334969.214
##        2083        2084        2085        2086        2087        2088
##  334969.214  334969.214  334969.214  334969.214  334969.214  334969.214
##        2089        2090        2091        2092        2093        2094
##  334969.214  334969.214  334969.214  -52020.869  351554.504  351554.504
##        2095        2096        2097        2098        2099        2100
##  351554.504  351554.504  351554.504  351554.504  351554.504  351554.504
##        2101        2102        2103        2104        2105        2106
##  351554.504  351554.504  301798.636  301798.636  301798.636  301798.636
##        2107        2108        2109        2110        2111        2112
##   30905.577   30905.577  102775.164   86189.875   86189.875  191230.040
##        2113        2114        2115        2116        2117        2118
##  191230.040  191230.040  191230.040  467651.529  467651.529  467651.529
##        2119        2120        2121        2122        2123        2124
##  467651.529  467651.529  467651.529  467651.529  467651.529 1225046.407
##        2125        2126        2127        2128        2129        2130
## 1225046.407 1225046.407 1225046.407 1225046.407 1225046.407  804885.745
##        2131        2132        2133        2134        2135        2136
##  804885.745  804885.745  804885.745  804885.745  804885.745  804885.745
##        2137        2138        2139        2140        2141        2142
##  804885.745  804885.745  804885.745  804885.745  804885.745  804885.745
##        2143        2144        2145        2146        2147        2148
##  804885.745  804885.745   -7793.431   -7793.431   -7793.431  357082.933
##        2149        2150        2151        2152        2153        2154
##  357082.933  357082.933  357082.933  357082.933  357082.933  357082.933
##        2155        2156        2157        2158        2159        2160
##  357082.933  357082.933  147002.602  147002.602  274156.487  274156.487
##        2161        2162        2163        2164        2165        2166
##  274156.487  274156.487  274156.487  274156.487  274156.487  274156.487
##        2167        2168        2169        2170        2171        2172
##  185701.611  185701.611  185701.611  185701.611  357082.933  357082.933
##        2173        2174        2175        2176        2177        2178
##  357082.933  357082.933  357082.933  357082.933  357082.933  357082.933
##        2179        2180        2181        2182        2183        2184
##  357082.933  368139.793  368139.793  368139.793  368139.793  368139.793
##        2185        2186        2187        2188        2189        2190
##  368139.793   41962.437   41962.437   41962.437   41962.437   41962.437
##        2191        2192        2193        2194        2195        2196
##  218872.189  218872.189   58547.726   58547.726   58547.726   58547.726
##        2197        2198        2199        2200        2201        2202
##  661146.571  661146.571  661146.571  661146.571  661146.571  661146.571
##        2203        2204        2205        2206        2207        2208
##  661146.571  661146.571  661146.571  661146.571  661146.571  661146.571
##        2209        2210        2211        2212        2213        2214
##  268628.057  268628.057  268628.057  -40964.010  -40964.010  124888.883
##        2215        2216        2217        2218        2219        2220
```

```
##  202286.900  202286.900  202286.900  202286.900  202286.900  202286.900
##        2221        2222        2223        2224        2225        2226
##  202286.900  594805.413  594805.413  180173.181  180173.181  180173.181
##        2227        2228        2229        2230        2231        2232
##  180173.181  180173.181  180173.181  180173.181  124888.883  124888.883
##        2233        2234        2235        2236        2237        2238
##  274156.487  274156.487  274156.487  274156.487  357082.933  357082.933
##        2239        2240        2241        2242        2243        2244
##  357082.933  224400.619  218872.189  218872.189  218872.189  229929.049
##        2245        2246        2247        2248        2249        2250
##  229929.049  522935.826  522935.826  362611.363  362611.363  362611.363
##        2251        2252        2253        2254        2255        2256
##  362611.363  362611.363  362611.363  362611.363  362611.363  362611.363
##        2257        2258        2259        2260        2261        2262
##  362611.363  362611.363  362611.363  362611.363  417895.661  417895.661
##        2263        2264        2265        2266        2267        2268
##  417895.661  417895.661  417895.661  417895.661  417895.661  417895.661
##        2269        2270        2271        2272        2273        2274
##  246514.338  246514.338  246514.338  246514.338  417895.661  417895.661
##        2275        2276        2277        2278        2279        2280
##  417895.661  417895.661  733016.158  733016.158  733016.158  733016.158
##        2281        2282        2283        2284        2285        2286
##  733016.158  733016.158  733016.158  733016.158  733016.158  733016.158
##        2287        2288        2289        2290        2291        2292
##  733016.158  733016.158  733016.158  733016.158 1014966.076 1014966.076
##        2293        2294        2295        2296        2297        2298
## 1014966.076 1014966.076 1014966.076 1014966.076 1014966.076 1014966.076
##        2299        2300        2301        2302        2303        2304
## 1014966.076   86189.875   86189.875   86189.875   86189.875  218872.189
##        2305        2306        2307        2308        2309        2310
##  218872.189  108303.594  108303.594  108303.594  108303.594  390253.512
##        2311        2312        2313        2314        2315        2316
##  390253.512  390253.512  390253.512  390253.512  390253.512  390253.512
##        2317        2318        2319        2320        2321        2322
##  390253.512  390253.512  390253.512  390253.512  390253.512  390253.512
##        2323        2324        2325        2326        2327        2328
##  390253.512  -40964.010  235457.478  235457.478  235457.478  235457.478
##        2329        2330        2331        2332        2333        2334
##  235457.478  235457.478  235457.478  -18850.291  -18850.291  -18850.291
##        2335        2336        2337        2338        2339        2340
##  -18850.291  766186.736  766186.736  766186.736  766186.736  766186.736
##        2341        2342        2343        2344        2345        2346
##  766186.736  180173.181  180173.181  180173.181  180173.181  180173.181
##        2347        2348        2349        2350        2351        2352
##  357082.933  357082.933  357082.933  357082.933  357082.933  522935.826
##        2353        2354        2355        2356        2357        2358
```

```
##  522935.826  522935.826  522935.826  246514.338  246514.338  246514.338
##        2359        2360        2361        2362        2363        2364
##  246514.338  246514.338  246514.338  246514.338   25377.147   25377.147
##        2365        2366        2367        2368        2369        2370
##   25377.147   25377.147   25377.147   25377.147   25377.147   25377.147
##        2371        2372        2373        2374        2375        2376
##   25377.147   25377.147   25377.147   25377.147  567163.265  567163.265
##        2377        2378        2379        2380        2381        2382
##  567163.265  567163.265  567163.265  567163.265  567163.265  567163.265
##        2383        2384        2385        2386        2387        2388
##  379196.652  379196.652  379196.652  379196.652  379196.652  379196.652
##        2389        2390        2391        2392        2393        2394
##  379196.652  185701.611  185701.611  185701.611  185701.611  185701.611
##        2395        2396        2397        2398        2399        2400
##  185701.611  185701.611  185701.611  185701.611  185701.611  301798.636
##        2401        2402        2403        2404        2405        2406
##  301798.636  301798.636  301798.636  301798.636  301798.636  301798.636
##        2407        2408        2409        2410        2411        2412
##  301798.636  301798.636  301798.636  301798.636  301798.636  301798.636
##        2413        2414        2415        2416        2417        2418
##   58547.726   58547.726   58547.726   58547.726   58547.726  285213.346
##        2419        2420        2421        2422        2423        2424
##  368139.793  141474.172  141474.172  141474.172  141474.172  307327.065
##        2425        2426        2427        2428        2429        2430
##  307327.065  307327.065  307327.065  307327.065  522935.826  522935.826
##        2431        2432        2433        2434        2435        2436
##  522935.826  334969.214  334969.214  334969.214  334969.214  334969.214
##        2437        2438        2439        2440        2441        2442
##  334969.214  334969.214  334969.214  334969.214  191230.040  672203.430
##        2443        2444        2445        2446        2447        2448
##  672203.430  672203.430  672203.430  672203.430  672203.430  672203.430
##        2449        2450        2451        2452        2453        2454
##  672203.430  672203.430  672203.430   14320.288   14320.288  158059.462
##        2455        2456        2457        2458        2459        2460
##  158059.462  158059.462  158059.462  158059.462  158059.462   75133.015
##        2461        2462        2463        2464        2465        2466
##   75133.015   75133.015  351554.504  351554.504  351554.504  351554.504
##        2467        2468        2469        2470        2471        2472
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        2473        2474        2475        2476        2477        2478
##  130417.313  130417.313  130417.313  130417.313  130417.313  340497.644
##        2479        2480        2481        2482        2483        2484
##  340497.644  340497.644  340497.644  340497.644  340497.644  340497.644
##        2485        2486        2487        2488        2489        2490
##  340497.644  340497.644  340497.644  340497.644  274156.487  274156.487
##        2491        2492        2493        2494        2495        2496
```

```
##  274156.487  274156.487  274156.487  274156.487  274156.487  274156.487
##        2497        2498        2499        2500        2501        2502
##  274156.487  274156.487  246514.338  804885.745  804885.745  965210.208
##        2503        2504        2505        2506        2507        2508
##  965210.208  965210.208  965210.208  965210.208  965210.208  965210.208
##        2509        2510        2511        2512        2513        2514
##  965210.208  965210.208  965210.208  965210.208  965210.208  965210.208
##        2515        2516        2517        2518        2519        2520
##  965210.208  965210.208  965210.208  965210.208  965210.208  965210.208
##        2521        2522        2523        2524        2525        2526
##  965210.208  533992.686  533992.686  533992.686  533992.686  533992.686
##        2527        2528        2529        2530        2531        2532
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        2533        2534        2535        2536        2537        2538
##  533992.686  533992.686  533992.686  533992.686  533992.686  390253.512
##        2539        2540        2541        2542        2543        2544
##  390253.512  108303.594  108303.594  108303.594  108303.594  108303.594
##        2545        2546        2547        2548        2549        2550
## 1407484.589 1407484.589 1407484.589 1407484.589 1407484.589 1407484.589
##        2551        2552        2553        2554        2555        2556
## 1407484.589 1407484.589 1407484.589 1407484.589 1407484.589  152531.032
##        2557        2558        2559        2560        2561        2562
##  152531.032  113832.024  113832.024  113832.024   30905.577   30905.577
##        2563        2564        2565        2566        2567        2568
##   30905.577   30905.577   58547.726   58547.726   58547.726   58547.726
##        2569        2570        2571        2572        2573        2574
##   58547.726   58547.726  334969.214  511878.967  511878.967   64076.156
##        2575        2576        2577        2578        2579        2580
##   47490.866   -2265.002   -2265.002   -2265.002  517407.397  390253.512
##        2581        2582        2583        2584        2585        2586
##  274156.487  274156.487  274156.487  274156.487  274156.487  274156.487
##        2587        2588        2589        2590        2591        2592
##  274156.487  274156.487  274156.487  274156.487  274156.487  113832.024
##        2593        2594        2595        2596        2597        2598
##  113832.024  113832.024  113832.024  263099.627  263099.627  390253.512
##        2599        2600        2601        2602        2603        2604
##  390253.512  390253.512  152531.032  152531.032  368139.793  368139.793
##        2605        2606        2607        2608        2609        2610
##  368139.793  368139.793  368139.793  368139.793  368139.793  368139.793
##        2611        2612        2613        2614        2615        2616
##  368139.793  368139.793  368139.793  368139.793  699845.579  699845.579
##        2617        2618        2619        2620        2621        2622
##  699845.579  782772.026  782772.026  782772.026  572691.694  572691.694
##        2623        2624        2625        2626        2627        2628
##  572691.694  572691.694  572691.694  898869.051  898869.051  898869.051
##        2629        2630        2631        2632        2633        2634
```

```
##  898869.051  898869.051  898869.051  898869.051  898869.051  898869.051
##        2635        2636        2637        2638        2639        2640
##  898869.051  898869.051  898869.051  898869.051  440009.380  440009.380
##        2641        2642        2643        2644        2645        2646
##  440009.380  119360.453  119360.453  119360.453  119360.453  119360.453
##        2647        2648        2649        2650        2651        2652
##  119360.453  119360.453  119360.453  119360.453  207815.330  207815.330
##        2653        2654        2655        2656        2657        2658
##  207815.330  207815.330  207815.330  207815.330  207815.330  207815.330
##        2659        2660        2661        2662        2663        2664
## 1258216.986 1258216.986 1258216.986 1258216.986   -7793.431   -7793.431
##        2665        2666        2667        2668        2669        2670
##   -7793.431   75133.015  274156.487  274156.487  274156.487  274156.487
##        2671        2672        2673        2674        2675        2676
##  274156.487  147002.602  147002.602  147002.602  147002.602  124888.883
##        2677        2678        2679        2680        2681        2682
##  124888.883   69604.585   69604.585   69604.585   69604.585   69604.585
##        2683        2684        2685        2686        2687        2688
##   69604.585   69604.585  119360.453  147002.602  108303.594  108303.594
##        2689        2690        2691        2692        2693        2694
##  108303.594  108303.594  108303.594  108303.594  108303.594  108303.594
##        2695        2696        2697        2698        2699        2700
##  285213.346  285213.346  285213.346  285213.346  285213.346  285213.346
##        2701        2702        2703        2704        2705        2706
##  285213.346  285213.346  285213.346  141474.172  141474.172  141474.172
##        2707        2708        2709        2710        2711        2712
##  141474.172  141474.172  141474.172  141474.172  141474.172  141474.172
##        2713        2714        2715        2716        2717        2718
##  141474.172  141474.172  141474.172  141474.172  782772.026  782772.026
##        2719        2720        2721        2722        2723        2724
##  782772.026  782772.026  782772.026  782772.026  782772.026  782772.026
##        2725        2726        2727        2728        2729        2730
##  782772.026  102775.164  102775.164 1075778.803 1075778.803 1075778.803
##        2731        2732        2733        2734        2735        2736
## 1075778.803 1075778.803  240985.908  240985.908  240985.908  240985.908
##        2737        2738        2739        2740        2741        2742
##  263099.627  263099.627  263099.627  147002.602  147002.602  147002.602
##        2743        2744        2745        2746        2747        2748
##  147002.602  147002.602  147002.602  545049.546  545049.546  545049.546
##        2749        2750        2751        2752        2753        2754
##  545049.546  545049.546  545049.546  545049.546  340497.644  340497.644
##        2755        2756        2757        2758        2759        2760
##  340497.644  340497.644  340497.644  340497.644  340497.644  340497.644
##        2761        2762        2763        2764        2765        2766
##  340497.644  340497.644  340497.644  340497.644  340497.644   47490.866
##        2767        2768        2769        2770        2771        2772
```

```
##   47490.866  207815.330  158059.462  556106.405  556106.405  556106.405
##        2773        2774        2775        2776        2777        2778
##  556106.405  556106.405  556106.405  556106.405  556106.405  141474.172
##        2779        2780        2781        2782        2783        2784
##  163587.892  163587.892  163587.892  163587.892  163587.892  169116.321
##        2785        2786        2787        2788        2789        2790
##  169116.321  169116.321   64076.156   64076.156  533992.686  533992.686
##        2791        2792        2793        2794        2795        2796
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        2797        2798        2799        2800        2801        2802
##  533992.686  533992.686  533992.686  533992.686  533992.686  202286.900
##        2803        2804        2805        2806        2807        2808
##  417895.661  417895.661  417895.661  417895.661  268628.057  268628.057
##        2809        2810        2811        2812        2813        2814
##  390253.512  390253.512  390253.512  301798.636  301798.636  301798.636
##        2815        2816        2817        2818        2819        2820
##  301798.636  301798.636   41962.437   41962.437   41962.437   41962.437
##        2821        2822        2823        2824        2825        2826
##  290741.776  290741.776  290741.776  290741.776  290741.776  782772.026
##        2827        2828        2829        2830        2831        2832
##  782772.026  782772.026  782772.026  782772.026  782772.026  782772.026
##        2833        2834        2835        2836        2837        2838
##  782772.026  782772.026  782772.026  782772.026  782772.026  782772.026
##        2839        2840        2841        2842        2843        2844
##  782772.026  782772.026  240985.908  240985.908  373668.223  373668.223
##        2845        2846        2847        2848        2849        2850
##  373668.223  373668.223  373668.223  373668.223  373668.223  373668.223
##        2851        2852        2853        2854        2855        2856
##  373668.223  373668.223  373668.223  373668.223  373668.223  373668.223
##        2857        2858        2859        2860        2861        2862
##  373668.223  373668.223  373668.223  373668.223  174644.751  445537.810
##        2863        2864        2865        2866        2867        2868
##  445537.810  445537.810  445537.810  445537.810   36434.007  423424.091
##        2869        2870        2871        2872        2873        2874
##  423424.091  423424.091  -29907.150  368139.793  368139.793  235457.478
##        2875        2876        2877        2878        2879        2880
##  235457.478  235457.478   53019.296   53019.296   53019.296  -13321.861
##        2881        2882        2883        2884        2885        2886
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        2887        2888        2889        2890        2891        2892
##  533992.686  533992.686  533992.686  -18850.291  -18850.291  -18850.291
##        2893        2894        2895        2896        2897        2898
##  108303.594  368139.793  368139.793  368139.793  368139.793  368139.793
##        2899        2900        2901        2902        2903        2904
##  368139.793  368139.793  368139.793  368139.793  130417.313  130417.313
##        2905        2906        2907        2908        2909        2910
```

```
##  130417.313  130417.313  130417.313  185701.611  185701.611  185701.611
##        2911        2912        2913        2914        2915        2916
##  185701.611  185701.611  185701.611  185701.611  185701.611  185701.611
##        2917        2918        2919        2920        2921        2922
##  185701.611  185701.611  185701.611  185701.611  130417.313  661146.571
##        2923        2924        2925        2926        2927        2928
##  661146.571  661146.571  661146.571  661146.571  661146.571   75133.015
##        2929        2930        2931        2932        2933        2934
##   75133.015   75133.015  721959.298  445537.810  445537.810  445537.810
##        2935        2936        2937        2938        2939        2940
##  445537.810  445537.810  445537.810  445537.810  484236.818  484236.818
##        2941        2942        2943        2944        2945        2946
##  484236.818  484236.818  484236.818  484236.818  484236.818  484236.818
##        2947        2948        2949        2950        2951        2952
##  484236.818  484236.818  484236.818  141474.172  141474.172  141474.172
##        2953        2954        2955        2956        2957        2958
##  141474.172  141474.172  141474.172  141474.172  141474.172  141474.172
##        2959        2960        2961        2962        2963        2964
##  141474.172  141474.172  141474.172  141474.172  141474.172  113832.024
##        2965        2966        2967        2968        2969        2970
##  113832.024  113832.024  113832.024  915454.340  915454.340  915454.340
##        2971        2972        2973        2974        2975        2976
##  915454.340  915454.340  915454.340  915454.340  915454.340  915454.340
##        2977        2978        2979        2980        2981        2982
##  915454.340  915454.340  915454.340   14320.288   14320.288   14320.288
##        2983        2984        2985        2986        2987        2988
##   14320.288   14320.288  202286.900  202286.900  202286.900  202286.900
##        2989        2990        2991        2992        2993        2994
##  202286.900  202286.900  246514.338  246514.338  246514.338  246514.338
##        2995        2996        2997        2998        2999        3000
##  246514.338  246514.338  246514.338   91718.305   91718.305   91718.305
##        3001        3002        3003        3004        3005        3006
##   91718.305   91718.305   91718.305  395781.942  395781.942  395781.942
##        3007        3008        3009        3010        3011        3012
##  395781.942  395781.942  395781.942  395781.942  395781.942  395781.942
##        3013        3014        3015        3016        3017        3018
##  395781.942  395781.942  395781.942  395781.942  119360.453  119360.453
##        3019        3020        3021        3022        3023        3024
##  119360.453  340497.644  268628.057  268628.057  268628.057  268628.057
##        3025        3026        3027        3028        3029        3030
##  268628.057  174644.751  174644.751  174644.751  174644.751  174644.751
##        3031        3032        3033        3034        3035        3036
##  174644.751  174644.751  174644.751  174644.751  -85191.448    8791.858
##        3037        3038        3039        3040        3041        3042
##    8791.858    8791.858  163587.892  163587.892  163587.892  163587.892
##        3043        3044        3045        3046        3047        3048
```

```
##  793828.885  793828.885  793828.885  793828.885  793828.885  793828.885
##        3049        3050        3051        3052        3053        3054
##  793828.885  793828.885  793828.885  793828.885  368139.793  368139.793
##        3055        3056        3057        3058        3059        3060
##  368139.793  368139.793  440009.380  440009.380  357082.933  357082.933
##        3061        3062        3063        3064        3065        3066
##  357082.933  -18850.291  174644.751  174644.751  351554.504  351554.504
##        3067        3068        3069        3070        3071        3072
##  351554.504  351554.504  351554.504  351554.504  351554.504  351554.504
##        3073        3074        3075        3076        3077        3078
##  351554.504  351554.504   41962.437   41962.437  -85191.448  -85191.448
##        3079        3080        3081        3082        3083        3084
##  898869.051  898869.051  898869.051  898869.051  898869.051  898869.051
##        3085        3086        3087        3088        3089        3090
##  898869.051  898869.051  898869.051  898869.051  898869.051  898869.051
##        3091        3092        3093        3094        3095        3096
##  898869.051  898869.051  898869.051  898869.051  898869.051  898869.051
##        3097        3098        3099        3100        3101        3102
##  898869.051   30905.577  180173.181  180173.181  180173.181  180173.181
##        3103        3104        3105        3106        3107        3108
##   91718.305   91718.305   41962.437  318383.925  318383.925  318383.925
##        3109        3110        3111        3112        3113        3114
##  318383.925  318383.925  318383.925  318383.925  318383.925  318383.925
##        3115        3116        3117        3118        3119        3120
##  318383.925  744073.017  744073.017  744073.017  744073.017  744073.017
##        3121        3122        3123        3124        3125        3126
##  744073.017  744073.017  744073.017  744073.017  744073.017  854641.613
##        3127        3128        3129        3130        3131        3132
##  854641.613  854641.613  854641.613  854641.613  854641.613  854641.613
##        3133        3134        3135        3136        3137        3138
##  854641.613  854641.613  854641.613  854641.613  854641.613  854641.613
##        3139        3140        3141        3142        3143        3144
##  854641.613  854641.613  854641.613  854641.613  484236.818  484236.818
##        3145        3146        3147        3148        3149        3150
##  484236.818  484236.818  887812.191  887812.191  887812.191  887812.191
##        3151        3152        3153        3154        3155        3156
##  887812.191  887812.191  887812.191  887812.191  887812.191  887812.191
##        3157        3158        3159        3160        3161        3162
##  887812.191  887812.191    8791.858    8791.858  -63077.729  -63077.729
##        3163        3164        3165        3166        3167        3168
##  279684.917  279684.917  279684.917  279684.917  279684.917  279684.917
##        3169        3170        3171        3172        3173        3174
##  279684.917  279684.917  279684.917  279684.917  423424.091  423424.091
##        3175        3176        3177        3178        3179        3180
##  423424.091  423424.091  423424.091  169116.321  169116.321  169116.321
##        3181        3182        3183        3184        3185        3186
```

```
##  224400.619  224400.619  224400.619  224400.619  224400.619  224400.619
##        3187        3188        3189        3190        3191        3192
##  224400.619  224400.619  224400.619  224400.619  224400.619    3263.428
##        3193        3194        3195        3196        3197        3198
##    3263.428    3263.428    3263.428 1009437.646 1009437.646 1009437.646
##        3199        3200        3201        3202        3203        3204
## 1009437.646 1009437.646 1009437.646 1009437.646 1009437.646  346026.074
##        3205        3206        3207        3208        3209        3210
##  346026.074  346026.074  346026.074  346026.074  346026.074  346026.074
##        3211        3212        3213        3214        3215        3216
##  285213.346  285213.346  456594.669  456594.669  456594.669  456594.669
##        3217        3218        3219        3220        3221        3222
##  456594.669  456594.669  456594.669  456594.669  456594.669   19848.718
##        3223        3224        3225        3226        3227        3228
##   19848.718  -29907.150  301798.636   86189.875   86189.875   86189.875
##        3229        3230        3231        3232        3233        3234
##   86189.875  672203.430  672203.430  672203.430  655618.141  655618.141
##        3235        3236        3237        3238        3239        3240
##  655618.141  655618.141  655618.141  655618.141  655618.141  650089.711
##        3241        3242        3243        3244        3245        3246
##  650089.711  650089.711  650089.711  650089.711  650089.711  650089.711
##        3247        3248        3249        3250        3251        3252
##  650089.711  650089.711  650089.711  650089.711  650089.711  650089.711
##        3253        3254        3255        3256        3257        3258
##  650089.711  650089.711  650089.711  445537.810  445537.810  445537.810
##        3259        3260        3261        3262        3263        3264
##  445537.810  445537.810  445537.810   36434.007  368139.793  368139.793
##        3265        3266        3267        3268        3269        3270
##  368139.793  368139.793  368139.793  368139.793  -13321.861  528464.256
##        3271        3272        3273        3274        3275        3276
##  528464.256  528464.256  528464.256  528464.256  263099.627  263099.627
##        3277        3278        3279        3280        3281        3282
##  263099.627  473179.959  473179.959  473179.959  473179.959  473179.959
##        3283        3284        3285        3286        3287        3288
##  473179.959  473179.959  473179.959  473179.959  522935.826  522935.826
##        3289        3290        3291        3292        3293        3294
##  522935.826  522935.826  522935.826  522935.826  522935.826  522935.826
##        3295        3296        3297        3298        3299        3300
##  522935.826  522935.826  522935.826  522935.826  522935.826  522935.826
##        3301        3302        3303        3304        3305        3306
##  -24378.721  -24378.721  263099.627  263099.627  263099.627  263099.627
##        3307        3308        3309        3310        3311        3312
##  263099.627  263099.627  605862.273  605862.273  605862.273  605862.273
##        3313        3314        3315        3316        3317        3318
##  738544.587  738544.587  738544.587  738544.587  738544.587  738544.587
##        3319        3320        3321        3322        3323        3324
```

```
##  738544.587  738544.587  738544.587  738544.587  738544.587   19848.718
##        3325        3326        3327        3328        3329        3330
##  838056.323  838056.323  838056.323  838056.323  838056.323  838056.323
##        3331        3332        3333        3334        3335        3336
##  838056.323  838056.323  838056.323  838056.323  462123.099  462123.099
##        3337        3338        3339        3340        3341        3342
##  462123.099  462123.099  462123.099  462123.099  462123.099  462123.099
##        3343        3344        3345        3346        3347        3348
##  462123.099  462123.099  462123.099  135945.743  135945.743  240985.908
##        3349        3350        3351        3352        3353        3354
##  240985.908  240985.908  240985.908  240985.908  240985.908  135945.743
##        3355        3356        3357        3358        3359        3360
##  135945.743  135945.743  135945.743  135945.743  135945.743  141474.172
##        3361        3362        3363        3364        3365        3366
##  141474.172  141474.172  843584.753  843584.753  843584.753  843584.753
##        3367        3368        3369        3370        3371        3372
##  843584.753  843584.753  843584.753  843584.753  843584.753  843584.753
##        3373        3374        3375        3376        3377        3378
##  843584.753  495293.678  495293.678  495293.678  495293.678  495293.678
##        3379        3380        3381        3382        3383        3384
##  495293.678  495293.678  495293.678  495293.678  495293.678  495293.678
##        3385        3386        3387        3388        3389        3390
##  351554.504  351554.504  351554.504  351554.504  462123.099  462123.099
##        3391        3392        3393        3394        3395        3396
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        3397        3398        3399        3400        3401        3402
##  401310.372  401310.372  401310.372  401310.372  252042.768  252042.768
##        3403        3404        3405        3406        3407        3408
##  252042.768   14320.288   14320.288   14320.288  102775.164  102775.164
##        3409        3410        3411        3412        3413        3414
##  102775.164  102775.164  102775.164  102775.164  102775.164  102775.164
##        3415        3416        3417        3418        3419        3420
##  102775.164  102775.164   53019.296  257571.198  257571.198  268628.057
##        3421        3422        3423        3424        3425        3426
##  445537.810  445537.810  445537.810  412367.231  412367.231  412367.231
##        3427        3428        3429        3430        3431        3432
##  412367.231  412367.231  412367.231  412367.231  412367.231  412367.231
##        3433        3434        3435        3436        3437        3438
##  -35435.580  252042.768  252042.768  252042.768  252042.768  417895.661
##        3439        3440        3441        3442        3443        3444
##  417895.661  417895.661  417895.661  417895.661  417895.661   86189.875
##        3445        3446        3447        3448        3449        3450
##   86189.875   86189.875  451066.239  451066.239  451066.239  451066.239
##        3451        3452        3453        3454        3455        3456
##  451066.239  451066.239  451066.239  451066.239  451066.239  451066.239
##        3457        3458        3459        3460        3461        3462
```

```
##  451066.239  451066.239  379196.652  379196.652  379196.652  379196.652
##        3463        3464        3465        3466        3467        3468
##  644561.281  644561.281  644561.281  644561.281  644561.281  644561.281
##        3469        3470        3471        3472        3473        3474
##  644561.281  644561.281  644561.281  644561.281   53019.296   53019.296
##        3475        3476        3477        3478        3479        3480
##   53019.296    3263.428  373668.223  373668.223  373668.223  373668.223
##        3481        3482        3483        3484        3485        3486
##  373668.223  373668.223  373668.223  373668.223  373668.223  373668.223
##        3487        3488        3489        3490        3491        3492
##  373668.223  141474.172  135945.743  135945.743  135945.743  135945.743
##        3493        3494        3495        3496        3497        3498
##  135945.743  135945.743  135945.743  135945.743  135945.743  135945.743
##        3499        3500        3501        3502        3503        3504
## 1125534.671 1125534.671 1125534.671 1125534.671 1125534.671 1125534.671
##        3505        3506        3507        3508        3509        3510
## 1125534.671 1125534.671 1125534.671  312855.495  312855.495  312855.495
##        3511        3512        3513        3514        3515        3516
##  312855.495  312855.495  312855.495  312855.495 -13321.861  633504.422
##        3517        3518        3519        3520        3521        3522
##  633504.422  633504.422  633504.422  633504.422  633504.422  633504.422
##        3523        3524        3525        3526        3527        3528
##   75133.015   75133.015  218872.189  218872.189  218872.189   36434.007
##        3529        3530        3531        3532        3533        3534
##   36434.007   36434.007   36434.007   36434.007   36434.007   47490.866
##        3535        3536        3537        3538        3539        3540
##   47490.866  246514.338  246514.338  246514.338  108303.594  108303.594
##        3541        3542        3543        3544        3545        3546
##  108303.594  108303.594  108303.594  108303.594 -29907.150   64076.156
##        3547        3548        3549        3550        3551        3552
##   64076.156   64076.156   64076.156   64076.156   64076.156  240985.908
##        3553        3554        3555        3556        3557        3558
##  240985.908  240985.908  240985.908  240985.908  252042.768  252042.768
##        3559        3560        3561        3562        3563        3564
##  252042.768  860170.042  860170.042  860170.042  860170.042  860170.042
##        3565        3566        3567        3568        3569        3570
##  860170.042  860170.042  860170.042  860170.042  860170.042  860170.042
##        3571        3572        3573        3574        3575        3576
##  860170.042  860170.042  860170.042  860170.042  860170.042  860170.042
##        3577        3578        3579        3580        3581        3582
##   36434.007   91718.305  158059.462  158059.462  158059.462  158059.462
##        3583        3584        3585        3586        3587        3588
##  158059.462   36434.007   36434.007  207815.330  207815.330  207815.330
##        3589        3590        3591        3592        3593        3594
##  152531.032  152531.032   25377.147  285213.346  285213.346  285213.346
##        3595        3596        3597        3598        3599        3600
```

```
##  285213.346  285213.346  285213.346  285213.346   80661.445  113832.024
##        3601        3602        3603        3604        3605        3606
##  113832.024  113832.024  755129.877  755129.877  755129.877  755129.877
##        3607        3608        3609        3610        3611        3612
##  755129.877  755129.877  755129.877  755129.877  755129.877  893340.621
##        3613        3614        3615        3616        3617        3618
##  893340.621  893340.621  893340.621  893340.621  639032.852  639032.852
##        3619        3620        3621        3622        3623        3624
##  639032.852  639032.852  639032.852 1258216.986 1258216.986 1258216.986
##        3625        3626        3627        3628        3629        3630
## 1258216.986 1258216.986 1258216.986 1258216.986 1258216.986 1258216.986
##        3631        3632        3633        3634        3635        3636
##  318383.925  318383.925  318383.925  318383.925  318383.925  318383.925
##        3637        3638        3639        3640        3641        3642
##  650089.711  650089.711   25377.147   25377.147   25377.147   25377.147
##        3643        3644        3645        3646        3647        3648
##  677731.860  677731.860  677731.860  550577.975  550577.975  550577.975
##        3649        3650        3651        3652        3653        3654
##  550577.975  550577.975  550577.975  550577.975  550577.975  550577.975
##        3655        3656        3657        3658        3659        3660
##  550577.975  550577.975  550577.975  108303.594  108303.594  108303.594
##        3661        3662        3663        3664        3665        3666
##  533992.686  533992.686  533992.686   41962.437   41962.437   41962.437
##        3667        3668        3669        3670        3671        3672
##    3263.428  661146.571  661146.571  661146.571  661146.571  661146.571
##        3673        3674        3675        3676        3677        3678
##  661146.571  661146.571  661146.571  661146.571  661146.571  661146.571
##        3679        3680        3681        3682        3683        3684
##  661146.571  661146.571  661146.571  661146.571  661146.571  661146.571
##        3685        3686        3687        3688        3689        3690
##  661146.571  301798.636  301798.636  191230.040  191230.040  191230.040
##        3691        3692        3693        3694        3695        3696
##  191230.040  511878.967  511878.967  511878.967  511878.967  511878.967
##        3697        3698        3699        3700        3701        3702
##  511878.967  511878.967  511878.967  511878.967  511878.967  511878.967
##        3703        3704        3705        3706        3707        3708
##  511878.967  511878.967  511878.967  600333.843  600333.843  600333.843
##        3709        3710        3711        3712        3713        3714
##  600333.843  434480.950  434480.950  434480.950  434480.950  202286.900
##        3715        3716        3717        3718        3719        3720
##  202286.900  202286.900  312855.495  312855.495  312855.495  312855.495
##        3721        3722        3723        3724        3725        3726
##  312855.495  312855.495  312855.495  257571.198  257571.198  257571.198
##        3727        3728        3729        3730        3731        3732
##  102775.164  323912.355  323912.355  323912.355  323912.355  323912.355
##        3733        3734        3735        3736        3737        3738
```

```
##  323912.355  323912.355  323912.355  323912.355  274156.487   41962.437
##        3739        3740        3741        3742        3743        3744
##   41962.437  766186.736  766186.736  766186.736  766186.736  766186.736
##        3745        3746        3747        3748        3749        3750
##  766186.736  766186.736  180173.181  180173.181  180173.181  180173.181
##        3751        3752        3753        3754        3755        3756
##  180173.181  180173.181  180173.181  180173.181  180173.181  180173.181
##        3757        3758        3759        3760        3761        3762
##  180173.181  351554.504   80661.445   80661.445   80661.445   80661.445
##        3763        3764        3765        3766        3767        3768
##  224400.619  224400.619  224400.619  224400.619  334969.214  334969.214
##        3769        3770        3771        3772        3773        3774
##  334969.214  334969.214  334969.214  334969.214  334969.214  334969.214
##        3775        3776        3777        3778        3779        3780
##  334969.214  334969.214  334969.214  334969.214  334969.214  334969.214
##        3781        3782        3783        3784        3785        3786
##  334969.214  235457.478  235457.478  235457.478  235457.478  235457.478
##        3787        3788        3789        3790        3791        3792
##  235457.478  235457.478  235457.478  235457.478 -18850.291 -18850.291
##        3793        3794        3795        3796        3797        3798
## -18850.291  395781.942  395781.942  395781.942  395781.942  395781.942
##        3799        3800        3801        3802        3803        3804
##  395781.942  395781.942  395781.942  395781.942  395781.942  395781.942
##        3805        3806        3807        3808        3809        3810
##  727487.728  727487.728  727487.728  727487.728   41962.437   14320.288
##        3811        3812        3813        3814        3815        3816
##   14320.288   14320.288  119360.453  119360.453  119360.453  119360.453
##        3817        3818        3819        3820        3821        3822
##  119360.453  119360.453  119360.453  119360.453  119360.453  119360.453
##        3823        3824        3825        3826        3827        3828
##  373668.223  373668.223  373668.223  373668.223  373668.223  373668.223
##        3829        3830        3831        3832        3833        3834
##  373668.223   80661.445   80661.445   80661.445   80661.445  749601.447
##        3835        3836        3837        3838        3839        3840
##  749601.447  749601.447  749601.447  749601.447  749601.447  749601.447
##        3841        3842        3843        3844        3845        3846
##  749601.447   75133.015   75133.015   75133.015   75133.015  290741.776
##        3847        3848        3849        3850        3851        3852
##  290741.776  290741.776  290741.776  290741.776  290741.776  290741.776
##        3853        3854        3855        3856        3857        3858
##  290741.776  290741.776  290741.776  290741.776  290741.776  290741.776
##        3859        3860        3861        3862        3863        3864
##  290741.776  290741.776  290741.776  368139.793  368139.793  368139.793
##        3865        3866        3867        3868        3869        3870
##  368139.793  368139.793  368139.793   25377.147  346026.074  346026.074
##        3871        3872        3873        3874        3875        3876
```

```
##  346026.074  346026.074  346026.074  346026.074  346026.074  346026.074
##        3877        3878        3879        3880        3881        3882
##  346026.074  318383.925  318383.925  318383.925  318383.925  533992.686
##        3883        3884        3885        3886        3887        3888
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        3889        3890        3891        3892        3893        3894
##  533992.686  533992.686   91718.305   97246.734   97246.734  229929.049
##        3895        3896        3897        3898        3899        3900
##  229929.049  229929.049  229929.049  346026.074  346026.074  346026.074
##        3901        3902        3903        3904        3905        3906
##  346026.074  346026.074  346026.074  196758.470  196758.470  301798.636
##        3907        3908        3909        3910        3911        3912
##  301798.636  207815.330  207815.330   91718.305   91718.305  213343.759
##        3913        3914        3915        3916        3917        3918
##  213343.759  213343.759  318383.925  318383.925  318383.925  467651.529
##        3919        3920        3921        3922        3923        3924
##  467651.529  467651.529  467651.529  467651.529  467651.529  467651.529
##        3925        3926        3927        3928        3929        3930
##  467651.529  467651.529  440009.380  440009.380  440009.380  440009.380
##        3931        3932        3933        3934        3935        3936
##  440009.380  440009.380  440009.380  440009.380  440009.380  738544.587
##        3937        3938        3939        3940        3941        3942
##  738544.587  119360.453  119360.453  539521.116  539521.116  539521.116
##        3943        3944        3945        3946        3947        3948
##  539521.116  539521.116  539521.116  539521.116  539521.116  539521.116
##        3949        3950        3951        3952        3953        3954
##  539521.116  539521.116  539521.116  539521.116  539521.116  539521.116
##        3955        3956        3957        3958        3959        3960
##  539521.116  533992.686  533992.686  533992.686  533992.686  533992.686
##        3961        3962        3963        3964        3965        3966
##  533992.686  533992.686  533992.686  533992.686  533992.686  838056.323
##        3967        3968        3969        3970        3971        3972
##  838056.323  838056.323  838056.323  838056.323  838056.323  838056.323
##        3973        3974        3975        3976        3977        3978
##  838056.323  838056.323  838056.323  838056.323  838056.323  838056.323
##        3979        3980        3981        3982        3983        3984
##  838056.323  838056.323  351554.504  351554.504  351554.504  351554.504
##        3985        3986        3987        3988        3989        3990
##  351554.504  351554.504  351554.504  351554.504  240985.908  240985.908
##        3991        3992        3993        3994        3995        3996
##  240985.908  240985.908  240985.908   25377.147  434480.950  434480.950
##        3997        3998        3999        4000        4001        4002
##  434480.950  434480.950  434480.950  434480.950  517407.397  517407.397
##        4003        4004        4005        4006        4007        4008
##  517407.397  517407.397  517407.397  517407.397  517407.397  517407.397
##        4009        4010        4011        4012        4013        4014
```

```
##  517407.397  517407.397  517407.397  517407.397  517407.397  152531.032
##        4015        4016        4017        4018        4019        4020
##  -40964.010  -40964.010  -18850.291  -18850.291  213343.759  213343.759
##        4021        4022        4023        4024        4025        4026
##  213343.759  213343.759  213343.759  213343.759  213343.759  213343.759
##        4027        4028        4029        4030        4031        4032
##  -57549.299  777243.596  777243.596  777243.596  777243.596  777243.596
##        4033        4034        4035        4036        4037        4038
##  777243.596  777243.596  777243.596  777243.596  362611.363  362611.363
##        4039        4040        4041        4042        4043        4044
##  362611.363  362611.363  462123.099  462123.099  462123.099  119360.453
##        4045        4046        4047        4048        4049        4050
##  119360.453  119360.453  119360.453  119360.453  119360.453  119360.453
##        4051        4052        4053        4054        4055        4056
##  119360.453  119360.453  119360.453  119360.453  340497.644  340497.644
##        4057        4058        4059        4060        4061        4062
##  340497.644  196758.470  196758.470  -18850.291  -18850.291  -18850.291
##        4063        4064        4065        4066        4067        4068
##  -85191.448  611390.703  611390.703  611390.703  611390.703  -46492.440
##        4069        4070        4071        4072        4073        4074
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        4075        4076        4077        4078        4079        4080
##  694317.149  694317.149  694317.149  694317.149  694317.149  716430.868
##        4081        4082        4083        4084        4085        4086
##  716430.868  -52020.869  301798.636  301798.636  301798.636  301798.636
##        4087        4088        4089        4090        4091        4092
##  301798.636  301798.636  301798.636  301798.636  301798.636  301798.636
##        4093        4094        4095        4096        4097        4098
##  329440.785  329440.785  329440.785  329440.785  329440.785  329440.785
##        4099        4100        4101        4102        4103        4104
##  329440.785  329440.785  329440.785  329440.785  329440.785  329440.785
##        4105        4106        4107        4108        4109        4110
##  871226.902  871226.902  871226.902  871226.902  871226.902  871226.902
##        4111        4112        4113        4114        4115        4116
##  871226.902  871226.902  876755.332  876755.332  876755.332  876755.332
##        4117        4118        4119        4120        4121        4122
##  876755.332  876755.332  876755.332  876755.332  876755.332  876755.332
##        4123        4124        4125        4126        4127        4128
##  876755.332  876755.332  876755.332  876755.332  705374.009  705374.009
##        4129        4130        4131        4132        4133        4134
##  705374.009  705374.009  705374.009  705374.009  705374.009  705374.009
##        4135        4136        4137        4138        4139        4140
##  705374.009  705374.009  655618.141  655618.141  655618.141  655618.141
##        4141        4142        4143        4144        4145        4146
##  655618.141  655618.141  655618.141  655618.141  655618.141  655618.141
##        4147        4148        4149        4150        4151        4152
```

```
##  655618.141  655618.141  655618.141  655618.141  655618.141  655618.141
##        4153        4154        4155        4156        4157        4158
##  655618.141  174644.751  174644.751  174644.751  174644.751  412367.231
##        4159        4160        4161        4162        4163        4164
##  412367.231  412367.231  412367.231  412367.231  412367.231  412367.231
##        4165        4166        4167        4168        4169        4170
##  412367.231  412367.231  412367.231  412367.231  130417.313  130417.313
##        4171        4172        4173        4174        4175        4176
##   53019.296   53019.296   53019.296   97246.734   97246.734   97246.734
##        4177        4178        4179        4180        4181        4182
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        4183        4184        4185        4186        4187        4188
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        4189        4190        4191        4192        4193        4194
##  694317.149  130417.313  130417.313  130417.313  285213.346  285213.346
##        4195        4196        4197        4198        4199        4200
##  285213.346  285213.346  285213.346  285213.346  285213.346  285213.346
##        4201        4202        4203        4204        4205        4206
##  285213.346  285213.346  285213.346  285213.346  285213.346  285213.346
##        4207        4208        4209        4210        4211        4212
##   41962.437   41962.437   41962.437   97246.734   97246.734  163587.892
##        4213        4214        4215        4216        4217        4218
##  384725.082  384725.082  384725.082  384725.082  384725.082  384725.082
##        4219        4220        4221        4222        4223        4224
##  384725.082  384725.082  384725.082  384725.082  384725.082  384725.082
##        4225        4226        4227        4228        4229        4230
##   -7793.431   -7793.431  163587.892  163587.892  163587.892  163587.892
##        4231        4232        4233        4234        4235        4236
##  163587.892  163587.892  163587.892  163587.892  163587.892  163587.892
##        4237        4238        4239        4240        4241        4242
##  124888.883  594805.413  594805.413  594805.413  594805.413  594805.413
##        4243        4244        4245        4246        4247        4248
##  594805.413  594805.413  594805.413  594805.413  583748.554  583748.554
##        4249        4250        4251        4252        4253        4254
##  583748.554  583748.554  583748.554  583748.554  583748.554  583748.554
##        4255        4256        4257        4258        4259        4260
##  583748.554  583748.554  583748.554  583748.554  583748.554  583748.554
##        4261        4262        4263        4264        4265        4266
##  633504.422  633504.422  633504.422  633504.422  633504.422  633504.422
##        4267        4268        4269        4270        4271        4272
##  633504.422  633504.422  633504.422  633504.422  633504.422  633504.422
##        4273        4274        4275        4276        4277        4278
##  -85191.448  -85191.448  500822.107  500822.107  500822.107  500822.107
##        4279        4280        4281        4282        4283        4284
##  500822.107  500822.107  500822.107  500822.107  500822.107  500822.107
##        4285        4286        4287        4288        4289        4290
```

```
##  500822.107  500822.107  357082.933  357082.933  357082.933  357082.933
##        4291        4292        4293        4294        4295        4296
##   14320.288   14320.288  135945.743  683260.290  683260.290 1142119.961
##        4297        4298        4299        4300        4301        4302
## 1142119.961 1142119.961 1142119.961 1142119.961 1142119.961 1142119.961
##        4303        4304        4305        4306        4307        4308
##  384725.082 1390899.300 1390899.300 1390899.300 1390899.300 1390899.300
##        4309        4310        4311        4312        4313        4314
## 1390899.300 1390899.300 1390899.300 1390899.300 1390899.300 1390899.300
##        4315        4316        4317        4318        4319        4320
## 1390899.300 1390899.300   97246.734   97246.734   97246.734   97246.734
##        4321        4322        4323        4324        4325        4326
##   97246.734   97246.734   97246.734   97246.734   97246.734  158059.462
##        4327        4328        4329        4330        4331        4332
##  511878.967  511878.967  511878.967  511878.967  511878.967  511878.967
##        4333        4334        4335        4336        4337        4338
##  511878.967  511878.967  511878.967  511878.967  511878.967  511878.967
##        4339        4340        4341        4342        4343        4344
##  511878.967  511878.967  511878.967  357082.933  451066.239  451066.239
##        4345        4346        4347        4348        4349        4350
##  451066.239  451066.239  451066.239  451066.239  451066.239  451066.239
##        4351        4352        4353        4354        4355        4356
##  451066.239  451066.239  451066.239    8791.858   36434.007   36434.007
##        4357        4358        4359        4360        4361        4362
##   36434.007  362611.363  218872.189  218872.189  218872.189   19848.718
##        4363        4364        4365        4366        4367        4368
##   19848.718   19848.718   19848.718   19848.718  705374.009  705374.009
##        4369        4370        4371        4372        4373        4374
##  705374.009  705374.009  705374.009  705374.009  705374.009  705374.009
##        4375        4376        4377        4378        4379        4380
##  594805.413  301798.636  108303.594  108303.594  108303.594  108303.594
##        4381        4382        4383        4384        4385        4386
##  108303.594  445537.810  445537.810  445537.810  445537.810  445537.810
##        4387        4388        4389        4390        4391        4392
##  445537.810  445537.810  379196.652  379196.652  379196.652  379196.652
##        4393        4394        4395        4396        4397        4398
##  379196.652  379196.652  379196.652  158059.462  158059.462  158059.462
##        4399        4400        4401        4402        4403        4404
##  141474.172  141474.172  627975.992  627975.992  627975.992  627975.992
##        4405        4406        4407        4408        4409        4410
##  627975.992  627975.992  627975.992  749601.447  749601.447  749601.447
##        4411        4412        4413        4414        4415        4416
##  749601.447  749601.447  749601.447  749601.447  749601.447  749601.447
##        4417        4418        4419        4420        4421        4422
##  749601.447  749601.447  749601.447  749601.447  202286.900  202286.900
##        4423        4424        4425        4426        4427        4428
```

```
##  202286.900  202286.900  202286.900  147002.602  147002.602  147002.602
##        4429        4430        4431        4432        4433        4434
##  147002.602  147002.602   36434.007   36434.007   36434.007   36434.007
##        4435        4436        4437        4438        4439        4440
##  124888.883  124888.883  124888.883  152531.032  152531.032  152531.032
##        4441        4442        4443        4444        4445        4446
##  152531.032  152531.032  152531.032  152531.032  152531.032  152531.032
##        4447        4448        4449        4450        4451        4452
##  152531.032 1418541.449 1418541.449 1418541.449 1418541.449 1418541.449
##        4453        4454        4455        4456        4457        4458
## 1418541.449 1418541.449 1418541.449 1418541.449 1418541.449 1418541.449
##        4459        4460        4461        4462        4463        4464
## 1418541.449 1418541.449 1418541.449 1418541.449  462123.099  462123.099
##        4465        4466        4467        4468        4469        4470
##  462123.099  462123.099  462123.099  462123.099  462123.099  462123.099
##        4471        4472        4473        4474        4475        4476
##  462123.099  462123.099    3263.428    3263.428    3263.428    3263.428
##        4477        4478        4479        4480        4481        4482
##  937568.059  937568.059  937568.059  937568.059  937568.059  937568.059
##        4483        4484        4485        4486        4487        4488
##  937568.059  937568.059  937568.059  937568.059  937568.059  937568.059
##        4489        4490        4491        4492        4493        4494
##  937568.059  937568.059  937568.059  937568.059   19848.718   19848.718
##        4495        4496        4497        4498        4499        4500
##  307327.065  307327.065  307327.065  307327.065  307327.065  307327.065
##        4501        4502        4503        4504        4505        4506
##  307327.065  307327.065  307327.065  290741.776  290741.776   69604.585
##        4507        4508        4509        4510        4511        4512
##   69604.585  782772.026  782772.026  782772.026  782772.026  782772.026
##        4513        4514        4515        4516        4517        4518
##  782772.026  782772.026  782772.026  782772.026  782772.026  782772.026
##        4519        4520        4521        4522        4523        4524
##  782772.026  782772.026  782772.026   41962.437  130417.313  130417.313
##        4525        4526        4527        4528        4529        4530
##  108303.594  108303.594  611390.703  611390.703  611390.703  611390.703
##        4531        4532        4533        4534        4535        4536
##  611390.703  611390.703  611390.703  611390.703  611390.703  611390.703
##        4537        4538        4539        4540        4541        4542
##  611390.703  611390.703  611390.703  611390.703  611390.703  611390.703
##        4543        4544        4545        4546        4547        4548
##  611390.703  611390.703  180173.181  180173.181  180173.181  180173.181
##        4549        4550        4551        4552        4553        4554
##  180173.181  180173.181  180173.181  180173.181  180173.181  180173.181
##        4555        4556        4557        4558        4559        4560
##  180173.181  119360.453  119360.453   69604.585  191230.040  191230.040
##        4561        4562        4563        4564        4565        4566
```

```
##  191230.040  191230.040  191230.040  694317.149  694317.149  694317.149
##        4567        4568        4569        4570        4571        4572
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        4573        4574        4575        4576        4577        4578
##  694317.149  257571.198  257571.198  257571.198  257571.198  257571.198
##        4579        4580        4581        4582        4583        4584
##  257571.198  257571.198  257571.198  257571.198   25377.147   25377.147
##        4585        4586        4587        4588        4589        4590
##  141474.172  351554.504  351554.504  351554.504  351554.504  351554.504
##        4591        4592        4593        4594        4595        4596
##  351554.504  152531.032  152531.032  301798.636  301798.636  301798.636
##        4597        4598        4599        4600        4601        4602
##  301798.636  301798.636  301798.636  301798.636  301798.636  301798.636
##        4603        4604        4605        4606        4607        4608
##  301798.636  301798.636  301798.636  301798.636  224400.619  224400.619
##        4609        4610        4611        4612        4613        4614
##  224400.619  224400.619  224400.619  224400.619  224400.619  224400.619
##        4615        4616        4617        4618        4619        4620
##  224400.619  224400.619  224400.619   97246.734   97246.734  379196.652
##        4621        4622        4623        4624        4625        4626
##  379196.652  379196.652  379196.652  379196.652  379196.652  379196.652
##        4627        4628        4629        4630        4631        4632
##   30905.577   30905.577   36434.007   36434.007   14320.288   75133.015
##        4633        4634        4635        4636        4637        4638
##   75133.015   75133.015   75133.015   75133.015  158059.462  158059.462
##        4639        4640        4641        4642        4643        4644
##  158059.462  158059.462  158059.462  -85191.448   14320.288  -18850.291
##        4645        4646        4647        4648        4649        4650
##  257571.198  257571.198  257571.198  257571.198  257571.198  257571.198
##        4651        4652        4653        4654        4655        4656
##  257571.198  799357.315  141474.172  141474.172  141474.172  141474.172
##        4657        4658        4659        4660        4661        4662
##  141474.172  141474.172  141474.172  141474.172  274156.487  274156.487
##        4663        4664        4665        4666        4667        4668
##  274156.487  351554.504  351554.504  351554.504  351554.504  351554.504
##        4669        4670        4671        4672        4673        4674
##  351554.504  351554.504  351554.504   91718.305   91718.305  799357.315
##        4675        4676        4677        4678        4679        4680
##  799357.315  799357.315  799357.315  799357.315  799357.315  799357.315
##        4681        4682        4683        4684        4685        4686
##  799357.315  799357.315  799357.315  799357.315  799357.315  799357.315
##        4687        4688        4689        4690        4691        4692
##  799357.315  799357.315  799357.315  799357.315  799357.315  799357.315
##        4693        4694        4695        4696        4697        4698
##   25377.147  810414.174  810414.174  810414.174  810414.174  810414.174
##        4699        4700        4701        4702        4703        4704
```

```
##  130417.313  130417.313  130417.313  130417.313  130417.313  130417.313
##        4705        4706        4707        4708        4709        4710
##   58547.726   58547.726   58547.726  218872.189   91718.305   91718.305
##        4711        4712        4713        4714        4715        4716
##   91718.305   91718.305  301798.636  301798.636  301798.636   58547.726
##        4717        4718        4719        4720        4721        4722
##   58547.726  522935.826  522935.826  522935.826  522935.826  301798.636
##        4723        4724        4725        4726        4727        4728
##  301798.636  301798.636  301798.636  301798.636  301798.636  301798.636
##        4729        4730        4731        4732        4733        4734
##  301798.636  815942.604  815942.604  815942.604  -24378.721   91718.305
##        4735        4736        4737        4738        4739        4740
##   91718.305   91718.305  108303.594  108303.594  279684.917  279684.917
##        4741        4742        4743        4744        4745        4746
##  279684.917  279684.917  279684.917  279684.917  279684.917  279684.917
##        4747        4748        4749        4750        4751        4752
##  279684.917  279684.917  279684.917  279684.917  213343.759  213343.759
##        4753        4754        4755        4756        4757        4758
##  213343.759  213343.759  213343.759  213343.759  213343.759  213343.759
##        4759        4760        4761        4762        4763        4764
##  213343.759  213343.759  213343.759  213343.759  213343.759  113832.024
##        4765        4766        4767        4768        4769        4770
##  113832.024  119360.453   75133.015   86189.875   86189.875   86189.875
##        4771        4772        4773        4774        4775        4776
##  -13321.861  -13321.861  -13321.861  -13321.861  733016.158  733016.158
##        4777        4778        4779        4780        4781        4782
##  733016.158  733016.158  733016.158  733016.158  733016.158  733016.158
##        4783        4784        4785        4786        4787        4788
##  733016.158  733016.158  733016.158  733016.158  329440.785  329440.785
##        4789        4790        4791        4792        4793        4794
##  329440.785  329440.785  329440.785  329440.785  329440.785  202286.900
##        4795        4796        4797        4798        4799        4800
##  202286.900  202286.900  202286.900  202286.900  202286.900  202286.900
##        4801        4802        4803        4804        4805        4806
##  202286.900  202286.900  202286.900  202286.900  202286.900  202286.900
##        4807        4808        4809        4810        4811        4812
##  307327.065  307327.065  307327.065  307327.065  307327.065  307327.065
##        4813        4814        4815        4816        4817        4818
##  307327.065  307327.065   80661.445  102775.164  102775.164  102775.164
##        4819        4820        4821        4822        4823        4824
##  102775.164  301798.636  301798.636  301798.636  301798.636  301798.636
##        4825        4826        4827        4828        4829        4830
##  301798.636   80661.445   80661.445   80661.445   80661.445   80661.445
##        4831        4832        4833        4834        4835        4836
##   80661.445   80661.445  489765.248  489765.248  489765.248  489765.248
##        4837        4838        4839        4840        4841        4842
```

```
##  489765.248  489765.248  196758.470  196758.470  196758.470   30905.577
##        4843        4844        4845        4846        4847        4848
##  445537.810  445537.810  445537.810  445537.810  445537.810  445537.810
##        4849        4850        4851        4852        4853        4854
##  445537.810  445537.810  445537.810  445537.810   69604.585   69604.585
##        4855        4856        4857        4858        4859        4860
## 1578865.912 1578865.912   64076.156  296270.206  296270.206  296270.206
##        4861        4862        4863        4864        4865        4866
##  296270.206  296270.206  296270.206  296270.206  -57549.299  -35435.580
##        4867        4868        4869        4870        4871        4872
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        4873        4874        4875        4876        4877        4878
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        4879        4880        4881        4882        4883        4884
##  533992.686  456594.669  456594.669  456594.669  572691.694  572691.694
##        4885        4886        4887        4888        4889        4890
##  572691.694  572691.694  572691.694  572691.694  572691.694  285213.346
##        4891        4892        4893        4894        4895        4896
##  285213.346  285213.346  285213.346  285213.346  285213.346  285213.346
##        4897        4898        4899        4900        4901        4902
##  285213.346  285213.346   80661.445   80661.445   80661.445   47490.866
##        4903        4904        4905        4906        4907        4908
##   47490.866   47490.866   64076.156   64076.156   64076.156  -35435.580
##        4909        4910        4911        4912        4913        4914
##  -35435.580  191230.040  191230.040  191230.040  246514.338  246514.338
##        4915        4916        4917        4918        4919        4920
##  246514.338  721959.298  721959.298  721959.298  721959.298  721959.298
##        4921        4922        4923        4924        4925        4926
##  721959.298  721959.298  390253.512  246514.338  -85191.448  -85191.448
##        4927        4928        4929        4930        4931        4932
##  -85191.448  246514.338  246514.338  246514.338  246514.338  246514.338
##        4933        4934        4935        4936        4937        4938
##  484236.818  484236.818  484236.818   -7793.431   -7793.431  163587.892
##        4939        4940        4941        4942        4943        4944
##  163587.892  163587.892  163587.892  401310.372  401310.372  401310.372
##        4945        4946        4947        4948        4949        4950
##  401310.372  401310.372  401310.372  611390.703  611390.703  611390.703
##        4951        4952        4953        4954        4955        4956
##  611390.703  611390.703  611390.703  611390.703  611390.703  611390.703
##        4957        4958        4959        4960        4961        4962
##  611390.703  611390.703  611390.703  611390.703  611390.703  611390.703
##        4963        4964        4965        4966        4967        4968
##  484236.818  484236.818  484236.818  484236.818  484236.818  484236.818
##        4969        4970        4971        4972        4973        4974
##  484236.818  484236.818  484236.818  163587.892  163587.892  174644.751
##        4975        4976        4977        4978        4979        4980
```

```
##  174644.751  174644.751  567163.265  567163.265  567163.265  567163.265
##        4981        4982        4983        4984        4985        4986
##  567163.265  567163.265  567163.265  567163.265  567163.265  567163.265
##        4987        4988        4989        4990        4991        4992
##  567163.265  567163.265  567163.265  567163.265  -40964.010  694317.149
##        4993        4994        4995        4996        4997        4998
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        4999        5000        5001        5002        5003        5004
##  252042.768  180173.181  180173.181  180173.181  395781.942  395781.942
##        5005        5006        5007        5008        5009        5010
##  395781.942  395781.942  395781.942  395781.942  395781.942  318383.925
##        5011        5012        5013        5014        5015        5016
##  318383.925  318383.925  318383.925  318383.925  318383.925  318383.925
##        5017        5018        5019        5020        5021        5022
##  318383.925  318383.925  318383.925  318383.925  716430.868  716430.868
##        5023        5024        5025        5026        5027        5028
##  716430.868  716430.868  716430.868  716430.868  716430.868  257571.198
##        5029        5030        5031        5032        5033        5034
##  102775.164  102775.164  102775.164  102775.164  102775.164  102775.164
##        5035        5036        5037        5038        5039        5040
##  185701.611  -24378.721  -24378.721  462123.099  462123.099  462123.099
##        5041        5042        5043        5044        5045        5046
##  462123.099  462123.099  462123.099  462123.099  462123.099  462123.099
##        5047        5048        5049        5050        5051        5052
##  462123.099  462123.099  462123.099  462123.099  462123.099  462123.099
##        5053        5054        5055        5056        5057        5058
##  229929.049  229929.049  229929.049  229929.049  229929.049  229929.049
##        5059        5060        5061        5062        5063        5064
##  229929.049  229929.049  229929.049  229929.049   36434.007  467651.529
##        5065        5066        5067        5068        5069        5070
##  467651.529  467651.529  467651.529  467651.529  467651.529  467651.529
##        5071        5072        5073        5074        5075        5076
##  556106.405  556106.405  556106.405  556106.405  556106.405  556106.405
##        5077        5078        5079        5080        5081        5082
##  556106.405  556106.405  130417.313  -85191.448  -85191.448  871226.902
##        5083        5084        5085        5086        5087        5088
##  257571.198  257571.198  -35435.580  334969.214  334969.214  622447.562
##        5089        5090        5091        5092        5093        5094
##  622447.562  622447.562  622447.562  622447.562  622447.562  622447.562
##        5095        5096        5097        5098        5099        5100
##  622447.562  473179.959  473179.959  473179.959  473179.959  473179.959
##        5101        5102        5103        5104        5105        5106
##  473179.959  473179.959  473179.959  473179.959  473179.959  473179.959
##        5107        5108        5109        5110        5111        5112
##  108303.594  108303.594  478708.388  478708.388  478708.388  478708.388
##        5113        5114        5115        5116        5117        5118
```

```
##  478708.388  478708.388  102775.164  102775.164  102775.164  163587.892
##        5119        5120        5121        5122        5123        5124
##  163587.892  511878.967  511878.967  511878.967  511878.967  511878.967
##        5125        5126        5127        5128        5129        5130
##  511878.967  511878.967  511878.967  511878.967  511878.967   47490.866
##        5131        5132        5133        5134        5135        5136
##  246514.338  246514.338  213343.759  213343.759  213343.759  -29907.150
##        5137        5138        5139        5140        5141        5142
##   80661.445   80661.445   80661.445   80661.445   80661.445  -29907.150
##        5143        5144        5145        5146        5147        5148
##   97246.734  218872.189  218872.189  218872.189  218872.189  191230.040
##        5149        5150        5151        5152        5153        5154
##  191230.040  202286.900  102775.164  224400.619  224400.619  224400.619
##        5155        5156        5157        5158        5159        5160
##  224400.619  804885.745  804885.745  804885.745  804885.745  130417.313
##        5161        5162        5163        5164        5165        5166
##  130417.313  130417.313  130417.313  130417.313  -85191.448   -2265.002
##        5167        5168        5169        5170        5171        5172
##  616919.133  616919.133  616919.133  616919.133  616919.133  616919.133
##        5173        5174        5175        5176        5177        5178
##  616919.133  616919.133  616919.133  616919.133  169116.321  169116.321
##        5179        5180        5181        5182        5183        5184
##  102775.164  130417.313  130417.313  451066.239  451066.239  451066.239
##        5185        5186        5187        5188        5189        5190
##  451066.239  451066.239  451066.239  451066.239  451066.239  451066.239
##        5191        5192        5193        5194        5195        5196
##  451066.239  661146.571  661146.571  661146.571  661146.571  661146.571
##        5197        5198        5199        5200        5201        5202
##  661146.571  661146.571  661146.571  661146.571  661146.571  661146.571
##        5203        5204        5205        5206        5207        5208
##  661146.571  661146.571  290741.776  290741.776  130417.313  130417.313
##        5209        5210        5211        5212        5213        5214
##  130417.313  301798.636  301798.636  301798.636  434480.950  351554.504
##        5215        5216        5217        5218        5219        5220
##  351554.504  351554.504  351554.504  351554.504  572691.694  572691.694
##        5221        5222        5223        5224        5225        5226
##  572691.694  572691.694  572691.694  572691.694  572691.694  572691.694
##        5227        5228        5229        5230        5231        5232
##  893340.621  893340.621  893340.621  893340.621  893340.621  893340.621
##        5233        5234        5235        5236        5237        5238
##  893340.621  224400.619  224400.619  224400.619  224400.619  224400.619
##        5239        5240        5241        5242        5243        5244
##  224400.619   -7793.431   -7793.431  268628.057  268628.057  268628.057
##        5245        5246        5247        5248        5249        5250
##  268628.057  268628.057  268628.057  528464.256  528464.256  113832.024
##        5251        5252        5253        5254        5255        5256
```

```
##  113832.024  113832.024  113832.024  738544.587  738544.587  738544.587
##        5257        5258        5259        5260        5261        5262
##  738544.587  738544.587  738544.587  738544.587  738544.587  738544.587
##        5263        5264        5265        5266        5267        5268
##  738544.587  738544.587  738544.587  738544.587  738544.587  738544.587
##        5269        5270        5271        5272        5273        5274
##  738544.587  738544.587   25377.147   25377.147   25377.147   25377.147
##        5275        5276        5277        5278        5279        5280
##   25377.147   25377.147 1213989.548 1213989.548 1213989.548 1213989.548
##        5281        5282        5283        5284        5285        5286
## 1213989.548 1213989.548  556106.405  556106.405  556106.405  556106.405
##        5287        5288        5289        5290        5291        5292
##  556106.405  556106.405  556106.405  556106.405  124888.883  124888.883
##        5293        5294        5295        5296        5297        5298
##  124888.883  263099.627  263099.627  263099.627  263099.627  263099.627
##        5299        5300        5301        5302        5303        5304
##  263099.627  263099.627  263099.627  263099.627  263099.627  263099.627
##        5305        5306        5307        5308        5309        5310
##  263099.627  263099.627  263099.627  263099.627  263099.627   30905.577
##        5311        5312        5313        5314        5315        5316
##  108303.594  108303.594  108303.594  108303.594  440009.380  440009.380
##        5317        5318        5319        5320        5321        5322
##  440009.380  440009.380  312855.495  312855.495  312855.495  312855.495
##        5323        5324        5325        5326        5327        5328
##  312855.495  368139.793  163587.892  163587.892  163587.892  871226.902
##        5329        5330        5331        5332        5333        5334
##  871226.902  871226.902  871226.902  871226.902  871226.902  871226.902
##        5335        5336        5337        5338        5339        5340
##  871226.902  871226.902  871226.902  235457.478  235457.478  235457.478
##        5341        5342        5343        5344        5345        5346
##  235457.478  274156.487  274156.487  274156.487  274156.487  417895.661
##        5347        5348        5349        5350        5351        5352
##  417895.661  417895.661  417895.661  417895.661  417895.661  417895.661
##        5353        5354        5355        5356        5357        5358
##  417895.661  417895.661  417895.661  926511.200  926511.200  926511.200
##        5359        5360        5361        5362        5363        5364
##  926511.200  926511.200  926511.200  926511.200  926511.200  926511.200
##        5365        5366        5367        5368        5369        5370
##  926511.200  710902.439  710902.439  710902.439  710902.439  710902.439
##        5371        5372        5373        5374        5375        5376
##  710902.439  710902.439  710902.439  710902.439  710902.439  710902.439
##        5377        5378        5379        5380        5381        5382
##  710902.439  174644.751  174644.751  174644.751  174644.751  174644.751
##        5383        5384        5385        5386        5387        5388
##  174644.751  174644.751  174644.751  174644.751  384725.082  384725.082
##        5389        5390        5391        5392        5393        5394
```

```
##  384725.082  384725.082  384725.082  384725.082  384725.082  384725.082
##        5395        5396        5397        5398        5399        5400
##  384725.082  384725.082  384725.082  229929.049  229929.049  -35435.580
##        5401        5402        5403        5404        5405        5406
##  334969.214  334969.214  334969.214  279684.917  279684.917  279684.917
##        5407        5408        5409        5410        5411        5412
##  279684.917  279684.917  279684.917  279684.917  279684.917  252042.768
##        5413        5414        5415        5416        5417        5418
##  252042.768  252042.768  252042.768  252042.768  252042.768  252042.768
##        5419        5420        5421        5422        5423        5424
##   14320.288   41962.437   41962.437   41962.437   25377.147  346026.074
##        5425        5426        5427        5428        5429        5430
##  346026.074  346026.074  279684.917  279684.917  279684.917  279684.917
##        5431        5432        5433        5434        5435        5436
##  279684.917  279684.917  279684.917  279684.917  517407.397  517407.397
##        5437        5438        5439        5440        5441        5442
##  517407.397  517407.397  517407.397  -18850.291   36434.007   36434.007
##        5443        5444        5445        5446        5447        5448
##   36434.007   36434.007   36434.007   36434.007   36434.007   36434.007
##        5449        5450        5451        5452        5453        5454
##   36434.007   36434.007  799357.315  799357.315  799357.315  799357.315
##        5455        5456        5457        5458        5459        5460
##  799357.315  799357.315  799357.315  799357.315  799357.315  799357.315
##        5461        5462        5463        5464        5465        5466
##  799357.315  799357.315  799357.315  799357.315  312855.495  312855.495
##        5467        5468        5469        5470        5471        5472
##  312855.495  312855.495  312855.495  312855.495  312855.495  312855.495
##        5473        5474        5475        5476        5477        5478
##  312855.495  312855.495  312855.495  312855.495  191230.040  191230.040
##        5479        5480        5481        5482        5483        5484
##  191230.040  191230.040  191230.040  191230.040  191230.040  191230.040
##        5485        5486        5487        5488        5489        5490
##  191230.040  191230.040  296270.206  296270.206  545049.546  545049.546
##        5491        5492        5493        5494        5495        5496
##  545049.546  545049.546  191230.040  169116.321  169116.321 1296915.994
##        5497        5498        5499        5500        5501        5502
## 1296915.994 1296915.994 1296915.994 1296915.994 1296915.994  323912.355
##        5503        5504        5505        5506        5507        5508
##  323912.355  323912.355  323912.355  323912.355  323912.355  323912.355
##        5509        5510        5511        5512        5513        5514
##  323912.355  323912.355  688788.720  688788.720  688788.720  688788.720
##        5515        5516        5517        5518        5519        5520
##  688788.720  688788.720   36434.007   36434.007  981795.497  981795.497
##        5521        5522        5523        5524        5525        5526
##  981795.497  981795.497  981795.497  981795.497  981795.497  981795.497
##        5527        5528        5529        5530        5531        5532
```

```
##  981795.497  981795.497  981795.497  755129.877  755129.877  755129.877
##        5533        5534        5535        5536        5537        5538
##  755129.877  755129.877  755129.877  755129.877  755129.877  755129.877
##        5539        5540        5541        5542        5543        5544
##  755129.877  755129.877  755129.877  755129.877  755129.877  755129.877
##        5545        5546        5547        5548        5549        5550
##  755129.877  257571.198  257571.198  257571.198  838056.323    8791.858
##        5551        5552        5553        5554        5555        5556
##   69604.585   69604.585   69604.585   69604.585   69604.585   69604.585
##        5557        5558        5559        5560        5561        5562
##   69604.585   69604.585   69604.585   69604.585   69604.585   69604.585
##        5563        5564        5565        5566        5567        5568
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        5569        5570        5571        5572        5573        5574
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        5575        5576        5577        5578        5579        5580
##  533992.686  191230.040  191230.040  191230.040  191230.040   64076.156
##        5581        5582        5583        5584        5585        5586
##   64076.156  196758.470  196758.470  196758.470  196758.470  196758.470
##        5587        5588        5589        5590        5591        5592
##  196758.470  296270.206  296270.206  296270.206  296270.206  296270.206
##        5593        5594        5595        5596        5597        5598
##  296270.206  213343.759  213343.759  213343.759  594805.413  594805.413
##        5599        5600        5601        5602        5603        5604
##  594805.413  594805.413  594805.413  594805.413  594805.413  594805.413
##        5605        5606        5607        5608        5609        5610
##  594805.413  594805.413  594805.413  594805.413  594805.413  594805.413
##        5611        5612        5613        5614        5615        5616
##  594805.413  876755.332  876755.332  876755.332  876755.332  876755.332
##        5617        5618        5619        5620        5621        5622
##  876755.332  876755.332  876755.332  876755.332  876755.332  876755.332
##        5623        5624        5625        5626        5627        5628
##  876755.332  876755.332  876755.332  876755.332  970738.638  970738.638
##        5629        5630        5631        5632        5633        5634
##  970738.638  970738.638  970738.638  970738.638  970738.638  970738.638
##        5635        5636        5637        5638        5639        5640
##  517407.397  517407.397  517407.397  517407.397  517407.397  517407.397
##        5641        5642        5643        5644        5645        5646
##  517407.397  583748.554  583748.554  583748.554  583748.554  583748.554
##        5647        5648        5649        5650        5651        5652
##  583748.554  583748.554  583748.554  583748.554  583748.554  290741.776
##        5653        5654        5655        5656        5657        5658
##  263099.627  263099.627  263099.627  263099.627  263099.627  263099.627
##        5659        5660        5661        5662        5663        5664
##  263099.627  263099.627  263099.627  263099.627  257571.198  257571.198
##        5665        5666        5667        5668        5669        5670
```

```
##  257571.198  257571.198  257571.198  257571.198  257571.198  257571.198
##        5671        5672        5673        5674        5675        5676
##  672203.430  672203.430  672203.430  672203.430  672203.430  672203.430
##        5677        5678        5679        5680        5681        5682
##  672203.430  672203.430  672203.430  224400.619  224400.619  224400.619
##        5683        5684        5685        5686        5687        5688
##  224400.619  224400.619  224400.619  224400.619  224400.619  224400.619
##        5689        5690        5691        5692        5693        5694
##  224400.619  622447.562  622447.562  622447.562  622447.562  622447.562
##        5695        5696        5697        5698        5699        5700
##  196758.470  152531.032  152531.032  152531.032  152531.032  152531.032
##        5701        5702        5703        5704        5705        5706
##  152531.032  677731.860  677731.860  677731.860  677731.860  677731.860
##        5707        5708        5709        5710        5711        5712
##  677731.860  677731.860  677731.860  296270.206  296270.206  296270.206
##        5713        5714        5715        5716        5717        5718
##  296270.206  296270.206  268628.057  268628.057  268628.057  268628.057
##        5719        5720        5721        5722        5723        5724
##  268628.057  268628.057  268628.057  268628.057  268628.057  268628.057
##        5725        5726        5727        5728        5729        5730
##   14320.288   14320.288   14320.288   14320.288   14320.288   14320.288
##        5731        5732        5733        5734        5735        5736
##   14320.288   14320.288   14320.288   14320.288  246514.338  246514.338
##        5737        5738        5739        5740        5741        5742
##  246514.338  246514.338  246514.338  467651.529  467651.529  467651.529
##        5743        5744        5745        5746        5747        5748
##  467651.529   25377.147  202286.900  202286.900  202286.900  202286.900
##        5749        5750        5751        5752        5753        5754
##  202286.900  202286.900  202286.900  202286.900   14320.288   14320.288
##        5755        5756        5757        5758        5759        5760
##   14320.288   14320.288   14320.288   14320.288  119360.453  119360.453
##        5761        5762        5763        5764        5765        5766
##   80661.445   80661.445  412367.231  412367.231  412367.231  412367.231
##        5767        5768        5769        5770        5771        5772
##  412367.231  898869.051  898869.051  898869.051  898869.051  898869.051
##        5773        5774        5775        5776        5777        5778
##  268628.057  268628.057  268628.057  169116.321  169116.321  169116.321
##        5779        5780        5781        5782        5783        5784
##  169116.321  169116.321  169116.321  169116.321  169116.321  -85191.448
##        5785        5786        5787        5788        5789        5790
##   97246.734   97246.734   97246.734  207815.330  207815.330  207815.330
##        5791        5792        5793        5794        5795        5796
##  207815.330  346026.074  346026.074  346026.074  346026.074  578220.124
##        5797        5798        5799        5800        5801        5802
##  578220.124  578220.124  578220.124  578220.124  578220.124  578220.124
##        5803        5804        5805        5806        5807        5808
```

```
##  578220.124  578220.124  578220.124  578220.124  578220.124  578220.124
##        5809        5810        5811        5812        5813        5814
##  578220.124  578220.124  578220.124   -7793.431  158059.462  158059.462
##        5815        5816        5817        5818        5819        5820
##  158059.462  357082.933  357082.933  357082.933  357082.933  357082.933
##        5821        5822        5823        5824        5825        5826
##  357082.933  357082.933  357082.933  357082.933  357082.933  357082.933
##        5827        5828        5829        5830        5831        5832
##  998380.787  998380.787  998380.787  998380.787  998380.787  998380.787
##        5833        5834        5835        5836        5837        5838
##  998380.787  998380.787  998380.787  998380.787  998380.787  998380.787
##        5839        5840        5841        5842        5843        5844
##  998380.787  998380.787  285213.346  285213.346   64076.156   64076.156
##        5845        5846        5847        5848        5849        5850
##   64076.156   64076.156   64076.156   64076.156   64076.156   64076.156
##        5851        5852        5853        5854        5855        5856
##   64076.156   64076.156   64076.156   64076.156   64076.156  434480.950
##        5857        5858        5859        5860        5861        5862
##  434480.950  434480.950  434480.950  434480.950  434480.950  434480.950
##        5863        5864        5865        5866        5867        5868
##  434480.950  434480.950  434480.950  522935.826  522935.826  522935.826
##        5869        5870        5871        5872        5873        5874
##  522935.826  323912.355  323912.355  323912.355  323912.355  323912.355
##        5875        5876        5877        5878        5879        5880
##  323912.355  323912.355  323912.355  323912.355  323912.355  323912.355
##        5881        5882        5883        5884        5885        5886
##  323912.355  323912.355  147002.602  -85191.448  401310.372  401310.372
##        5887        5888        5889        5890        5891        5892
##  401310.372  401310.372  401310.372  218872.189  218872.189  218872.189
##        5893        5894        5895        5896        5897        5898
##  218872.189  218872.189  312855.495  312855.495  312855.495  312855.495
##        5899        5900        5901        5902        5903        5904
##   -2265.002  180173.181  185701.611  185701.611  185701.611  213343.759
##        5905        5906        5907        5908        5909        5910
##  213343.759  213343.759  213343.759  213343.759  213343.759  213343.759
##        5911        5912        5913        5914        5915        5916
##  213343.759  213343.759  213343.759  213343.759  207815.330  207815.330
##        5917        5918        5919        5920        5921        5922
##  207815.330  207815.330  207815.330  207815.330  207815.330  428952.520
##        5923        5924        5925        5926        5927        5928
##  428952.520  428952.520  428952.520  428952.520  428952.520  428952.520
##        5929        5930        5931        5932        5933        5934
##  428952.520  428952.520  102775.164  102775.164  102775.164   41962.437
##        5935        5936        5937        5938        5939        5940
##   41962.437   41962.437   41962.437   41962.437  -18850.291  -18850.291
##        5941        5942        5943        5944        5945        5946
```

```
##  -18850.291  -18850.291   80661.445   80661.445   80661.445   80661.445
##        5947        5948        5949        5950        5951        5952
##  666675.000  666675.000  666675.000  666675.000  666675.000  666675.000
##        5953        5954        5955        5956        5957        5958
##  666675.000  666675.000  169116.321   47490.866   47490.866   47490.866
##        5959        5960        5961        5962        5963        5964
##  782772.026  782772.026  782772.026  782772.026  782772.026  782772.026
##        5965        5966        5967        5968        5969        5970
##  329440.785  329440.785  329440.785  329440.785  329440.785  329440.785
##        5971        5972        5973        5974        5975        5976
##  329440.785  329440.785  329440.785  -40964.010  445537.810  445537.810
##        5977        5978        5979        5980        5981        5982
##  445537.810  445537.810  445537.810  445537.810  445537.810  445537.810
##        5983        5984        5985        5986        5987        5988
##  445537.810  445537.810  445537.810  445537.810  445537.810  473179.959
##        5989        5990        5991        5992        5993        5994
##  473179.959  473179.959  473179.959  473179.959  473179.959  473179.959
##        5995        5996        5997        5998        5999        6000
##  473179.959  473179.959  473179.959  473179.959  246514.338  246514.338
##        6001        6002        6003        6004        6005        6006
##  246514.338  246514.338  246514.338  246514.338  246514.338  246514.338
##        6007        6008        6009        6010        6011        6012
##  246514.338  246514.338  246514.338   53019.296   69604.585   69604.585
##        6013        6014        6015        6016        6017        6018
##   69604.585   69604.585  605862.273  605862.273  605862.273  605862.273
##        6019        6020        6021        6022        6023        6024
##  605862.273  605862.273  605862.273  605862.273  605862.273  605862.273
##        6025        6026        6027        6028        6029        6030
##  605862.273  605862.273  605862.273  605862.273  605862.273  605862.273
##        6031        6032        6033        6034        6035        6036
##  605862.273  605862.273  533992.686  533992.686  533992.686  533992.686
##        6037        6038        6039        6040        6041        6042
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        6043        6044        6045        6046        6047        6048
##  533992.686  533992.686  533992.686  533992.686  533992.686  246514.338
##        6049        6050        6051        6052        6053        6054
##  246514.338  246514.338  246514.338  500822.107  500822.107  500822.107
##        6055        6056        6057        6058        6059        6060
##  500822.107  500822.107  500822.107  500822.107  500822.107  500822.107
##        6061        6062        6063        6064        6065        6066
##  500822.107  500822.107  500822.107  500822.107  500822.107  500822.107
##        6067        6068        6069        6070        6071        6072
##  500822.107  500822.107  500822.107  500822.107  323912.355  323912.355
##        6073        6074        6075        6076        6077        6078
##  323912.355  152531.032  152531.032  152531.032  152531.032  152531.032
##        6079        6080        6081        6082        6083        6084
```

```
##  152531.032  920982.770  920982.770  920982.770  920982.770  920982.770
##        6085        6086        6087        6088        6089        6090
##  920982.770  920982.770  920982.770  533992.686  533992.686  533992.686
##        6091        6092        6093        6094        6095        6096
##  533992.686  533992.686  533992.686  147002.602  147002.602  147002.602
##        6097        6098        6099        6100        6101        6102
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        6103        6104        6105        6106        6107        6108
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        6109        6110        6111        6112        6113        6114
##  379196.652  379196.652  379196.652  379196.652  379196.652  379196.652
##        6115        6116        6117        6118        6119        6120
##  379196.652  379196.652  500822.107  500822.107  500822.107  500822.107
##        6121        6122        6123        6124        6125        6126
##  965210.208  965210.208  965210.208  965210.208  965210.208  965210.208
##        6127        6128        6129        6130        6131        6132
##  965210.208  965210.208  965210.208  296270.206  296270.206  296270.206
##        6133        6134        6135        6136        6137        6138
##  594805.413  594805.413  594805.413  594805.413  -85191.448 1136591.531
##        6139        6140        6141        6142        6143        6144
##  395781.942  395781.942  395781.942  -29907.150  799357.315  799357.315
##        6145        6146        6147        6148        6149        6150
##  799357.315  799357.315  799357.315  799357.315  799357.315  799357.315
##        6151        6152        6153        6154        6155        6156
##  799357.315  799357.315  799357.315  799357.315  235457.478  235457.478
##        6157        6158        6159        6160        6161        6162
##  235457.478  235457.478  235457.478  235457.478  235457.478  235457.478
##        6163        6164        6165        6166        6167        6168
##  235457.478  235457.478  235457.478  235457.478  235457.478  235457.478
##        6169        6170        6171        6172        6173        6174
##  235457.478  235457.478  235457.478  235457.478   64076.156   64076.156
##        6175        6176        6177        6178        6179        6180
##   64076.156   64076.156   64076.156   80661.445   80661.445  351554.504
##        6181        6182        6183        6184        6185        6186
##  644561.281  644561.281  644561.281  644561.281  644561.281  644561.281
##        6187        6188        6189        6190        6191        6192
##  644561.281  644561.281  644561.281  163587.892  163587.892  163587.892
##        6193        6194        6195        6196        6197        6198
##  163587.892  163587.892  163587.892  263099.627  263099.627  263099.627
##        6199        6200        6201        6202        6203        6204
##  263099.627  357082.933  357082.933  357082.933  357082.933  357082.933
##        6205        6206        6207        6208        6209        6210
##  357082.933  357082.933  235457.478  235457.478  235457.478  235457.478
##        6211        6212        6213        6214        6215        6216
##  235457.478  235457.478  235457.478  235457.478  235457.478  235457.478
##        6217        6218        6219        6220        6221        6222
```

```
##  235457.478  235457.478  235457.478   19848.718   19848.718   19848.718
##        6223        6224        6225        6226        6227        6228
##  329440.785  329440.785  329440.785  329440.785  522935.826   97246.734
##        6229        6230        6231        6232        6233        6234
##   97246.734   97246.734   97246.734  124888.883  556106.405  556106.405
##        6235        6236        6237        6238        6239        6240
##  556106.405  556106.405  556106.405  556106.405  556106.405  550577.975
##        6241        6242        6243        6244        6245        6246
##  550577.975  550577.975  550577.975  550577.975  550577.975  550577.975
##        6247        6248        6249        6250        6251        6252
##   19848.718   19848.718   19848.718  102775.164  102775.164  102775.164
##        6253        6254        6255        6256        6257        6258
##  434480.950  434480.950  434480.950  434480.950  434480.950  434480.950
##        6259        6260        6261        6262        6263        6264
##  434480.950  434480.950  434480.950  180173.181  318383.925  318383.925
##        6265        6266        6267        6268        6269        6270
##  318383.925  318383.925  318383.925  169116.321  169116.321  113832.024
##        6271        6272        6273        6274        6275        6276
##  113832.024  113832.024  860170.042  860170.042  860170.042  860170.042
##        6277        6278        6279        6280        6281        6282
##  860170.042  860170.042  860170.042  860170.042  860170.042  860170.042
##        6283        6284        6285        6286        6287        6288
##  860170.042  860170.042  290741.776  290741.776  290741.776  290741.776
##        6289        6290        6291        6292        6293        6294
##  290741.776  290741.776  290741.776  290741.776  567163.265  567163.265
##        6295        6296        6297        6298        6299        6300
##  567163.265  412367.231  412367.231  412367.231  412367.231  412367.231
##        6301        6302        6303        6304        6305        6306
##  412367.231  323912.355  323912.355  323912.355  323912.355  323912.355
##        6307        6308        6309        6310        6311        6312
##  920982.770  920982.770  920982.770  920982.770  473179.959  473179.959
##        6313        6314        6315        6316        6317        6318
##  473179.959  456594.669  456594.669  -68606.159  -35435.580  -35435.580
##        6319        6320        6321        6322        6323        6324
##  511878.967  511878.967  511878.967  511878.967  511878.967  511878.967
##        6325        6326        6327        6328        6329        6330
##  511878.967  511878.967  511878.967  511878.967  511878.967  511878.967
##        6331        6332        6333        6334        6335        6336
##  279684.917  279684.917  733016.158  733016.158  539521.116  539521.116
##        6337        6338        6339        6340        6341        6342
##  539521.116  539521.116  539521.116  539521.116  539521.116  539521.116
##        6343        6344        6345        6346        6347        6348
##  539521.116  462123.099  462123.099  462123.099  462123.099  462123.099
##        6349        6350        6351        6352        6353        6354
##  462123.099  462123.099  462123.099  412367.231  412367.231  412367.231
##        6355        6356        6357        6358        6359        6360
```

```
##  412367.231  412367.231  412367.231   58547.726   58547.726  229929.049
##        6361        6362        6363        6364        6365        6366
##  229929.049  229929.049  229929.049  456594.669  456594.669  456594.669
##        6367        6368        6369        6370        6371        6372
##  456594.669  456594.669  456594.669  456594.669  456594.669  456594.669
##        6373        6374        6375        6376        6377        6378
##  456594.669  456594.669  456594.669  456594.669  174644.751  307327.065
##        6379        6380        6381        6382        6383        6384
##  307327.065  307327.065  307327.065  307327.065  307327.065  113832.024
##        6385        6386        6387        6388        6389        6390
##  390253.512  390253.512  390253.512  390253.512  390253.512  185701.611
##        6391        6392        6393        6394        6395        6396
##  185701.611  185701.611  185701.611  185701.611  185701.611  185701.611
##        6397        6398        6399        6400        6401        6402
##  185701.611  185701.611  185701.611  185701.611  185701.611  340497.644
##        6403        6404        6405        6406        6407        6408
##  340497.644  340497.644  268628.057  268628.057  268628.057  268628.057
##        6409        6410        6411        6412        6413        6414
##  705374.009  705374.009  705374.009  705374.009  705374.009  705374.009
##        6415        6416        6417        6418        6419        6420
##  705374.009  705374.009  705374.009  705374.009  705374.009  705374.009
##        6421        6422        6423        6424        6425        6426
##  705374.009  705374.009  705374.009  705374.009  705374.009  517407.397
##        6427        6428        6429        6430        6431        6432
##  -52020.869  -52020.869  -13321.861  -13321.861  124888.883  124888.883
##        6433        6434        6435        6436        6437        6438
##  124888.883  124888.883   36434.007   36434.007   86189.875   86189.875
##        6439        6440        6441        6442        6443        6444
##   86189.875  462123.099  462123.099  462123.099  462123.099  462123.099
##        6445        6446        6447        6448        6449        6450
##  462123.099  462123.099  -24378.721  -24378.721  141474.172  539521.116
##        6451        6452        6453        6454        6455        6456
##  539521.116  539521.116  539521.116  539521.116  539521.116  539521.116
##        6457        6458        6459        6460        6461        6462
##  539521.116  539521.116  539521.116  539521.116  539521.116  539521.116
##        6463        6464        6465        6466        6467        6468
##  539521.116  539521.116  440009.380  440009.380  440009.380  440009.380
##        6469        6470        6471        6472        6473        6474
##  290741.776  290741.776  290741.776  290741.776  290741.776  290741.776
##        6475        6476        6477        6478        6479        6480
##  290741.776  290741.776  290741.776  290741.776  290741.776  180173.181
##        6481        6482        6483        6484        6485        6486
##  180173.181  180173.181  196758.470  196758.470  196758.470  196758.470
##        6487        6488        6489        6490        6491        6492
##  196758.470  196758.470  196758.470  522935.826  522935.826  522935.826
##        6493        6494        6495        6496        6497        6498
```

```
##  522935.826  522935.826  246514.338  246514.338  246514.338  246514.338
##        6499        6500        6501        6502        6503        6504
##  246514.338  246514.338 -18850.291 -18850.291  401310.372  401310.372
##        6505        6506        6507        6508        6509        6510
##  401310.372  401310.372  401310.372  401310.372  401310.372  401310.372
##        6511        6512        6513        6514        6515        6516
##  401310.372  401310.372  401310.372  495293.678  495293.678  495293.678
##        6517        6518        6519        6520        6521        6522
##  495293.678  495293.678  495293.678  495293.678  495293.678   64076.156
##        6523        6524        6525        6526        6527        6528
##   64076.156  368139.793  368139.793  368139.793  368139.793  368139.793
##        6529        6530        6531        6532        6533        6534
##   91718.305   91718.305   91718.305   91718.305  661146.571  196758.470
##        6535        6536        6537        6538        6539        6540
##  196758.470  196758.470  196758.470  196758.470  196758.470  196758.470
##        6541        6542        6543        6544        6545        6546
##  174644.751  174644.751  174644.751  174644.751  174644.751  174644.751
##        6547        6548        6549        6550        6551        6552
##  174644.751  174644.751  174644.751  174644.751  174644.751  174644.751
##        6553        6554        6555        6556        6557        6558
##  174644.751 1059193.514 1059193.514 1059193.514 1059193.514 1059193.514
##        6559        6560        6561        6562        6563        6564
## 1059193.514 1059193.514 1059193.514 1059193.514 1059193.514 1059193.514
##        6565        6566        6567        6568        6569        6570
## 1059193.514 1059193.514 1059193.514 1059193.514 1059193.514 1059193.514
##        6571        6572        6573        6574        6575        6576
## 1059193.514 1059193.514 1059193.514  240985.908  556106.405  556106.405
##        6577        6578        6579        6580        6581        6582
##  556106.405  556106.405  312855.495  312855.495  451066.239  451066.239
##        6583        6584        6585        6586        6587        6588
##  451066.239  451066.239  451066.239   53019.296   91718.305   91718.305
##        6589        6590        6591        6592        6593        6594
##   91718.305   91718.305   91718.305   91718.305   91718.305  -85191.448
##        6595        6596        6597        6598        6599        6600
##  108303.594  108303.594  108303.594  108303.594  108303.594   30905.577
##        6601        6602        6603        6604        6605        6606
##   30905.577   30905.577   30905.577  -13321.861  -13321.861  357082.933
##        6607        6608        6609        6610        6611        6612
##  357082.933  357082.933  357082.933  357082.933  357082.933  650089.711
##        6613        6614        6615        6616        6617        6618
##  650089.711  650089.711  650089.711  650089.711  650089.711  650089.711
##        6619        6620        6621        6622        6623        6624
##  650089.711  650089.711  650089.711  650089.711  650089.711  650089.711
##        6625        6626        6627        6628        6629        6630
##  650089.711  -29907.150  -29907.150   80661.445  578220.124  578220.124
##        6631        6632        6633        6634        6635        6636
```

```
##  578220.124  578220.124  578220.124  578220.124  578220.124  578220.124
##        6637        6638        6639        6640        6641        6642
##  578220.124  578220.124  578220.124  550577.975  550577.975  550577.975
##        6643        6644        6645        6646        6647        6648
##  661146.571  661146.571  661146.571  661146.571  661146.571  661146.571
##        6649        6650        6651        6652        6653        6654
##  661146.571  661146.571  661146.571  661146.571  882283.761  882283.761
##        6655        6656        6657        6658        6659        6660
##  882283.761  882283.761  882283.761 1120006.242 1120006.242 1120006.242
##        6661        6662        6663        6664        6665        6666
## 1120006.242 1120006.242 1120006.242   14320.288   14320.288  147002.602
##        6667        6668        6669        6670        6671        6672
##  124888.883  124888.883  124888.883  124888.883  124888.883  124888.883
##        6673        6674        6675        6676        6677        6678
##  124888.883  124888.883  124888.883  124888.883  124888.883  124888.883
##        6679        6680        6681        6682        6683        6684
##  373668.223  373668.223  373668.223  373668.223  373668.223  373668.223
##        6685        6686        6687        6688        6689        6690
##  373668.223  373668.223  373668.223  462123.099  462123.099  462123.099
##        6691        6692        6693        6694        6695        6696
##  462123.099  462123.099  644561.281  644561.281  644561.281  644561.281
##        6697        6698        6699        6700        6701        6702
##  644561.281  644561.281  644561.281  644561.281  644561.281  644561.281
##        6703        6704        6705        6706        6707        6708
##  644561.281  644561.281  644561.281  644561.281  644561.281  644561.281
##        6709        6710        6711        6712        6713        6714
##  196758.470 1225046.407 1225046.407 1225046.407 1225046.407 1225046.407
##        6715        6716        6717        6718        6719        6720
## 1225046.407 1225046.407 1225046.407 1225046.407 1225046.407 1225046.407
##        6721        6722        6723        6724        6725        6726
## 1225046.407 1225046.407   14320.288   14320.288   80661.445   80661.445
##        6727        6728        6729        6730        6731        6732
##  224400.619  224400.619  224400.619  224400.619  224400.619  224400.619
##        6733        6734        6735        6736        6737        6738
##   69604.585   69604.585   69604.585  412367.231  169116.321  169116.321
##        6739        6740        6741        6742        6743        6744
##  169116.321  169116.321  169116.321  169116.321  169116.321  169116.321
##        6745        6746        6747        6748        6749        6750
##  434480.950  434480.950  434480.950  213343.759  213343.759  213343.759
##        6751        6752        6753        6754        6755        6756
##  213343.759  213343.759  213343.759  213343.759  213343.759  213343.759
##        6757        6758        6759        6760        6761        6762
##  213343.759  213343.759  384725.082  384725.082  384725.082  384725.082
##        6763        6764        6765        6766        6767        6768
##  384725.082  384725.082  384725.082  384725.082  384725.082  384725.082
##        6769        6770        6771        6772        6773        6774
```

```
##  384725.082  384725.082  384725.082  384725.082  384725.082  163587.892
##        6775        6776        6777        6778        6779        6780
##  163587.892  163587.892  163587.892  163587.892  163587.892  113832.024
##        6781        6782        6783        6784        6785        6786
##  113832.024  113832.024  561634.835  561634.835  561634.835  135945.743
##        6787        6788        6789        6790        6791        6792
##   69604.585   69604.585   -7793.431   -7793.431   -7793.431  290741.776
##        6793        6794        6795        6796        6797        6798
##  290741.776  290741.776  290741.776  290741.776  290741.776  290741.776
##        6799        6800        6801        6802        6803        6804
##  290741.776  290741.776  290741.776  290741.776  290741.776  290741.776
##        6805        6806        6807        6808        6809        6810
##  290741.776  290741.776  406838.801  406838.801  406838.801  406838.801
##        6811        6812        6813        6814        6815        6816
##  147002.602  147002.602  147002.602  147002.602  147002.602  147002.602
##        6817        6818        6819        6820        6821        6822
##  147002.602  147002.602  147002.602  180173.181  180173.181  180173.181
##        6823        6824        6825        6826        6827        6828
##  180173.181  180173.181  180173.181  180173.181  -85191.448  141474.172
##        6829        6830        6831        6832        6833        6834
##  141474.172  141474.172  113832.024  113832.024  268628.057  268628.057
##        6835        6836        6837        6838        6839        6840
##  268628.057  268628.057  268628.057  268628.057  268628.057  268628.057
##        6841        6842        6843        6844        6845        6846
##  268628.057  268628.057  268628.057  268628.057  174644.751  174644.751
##        6847        6848        6849        6850        6851        6852
##  174644.751  417895.661  417895.661  417895.661  417895.661  417895.661
##        6853        6854        6855        6856        6857        6858
##  417895.661  417895.661  417895.661  749601.447  749601.447  749601.447
##        6859        6860        6861        6862        6863        6864
##  749601.447  412367.231  412367.231  412367.231  412367.231  412367.231
##        6865        6866        6867        6868        6869        6870
##  412367.231  771715.166  771715.166  771715.166  771715.166  771715.166
##        6871        6872        6873        6874        6875        6876
##  771715.166  771715.166  771715.166  771715.166  771715.166  771715.166
##        6877        6878        6879        6880        6881        6882
##  771715.166  771715.166  771715.166  771715.166  771715.166  771715.166
##        6883        6884        6885        6886        6887        6888
##  158059.462  158059.462  158059.462  158059.462  158059.462  158059.462
##        6889        6890        6891        6892        6893        6894
##  622447.562  622447.562  622447.562  622447.562  622447.562  622447.562
##        6895        6896        6897        6898        6899        6900
##  622447.562   80661.445   80661.445   80661.445   41962.437   41962.437
##        6901        6902        6903        6904        6905        6906
##   41962.437  296270.206  296270.206  296270.206  296270.206  296270.206
##        6907        6908        6909        6910        6911        6912
```

```
##  296270.206  296270.206  296270.206  506350.537  506350.537  506350.537
##        6913        6914        6915        6916        6917        6918
##  506350.537  506350.537  506350.537  506350.537  506350.537  506350.537
##        6919        6920        6921        6922        6923        6924
##  506350.537   14320.288   41962.437   41962.437  937568.059  937568.059
##        6925        6926        6927        6928        6929        6930
##  937568.059  937568.059  937568.059  937568.059  937568.059  937568.059
##        6931        6932        6933        6934        6935        6936
##  937568.059  937568.059  937568.059  937568.059  937568.059  185701.611
##        6937        6938        6939        6940        6941        6942
##  185701.611  185701.611  185701.611  185701.611  185701.611  185701.611
##        6943        6944        6945        6946        6947        6948
##  185701.611  185701.611  185701.611  185701.611  185701.611  246514.338
##        6949        6950        6951        6952        6953        6954
##  246514.338  246514.338  528464.256  528464.256  528464.256  528464.256
##        6955        6956        6957        6958        6959        6960
##  815942.604  815942.604  815942.604  815942.604  815942.604  815942.604
##        6961        6962        6963        6964        6965        6966
##  815942.604  815942.604  815942.604  130417.313  130417.313  130417.313
##        6967        6968        6969        6970        6971        6972
##  451066.239  451066.239  451066.239  451066.239  451066.239  451066.239
##        6973        6974        6975        6976        6977        6978
##  451066.239  611390.703  611390.703  611390.703  611390.703  611390.703
##        6979        6980        6981        6982        6983        6984
##  611390.703  611390.703  185701.611  185701.611   75133.015   75133.015
##        6985        6986        6987        6988        6989        6990
##   75133.015   75133.015   75133.015   75133.015  -40964.010  174644.751
##        6991        6992        6993        6994        6995        6996
##  174644.751  174644.751  174644.751  213343.759  213343.759  213343.759
##        6997        6998        6999        7000        7001        7002
##  213343.759  213343.759  213343.759  213343.759  213343.759  213343.759
##        7003        7004        7005        7006        7007        7008
##  213343.759  213343.759  213343.759  213343.759  213343.759  213343.759
##        7009        7010        7011        7012        7013        7014
##  572691.694  572691.694  572691.694   -7793.431  263099.627  263099.627
##        7015        7016        7017        7018        7019        7020
##  263099.627  263099.627  727487.728  727487.728  727487.728  727487.728
##        7021        7022        7023        7024        7025        7026
##  533992.686  533992.686  533992.686  533992.686  533992.686  533992.686
##        7027        7028        7029        7030        7031        7032
##  533992.686  533992.686  506350.537  506350.537  506350.537  506350.537
##        7033        7034        7035        7036        7037        7038
##  506350.537  506350.537  506350.537  506350.537  506350.537  506350.537
##        7039        7040        7041        7042        7043        7044
##  506350.537  174644.751  174644.751  174644.751  174644.751  174644.751
##        7045        7046        7047        7048        7049        7050
```

```
##  174644.751  174644.751  174644.751  522935.826  522935.826  522935.826
##        7051        7052        7053        7054        7055        7056
##  522935.826  329440.785  329440.785  329440.785  329440.785  329440.785
##        7057        7058        7059        7060        7061        7062
##  329440.785  329440.785  329440.785  329440.785  329440.785 1003909.216
##        7063        7064        7065        7066        7067        7068
## 1003909.216 1003909.216 1003909.216 1003909.216 1003909.216 1003909.216
##        7069        7070        7071        7072        7073        7074
## 1003909.216 1003909.216 1003909.216 1003909.216 1003909.216 1003909.216
##        7075        7076        7077        7078        7079        7080
## 1003909.216 1003909.216 1003909.216 1003909.216 1003909.216 1003909.216
##        7081        7082        7083        7084        7085        7086
## 1003909.216  738544.587  373668.223  373668.223  373668.223  373668.223
##        7087        7088        7089        7090        7091        7092
##  373668.223  373668.223  373668.223  373668.223  373668.223  373668.223
##        7093        7094        7095        7096        7097        7098
##  373668.223  290741.776  804885.745  804885.745  804885.745  804885.745
##        7099        7100        7101        7102        7103        7104
##  804885.745  804885.745  804885.745   41962.437   41962.437   41962.437
##        7105        7106        7107        7108        7109        7110
##  124888.883  124888.883  124888.883  124888.883   25377.147   25377.147
##        7111        7112        7113        7114        7115        7116
##   25377.147  384725.082  384725.082  384725.082  384725.082  384725.082
##        7117        7118        7119        7120        7121        7122
##  185701.611  185701.611  185701.611  185701.611  185701.611  185701.611
##        7123        7124        7125        7126        7127        7128
##  -46492.440  -46492.440  -46492.440  213343.759  213343.759  158059.462
##        7129        7130        7131        7132        7133        7134
##  158059.462  158059.462  158059.462  158059.462  158059.462  158059.462
##        7135        7136        7137        7138        7139        7140
##  158059.462  158059.462  158059.462  158059.462  158059.462  346026.074
##        7141        7142        7143        7144        7145        7146
##  346026.074  346026.074  346026.074  163587.892  163587.892  163587.892
##        7147        7148        7149        7150        7151        7152
##  163587.892  163587.892  163587.892  163587.892  163587.892   19848.718
##        7153        7154        7155        7156        7157        7158
##   19848.718   19848.718   19848.718   19848.718   19848.718   19848.718
##        7159        7160        7161        7162        7163        7164
##  390253.512  390253.512  522935.826  522935.826  522935.826  522935.826
##        7165        7166        7167        7168        7169        7170
##  522935.826  589276.984  589276.984  589276.984  589276.984  478708.388
##        7171        7172        7173        7174        7175        7176
##  478708.388  478708.388  898869.051  898869.051  898869.051  390253.512
##        7177        7178        7179        7180        7181        7182
##  390253.512  390253.512  390253.512  390253.512  390253.512  390253.512
##        7183        7184        7185        7186        7187        7188
```

```
##  390253.512  390253.512  390253.512  390253.512  390253.512  390253.512
##        7189        7190        7191        7192        7193        7194
##   91718.305   91718.305   91718.305  274156.487  274156.487  274156.487
##        7195        7196        7197        7198        7199        7200
##  274156.487  274156.487  274156.487  274156.487  274156.487  274156.487
##        7201        7202        7203        7204        7205        7206
##  329440.785  329440.785  329440.785  329440.785  257571.198  257571.198
##        7207        7208        7209        7210        7211        7212
##  257571.198  257571.198  257571.198  130417.313  130417.313  130417.313
##        7213        7214        7215        7216        7217        7218
##  130417.313  130417.313  130417.313  307327.065  307327.065  307327.065
##        7219        7220        7221        7222        7223        7224
##  -46492.440  124888.883  124888.883  124888.883   91718.305   91718.305
##        7225        7226        7227        7228        7229        7230
##   91718.305   91718.305  235457.478  235457.478  235457.478  235457.478
##        7231        7232        7233        7234        7235        7236
##  235457.478  240985.908  240985.908  240985.908  240985.908  240985.908
##        7237        7238        7239        7240        7241        7242
##  755129.877  119360.453  119360.453  119360.453  119360.453  119360.453
##        7243        7244        7245        7246        7247        7248
##  119360.453  119360.453  119360.453  119360.453  119360.453  119360.453
##        7249        7250        7251        7252        7253        7254
##  119360.453  119360.453  108303.594  108303.594  108303.594  108303.594
##        7255        7256        7257        7258        7259        7260
##  108303.594  528464.256  528464.256  528464.256  528464.256  528464.256
##        7261        7262        7263        7264        7265        7266
##  528464.256  528464.256  528464.256  528464.256  528464.256  528464.256
##        7267        7268        7269        7270        7271        7272
##  528464.256  528464.256  528464.256  545049.546  545049.546   97246.734
##        7273        7274        7275        7276        7277        7278
##   97246.734   97246.734   97246.734   97246.734   25377.147   25377.147
##        7279        7280        7281        7282        7283        7284
##  130417.313  130417.313  130417.313  130417.313  130417.313  130417.313
##        7285        7286        7287        7288        7289        7290
##  130417.313  130417.313  130417.313  130417.313  130417.313  130417.313
##        7291        7292        7293        7294        7295        7296
##  130417.313  246514.338  -13321.861   14320.288  163587.892  163587.892
##        7297        7298        7299        7300        7301        7302
##  163587.892  163587.892  163587.892  229929.049  838056.323  838056.323
##        7303        7304        7305        7306        7307        7308
##  357082.933  357082.933  357082.933  357082.933  357082.933  357082.933
##        7309        7310        7311        7312        7313        7314
##  357082.933  357082.933  357082.933  357082.933  357082.933  357082.933
##        7315        7316        7317        7318        7319        7320
##    8791.858    8791.858    8791.858    8791.858  113832.024  113832.024
##        7321        7322        7323        7324        7325        7326
```

```
##  113832.024  113832.024  113832.024   19848.718   14320.288   14320.288
##        7327        7328        7329        7330        7331        7332
##   14320.288  213343.759  213343.759  766186.736  766186.736  766186.736
##        7333        7334        7335        7336        7337        7338
##  766186.736  307327.065  307327.065  307327.065  307327.065  307327.065
##        7339        7340        7341        7342        7343        7344
##  307327.065  307327.065   41962.437   41962.437   41962.437   41962.437
##        7345        7346        7347        7348        7349        7350
##   41962.437   41962.437   41962.437  832527.894  832527.894  832527.894
##        7351        7352        7353        7354        7355        7356
##  832527.894  832527.894  832527.894  832527.894  832527.894  832527.894
##        7357        7358        7359        7360        7361        7362
##  832527.894  832527.894  832527.894  832527.894  832527.894  832527.894
##        7363        7364        7365        7366        7367        7368
##  832527.894  832527.894  235457.478  312855.495  312855.495  312855.495
##        7369        7370        7371        7372        7373        7374
##  312855.495  312855.495  312855.495  312855.495  312855.495  312855.495
##        7375        7376        7377        7378        7379        7380
##  312855.495  312855.495  312855.495  312855.495    3263.428    3263.428
##        7381        7382        7383        7384        7385        7386
##  257571.198  240985.908  240985.908  240985.908  240985.908  240985.908
##        7387        7388        7389        7390        7391        7392
##  240985.908  240985.908  240985.908  240985.908  965210.208  965210.208
##        7393        7394        7395        7396        7397        7398
##  965210.208  965210.208  965210.208  965210.208  965210.208  965210.208
##        7399        7400        7401        7402        7403        7404
##  965210.208  965210.208  965210.208  965210.208  627975.992  627975.992
##        7405        7406        7407        7408        7409        7410
##  627975.992  627975.992  627975.992  627975.992  627975.992  627975.992
##        7411        7412        7413        7414        7415        7416
##  627975.992  627975.992  627975.992  627975.992   47490.866  108303.594
##        7417        7418        7419        7420        7421        7422
##  108303.594  108303.594  108303.594  108303.594  185701.611  185701.611
##        7423        7424        7425        7426        7427        7428
##  185701.611  605862.273  605862.273  605862.273  605862.273  605862.273
##        7429        7430        7431        7432        7433        7434
##  346026.074  346026.074  346026.074  113832.024  113832.024  113832.024
##        7435        7436        7437        7438        7439        7440
##  113832.024  113832.024  384725.082  384725.082  384725.082  384725.082
##        7441        7442        7443        7444        7445        7446
##    3263.428  -18850.291  -18850.291  -18850.291   -2265.002   -2265.002
##        7447        7448        7449        7450        7451        7452
##   -2265.002   -2265.002  207815.330  207815.330  207815.330  926511.200
##        7453        7454        7455        7456        7457        7458
##  926511.200  926511.200  926511.200  926511.200  926511.200  926511.200
##        7459        7460        7461        7462        7463        7464
```

```
##   64076.156   19848.718   19848.718  860170.042  860170.042  860170.042
##        7465        7466        7467        7468        7469        7470
##  860170.042  860170.042  860170.042  860170.042  860170.042  860170.042
##        7471        7472        7473        7474        7475        7476
##  860170.042  860170.042  860170.042  860170.042  860170.042   19848.718
##        7477        7478        7479        7480        7481        7482
##   19848.718   19848.718   19848.718   19848.718   19848.718  583748.554
##        7483        7484        7485        7486        7487        7488
##  583748.554  583748.554   69604.585   69604.585   64076.156  528464.256
##        7489        7490        7491        7492        7493        7494
##  528464.256  528464.256  484236.818  484236.818  484236.818  484236.818
##        7495        7496        7497        7498        7499        7500
##  484236.818  484236.818  484236.818  484236.818  484236.818  484236.818
##        7501        7502        7503        7504        7505        7506
##  484236.818  484236.818  484236.818  484236.818  484236.818 1075778.803
##        7507        7508        7509        7510        7511        7512
## 1075778.803 1075778.803 1075778.803 1075778.803  838056.323  838056.323
##        7513        7514        7515        7516        7517        7518
##  838056.323  838056.323  428952.520  428952.520  428952.520  428952.520
##        7519        7520        7521        7522        7523        7524
##  428952.520  428952.520  428952.520  428952.520  428952.520  428952.520
##        7525        7526        7527        7528        7529        7530
##  428952.520  428952.520   86189.875   86189.875  423424.091  423424.091
##        7531        7532        7533        7534        7535        7536
##  423424.091  423424.091  423424.091  423424.091  141474.172  141474.172
##        7537        7538        7539        7540        7541        7542
##  141474.172  169116.321  169116.321  169116.321  169116.321  169116.321
##        7543        7544        7545        7546        7547        7548
##  279684.917  279684.917  279684.917  279684.917  279684.917  124888.883
##        7549        7550        7551        7552        7553        7554
##  124888.883  191230.040  384725.082  -85191.448  163587.892  163587.892
##        7555        7556        7557        7558        7559        7560
##  163587.892  163587.892  169116.321  699845.579  699845.579  699845.579
##        7561        7562        7563        7564        7565        7566
##  699845.579  699845.579  699845.579  699845.579  699845.579  699845.579
##        7567        7568        7569        7570        7571        7572
##  699845.579 1081307.233 1081307.233 1081307.233 1081307.233 1081307.233
##        7573        7574        7575        7576        7577        7578
##  301798.636  301798.636  301798.636  301798.636  301798.636  301798.636
##        7579        7580        7581        7582        7583        7584
##  301798.636 1059193.514 1059193.514 1059193.514 1059193.514 1059193.514
##        7585        7586        7587        7588        7589        7590
## 1059193.514 1059193.514  158059.462  158059.462  158059.462  158059.462
##        7591        7592        7593        7594        7595        7596
##  334969.214  334969.214  334969.214  334969.214  334969.214  -29907.150
##        7597        7598        7599        7600        7601        7602
```

```
##  163587.892  163587.892  522935.826  522935.826  522935.826  522935.826
##        7603        7604        7605        7606        7607        7608
##  522935.826  522935.826  522935.826  522935.826  522935.826  522935.826
##        7609        7610        7611        7612        7613        7614
##  522935.826  522935.826  522935.826  185701.611  185701.611  185701.611
##        7615        7616        7617        7618        7619        7620
##  185701.611  185701.611  318383.925  185701.611  185701.611  185701.611
##        7621        7622        7623        7624        7625        7626
##  185701.611  185701.611   25377.147  323912.355  323912.355  323912.355
##        7627        7628        7629        7630        7631        7632
##  323912.355  517407.397  517407.397  517407.397  517407.397  517407.397
##        7633        7634        7635        7636        7637        7638
##  517407.397  517407.397  180173.181  180173.181  180173.181  489765.248
##        7639        7640        7641        7642        7643        7644
##  489765.248  489765.248  489765.248  489765.248  489765.248  489765.248
##        7645        7646        7647        7648        7649        7650
##  489765.248  489765.248  489765.248  489765.248  489765.248  489765.248
##        7651        7652        7653        7654        7655        7656
##  257571.198  257571.198  257571.198  257571.198  257571.198  257571.198
##        7657        7658        7659        7660        7661        7662
##  954153.348  954153.348  954153.348  954153.348  954153.348  954153.348
##        7663        7664        7665        7666        7667        7668
##  954153.348  954153.348  954153.348  954153.348  954153.348  954153.348
##        7669        7670        7671        7672        7673        7674
##  705374.009  705374.009  705374.009  705374.009  705374.009  705374.009
##        7675        7676        7677        7678        7679        7680
##  705374.009  705374.009  705374.009  705374.009  257571.198  257571.198
##        7681        7682        7683        7684        7685        7686
##  257571.198  257571.198  257571.198  257571.198  257571.198  257571.198
##        7687        7688        7689        7690        7691        7692
##  257571.198  257571.198  257571.198  141474.172  141474.172  141474.172
##        7693        7694        7695        7696        7697        7698
##  141474.172  141474.172  141474.172  141474.172  495293.678  495293.678
##        7699        7700        7701        7702        7703        7704
##  495293.678  495293.678  495293.678  495293.678  954153.348  954153.348
##        7705        7706        7707        7708        7709        7710
##  954153.348  954153.348  954153.348  954153.348  954153.348  954153.348
##        7711        7712        7713        7714        7715        7716
##  954153.348  340497.644  340497.644  340497.644  528464.256  528464.256
##        7717        7718        7719        7720        7721        7722
##  528464.256  528464.256  528464.256  528464.256  528464.256  -85191.448
##        7723        7724        7725        7726        7727        7728
##   97246.734   97246.734   97246.734    8791.858    8791.858    8791.858
##        7729        7730        7731        7732        7733        7734
##    8791.858    8791.858    8791.858    8791.858    8791.858    8791.858
##        7735        7736        7737        7738        7739        7740
```

```
##    8791.858  290741.776  290741.776  290741.776  290741.776  290741.776
##        7741        7742        7743        7744        7745        7746
##  290741.776  290741.776  290741.776  290741.776  290741.776  268628.057
##        7747        7748        7749        7750        7751        7752
##  268628.057  268628.057  268628.057  268628.057  268628.057  268628.057
##        7753        7754        7755        7756        7757        7758
##  268628.057  351554.504  351554.504  351554.504  351554.504  351554.504
##        7759        7760        7761        7762        7763        7764
##  351554.504  351554.504  351554.504  826999.464  826999.464  826999.464
##        7765        7766        7767        7768        7769        7770
##   80661.445  578220.124  578220.124  578220.124  578220.124  578220.124
##        7771        7772        7773        7774        7775        7776
##  489765.248  489765.248  147002.602  147002.602  147002.602  147002.602
##        7777        7778        7779        7780        7781        7782
##  301798.636  428952.520  428952.520  428952.520  428952.520  428952.520
##        7783        7784        7785        7786        7787        7788
##  428952.520  428952.520  428952.520  428952.520  428952.520  428952.520
##        7789        7790        7791        7792        7793        7794
##  428952.520  428952.520  428952.520   -7793.431   -7793.431   -7793.431
##        7795        7796        7797        7798        7799        7800
##  -29907.150  -29907.150   25377.147   25377.147   25377.147   25377.147
##        7801        7802        7803        7804        7805        7806
##   25377.147  240985.908  240985.908  240985.908  -57549.299  -57549.299
##        7807        7808        7809        7810        7811        7812
##  -57549.299   25377.147  268628.057  268628.057  268628.057  268628.057
##        7813        7814        7815        7816        7817        7818
##  268628.057  268628.057  268628.057  268628.057  268628.057   58547.726
##        7819        7820        7821        7822        7823        7824
##  102775.164  102775.164  102775.164  102775.164  616919.133  616919.133
##        7825        7826        7827        7828        7829        7830
##   30905.577   53019.296   53019.296   53019.296  279684.917  279684.917
##        7831        7832        7833        7834        7835        7836
##  152531.032  152531.032   86189.875   86189.875   86189.875   86189.875
##        7837        7838        7839        7840        7841        7842
##   86189.875   86189.875   86189.875   86189.875   86189.875   86189.875
##        7843        7844        7845        7846        7847        7848
##   86189.875   86189.875  -85191.448  340497.644  340497.644   14320.288
##        7849        7850        7851        7852        7853        7854
##   64076.156   64076.156  683260.290  683260.290  683260.290  655618.141
##        7855        7856        7857        7858        7859        7860
##  655618.141  655618.141  655618.141  655618.141  578220.124  578220.124
##        7861        7862        7863        7864        7865        7866
##  578220.124  578220.124  578220.124  578220.124  578220.124  578220.124
##        7867        7868        7869        7870        7871        7872
##  578220.124  578220.124  578220.124  578220.124  578220.124  -18850.291
##        7873        7874        7875        7876        7877        7878
```

```
##  694317.149  627975.992  627975.992  627975.992  627975.992  307327.065
##        7879        7880        7881        7882        7883        7884
##  307327.065  307327.065  307327.065  307327.065  307327.065   14320.288
##        7885        7886        7887        7888        7889        7890
##   14320.288  -29907.150  473179.959  473179.959  240985.908  240985.908
##        7891        7892        7893        7894        7895        7896
##  240985.908  240985.908  240985.908  240985.908  240985.908  240985.908
##        7897        7898        7899        7900        7901        7902
##  240985.908  240985.908  240985.908  240985.908  727487.728   36434.007
##        7903        7904        7905        7906        7907        7908
##   36434.007  318383.925  318383.925  318383.925   36434.007   36434.007
##        7909        7910        7911        7912        7913        7914
##   36434.007  240985.908  240985.908  240985.908  240985.908  240985.908
##        7915        7916        7917        7918        7919        7920
##  240985.908  240985.908   91718.305   91718.305   91718.305   91718.305
##        7921        7922        7923        7924        7925        7926
##   91718.305   91718.305  484236.818  484236.818  484236.818  484236.818
##        7927        7928        7929        7930        7931        7932
##  484236.818  484236.818  484236.818  484236.818  484236.818  484236.818
##        7933        7934        7935        7936        7937        7938
##  484236.818  484236.818   -2265.002   -2265.002   30905.577   30905.577
##        7939        7940        7941        7942        7943        7944
##  -29907.150  296270.206  296270.206  296270.206  296270.206  296270.206
##        7945        7946        7947        7948        7949        7950
##  296270.206  296270.206  -74134.589  -74134.589   36434.007   36434.007
##        7951        7952        7953        7954        7955        7956
##   14320.288  318383.925  318383.925  318383.925  318383.925  318383.925
##        7957        7958        7959        7960        7961        7962
##  318383.925  318383.925  323912.355  323912.355  323912.355  473179.959
##        7963        7964        7965        7966        7967        7968
##  473179.959   80661.445   69604.585   30905.577   30905.577   97246.734
##        7969        7970        7971        7972        7973        7974
##   97246.734  821471.034  821471.034  412367.231  412367.231  412367.231
##        7975        7976        7977        7978        7979        7980
##  412367.231  412367.231  412367.231  412367.231  412367.231  412367.231
##        7981        7982        7983        7984        7985        7986
##  412367.231  412367.231  412367.231   36434.007   36434.007   36434.007
##        7987        7988        7989        7990        7991        7992
##  373668.223  373668.223  373668.223  622447.562  -63077.729  -63077.729
##        7993        7994        7995        7996        7997        7998
##  147002.602  147002.602  -18850.291  -18850.291  334969.214  334969.214
##        7999        8000        8001        8002        8003        8004
##  334969.214  141474.172  141474.172  141474.172  130417.313  130417.313
##        8005        8006        8007        8008        8009        8010
##  130417.313  130417.313  274156.487  274156.487  274156.487  274156.487
##        8011        8012        8013        8014        8015        8016
```

```
##  274156.487  274156.487   47490.866   41962.437  174644.751  174644.751
##        8017        8018        8019        8020        8021        8022
##  174644.751  174644.751  174644.751  174644.751  174644.751  174644.751
##        8023        8024        8025        8026        8027        8028
##  174644.751  174644.751  174644.751  174644.751  174644.751  174644.751
##        8029        8030        8031        8032        8033        8034
##  224400.619  224400.619  224400.619  224400.619  224400.619   58547.726
##        8035        8036        8037        8038        8039        8040
##   58547.726   58547.726  218872.189  218872.189  218872.189  218872.189
##        8041        8042        8043        8044        8045        8046
##  218872.189  218872.189  218872.189   19848.718  428952.520  428952.520
##        8047        8048        8049        8050        8051        8052
##  428952.520  428952.520  710902.439  196758.470  196758.470  196758.470
##        8053        8054        8055        8056        8057        8058
##  196758.470  196758.470  334969.214  334969.214  334969.214  334969.214
##        8059        8060        8061        8062        8063        8064
##  334969.214  334969.214  716430.868   64076.156   64076.156  135945.743
##        8065        8066        8067        8068        8069        8070
##  135945.743  135945.743  135945.743  135945.743  323912.355  323912.355
##        8071        8072        8073        8074        8075        8076
##  323912.355  323912.355  323912.355  323912.355  323912.355  323912.355
##        8077        8078        8079        8080        8081        8082
##  323912.355  323912.355   80661.445   80661.445   80661.445   80661.445
##        8083        8084        8085        8086        8087        8088
##  252042.768  252042.768  252042.768  252042.768  252042.768  252042.768
##        8089        8090        8091        8092        8093        8094
##  252042.768  252042.768  252042.768  252042.768  252042.768  517407.397
##        8095        8096        8097        8098        8099        8100
##  517407.397  517407.397  517407.397  517407.397  517407.397  517407.397
##        8101        8102        8103        8104        8105        8106
##  517407.397  517407.397  517407.397  517407.397  517407.397  517407.397
##        8107        8108        8109        8110        8111        8112
##  716430.868  716430.868  716430.868  716430.868  716430.868  716430.868
##        8113        8114        8115        8116        8117        8118
##  716430.868  716430.868  716430.868  716430.868  716430.868  716430.868
##        8119        8120        8121        8122        8123        8124
##  716430.868  716430.868  716430.868  716430.868  716430.868  605862.273
##        8125        8126        8127        8128        8129        8130
##  605862.273  605862.273  605862.273  605862.273  605862.273  605862.273
##        8131        8132        8133        8134        8135        8136
##  605862.273  605862.273  605862.273  605862.273  605862.273  605862.273
##        8137        8138        8139        8140        8141        8142
##  605862.273   36434.007   36434.007   36434.007  224400.619  224400.619
##        8143        8144        8145        8146        8147        8148
##  224400.619  224400.619  224400.619   80661.445   97246.734   97246.734
##        8149        8150        8151        8152        8153        8154
```

```
##   25377.147   25377.147  705374.009  705374.009  705374.009  705374.009
##        8155        8156        8157        8158        8159        8160
##  705374.009  705374.009  705374.009  -52020.869   69604.585   69604.585
##        8161        8162        8163        8164        8165        8166
##  733016.158  733016.158  257571.198  257571.198  257571.198  257571.198
##        8167        8168        8169        8170        8171        8172
##  257571.198  290741.776  290741.776  290741.776  290741.776  290741.776
##        8173        8174        8175        8176        8177        8178
##  290741.776  290741.776  290741.776  290741.776  290741.776  290741.776
##        8179        8180        8181        8182        8183        8184
##  263099.627  263099.627  263099.627  263099.627  196758.470  196758.470
##        8185        8186        8187        8188        8189        8190
##  296270.206  296270.206  296270.206  296270.206  296270.206  296270.206
##        8191        8192        8193        8194        8195        8196
##  296270.206  296270.206   75133.015   75133.015   75133.015   64076.156
##        8197        8198        8199        8200        8201        8202
##   64076.156  351554.504  351554.504  351554.504  351554.504  351554.504
##        8203        8204        8205        8206        8207        8208
##  351554.504  351554.504  351554.504  351554.504  351554.504  202286.900
##        8209        8210        8211        8212        8213        8214
##  202286.900  202286.900  202286.900  351554.504  351554.504  351554.504
##        8215        8216        8217        8218        8219        8220
##  351554.504  351554.504  351554.504  351554.504  926511.200  926511.200
##        8221        8222        8223        8224        8225        8226
##  926511.200  926511.200  926511.200  926511.200  926511.200  849113.183
##        8227        8228        8229        8230        8231        8232
##  849113.183  849113.183  849113.183  849113.183  849113.183  849113.183
##        8233        8234        8235        8236        8237        8238
##  849113.183  849113.183  849113.183  849113.183  849113.183  849113.183
##        8239        8240        8241        8242        8243        8244
##  849113.183  849113.183  849113.183   41962.437   41962.437  605862.273
##        8245        8246        8247        8248        8249        8250
##  605862.273  605862.273  605862.273  290741.776  290741.776  290741.776
##        8251        8252        8253        8254        8255        8256
##  290741.776  141474.172  141474.172  141474.172  141474.172  141474.172
##        8257        8258        8259        8260        8261        8262
##  141474.172  141474.172   69604.585   69604.585   41962.437   91718.305
##        8263        8264        8265        8266        8267        8268
##  390253.512  390253.512  390253.512  390253.512  390253.512  390253.512
##        8269        8270        8271        8272        8273        8274
##  390253.512  390253.512  390253.512  390253.512  312855.495  312855.495
##        8275        8276        8277        8278        8279        8280
##  312855.495  312855.495  312855.495  312855.495  312855.495  312855.495
##        8281        8282        8283        8284        8285        8286
##  312855.495  312855.495  312855.495  312855.495  312855.495   64076.156
##        8287        8288        8289        8290        8291        8292
```

```
##   36434.007   36434.007   36434.007   36434.007   36434.007   36434.007
##        8293        8294        8295        8296        8297        8298
##   36434.007   91718.305   91718.305   91718.305   91718.305  473179.959
##        8299        8300        8301        8302        8303        8304
##  473179.959  473179.959  473179.959  473179.959   80661.445   80661.445
##        8305        8306        8307        8308        8309        8310
##  639032.852  639032.852  639032.852  639032.852  639032.852  854641.613
##        8311        8312        8313        8314        8315        8316
##  854641.613  854641.613  854641.613  854641.613  854641.613  854641.613
##        8317        8318        8319        8320        8321        8322
##  854641.613  854641.613  854641.613  854641.613  854641.613  854641.613
##        8323        8324        8325        8326        8327        8328
##   47490.866   47490.866   47490.866   -7793.431   -7793.431   -7793.431
##        8329        8330        8331        8332        8333        8334
##  -24378.721  -24378.721  -24378.721  -24378.721   47490.866   47490.866
##        8335        8336        8337        8338        8339        8340
##   47490.866  959681.778  959681.778  959681.778  959681.778  959681.778
##        8341        8342        8343        8344        8345        8346
##  959681.778  959681.778  959681.778  959681.778  959681.778  959681.778
##        8347        8348        8349        8350        8351        8352
##  959681.778  959681.778  959681.778  959681.778  655618.141  655618.141
##        8353        8354        8355        8356        8357        8358
##  655618.141  655618.141  655618.141  655618.141  655618.141  655618.141
##        8359        8360        8361        8362        8363        8364
##  655618.141  655618.141  655618.141  655618.141  357082.933  357082.933
##        8365        8366        8367        8368        8369        8370
##  357082.933   36434.007  329440.785  329440.785  329440.785  329440.785
##        8371        8372        8373        8374        8375        8376
##  329440.785  329440.785  329440.785  169116.321  169116.321  169116.321
##        8377        8378        8379        8380        8381        8382
##  169116.321  169116.321  644561.281  644561.281  644561.281  644561.281
##        8383        8384        8385        8386        8387        8388
##  644561.281  644561.281  290741.776  290741.776  290741.776  290741.776
##        8389        8390        8391        8392        8393        8394
##  290741.776  611390.703  611390.703  611390.703  611390.703  611390.703
##        8395        8396        8397        8398        8399        8400
##  611390.703  611390.703  611390.703  611390.703  611390.703  511878.967
##        8401        8402        8403        8404        8405        8406
##  511878.967  511878.967  511878.967  511878.967   97246.734   97246.734
##        8407        8408        8409        8410        8411        8412
##   97246.734    8791.858    8791.858    8791.858    8791.858    8791.858
##        8413        8414        8415        8416        8417        8418
##    8791.858    8791.858    8791.858  390253.512  390253.512  390253.512
##        8419        8420        8421        8422        8423        8424
##  390253.512  390253.512  390253.512  390253.512  390253.512  390253.512
##        8425        8426        8427        8428        8429        8430
```

```
##  390253.512   36434.007  274156.487  274156.487  274156.487  274156.487
##        8431        8432        8433        8434        8435        8436
##  152531.032  152531.032  152531.032  152531.032  152531.032  379196.652
##        8437        8438        8439        8440        8441        8442
##  379196.652  379196.652  379196.652  379196.652  379196.652  379196.652
##        8443        8444        8445        8446        8447        8448
##  379196.652  379196.652   41962.437  694317.149  694317.149  694317.149
##        8449        8450        8451        8452        8453        8454
##  694317.149  694317.149  694317.149  694317.149  694317.149  694317.149
##        8455        8456        8457        8458        8459        8460
##  694317.149  191230.040  191230.040   14320.288   69604.585   69604.585
##        8461        8462        8463        8464        8465        8466
##   47490.866  733016.158   53019.296   53019.296   53019.296   64076.156
##        8467        8468        8469        8470        8471        8472
##   64076.156  263099.627  263099.627  213343.759  213343.759  252042.768
##        8473        8474        8475        8476        8477        8478
##  252042.768  252042.768  174644.751  174644.751  522935.826  522935.826
##        8479        8480        8481        8482        8483        8484
##  522935.826  522935.826  522935.826  522935.826  522935.826  522935.826
##        8485        8486        8487        8488        8489        8490
##  522935.826  522935.826  229929.049   75133.015   75133.015  616919.133
##        8491        8492        8493        8494        8495        8496
##  616919.133  616919.133  616919.133  616919.133  616919.133  616919.133
##        8497        8498        8499        8500        8501        8502
##  616919.133  616919.133   47490.866   47490.866   47490.866   47490.866
##        8503        8504        8505        8506        8507        8508
##  362611.363  362611.363  362611.363  362611.363  362611.363  362611.363
##        8509        8510        8511        8512        8513        8514
##  323912.355  323912.355  323912.355  323912.355  323912.355  323912.355
##        8515        8516        8517        8518        8519        8520
##  323912.355  323912.355  323912.355  323912.355  240985.908  240985.908
##        8521        8522        8523        8524        8525        8526
##  240985.908  240985.908  240985.908  240985.908  240985.908  240985.908
##        8527        8528        8529        8530        8531        8532
##  357082.933  655618.141  655618.141  655618.141  655618.141  655618.141
##        8533        8534        8535        8536        8537        8538
##  655618.141  655618.141  655618.141  655618.141  655618.141  655618.141
##        8539        8540        8541        8542        8543        8544
##  655618.141  655618.141  655618.141  655618.141  655618.141  655618.141
##        8545        8546        8547        8548        8549        8550
##  655618.141  655618.141   36434.007   36434.007   36434.007   36434.007
##        8551        8552        8553        8554        8555        8556
##   36434.007  218872.189   41962.437   64076.156   64076.156  229929.049
##        8557        8558        8559        8560        8561        8562
##  229929.049  229929.049  229929.049  229929.049  229929.049  229929.049
##        8563        8564        8565        8566        8567        8568
```

```
##  229929.049  229929.049  229929.049  229929.049  943096.489  943096.489
##        8569        8570        8571        8572        8573        8574
##  943096.489  943096.489  943096.489  943096.489  943096.489   53019.296
##        8575        8576        8577        8578        8579        8580
##   53019.296    3263.428   64076.156   64076.156   64076.156   64076.156
##        8581        8582        8583        8584        8585        8586
##  428952.520  428952.520  428952.520  428952.520  428952.520  428952.520
##        8587        8588        8589        8590        8591        8592
##  428952.520  428952.520  428952.520  428952.520  428952.520  362611.363
##        8593        8594        8595        8596        8597        8598
##  362611.363  362611.363  362611.363  362611.363  362611.363  362611.363
##        8599        8600        8601        8602        8603        8604
##  362611.363  362611.363  362611.363  362611.363  362611.363  362611.363
##        8605        8606        8607        8608        8609        8610
##  202286.900  202286.900  202286.900  202286.900  202286.900  202286.900
##        8611        8612        8613        8614        8615        8616
##  202286.900   19848.718   19848.718  550577.975  550577.975  550577.975
##        8617        8618        8619        8620        8621        8622
##  550577.975  550577.975  550577.975  550577.975  550577.975  550577.975
##        8623        8624        8625        8626        8627        8628
##  550577.975  550577.975  550577.975  550577.975  340497.644  340497.644
##        8629        8630        8631        8632        8633        8634
##  340497.644  340497.644  340497.644  340497.644  340497.644  340497.644
##        8635        8636        8637        8638        8639        8640
##  340497.644   69604.585   69604.585  340497.644  340497.644  340497.644
##        8641        8642        8643        8644        8645        8646
##  340497.644  406838.801  406838.801  406838.801  406838.801  406838.801
##        8647        8648        8649        8650        8651        8652
##  406838.801  406838.801  406838.801  406838.801  406838.801  406838.801
##        8653        8654        8655        8656        8657        8658
##  992852.357  992852.357  992852.357  992852.357  992852.357  992852.357
##        8659        8660        8661        8662        8663        8664
##  992852.357  351554.504  113832.024  423424.091  423424.091  423424.091
##        8665        8666        8667        8668        8669        8670
##  423424.091  423424.091  423424.091  423424.091   47490.866   47490.866
##        8671        8672        8673        8674        8675        8676
##  572691.694  572691.694  572691.694  572691.694  572691.694  572691.694
##        8677        8678        8679        8680        8681        8682
##  572691.694  572691.694  572691.694  572691.694  572691.694  572691.694
##        8683        8684        8685        8686        8687        8688
##  572691.694   53019.296  688788.720  688788.720  688788.720  384725.082
##        8689        8690        8691        8692        8693        8694
##  384725.082  384725.082  384725.082  384725.082  384725.082  384725.082
##        8695        8696        8697        8698        8699        8700
##  384725.082  384725.082  384725.082  384725.082 -29907.150  268628.057
##        8701        8702        8703        8704        8705        8706
```

```
##  268628.057  268628.057  268628.057  268628.057  268628.057  268628.057
##        8707        8708        8709        8710        8711        8712
##  268628.057  268628.057  268628.057  268628.057  268628.057  268628.057
##        8713        8714        8715        8716        8717        8718
##  213343.759  213343.759  108303.594  108303.594  108303.594  108303.594
##        8719        8720        8721        8722        8723        8724
##  108303.594 1147648.390 1147648.390 1147648.390  180173.181  180173.181
##        8725        8726        8727        8728        8729        8730
##  180173.181  180173.181  180173.181  180173.181  180173.181  390253.512
##        8731        8732        8733        8734        8735        8736
##  390253.512  390253.512  390253.512  390253.512  390253.512   69604.585
##        8737        8738        8739        8740        8741        8742
##   69604.585  -46492.440  -46492.440  -46492.440  180173.181  180173.181
##        8743        8744        8745        8746        8747        8748
##  180173.181  180173.181  180173.181  180173.181  180173.181   75133.015
##        8749        8750        8751        8752        8753        8754
##   75133.015   75133.015   75133.015  141474.172  141474.172  141474.172
##        8755        8756        8757        8758        8759        8760
##  323912.355  323912.355  323912.355  323912.355  323912.355  323912.355
##        8761        8762        8763        8764        8765        8766
##  323912.355  174644.751  174644.751  174644.751  174644.751  174644.751
##        8767        8768        8769        8770        8771        8772
##  174644.751  174644.751  174644.751  174644.751   19848.718   19848.718
##        8773        8774        8775        8776        8777        8778
##   19848.718  528464.256  528464.256  528464.256  528464.256  528464.256
##        8779        8780        8781        8782        8783        8784
##  528464.256  528464.256  528464.256  528464.256  528464.256  528464.256
##        8785        8786        8787        8788        8789        8790
##  528464.256  528464.256  528464.256  528464.256  462123.099  462123.099
##        8791        8792        8793        8794        8795        8796
##  462123.099  462123.099  462123.099  462123.099  462123.099  462123.099
##        8797        8798        8799        8800        8801        8802
##  627975.992  627975.992  627975.992  -13321.861  108303.594  108303.594
##        8803        8804        8805        8806        8807        8808
##  113832.024  113832.024  113832.024  113832.024  108303.594  108303.594
##        8809        8810        8811        8812        8813        8814
##  108303.594  108303.594  108303.594  108303.594  202286.900  202286.900
##        8815        8816        8817        8818        8819        8820
##  202286.900  202286.900  202286.900  379196.652  379196.652  379196.652
##        8821        8822        8823        8824        8825        8826
##  379196.652  379196.652  379196.652  379196.652  379196.652  379196.652
##        8827        8828        8829        8830        8831        8832
##  379196.652  379196.652  379196.652  379196.652  379196.652  379196.652
##        8833        8834        8835        8836        8837        8838
##  135945.743  135945.743  329440.785  329440.785  158059.462  158059.462
##        8839        8840        8841        8842        8843        8844
```

```
##  158059.462  135945.743  135945.743  583748.554  583748.554  583748.554
##        8845        8846        8847        8848        8849        8850
##  583748.554  583748.554  583748.554  357082.933  357082.933  710902.439
##        8851        8852        8853        8854        8855        8856
##  710902.439  710902.439  710902.439  710902.439  710902.439  710902.439
##        8857        8858        8859        8860        8861        8862
##  710902.439  799357.315  799357.315  799357.315  799357.315  799357.315
##        8863        8864        8865        8866        8867        8868
##  799357.315  799357.315  799357.315  799357.315  373668.223  373668.223
##        8869        8870        8871        8872        8873        8874
##  191230.040  191230.040  318383.925  318383.925  318383.925  318383.925
##        8875        8876        8877        8878        8879        8880
##  318383.925  318383.925  318383.925  318383.925  318383.925  318383.925
##        8881        8882        8883        8884        8885        8886
##  318383.925  318383.925  318383.925  318383.925  318383.925  318383.925
##        8887        8888        8889        8890        8891        8892
##   75133.015   75133.015   75133.015   75133.015  412367.231  412367.231
##        8893        8894        8895        8896        8897        8898
##  412367.231  412367.231  412367.231  412367.231  412367.231   75133.015
##        8899        8900        8901        8902        8903        8904
##  163587.892  163587.892  163587.892  163587.892  163587.892  163587.892
##        8905        8906        8907        8908        8909        8910
##  163587.892  163587.892  163587.892  163587.892  163587.892  473179.959
##        8911        8912        8913        8914        8915        8916
##  473179.959  473179.959  473179.959  473179.959   80661.445   80661.445
##        8917        8918        8919        8920        8921        8922
##   80661.445   80661.445  113832.024  113832.024  113832.024  489765.248
##        8923        8924        8925        8926        8927        8928
##  489765.248  489765.248  489765.248  489765.248  489765.248  489765.248
##        8929        8930        8931        8932        8933        8934
##  489765.248  489765.248  489765.248  489765.248   19848.718   19848.718
##        8935        8936        8937        8938        8939        8940
##  -13321.861  196758.470  196758.470  196758.470  196758.470  135945.743
##        8941        8942        8943        8944        8945        8946
##  135945.743  135945.743  135945.743  135945.743  135945.743  135945.743
##        8947        8948        8949        8950        8951        8952
##  135945.743  135945.743  -85191.448   69604.585   97246.734   97246.734
##        8953        8954        8955        8956        8957        8958
##  782772.026  782772.026  782772.026  782772.026  782772.026  782772.026
##        8959        8960        8961        8962        8963        8964
##  782772.026  207815.330  207815.330  207815.330  207815.330  207815.330
##        8965        8966        8967        8968        8969        8970
##  207815.330  207815.330  207815.330  207815.330 1131063.101 1131063.101
##        8971        8972        8973        8974        8975        8976
## 1131063.101 1131063.101 1131063.101 1131063.101 1131063.101 1131063.101
##        8977        8978        8979        8980        8981        8982
```

```
## 1131063.101 1131063.101 1131063.101 1131063.101 1131063.101 1131063.101
##        8983        8984        8985        8986        8987        8988
## 1131063.101 1131063.101  207815.330  207815.330  207815.330  207815.330
##        8989        8990        8991        8992        8993        8994
##  207815.330  434480.950  434480.950  434480.950  434480.950  644561.281
##        8995        8996        8997        8998        8999        9000
##  644561.281  644561.281  644561.281  644561.281  644561.281  882283.761
##        9001        9002        9003        9004        9005        9006
##  882283.761  882283.761  882283.761  882283.761  882283.761  882283.761
##        9007        9008        9009        9010        9011        9012
##  882283.761  882283.761  882283.761  882283.761  246514.338  246514.338
##        9013        9014        9015        9016        9017        9018
##  246514.338  246514.338 1009437.646 1009437.646 1009437.646 1009437.646
##        9019        9020        9021        9022        9023        9024
## 1009437.646 1009437.646 1009437.646  207815.330  207815.330  207815.330
##        9025        9026        9027        9028        9029        9030
##  207815.330  207815.330  207815.330  207815.330  207815.330  229929.049
##        9031        9032        9033        9034        9035        9036
##  229929.049  229929.049  229929.049  229929.049  229929.049  229929.049
##        9037        9038        9039        9040        9041        9042
##  229929.049  229929.049  229929.049  229929.049  229929.049  229929.049
##        9043        9044        9045        9046        9047        9048
##  229929.049  572691.694  572691.694  572691.694  572691.694  572691.694
##        9049        9050        9051        9052        9053        9054
##  572691.694  572691.694  572691.694  572691.694  572691.694  572691.694
##        9055        9056        9057        9058        9059        9060
##  572691.694  572691.694  572691.694  572691.694  334969.214  334969.214
##        9061        9062        9063        9064        9065        9066
##  334969.214  334969.214  334969.214  710902.439  710902.439  710902.439
##        9067        9068        9069        9070        9071        9072
##  710902.439  710902.439  710902.439  710902.439  710902.439  710902.439
##        9073        9074        9075        9076        9077        9078
##  710902.439  710902.439  710902.439  710902.439  710902.439  965210.208
##        9079        9080        9081        9082        9083        9084
##  965210.208  965210.208  965210.208  965210.208  965210.208  965210.208
##        9085        9086        9087        9088        9089        9090
##  965210.208  -29907.150  169116.321  169116.321  169116.321  174644.751
##        9091        9092        9093        9094        9095        9096
##  174644.751  174644.751  174644.751  174644.751  174644.751  174644.751
##        9097        9098        9099        9100        9101        9102
##  174644.751  174644.751  174644.751  263099.627  263099.627  263099.627
##        9103        9104        9105        9106        9107        9108
##  263099.627  263099.627  263099.627  263099.627  263099.627  229929.049
##        9109        9110        9111        9112        9113        9114
##  229929.049  710902.439  710902.439  710902.439  710902.439   19848.718
##        9115        9116        9117        9118        9119        9120
```

```
##  434480.950  434480.950  434480.950  434480.950  434480.950  434480.950
##        9121        9122        9123        9124        9125        9126
##  434480.950  434480.950   75133.015  296270.206  296270.206   53019.296
##        9127        9128        9129        9130        9131        9132
##  312855.495  312855.495  312855.495  312855.495  312855.495  312855.495
##        9133        9134        9135        9136        9137        9138
##  312855.495  312855.495  312855.495  312855.495  268628.057  268628.057
##        9139        9140        9141        9142        9143        9144
##  268628.057  268628.057  268628.057  268628.057  268628.057  268628.057
##        9145        9146        9147        9148        9149        9150
##  268628.057  268628.057  268628.057  268628.057  268628.057  268628.057
##        9151        9152        9153        9154        9155        9156
##   91718.305   64076.156  -85191.448  135945.743  135945.743  550577.975
##        9157        9158        9159        9160        9161        9162
##  550577.975  550577.975  550577.975  550577.975  550577.975  550577.975
##        9163        9164        9165        9166        9167        9168
##  191230.040  191230.040 1059193.514 1059193.514 1059193.514 1059193.514
##        9169        9170        9171        9172        9173        9174
## 1059193.514 1059193.514 1059193.514 1059193.514 1059193.514 1059193.514
##        9175        9176        9177        9178        9179        9180
## 1059193.514 1059193.514  395781.942  395781.942  395781.942  395781.942
##        9181        9182        9183        9184        9185        9186
##  395781.942  395781.942  395781.942  395781.942  395781.942  395781.942
##        9187        9188        9189        9190        9191        9192
##  395781.942  395781.942  395781.942  395781.942  274156.487  274156.487
##        9193        9194        9195        9196        9197        9198
##  274156.487  274156.487  583748.554  583748.554  583748.554  583748.554
##        9199        9200        9201        9202        9203        9204
##  583748.554  583748.554  583748.554  583748.554  583748.554  583748.554
##        9205        9206        9207        9208        9209        9210
##   97246.734  252042.768  252042.768  252042.768  252042.768  169116.321
##        9211        9212        9213        9214        9215        9216
##  169116.321   30905.577  605862.273  605862.273  605862.273  605862.273
##        9217        9218        9219        9220        9221        9222
##  605862.273  605862.273  605862.273  605862.273  605862.273  605862.273
##        9223        9224        9225        9226        9227        9228
## 1186347.399 1186347.399 1186347.399 1186347.399 1186347.399 1186347.399
##        9229        9230        9231        9232        9233        9234
## 1186347.399 1186347.399 1186347.399 1186347.399  666675.000  666675.000
##        9235        9236        9237        9238        9239        9240
##  666675.000  666675.000  666675.000  666675.000  666675.000  666675.000
##        9241        9242        9243        9244        9245        9246
##  666675.000  666675.000  666675.000  666675.000  666675.000  666675.000
##        9247        9248        9249        9250        9251        9252
##  666675.000  451066.239  451066.239  451066.239  451066.239  451066.239
##        9253        9254        9255        9256        9257        9258
```

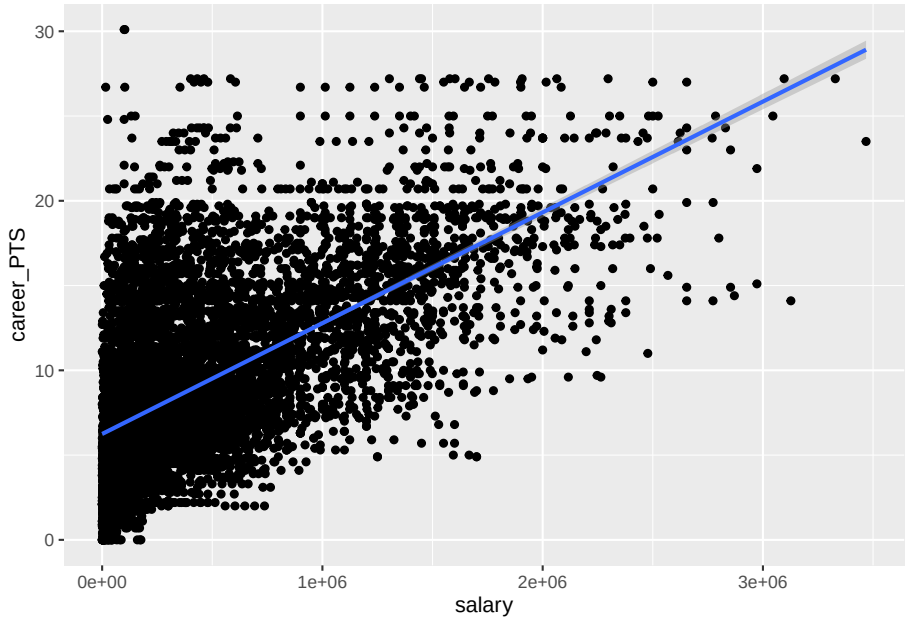
```
##  451066.239  451066.239  451066.239  451066.239  445537.810  445537.810
##        9259        9260        9261        9262        9263        9264
##  445537.810  229929.049  229929.049  -29907.150  -29907.150  -29907.150
##        9265        9266        9267        9268        9269        9270
##   30905.577   30905.577   97246.734   97246.734  384725.082  240985.908
##        9271        9272        9273        9274        9275        9276
##  240985.908  240985.908  240985.908  240985.908  655618.141  655618.141
##        9277        9278        9279        9280        9281        9282
##  655618.141  655618.141  655618.141  655618.141  312855.495  312855.495
##        9283        9284        9285        9286        9287        9288
##  240985.908  240985.908  240985.908  240985.908  240985.908  240985.908
##        9289        9290        9291        9292        9293        9294
##   64076.156   64076.156   64076.156   64076.156   64076.156   64076.156
##        9295        9296        9297        9298        9299        9300
##  307327.065  307327.065  307327.065  307327.065  -85191.448  274156.487
##        9301        9302        9303        9304        9305        9306
##  274156.487  274156.487  274156.487  274156.487  274156.487   53019.296
##        9307        9308        9309        9310        9311        9312
##   53019.296  987323.927  987323.927  987323.927  987323.927  368139.793
##        9313        9314        9315        9316        9317        9318
##  368139.793  368139.793  368139.793  368139.793  368139.793  368139.793
##        9319        9320        9321        9322        9323        9324
##  368139.793  368139.793  368139.793  368139.793  257571.198  257571.198
##        9325        9326        9327        9328        9329        9330
##  257571.198  257571.198  257571.198  257571.198  257571.198  257571.198
##        9331        9332        9333        9334        9335        9336
##  257571.198  257571.198 1285859.135 1285859.135  633504.422  633504.422
##        9337        9338        9339        9340        9341        9342
##   53019.296   53019.296   53019.296   53019.296   53019.296   53019.296
##        9343        9344        9345        9346        9347        9348
##   53019.296   53019.296   53019.296   53019.296  235457.478  235457.478
##        9349        9350        9351        9352        9353        9354
##  235457.478  235457.478  235457.478  235457.478  235457.478  235457.478
##        9355        9356        9357        9358        9359        9360
##  235457.478  235457.478  412367.231  412367.231  412367.231  412367.231
##        9361        9362        9363        9364        9365        9366
##  412367.231  412367.231  412367.231  412367.231  412367.231  412367.231
##        9367        9368        9369        9370        9371        9372
##  257571.198  257571.198  257571.198   41962.437   41962.437  622447.562
##        9373        9374        9375        9376        9377        9378
##  528464.256  528464.256  528464.256  528464.256  528464.256  528464.256
##        9379        9380        9381        9382        9383        9384
##  528464.256  528464.256  528464.256  815942.604  815942.604  815942.604
##        9385        9386        9387        9388        9389        9390
##  815942.604  815942.604  815942.604  815942.604  815942.604  815942.604
##        9391        9392        9393        9394        9395        9396
```

```
##  815942.604  815942.604  815942.604  815942.604  815942.604  815942.604
##        9397        9398        9399        9400        9401        9402
##  815942.604  406838.801  406838.801  406838.801  406838.801  406838.801
##        9403        9404        9405        9406        9407        9408
##  406838.801  406838.801  406838.801  185701.611  185701.611  185701.611
##        9409        9410        9411        9412        9413        9414
##  185701.611  185701.611  185701.611  185701.611  185701.611  185701.611
##        9415        9416        9417        9418        9419        9420
##  390253.512  390253.512  390253.512  390253.512  390253.512  390253.512
##        9421        9422        9423        9424        9425        9426
##  390253.512  390253.512  390253.512   75133.015   75133.015   75133.015
##        9427        9428        9429        9430        9431        9432
##  511878.967  522935.826  522935.826  522935.826  318383.925  318383.925
##        9433        9434        9435        9436        9437        9438
##  318383.925  318383.925  495293.678  495293.678  495293.678  495293.678
##        9439        9440        9441        9442        9443        9444
##  495293.678  495293.678  495293.678  495293.678  495293.678  495293.678
##        9445        9446        9447        9448        9449        9450
##  495293.678  495293.678  495293.678  495293.678  440009.380  440009.380
##        9451        9452        9453        9454        9455        9456
##  440009.380  124888.883  124888.883  124888.883  279684.917  279684.917
##        9457        9458        9459        9460        9461        9462
##  279684.917  279684.917  279684.917  279684.917  279684.917  279684.917
##        9463        9464        9465        9466        9467        9468
##  279684.917  279684.917  169116.321  169116.321  102775.164  102775.164
##        9469        9470        9471        9472        9473        9474
##  583748.554  583748.554  583748.554  583748.554  583748.554  583748.554
##        9475        9476        9477        9478        9479        9480
##  583748.554   80661.445   80661.445   80661.445   80661.445   80661.445
##        9481        9482        9483        9484        9485        9486
##  699845.579  699845.579  699845.579  699845.579  699845.579  699845.579
##        9487        9488        9489        9490        9491        9492
##  699845.579  699845.579  699845.579  699845.579  699845.579  699845.579
##        9493        9494        9495        9496        9497        9498
##  699845.579  644561.281  644561.281  644561.281  644561.281  644561.281
##        9499        9500        9501        9502        9503        9504
##  644561.281  644561.281  644561.281  644561.281  644561.281  644561.281
##        9505        9506        9507        9508        9509        9510
##  644561.281  644561.281  644561.281  495293.678  495293.678  495293.678
##        9511        9512        9513        9514        9515        9516
##  495293.678  495293.678  495293.678  495293.678  495293.678  495293.678
##        9517        9518        9519        9520        9521        9522
##  495293.678  495293.678  495293.678  495293.678  224400.619  224400.619
##        9523        9524        9525        9526        9527        9528
##  224400.619  224400.619 -24378.721  522935.826  263099.627  263099.627
##        9529        9530        9531        9532        9533        9534
```

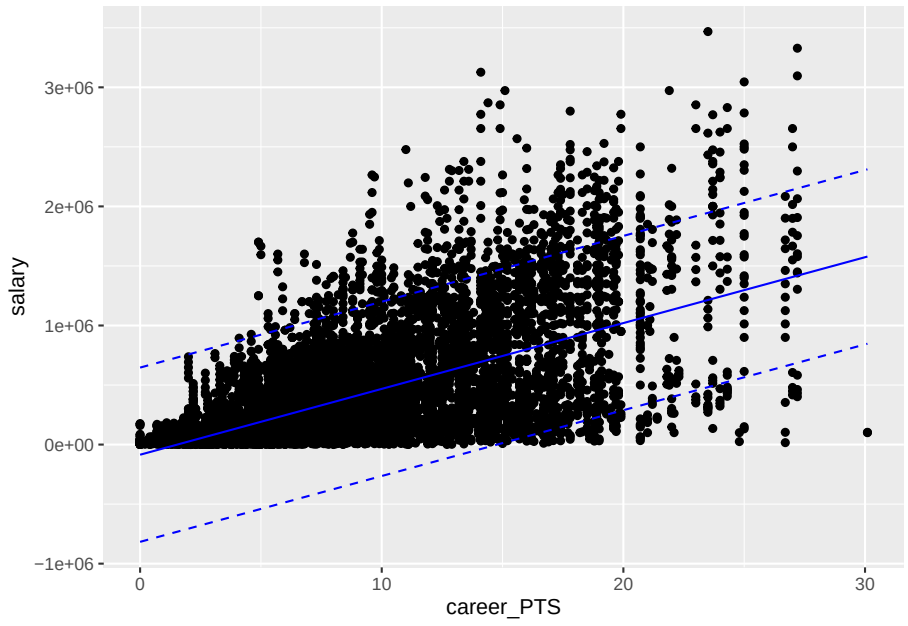
```
##  263099.627  263099.627  263099.627  346026.074  346026.074  346026.074
##        9535        9536        9537        9538        9539        9540
##  346026.074  346026.074  346026.074  346026.074  196758.470  196758.470
##        9541        9542        9543        9544        9545        9546
##  196758.470  196758.470  196758.470  196758.470  196758.470  147002.602
##        9547        9548        9549        9550        9551        9552
##  147002.602  147002.602  235457.478  235457.478  235457.478  235457.478
##        9553        9554        9555        9556        9557        9558
##  235457.478  235457.478  163587.892  163587.892  163587.892  163587.892
##        9559        9560        9561        9562        9563        9564
##  163587.892  163587.892  224400.619  224400.619  224400.619  224400.619
##        9565        9566        9567        9568        9569        9570
##  224400.619  224400.619  224400.619  224400.619  307327.065  307327.065
##        9571        9572        9573        9574        9575        9576
##  307327.065  307327.065  246514.338  246514.338  246514.338  246514.338
##        9577        9578        9579        9580        9581        9582
##  567163.265  567163.265  567163.265  567163.265  567163.265  334969.214
##        9583        9584        9585        9586        9587        9588
##  152531.032  224400.619  224400.619  224400.619  406838.801  406838.801
##        9589        9590        9591        9592        9593        9594
##  406838.801  -85191.448  -85191.448   91718.305   91718.305   91718.305
##        9595        9596        9597        9598        9599        9600
##   91718.305  147002.602   25377.147   25377.147   25377.147   25377.147
##        9601        9602        9603        9604        9605        9606
##  207815.330  207815.330   58547.726   58547.726   58547.726   58547.726
##        9607        9608        9609        9610        9611        9612
##   58547.726   58547.726  141474.172  141474.172  141474.172  141474.172
##        9613        9614        9615        9616        9617        9618
##  218872.189  218872.189  218872.189  213343.759  213343.759  213343.759
##        9619        9620        9621        9622        9623        9624
##  213343.759  213343.759  213343.759  191230.040  191230.040  301798.636
##        9625        9626        9627        9628        9629        9630
##  301798.636  301798.636  301798.636  301798.636  301798.636  301798.636
##        9631        9632        9633        9634        9635        9636
##  301798.636  301798.636  301798.636  119360.453  119360.453  323912.355
##        9637        9638        9639        9640        9641        9642
##  323912.355  323912.355  379196.652  379196.652  379196.652  379196.652
##        9643        9644        9645        9646        9647        9648
##  379196.652  379196.652  379196.652  379196.652  379196.652  379196.652
##        9649        9650        9651        9652        9653        9654
##  379196.652  379196.652  130417.313  130417.313  130417.313  130417.313
##        9655        9656        9657        9658        9659        9660
##  357082.933  357082.933  357082.933  357082.933  357082.933  357082.933
##        9661        9662        9663        9664        9665        9666
##  357082.933  357082.933  357082.933  357082.933  357082.933  451066.239
##        9667        9668        9669        9670        9671        9672
```

```
##  528464.256  528464.256  528464.256  528464.256  528464.256  528464.256
##        9673        9674        9675        9676        9677        9678
##   41962.437  102775.164  102775.164  102775.164  152531.032  545049.546
##        9679        9680        9681        9682        9683        9684
##  545049.546  545049.546  545049.546  545049.546  545049.546  545049.546
##        9685        9686        9687        9688        9689        9690
##  545049.546  545049.546  545049.546  545049.546  235457.478  235457.478
##        9691        9692        9693        9694        9695        9696
##  235457.478  235457.478  655618.141  655618.141  655618.141  655618.141
##        9697        9698        9699        9700        9701        9702
##  655618.141  655618.141  655618.141  655618.141  655618.141  655618.141
##        9703        9704        9705        9706        9707        9708
##  655618.141   36434.007  -52020.869  368139.793  368139.793  368139.793
##        9709        9710        9711        9712        9713        9714
##  368139.793  368139.793  -18850.291  301798.636  301798.636  301798.636
##        9715        9716        9717        9718        9719        9720
##  301798.636  301798.636  301798.636  158059.462  158059.462  158059.462
##        9721        9722        9723        9724        9725        9726
##  158059.462  158059.462  -18850.291  174644.751  174644.751  268628.057
##        9727        9728
##  296270.206  296270.206
```

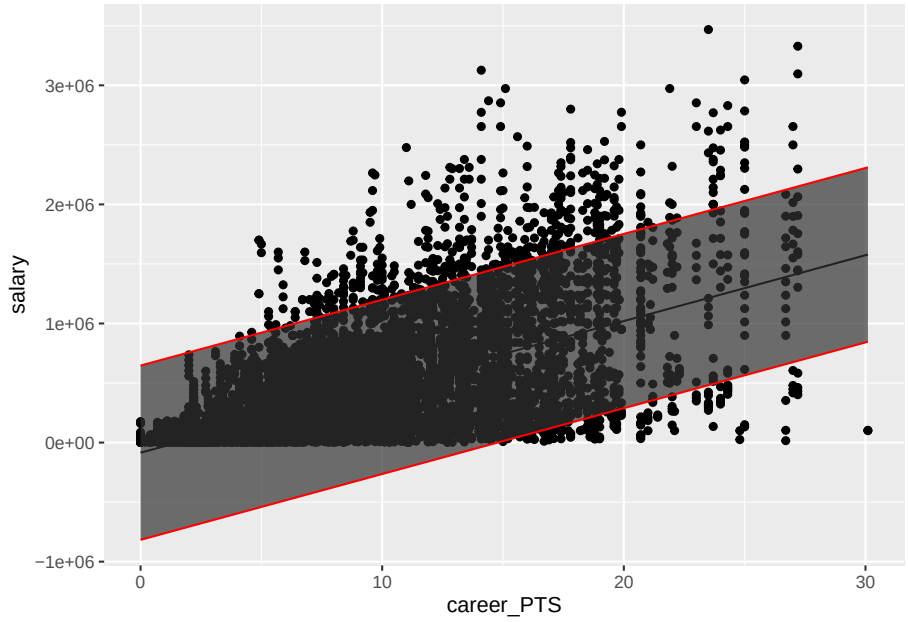
```
# using geom_smooth, method=lm will automatically plot the confidence intervals
data_nba %>%
  ggplot(aes(x=salary, y=career_PTS)) +
  geom_point() +
  geom_smooth(method = "lm")
```




```
# let's get prediction intervals and add them to our dataset
data_nba_predict <- cbind(data_nba, predict(model2, interval = c("prediction")))

data_nba_predict %>%
  ggplot(aes(x= career_PTS, y=salary)) +
  geom_point() +
  geom_line(aes(x=career_PTS, y=fit),
            col="blue") +
  geom_line(aes(y=lwr),
                col="blue",
                linetype="dashed") +
  geom_line(aes(y=upr),
                col="blue",
                linetype="dashed")
```



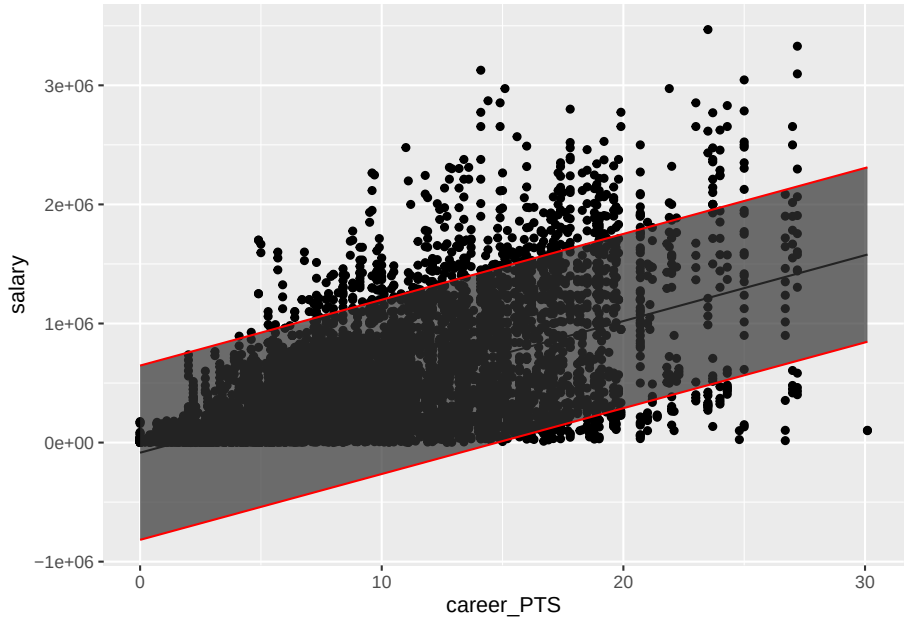

```
# same using geom_ribbon
data_nba_predict %>%
ggplot(aes(x=career_PTS,y=salary))+
  geom_point()+
  geom_line(aes(x=career_PTS,y=fit))+
  geom_ribbon(aes(ymax=upr,ymin=lwr),color="red",alpha=0.7)
```




```
## do this with tidy approach

# get confidnce interval for estimate
tidy(model2, conf.int = TRUE)
```

```
## # A tibble: 2 x 7
##   term        estimate std.error statistic  p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
## 1 (Intercept)  -85191.     7642.     -11.1 1.09e-28 -100171.   -70212.
## 2 career_PTS    55284.      745.      74.2 0          53823.    56745.
```

```
# get predictive intervals
augment(model2, interval= "prediction") %>%
  ggplot(aes(x=career_PTS,y=salary))+
  geom_point()+
  geom_line(aes(x=career_PTS,y=.fitted))+
  geom_ribbon(aes(ymax=.upper,ymin=.lower),color="red",alpha=0.7)
```



Now, let's compare the prediction interval for 2 differnt models.To compare the performance of models, you can use the r-squared (which measures how much of the variation in the outcome can be explained by your set of independent variables) and the mean squared error. For more background on both measures see here: https://vitalflux.com/mean-square-error-r-squared-which-one-to-use/

The Mean squared error (MSE) represents the error of the estimator or predictive model created based on the given set of observations in the sample. It measures the average squared difference between the predicted values and the actual values, quantifying the discrepancy between the model's predictions and the true observations. The lower the MSE, the better the model predictive accuracy, and, the better the regression model is.

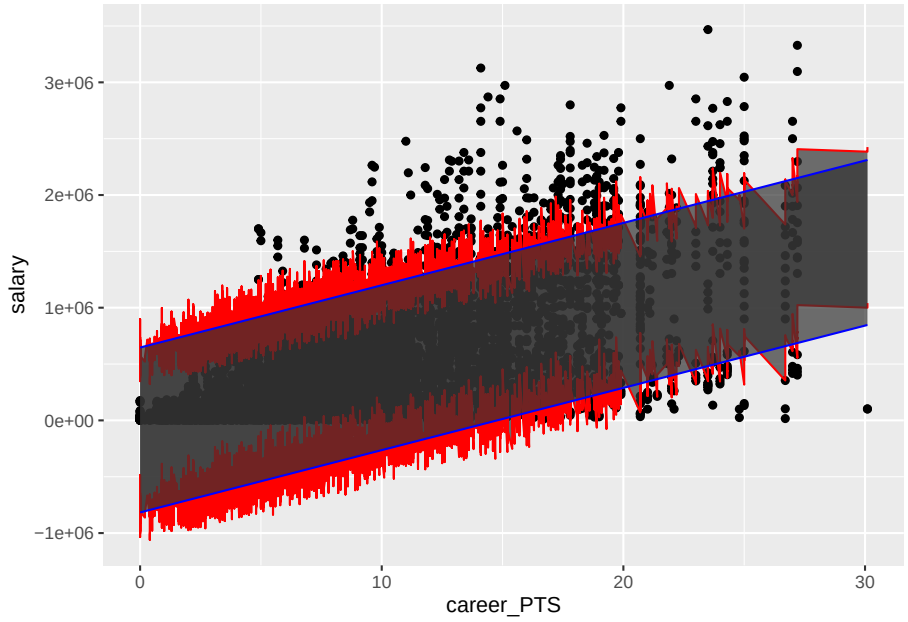
```
model1 <- lm(salary ~ career_PTS + position_rec + season_start +
             age, data = data_nba)
model2 <- lm(salary ~ career_PTS, data = data_nba)

# compared R-squared/ adjusted R-squared
glance(model1)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared   sigma statistic p.value    df   logLik    AIC    BIC
##       <dbl>         <dbl>   <dbl>     <dbl>   <dbl> <dbl>    <dbl>  <dbl>  <dbl>
## 1     0.432         0.431 352361.      308.       0    24 -138041. 2.76e5 2.76e5
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
glance(model2)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared   sigma statistic p.value    df   logLik    AIC    BIC
##       <dbl>         <dbl>   <dbl>     <dbl>   <dbl> <dbl>    <dbl>  <dbl>  <dbl>
## 1     0.361         0.361 373217.     5502.       0     1 -138612. 2.77e5 2.77e5
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
# compare prediction intervals
model1_predicted <- augment(model1, interval = "prediction")
model2_predicted <- augment(model2, interval = "prediction")

  ggplot(aes(x=career_PTS,y=salary), data = model1_predicted)+
  geom_point()+
  # geom_line(aes(x=career_pts,y=.fitted),
  #           color ="red", data = model1_predicted) +
  # geom_line(aes(x=career_pts,y=.fitted),
  #          color ="blue", data = model2_predicted) +
  geom_ribbon(aes(ymax=.upper,ymin=.lower),
              color="red",
              alpha=0.7,
              data = model1_predicted) +
  geom_ribbon(aes(ymax=.upper,ymin=.lower),
              color="blue",
              alpha=0.7,
              data = model2_predicted)
```




```
# see if there is a better way to do this

library(Metrics) # using rmse from Metrics library
# compare mean squared error
  rmse(model1_predicted$.fitted, data_nba$salary)
```

```
## [1] 351908.3
```

```
  rmse(model2_predicted$.fitted, data_nba$salary)
```

```
## [1] 373178.4
```

Looking at r2, the prediction intervals and rmse, model 1 clearly performs better than model 2.

11.2 Intro to Machine learning

We have now arrived at the entry gates to machine learning. We will conduct a very basic and simple machine learning routine using linear regression. The main difference between simple prediction and Machine Learning is that the sample is first divided into a training and a test dataset at random. The model

is than tuned based on, let's say, 80% of the sample. When the model is ready, it is tested based on the 20% remaining sample. The prediction produced with the model are then compared with the actual values in the test dataset. There is of course more nuance to all this, but this is basically the idea.

Let's use the "caret" package, a common package for machine learning.

```
library(caret)

# Create a train and test split
set.seed(123)  # For reproducibility
train_indices <- createDataPartition(data_nba$salary, p = 0.7, list = FALSE)
train_data <- data_nba[train_indices, ]
test_data <- data_nba[-train_indices, ]

# Create a train control object
ctrl <- trainControl(method = "none")

# Train a linear regression model using caret
model1 <- train(
  salary ~ career_PTS + position_rec + season_start + age,
  data = train_data,
  method = "lm",
  trControl = ctrl
)

model2 <- train(
  salary ~ career_PTS,
  data = train_data,
  method = "lm",
  trControl = ctrl
)

# Make predictions on the test set
predicted_salaries_m1 <- predict(model1, newdata = test_data)
predicted_salaries_m2 <- predict(model2, newdata = test_data)


# Calculate prediction errors (e.g., root mean squared error)
rmse1 <- sqrt(mean((predicted_salaries_m1 - test_data$salary)^2))
rmse2 <- sqrt(mean((predicted_salaries_m2 - test_data$salary)^2))

# Print the prediction errors
print(rmse1)
```

```
## [1] 341746.4
```

```
print(rmse2)
```

```
## [1] 360297.3
```

Now machine learning usually involves several more steps: + we can optimize how the variables (in ML language called features) enter the model (Preprocessing; transformations; diagnostics, see week X) + we can optimize which variables should even enter the model ("feature selection", see e.g. lasso regression) + we can optimize how the predictions of the model get evaluated ("training") + we can optimize which estimator or algorithm best predicts the outcome (lm model is just one option among many) + for other algorithms (e.g. random forests), we can also "tune" the model using hyper-parameters

11.3 More resources:

- Provide list of online resources to dig deeper
- see `tidymodels()` for another package for Machine Learning

Chapter 12

Prediction - Application

Here goes some texts.

12.1 Exercises

1. **Variable selection**: Use variable selection methods, such as stepwise selection or best subset selection, to identify a subset of predictor variables that provide the best fit for a multiple linear regression model. Compare the performance of different models and discuss their relative strengths and weaknesses.
2. **Interaction effects**: Include interaction terms in a multiple linear regression model to capture the effect of two or more predictor variables interacting with each other. Interpret the coefficients of the interaction terms and discuss their implications from a social science perspective.
3. **Polynomial regression**: Use polynomial regression to model non-linear relationships between predictor variables and the response variable. Fit a polynomial regression model, generate predictions, and assess the model's performance.
4. **Model comparison**: Use model comparison methods, such as adjusted R-squared or AIC, to compare the performance of different linear regression models. Select the best model based on these criteria and discuss its strengths and weaknesses.
5. **Tidy models**: Use the `broom` package to convert linear regression models into tidy data frames using functions such as `tidy()`, `glance()`, and `augment()`. Manipulate and visualize the tidy data frames to gain insights into the models and their performance.
6. **Prediction intervals**: Generate prediction intervals for new observations using a fitted linear regression model. Interpret the prediction intervals

and discuss their usefulness for making predictions in a social science context.

Chapter 13

Outlook

13.1 Summary of what was covered in course

13.2 Other outcome variables

- generalized linear models
- logistic
- poisson

13.3 Data structures

- multilevel

13.4 Where to go next

- econometrics
- machine learning
- geo-data
- web-scraping