

RStatsbook

Andreas Busjahn

2023-03-07

Table of contents

Preface	4
1 Introduction	5
2 Syntax rules / basic things to know about R	6
2.1 Script preparation / basic setup	6
2.2 Numeric operations	6
2.3 Variables	7
2.3.1 Variable names	7
2.3.2 Basic classes of data	8
2.3.3 Indexing variables	13
2.3.4 Usage of variables	14
2.4 Functions	15
2.5 More complex data types, created by functions	18
2.5.1 Matrix	18
2.5.2 Data frame	20
2.5.3 Tibble	23
2.5.4 List	46
3 Importing data	63
3.1 Import from text files (.txt, .csv)	63
3.2 Import from Excel	63
3.2.1 Tidy Excel files	63
3.2.2 Dirty Excel files	63
3.3 Import from SPSS	63
3.4 Import from SAS	63
4 Grouping of variables by type / distribution / use	64
4.1 Test for normal distribution	64
4.2 Picking column names and positions	64
5 Visualize data with ggplot	65
6 Descriptive statistics	123
7 Better tables with flextable	131

8 Simple test statistics	132
9 Intro to lm	153
9.1 Setup	153
9.2 Import / Preparation	154
9.3 Graphical exploration	154
9.4 Linear Models	159
9.4.1 Linear regression	159
9.4.2 ANOVA	164
9.4.3 LM with continuous AND categorical IV	170
9.4.4 Model exploration with package performance	178
10 Summary	180
References	181

Preface

This is a Quarto book.

To learn more about Quarto books visit <https://quarto.org/docs/books>.

...

1 Introduction

This booklet is meant as companion to my R / statistics seminars. It is NOT a complete guide to either R or statistical data analysis.

There are plenty of books available, check out e.g.

[R for Data Science](#)

[Modern Statistics for Modern Biology](#)

[Modern R with the tidyverse](#)

[ggplot2: Elegant Graphics for Data Analysis](#)

Chapters will follow my usual schedule, starting from R basics, introducing ggplot, data import, data preparation and cleaning, descriptive statistics, simple test statistics, then progression to linear models (regression/ANOVA), generalized linear models with logistic regression as example, linear mixed effect models, and machine learning.

2 Syntax rules / basic things to know about R

2.1 Script preparation / basic setup

At the beginning of (almost) every script we define packages to be used. This could be done by either

- checking if packages needed are installed and otherwise do so, followed by function `library(packagename)`

OR

- simplifying this using function `p_load()` from package pacman; if you want to create fool-prof scripts, check for pacman and install if needed.

```
if(!require(pacman, quietly = TRUE)){
  install.packages('pacman')
}
pacman::p_load(
  tidyverse, # metapackage
  wrappedtools, # my own tools package
  randomNames # used to create pseudo names
)
```

2.2 Numeric operations

```
#### simple calculations ####
2+5
```

```
[1] 7
```

```
3*5
```

```
[1] 15
```

```
15/3 #not 15:3!!
```

```
[1] 5
```

```
3^2
```

```
[1] 9
```

```
9^0.5
```

```
[1] 3
```

```
10%%3 #modulo
```

```
[1] 1
```

2.3 Variables

2.3.1 Variable names

Naming things is harder than you may expect. Try to be verbose and consistent in language and style. Commonly used are snake_case_style and CamelCaseStyle.

Decide about computer-friendly (syntactical) or human-friendly names, illegal names can be used inside backticks: ‘measure [unit]’. My preference is syntactical for script variables and humane for data variables, e.g. column names, print labels etc.

There are rules for valid syntactical names:

- UPPERCASE and lowercase are distinguished
- start with letter or symbols as . __ , but not with a number
- no mathematical symbols or brackets allowed

To store some value into a variable, use the assignment operator `<-` ; while it is possible to use `=` or `->` , this is rather unusual. Assignments are silent, so either a call of the variable, or `print()` / `cat()` function are needed to inspect. Alternatively, put brackets around assignment: (varname `<-` content).

```
### Variable names ####
test <- 1
test1 <- 1
# 1test <- 2 # wrong, would result in error
`1test` <- 2 # this would be possible
test_1<- 5
test.1 <- 2
`test-1` <- 6
`test(1)` <- 5
Test <- 'bla'
HereAreFilteredData <- '' #CamelCase
here_are_filtered_data <- "test" #snake_case
`Gewicht [kg]` <- 67
```

2.3.2 Basic classes of data

R is ‘guessing’ the suitable type of data from input. This should be checked after e.g. importing data! If elements of different classes are found, the more inclusive is used. There are functions to change / force a type if needed.

The `class()` function returns the class of an object, which determines how it behaves with respect to functions like `print()`. The class of an object can be changed by using generic functions and methods.

The `typeof()` function returns the basic data type of an object, which determines how it is stored in memory. The basic data type of an object cannot be changed.

The `str()` function shows class and examples of an object.

2.3.2.1 Guessed classes

```
float_num <- 123.456
class(float_num)
```

```
[1] "numeric"
```

```
typeof(float_num)
```

```
[1] "double"
```

```
int_num <- 123L # L specifies integer, guessing requires more values
class(int_num)
```

```
[1] "integer"
```

```
typeof(int_num)
```

```
[1] "integer"
```

```
result<-9^(1/2)
result
```

```
[1] 3
```

```
print(result)
```

```
[1] 3
```

```
cat(result)
```

```
3
```

```
char_var <- ' some words'
class(char_var)
```

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

```
typeof(char_var)

[1] "character"

logical_var <- TRUE
class(logical_var)

[1] "logical"

typeof(logical_var)

[1] "logical"

logical(length = 3)

[1] FALSE FALSE FALSE

# logicals usually are defined by conditions:
int_num < float_num

[1] TRUE

# all numbers are true but 0
as.logical(c(0,1,5,-7.45678)) # c() combines values into a vector

[1] FALSE  TRUE  TRUE  TRUE

factor_var <- factor(c("m","m","f","m","f","f","?"))
factor_var

[1] m m f m f f ?
Levels: ? f m
```

```
class(factor_var)

[1] "factor"

typeof(factor_var)

[1] "integer"

# factor definition can reorder, rename, and drop levels:
factor_var2 <- factor(c("m","m","f","m","f","f","?"),
                       levels=c("m","f"),
                       labels=c("male","female"))

factor_var2

[1] male   male   female male   female female <NA>
Levels: male female

# everything intended to group subjects or representing categories should be stored as factors
# packageforcats provides nice tools for factors

(date_var <- Sys.Date())

[1] "2023-07-07"

class(date_var)

[1] "Date"

typeof(date_var)

[1] "double"
```

```
# mixed classes
test2 <- c(1,2,'a','b')
class(test2)
```

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

2.3.2.2 Forcing / casting classes

Casting functions usually start with `as_`, when creating variables filled with NA, use casting functions or specific variants of NA to force type!

```
(test<-c(1,2,3,'a','b','c'))
```

```
[1] "1" "2" "3" "a" "b" "c"
```

```
(test_n<-as.numeric(test))
```

Warning: NAs durch Umwandlung erzeugt

```
[1] 1 2 3 NA NA NA
```

```
as.numeric(factor_var)
```

```
[1] 3 3 2 3 2 2 1
```

```
as.character(10:19)
```

```
[1] "10" "11" "12" "13" "14" "15" "16" "17" "18" "19"
```

```
# NAs
class(NA_real_)
```

```
[1] "numeric"
```

```
class(NA_integer_)

[1] "integer"

class(NA_character_)

[1] "character"

class(NA_Date_)

[1] "Date"
```

2.3.3 Indexing variables

The most general kind of indexing is by position, starting with 1. Negative numbers result in exclusion of position(s). Position indices are provided within square brackets. The index can be a variable instead of hard coded numbers.

```
(numbers1<-c(5,3,6,8,2,1))

[1] 5 3 6 8 2 1

numbers1[1]

[1] 5

numbers1[1:3]

[1] 5 3 6

numbers1[-1]

[1] 3 6 8 2 1
```

```
numbers2 <- 1:3  
numbers1[numbers2]
```

```
[1] 5 3 6
```

To get first or last entries, `head()` and `tail()` can be used. By default 6 entries are returned.

```
tail(x=numbers1, n = 1)
```

```
[1] 1
```

```
head(x = numbers1, n = 3)
```

```
[1] 5 3 6
```

2.3.4 Usage of variables

Variables are like placeholders for their content, so that you don't have to remember where you left things. Operations on variables are operations on their content. Changing the content of a variable does not automatically save those changes back to the variable, this needs to be done explicitly!

```
numbers1+100 # not stored anywhere, just printed
```

```
[1] 105 103 106 108 102 101
```

```
numbers1+numbers2 # why does this even work?
```

```
[1] 6 5 9 9 4 4
```

When combining variables of different length, the short one is recycled, so the `numbers2` is added to the first 3 elements of `numbers2`, then is reused and added to the remaining 3 elements. If the length of the longer is not a multiple of the shorter, there will be a warning.

```
c(2,4,6,8)+1
```

```
[1] 3 5 7 9
```

```
c(2,4,6,8)+c(1,2)
```

```
[1] 3 6 7 10
```

```
c(2,4,6,8)+c(1,2,3)
```

```
Warning in c(2, 4, 6, 8) + c(1, 2, 3): Länge des längeren Objektes  
ist kein Vielfaches der Länge des kürzeren Objektes
```

```
[1] 3 6 9 9
```

2.4 Functions

Functions have the same naming rules as variables, but the name is always followed by opening/closing round brackets, within those brackets function parameters/arguments can be specified to provide input or control behavior:

```
FunctionName(parameter1=x1,parameter2=x2,x3,...)
```

Most functions have named arguments, those argument names may be omitted as long as parameter values are supplied in the defined order. Arguments may have predefined default values, see `help!` Some functions like `c()` use unnamed arguments.

```
c('my','name') # unnamed
```

```
[1] "my"    "name"
```

```
?mean
```

```
starte den http Server für die Hilfe fertig
```

```

mean(x = c(3,5,7,NA)) #using default parameters

[1] NA

mean(x = c(3,5,7,NA),na.rm = TRUE) #overriding default parameter

[1] 5

mean(na.rm = TRUE, x=c(3,5,7,NA)) # changed order of arguments

[1] 5

mean(c(3,5,7,NA), na.rm = TRUE) # name of 1st argument omitted

[1] 5

sd(c(3,5,7,NA), na.rm = TRUE) # same logic as mean, partially the same arguments

[1] 2

median(1:100, TRUE) # does influence the readability of a function, careful!

[1] 50.5

```

Functions often just solve one problem or task, so often we need combine them. This can be done by nesting or piping. Piping makes reading/understanding scripts easier, as it shows order of functions:

```

# functions may be nested:
floor(
  as.numeric(
    Sys.Date() -

```

```

    as.Date('1985/12/10'))/
365.25)

[1] 37

# or (usually better) piped:
mtcars |>
  mutate(am=factor(am)) |>
  filter(vs==1) |>
  group_by(am) |>
  summarize(across(.cols = c(wt, mpg, qsec, disp),
                  .fns = mean)) |>
  pivot_longer(cols = -am, names_to = 'Measure') |>
  pivot_wider(id_cols = Measure, names_from = am,
              values_from = value)

```

```

# A tibble: 4 x 3
Measure      `0`   `1`
<chr>     <dbl> <dbl>
1 wt        3.19  2.03
2 mpg       20.7  28.4
3 qsec      20.0  18.7
4 disp      175.   89.8

```

If a sequence of functions is used often, combining them into a new function is advisable, e.g. this combination of descriptive and test statistics:

```

# can be combined into higher order functions:
compare2numvars(data = mtcars,
                 dep_vars = c('mpg','wt','qsec'),
                 indep_var = 'am',
                 add_n = TRUE,
                 gaussian = TRUE)

# A tibble: 3 x 5
Variable desc_all           `am 0`           `am 1`           p
<fct>   <chr>             <chr>             <chr>             <chr>
1 mpg     20 ± 6 [n=32]    17 ± 4 [n=19]    24 ± 6 [n=13]  0.001
2 wt      3.2 ± 1.0 [n=32] 3.8 ± 0.8 [n=19]  2.4 ± 0.6 [n=13] 0.001
3 qsec    18 ± 2 [n=32]    18 ± 2 [n=19]    17 ± 2 [n=13]  0.206

```

2.5 More complex data types, created by functions

2.5.1 Matrix

A matrix is a 2-dimensional data structure, where all elements are of the same class.

2.5.1.1 Creation

```
my1.Matrix<-  
  matrix(data=1:12,  
         # nrow=4, # this is not needed, as it can be derived from data  
         ncol=3,  
         byrow=T, # date are put into row 1 first  
         dimnames=list(paste0('row',1:4),  
                       paste0('col',1:3)))  
print(my1.Matrix)
```

```
    col1  col2  col3  
row1    1    2    3  
row2    4    5    6  
row3    7    8    9  
row4   10   11   12
```

```
data <- 1:100  
nrow <- 20  
matrix(data=data,  
       nrow=nrow,  
       byrow=F, # data are put into column 1 first  
       dimnames=list(paste0('row',1:nrow),  
                     paste0('col',1:(length(data)/nrow))))
```

```
    col1  col2  col3  col4  col5  
row1    1    21    41    61    81  
row2    2    22    42    62    82  
row3    3    23    43    63    83  
row4    4    24    44    64    84  
row5    5    25    45    65    85  
row6    6    26    46    66    86
```

```
row7    7   27   47   67   87  
row8    8   28   48   68   88  
row9    9   29   49   69   89  
row10   10  30   50   70   90  
row11   11  31   51   71   91  
row12   12  32   52   72   92  
row13   13  33   53   73   93  
row14   14  34   54   74   94  
row15   15  35   55   75   95  
row16   16  36   56   76   96  
row17   17  37   57   77   97  
row18   18  38   58   78   98  
row19   19  39   59   79   99  
row20   20  40   60   80   100
```

```
mdat <- matrix(c(1,2,3, 11,12,13),  
                 nrow = 2, ncol=3) #byrow=FALSE, not specified but default  
mdat
```

```
[,1] [,2] [,3]  
[1,]    1     3    12  
[2,]    2    11    13
```

2.5.1.2 Indexing

Addressing a matrix is done with [row_index, column_index]

```
my1.Matrix[2,3] # Index:[row,column]
```

```
[1] 6
```

```
my1.Matrix[2,] # all columns
```

```
col1 col2 col3  
4      5      6
```

```
my1.Matrix[,2] # all rows
```

```

row1 row2 row3 row4
 2     5     8    11

my1.Matrix[c(1,3),-2] # exclude column 2

      col1 col3
row1     1     3
row3     7     9

my1.Matrix[1,1]<-NA # Index can be used for writing as well

```

2.5.2 Data frame

A data frame has 2 dimensions, it can handle various data types (1 per columns). This structure is rather superseded by tibbles (see below).

2.5.2.1 Creation

Data frames are defined by creating and filling columns, functions can be used (and piped) to create content.

```

patientN<-15
(myTable<-data.frame(
  patientCode=paste0('pat',1:patientN),
  Var1=1, # gets recycled
  Var2=NA_Date_))

      patientCode Var1 Var2
1          pat1     1 <NA>
2          pat2     1 <NA>
3          pat3     1 <NA>
4          pat4     1 <NA>
5          pat5     1 <NA>
6          pat6     1 <NA>
7          pat7     1 <NA>
8          pat8     1 <NA>
9          pat9     1 <NA>

```

```

10      pat10    1 <NA>
11      pat11    1 <NA>
12      pat12    1 <NA>
13      pat13    1 <NA>
14      pat14    1 <NA>
15      pat15    1 <NA>

str(myTable)

'data.frame': 15 obs. of 3 variables:
$ patientCode: chr "pat1" "pat2" "pat3" "pat4" ...
$ Var1        : num 1 1 1 1 1 1 1 1 1 ...
$ Var2        : Date, format: NA NA ...

set.seed(101)
myTable<-data.frame(
  patientCode=paste0('pat',1:patientN),
  Age=runif(n=patientN,min=18,max=65) |> floor(),
  Sex=factor(rep(x=NA,times=patientN),
             levels=c('m','f')),
  sysRR=round(rnorm(n=patientN,mean=140,sd=10)))
head(myTable)

  patientCode Age  Sex sysRR
1      pat1  35 <NA>   142
2      pat2  20 <NA>   132
3      pat3  51 <NA>   122
4      pat4  48 <NA>   157
5      pat5  29 <NA>   144
6      pat6  32 <NA>   148

```

2.5.2.2 Indexing

Beside the numeric index, columns can be addressed by name. This can be done by either `dfname$colname` (for the content of a single column) or `dfname[,“colname”]` for 1 or more columns.

```
myTable[1:5,1]
```

```
[1] "pat1" "pat2" "pat3" "pat4" "pat5"

myTable$patientCode[1:5]

[1] "pat1" "pat2" "pat3" "pat4" "pat5"

myTable[1:5,"patientCode"] # returns vector of values for a single column, df otherwise

[1] "pat1" "pat2" "pat3" "pat4" "pat5"

myTable["patientCode"] # returns df

  patientCode
1      pat1
2      pat2
3      pat3
4      pat4
5      pat5
6      pat6
7      pat7
8      pat8
9      pat9
10     pat10
11     pat11
12     pat12
13     pat13
14     pat14
15     pat15

spalten<-c('Sex',"Age")
myTable[1:5,spalten]
```

```

      Sex Age
1 <NA> 35
2 <NA> 20
3 <NA> 51
4 <NA> 48
5 <NA> 29

myTable[1:5,c('patientCode','Age')]

```

```

patientCode Age
1       pat1 35
2       pat2 20
3       pat3 51
4       pat4 48
5       pat5 29

myTable[,1] <- paste0('Code',1:patientN)

```

2.5.3 Tibble

Tibbles are a modern and efficient data structure that extend data frames, providing enhanced features and performance for data manipulation and analysis.

2.5.3.1 Creation

```

patientN <- 200
rawdata <- tibble(
  PatID=paste('P',1:patientN), # as in data.frame
  Sex=sample(x = c('male','female'), # random generator
             size = patientN,replace = T,
             prob = c(.7,.3)),
  Ethnicity=sample(1:6,patientN,T,c(.01,.01,.05,.03,.75,.15)), # random assignments
  `Given name`=randomNames(n = patientN,
                            gender = Sex, # this is a reference to column Sex
                            ethnicity = Ethnicity,
                            which.names = 'first'),
  `Family name`=randomNames(n = patientN,
                            ethnicity = Ethnicity,

```

```

            which.names = 'last'),
Treatment=sample(c('Placebo','Verum'),patientN,T),
`sysRR (mmHg)`=round(rnorm(n=patientN,mean=140,sd=10))-
  (Treatment=='Verum')*15,
diaRR=round(rnorm(n=patientN,mean=80,sd=10))-
  (Treatment=='Verum')*10,
HR=round(rnorm(n=patientN,mean=90,sd=7)))
rawdata

# A tibble: 200 x 9
  PatID Sex   Ethnicity `Given name` `Family name` Treatment `sysRR (mmHg)`
  <chr> <chr>     <int> <chr>       <chr>           <chr>           <dbl>
1 P 1 male        5 Jackson    Vahlenkamp    Verum             99
2 P 2 male        5 Bradley    Sivers        Verum            134
3 P 3 female      5 Bre        Jerome        Placebo           129
4 P 4 female      5 Hannah     Knight        Placebo           144
5 P 5 male        5 Tell       Howard        Placebo           154
6 P 6 male        5 Christian  Cheesman     Verum            123
7 P 7 male        5 Travis     McClellan    Placebo           149
8 P 8 male        5 Tanner     Srebniak    Placebo           145
9 P 9 female      6 Aatika     el-Basa      Verum            118
10 P 10 male      5 Eric       Stephenson  Verum            110
# i 190 more rows
# i 2 more variables: diaRR <dbl>, HR <dbl>

colnames(rawdata)

[1] "PatID"          "Sex"            "Ethnicity"       "Given name"      "Family name"
[6] "Treatment"      "sysRR (mmHg)"    "diaRR"          "HR"

cn() # shortcut from wrappedtools

[1] "PatID"          "Sex"            "Ethnicity"       "Given name"      "Family name"
[6] "Treatment"      "sysRR (mmHg)"    "diaRR"          "HR"

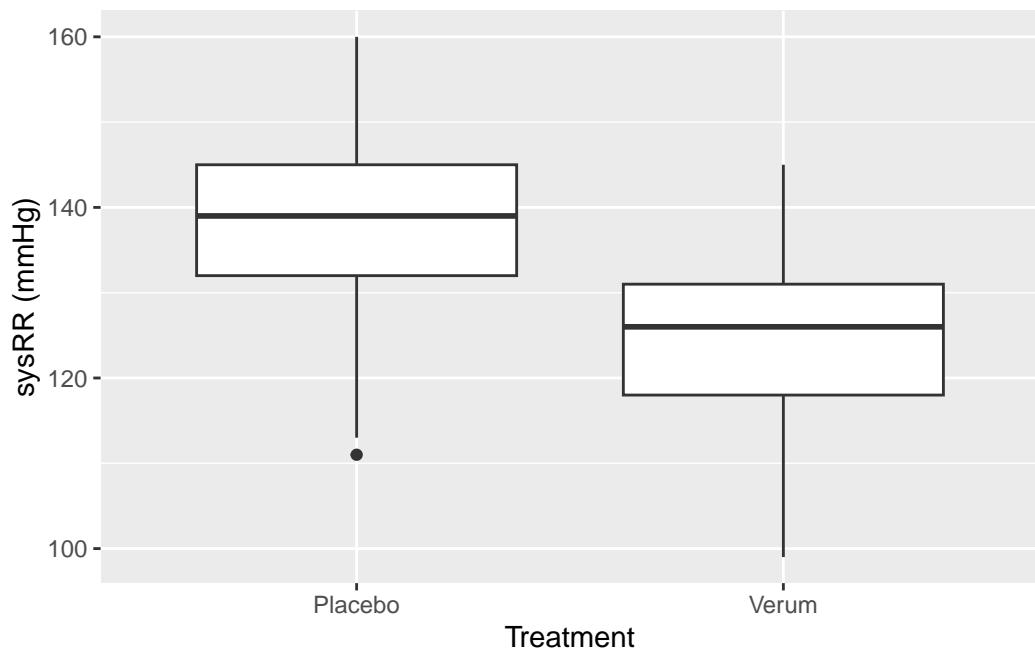
```

```

# example of data management for a tibble, recoding ethnicity:
rawdata <- rawdata |>
  mutate(Ethnicity=factor(
    Ethnicity, levels = 1:6,
    labels= c(
      'American Indian or Native Alaskan',
      'Asian or Pacific Islander',
      'Black (not Hispanic)',
      'Hispanic',
      'White (not Hispanic)',
      'Middle-Eastern, Arabic')))

# quick visual inspection
ggplot(rawdata,aes(x = Treatment,y = `sysRR (mmHg)`))+
  geom_boxplot()

```



2.5.3.2 Indexing

The same rules as for the data frame, but more consistent behavior.

```
rawdata[1:5,1:2]
```

```
# A tibble: 5 x 2
  PatID Sex
  <chr> <chr>
1 P 1   male
2 P 2   male
3 P 3   female
4 P 4   female
5 P 5   male
```

```
  rawdata[,6]
```

```
# A tibble: 200 x 1
  Treatment
  <chr>
1 Verum
2 Verum
3 Placebo
4 Placebo
5 Placebo
6 Verum
7 Placebo
8 Placebo
9 Verum
10 Verum
# i 190 more rows
```

```
  rawdata[6]
```

```
# A tibble: 200 x 1
  Treatment
  <chr>
1 Verum
2 Verum
3 Placebo
4 Placebo
5 Placebo
6 Verum
7 Placebo
8 Placebo
```

```
9 Verum  
10 Verum  
# i 190 more rows
```

```
rawdata[[6]]
```

```
[1] "Verum"    "Verum"    "Placebo"   "Placebo"   "Placebo"   "Verum"    "Placebo"  
[8] "Placebo"   "Verum"    "Verum"     "Placebo"   "Placebo"   "Placebo"   "Verum"  
[15] "Placebo"   "Placebo"   "Placebo"   "Verum"     "Placebo"   "Placebo"   "Verum"  
[22] "Verum"    "Verum"    "Verum"     "Verum"     "Placebo"   "Placebo"   "Verum"  
[29] "Verum"    "Placebo"   "Verum"     "Placebo"   "Placebo"   "Placebo"   "Verum"  
[36] "Verum"    "Placebo"   "Placebo"   "Placebo"   "Placebo"   "Placebo"   "Placebo"  
[43] "Placebo"   "Placebo"   "Verum"     "Placebo"   "Verum"     "Placebo"   "Verum"  
[50] "Placebo"   "Verum"    "Verum"     "Placebo"   "Verum"     "Verum"    "Verum"  
[57] "Placebo"   "Placebo"   "Placebo"   "Verum"     "Verum"     "Verum"    "Verum"  
[64] "Placebo"   "Placebo"   "Placebo"   "Placebo"   "Verum"     "Placebo"   "Placebo"  
[71] "Placebo"   "Verum"    "Placebo"   "Placebo"   "Placebo"   "Verum"    "Placebo"  
[78] "Placebo"   "Verum"    "Placebo"   "Verum"     "Placebo"   "Placebo"   "Placebo"  
[85] "Verum"    "Placebo"   "Placebo"   "Verum"     "Verum"     "Placebo"   "Verum"  
[92] "Placebo"   "Verum"    "Verum"     "Placebo"   "Verum"     "Placebo"   "Verum"  
[99] "Placebo"   "Verum"    "Placebo"   "Verum"     "Verum"     "Placebo"   "Verum"  
[106] "Verum"    "Placebo"   "Placebo"   "Placebo"   "Placebo"   "Placebo"   "Verum"  
[113] "Verum"    "Verum"    "Placebo"   "Verum"     "Placebo"   "Placebo"   "Verum"  
[120] "Verum"    "Placebo"   "Verum"     "Verum"     "Verum"     "Placebo"   "Placebo"  
[127] "Placebo"   "Placebo"   "Placebo"   "Placebo"   "Verum"     "Placebo"   "Verum"  
[134] "Placebo"   "Placebo"   "Verum"     "Verum"     "Verum"     "Placebo"   "Verum"  
[141] "Verum"    "Verum"    "Verum"     "Verum"     "Placebo"   "Placebo"   "Placebo"  
[148] "Verum"    "Placebo"   "Verum"     "Placebo"   "Placebo"   "Verum"    "Verum"  
[155] "Verum"    "Verum"    "Verum"     "Placebo"   "Verum"     "Placebo"   "Verum"  
[162] "Verum"    "Verum"    "Verum"     "Verum"     "Placebo"   "Placebo"   "Verum"  
[169] "Verum"    "Verum"    "Verum"     "Placebo"   "Placebo"   "Placebo"   "Verum"  
[176] "Verum"    "Verum"    "Placebo"   "Placebo"   "Verum"     "Verum"    "Placebo"  
[183] "Verum"    "Verum"    "Verum"     "Verum"     "Verum"     "Verum"    "Verum"  
[190] "Verum"    "Verum"    "Verum"     "Verum"     "Placebo"   "Verum"    "Placebo"  
[197] "Placebo"   "Placebo"   "Verum"     "Placebo"
```

```
rawdata$`Family name`
```

```
[1] "Vahlenkamp"        "Sivers"          "Jerome"
```

[4]	"Knight"	"Howard"	"Cheesman"
[7]	"McClellan"	"Srebniak"	"el-Basa"
[10]	"Stephenson"	"Reed"	"Montellano"
[13]	"al-Mansouri"	"Deeb"	"Scurto"
[16]	"Tebbe"	"Snyder"	"Schaffer"
[19]	"Sidrow"	"Miklin"	"el-Rashid"
[22]	"Welder"	"Moreland"	"al-Aydin"
[25]	"el-Amer"	"Lorey Mcmahon"	"Spalding"
[28]	"Gallegos"	"Ball"	"Perez"
[31]	"Goodburn"	"Brooks"	"Farris"
[34]	"Dyer"	"Taylor"	"Tucker"
[37]	"Munson"	"Mcbeth"	"Berg"
[40]	"St Clair"	"Ford"	"Whiston"
[43]	"Durocher-Harleman"	"Malmquist"	"Woolley"
[46]	"Fanning"	"Stark"	"McLaughlin"
[49]	"el-Qadir"	"Mitchell"	"Nixon"
[52]	"Tanner"	"Willison"	"Preciado"
[55]	"Wolf"	"Hall"	"Smith"
[58]	"Broyles"	"Dietrick"	"Thomas"
[61]	"De Alba"	"al-Odeh"	"Kingston"
[64]	"Turner"	"Collupy"	"Hoggard"
[67]	"el-Selim"	"Glaves"	"Holden"
[70]	"Seth"	"el-Tariq"	"Hardison"
[73]	"Bertsch"	"Abrams"	"Flathers"
[76]	"Jeffryes"	"Buchanan"	"Danielson"
[79]	"East"	"Fognani"	"el-Barakat"
[82]	"Freeman"	"Perez"	"Hilton"
[85]	"Anderson Casey"	"al-Huda"	"el-Rassi"
[88]	"Dillie"	"Heimbrock"	"Testroet"
[91]	"Harris"	"Moldenhauer"	"Hockley"
[94]	"Willis"	"Wicks"	"Rastall"
[97]	"Montgomery"	"Nelson"	"Lydeen"
[100]	"el-Abdi"	"Garner"	"al-Wahba"
[103]	"Leslie"	"Cottrell"	"Hershberger"
[106]	"el-Safar"	"Bellis"	"Truesdale"
[109]	"Reiher"	"Newton-Bustamante"	"Hasson"
[112]	"el-Murad"	"Allen"	"McDonald"
[115]	"Hartman"	"al-Malek"	"Schoneman"
[118]	"Williams"	"Crapsey"	"Levy"
[121]	"Sorenson"	"Mcnamee"	"Bennett"
[124]	"Tipton"	"el-Pour"	"Larson"
[127]	"Martinez"	"Blackney"	"Baldwin"
[130]	"Creamer"	"Lee"	"Putzstuck"

[133]	"el-Yamin"	"Hernandez"	"Montoya"
[136]	"Emmons"	"Wilson"	"Linman"
[139]	"Sharp"	"Pier"	"O'Leary"
[142]	"Woerpel"	"Newton"	"Geary"
[145]	"Swearnger"	"Vigil"	"Barela"
[148]	"Moss"	"al-Agha"	"Cerceda Martinez"
[151]	"Swan"	"Barber"	"Kuetemann"
[154]	"Miller"	"Byers"	"Romero"
[157]	"al-Greiss"	"Roybal"	"al-Bari"
[160]	"Maestas"	"Rodriguez"	"Aschenbrenner"
[163]	"el-Madani"	"al-Karimi"	"Winkelman"
[166]	"al-Masood"	"Swihart"	"Goodrich"
[169]	"Sullivan"	"Kessler"	"Jacquez"
[172]	"Shippley"	"Dyl"	"Wells"
[175]	"Hansen"	"Williams"	"Hernandez"
[178]	"Cain"	"Smith"	"Cooper"
[181]	"Fisher"	"McLaughlin"	"al-Samra"
[184]	"Kennis"	"al-Salih"	"al-Rasheed"
[187]	"el-Ayoob"	"Roberts"	"Mosher"
[190]	"Wernsman"	"Parrish"	"Davis"
[193]	"Brooks"	"al-Hassen"	"Hilzer"
[196]	"al-Raad"	"Benjamin"	"Hastings"
[199]	"Korosec"	"Funk"	

Differences in addressing data frames and tibbles:

- tibble and [always returns tibble
- tibble and [[always returns vector
- data.frame and [may return data.frame (if >1 column) or vector
- data.frame and [[always returns vector

```
rawdata_df <- as.data.frame(rawdata)
rawdata[2] #returns Tibble with 1 column
```

```
# A tibble: 200 x 1
  Sex
  <chr>
1 male
2 male
3 female
```

```

4 female
5 male
6 male
7 male
8 male
9 female
10 male
# i 190 more rows

rawdata[[2]] #returns vector

[1] "male"   "male"   "female" "female" "male"   "male"   "male"   "male"
[9] "female" "male"   "female" "male"   "male"   "male"   "male"   "female"
[17] "female" "male"   "female" "female" "female" "male"   "male"   "female"
[25] "female" "male"   "female" "male"   "female" "female" "male"   "male"
[33] "male"   "male"   "female" "male"   "female" "female" "female" "male"
[41] "female" "male"   "male"   "male"   "male"   "male"   "female" "female"
[49] "male"   "male"   "female" "male"   "male"   "male"   "male"   "male"
[57] "male"   "male"   "female" "female" "male"   "male"   "male"   "female"
[65] "female" "male"   "male"   "male"   "female" "female" "female" "female"
[73] "male"   "male"   "male"   "male"   "male"   "male"   "male"   "male"
[81] "female" "male"   "male"   "male"   "male"   "male"   "male"   "male"
[89] "male"   "male"   "male"   "male"   "female" "male"   "male"   "female"
[97] "male"   "female" "male"   "female" "male"   "male"   "male"   "male"
[105] "male"  "male"   "male"   "male"   "female" "male"   "male"   "male"
[113] "female" "male"   "female" "female" "male"   "female" "male"   "male"
[121] "female" "male"   "female" "male"   "male"   "male"   "female" "male"
[129] "female" "female" "female" "female" "female" "female" "male"   "female"
[137] "female" "male"   "male"   "male"   "male"   "male"   "male"   "male"
[145] "male"   "male"   "male"   "male"   "male"   "female" "male"   "male"
[153] "female" "male"   "male"   "male"   "male"   "male"   "male"   "female"
[161] "male"   "female" "male"   "female" "male"   "male"   "female" "male"
[169] "male"   "male"   "male"   "male"   "male"   "female" "male"   "male"
[177] "male"   "male"   "male"   "male"   "male"   "female" "male"   "female"
[185] "female" "male"   "male"   "male"   "male"   "male"   "male"   "male"
[193] "male"   "female" "male"   "male"   "female" "male"   "male"   "male"

```

rawdata[,2] #returns Tibble with 1 column

```

# A tibble: 200 x 1
  Sex
  <chr>
  1 male
  2 male
  3 female
  4 female
  5 male
  6 male
  7 male
  8 male
  9 female
 10 male
# i 190 more rows

  rawdata[,2:3] #returns tibble with 2 columns

# A tibble: 200 x 2
  Sex     Ethnicity
  <chr>   <fct>
  1 male   White (not Hispanic)
  2 male   White (not Hispanic)
  3 female White (not Hispanic)
  4 female White (not Hispanic)
  5 male   White (not Hispanic)
  6 male   White (not Hispanic)
  7 male   White (not Hispanic)
  8 male   White (not Hispanic)
  9 female Middle-Eastern, Arabic
 10 male   White (not Hispanic)
# i 190 more rows

  rawdata_df[2] #returns DF with 1 column

  Sex
  1   male
  2   male
  3 female
  4 female

```

5	male
6	male
7	male
8	male
9	female
10	male
11	female
12	male
13	male
14	male
15	male
16	female
17	female
18	male
19	female
20	female
21	female
22	male
23	male
24	female
25	female
26	male
27	female
28	male
29	female
30	female
31	male
32	male
33	male
34	male
35	female
36	male
37	female
38	female
39	female
40	male
41	female
42	male
43	male
44	male
45	male
46	male
47	female

48 female
49 male
50 male
51 female
52 male
53 male
54 male
55 male
56 male
57 male
58 male
59 female
60 female
61 male
62 male
63 male
64 female
65 female
66 male
67 male
68 male
69 female
70 female
71 female
72 female
73 male
74 male
75 male
76 male
77 male
78 male
79 male
80 male
81 female
82 male
83 male
84 male
85 male
86 male
87 male
88 male
89 male
90 male

91 male
92 male
93 female
94 male
95 male
96 female
97 male
98 female
99 male
100 female
101 male
102 male
103 male
104 male
105 male
106 male
107 male
108 male
109 female
110 male
111 male
112 male
113 female
114 male
115 female
116 female
117 male
118 female
119 male
120 male
121 female
122 male
123 female
124 male
125 male
126 male
127 female
128 male
129 female
130 female
131 female
132 female
133 female

134 female
135 male
136 female
137 female
138 male
139 male
140 male
141 male
142 male
143 male
144 male
145 male
146 male
147 male
148 male
149 male
150 female
151 male
152 male
153 female
154 male
155 male
156 male
157 male
158 male
159 male
160 female
161 male
162 female
163 male
164 female
165 male
166 male
167 female
168 male
169 male
170 male
171 male
172 male
173 male
174 female
175 male
176 male

```
177 male
178 male
179 male
180 male
181 male
182 female
183 male
184 female
185 female
186 male
187 male
188 male
189 male
190 male
191 male
192 male
193 male
194 female
195 male
196 male
197 female
198 male
199 male
200 male
```

```
rawdata_df[[2]] #returns vector
```

```
[1] "male"   "male"   "female" "female" "male"   "male"   "male"   "male"
[9] "female" "male"   "female" "male"   "male"   "male"   "male"   "female"
[17] "female" "male"   "female" "female" "female" "male"   "male"   "female"
[25] "female" "male"   "female" "male"   "female" "female" "male"   "male"
[33] "male"   "male"   "female" "male"   "female" "female" "female" "male"
[41] "female" "male"   "male"   "male"   "male"   "male"   "female" "female"
[49] "male"   "male"   "female" "male"   "male"   "male"   "male"   "male"
[57] "male"   "male"   "female" "female" "male"   "male"   "male"   "female"
[65] "female" "male"   "male"   "male"   "female" "female" "female" "female"
[73] "male"   "male"   "male"   "male"   "male"   "male"   "male"   "male"
[81] "female" "male"   "male"   "male"   "male"   "male"   "male"   "male"
[89] "male"   "male"   "male"   "male"   "female" "male"   "male"   "female"
[97] "male"   "female" "male"   "female" "male"   "male"   "male"   "male"
[105] "male"  "male"   "male"   "male"   "female" "male"   "male"   "male"
```

```

[113] "female" "male"   "female" "female" "male"   "female" "male"   "male"
[121] "female" "male"   "female" "male"   "male"   "male"   "female" "male"
[129] "female" "female" "female" "female" "female" "female" "male"   "female"
[137] "female" "male"   "male"   "male"   "male"   "male"   "male"   "male"
[145] "male"   "male"   "male"   "male"   "male"   "female" "male"   "male"
[153] "female" "male"   "male"   "male"   "male"   "male"   "male"   "female"
[161] "male"   "female" "male"   "female" "male"   "male"   "female" "male"
[169] "male"   "male"   "male"   "male"   "male"   "female" "male"   "male"
[177] "male"   "male"   "male"   "male"   "male"   "female" "male"   "female"
[185] "female" "male"   "male"   "male"   "male"   "male"   "male"   "male"
[193] "male"   "female" "male"   "male"   "female" "male"   "male"   "male"

```

```
rawdata_df[, 2] #returns vector
```

```

[1] "male"   "male"   "female" "female" "male"   "male"   "male"   "male"
[9] "female" "male"   "female" "male"   "male"   "male"   "male"   "female"
[17] "female" "male"   "female" "female" "female" "male"   "male"   "female"
[25] "female" "male"   "female" "male"   "female" "female" "male"   "male"
[33] "male"   "male"   "female" "male"   "female" "female" "female" "male"
[41] "female" "male"   "male"   "male"   "male"   "male"   "female" "female"
[49] "male"   "male"   "female" "male"   "male"   "male"   "male"   "male"
[57] "male"   "male"   "female" "female" "male"   "male"   "male"   "female"
[65] "female" "male"   "male"   "male"   "female" "female" "female" "female"
[73] "male"   "male"   "male"   "male"   "male"   "male"   "male"   "male"
[81] "female" "male"   "male"   "male"   "male"   "male"   "male"   "male"
[89] "male"   "male"   "male"   "male"   "female" "male"   "male"   "female"
[97] "male"   "female" "male"   "female" "male"   "male"   "male"   "male"
[105] "male"   "male"   "male"   "male"   "female" "male"   "male"   "male"
[113] "female" "male"   "female" "female" "male"   "female" "male"   "male"
[121] "female" "male"   "female" "male"   "male"   "male"   "female" "male"
[129] "female" "female" "female" "female" "female" "female" "male"   "female"
[137] "female" "male"   "male"   "male"   "male"   "male"   "male"   "male"
[145] "male"   "male"   "male"   "male"   "male"   "female" "male"   "male"
[153] "female" "male"   "male"   "male"   "male"   "male"   "male"   "female"
[161] "male"   "female" "male"   "female" "male"   "male"   "female" "male"
[169] "male"   "male"   "male"   "male"   "male"   "female" "male"   "male"
[177] "male"   "male"   "male"   "male"   "male"   "female" "male"   "female"
[185] "female" "male"   "male"   "male"   "male"   "male"   "male"   "male"
[193] "male"   "female" "male"   "male"   "female" "male"   "male"   "male"

```

```
rawdata_df[,2:3] #returns DF with 2 columns
```

	Sex	Ethnicity
1	male	White (not Hispanic)
2	male	White (not Hispanic)
3	female	White (not Hispanic)
4	female	White (not Hispanic)
5	male	White (not Hispanic)
6	male	White (not Hispanic)
7	male	White (not Hispanic)
8	male	White (not Hispanic)
9	female	Middle-Eastern, Arabic
10	male	White (not Hispanic)
11	female	White (not Hispanic)
12	male	Hispanic
13	male	Middle-Eastern, Arabic
14	male	White (not Hispanic)
15	male	White (not Hispanic)
16	female	White (not Hispanic)
17	female	White (not Hispanic)
18	male	White (not Hispanic)
19	female	White (not Hispanic)
20	female	White (not Hispanic)
21	female	Middle-Eastern, Arabic
22	male	White (not Hispanic)
23	male	White (not Hispanic)
24	female	Middle-Eastern, Arabic
25	female	Middle-Eastern, Arabic
26	male	White (not Hispanic)
27	female	White (not Hispanic)
28	male	Hispanic
29	female	White (not Hispanic)
30	female	Hispanic
31	male	White (not Hispanic)
32	male	Black (not Hispanic)
33	male	White (not Hispanic)
34	male	White (not Hispanic)
35	female	White (not Hispanic)
36	male	White (not Hispanic)
37	female	White (not Hispanic)
38	female	White (not Hispanic)

39	female	White (not Hispanic)
40	male	White (not Hispanic)
41	female	White (not Hispanic)
42	male	White (not Hispanic)
43	male	White (not Hispanic)
44	male	White (not Hispanic)
45	male	White (not Hispanic)
46	male	White (not Hispanic)
47	female	Asian or Pacific Islander
48	female	White (not Hispanic)
49	male	Middle-Eastern, Arabic
50	male	White (not Hispanic)
51	female	White (not Hispanic)
52	male	White (not Hispanic)
53	male	Black (not Hispanic)
54	male	White (not Hispanic)
55	male	White (not Hispanic)
56	male	White (not Hispanic)
57	male	White (not Hispanic)
58	male	White (not Hispanic)
59	female	White (not Hispanic)
60	female	White (not Hispanic)
61	male	White (not Hispanic)
62	male	Middle-Eastern, Arabic
63	male	White (not Hispanic)
64	female	White (not Hispanic)
65	female	White (not Hispanic)
66	male	White (not Hispanic)
67	male	Middle-Eastern, Arabic
68	male	White (not Hispanic)
69	female	White (not Hispanic)
70	female	Asian or Pacific Islander
71	female	Middle-Eastern, Arabic
72	female	White (not Hispanic)
73	male	White (not Hispanic)
74	male	White (not Hispanic)
75	male	White (not Hispanic)
76	male	White (not Hispanic)
77	male	White (not Hispanic)
78	male	White (not Hispanic)
79	male	White (not Hispanic)
80	male	White (not Hispanic)
81	female	Middle-Eastern, Arabic

82	male	White (not Hispanic)
83	male	White (not Hispanic)
84	male	White (not Hispanic)
85	male	White (not Hispanic)
86	male	Middle-Eastern, Arabic
87	male	Middle-Eastern, Arabic
88	male	White (not Hispanic)
89	male	White (not Hispanic)
90	male	White (not Hispanic)
91	male	White (not Hispanic)
92	male	White (not Hispanic)
93	female	White (not Hispanic)
94	male	White (not Hispanic)
95	male	White (not Hispanic)
96	female	White (not Hispanic)
97	male	Black (not Hispanic)
98	female	White (not Hispanic)
99	male	White (not Hispanic)
100	female	Middle-Eastern, Arabic
101	male	Black (not Hispanic)
102	male	Middle-Eastern, Arabic
103	male	White (not Hispanic)
104	male	White (not Hispanic)
105	male	White (not Hispanic)
106	male	Middle-Eastern, Arabic
107	male	White (not Hispanic)
108	male	White (not Hispanic)
109	female	White (not Hispanic)
110	male	White (not Hispanic)
111	male	White (not Hispanic)
112	male	Middle-Eastern, Arabic
113	female	White (not Hispanic)
114	male	White (not Hispanic)
115	female	White (not Hispanic)
116	female	Middle-Eastern, Arabic
117	male	White (not Hispanic)
118	female	White (not Hispanic)
119	male	White (not Hispanic)
120	male	White (not Hispanic)
121	female	White (not Hispanic)
122	male	White (not Hispanic)
123	female	White (not Hispanic)
124	male	White (not Hispanic)

125	male	Middle-Eastern, Arabic
126	male	White (not Hispanic)
127	female	White (not Hispanic)
128	male	White (not Hispanic)
129	female	White (not Hispanic)
130	female	White (not Hispanic)
131	female	American Indian or Native Alaskan
132	female	White (not Hispanic)
133	female	Middle-Eastern, Arabic
134	female	Hispanic
135	male	White (not Hispanic)
136	female	White (not Hispanic)
137	female	White (not Hispanic)
138	male	White (not Hispanic)
139	male	White (not Hispanic)
140	male	Black (not Hispanic)
141	male	White (not Hispanic)
142	male	White (not Hispanic)
143	male	White (not Hispanic)
144	male	White (not Hispanic)
145	male	Black (not Hispanic)
146	male	White (not Hispanic)
147	male	White (not Hispanic)
148	male	White (not Hispanic)
149	male	Middle-Eastern, Arabic
150	female	Hispanic
151	male	White (not Hispanic)
152	male	White (not Hispanic)
153	female	White (not Hispanic)
154	male	White (not Hispanic)
155	male	White (not Hispanic)
156	male	White (not Hispanic)
157	male	Middle-Eastern, Arabic
158	male	White (not Hispanic)
159	male	Middle-Eastern, Arabic
160	female	White (not Hispanic)
161	male	Hispanic
162	female	White (not Hispanic)
163	male	Middle-Eastern, Arabic
164	female	Middle-Eastern, Arabic
165	male	White (not Hispanic)
166	male	Middle-Eastern, Arabic
167	female	White (not Hispanic)

```

168 male White (not Hispanic)
169 male White (not Hispanic)
170 male White (not Hispanic)
171 male American Indian or Native Alaskan
172 male White (not Hispanic)
173 male White (not Hispanic)
174 female White (not Hispanic)
175 male White (not Hispanic)
176 male Black (not Hispanic)
177 male Hispanic
178 male White (not Hispanic)
179 male White (not Hispanic)
180 male White (not Hispanic)
181 male White (not Hispanic)
182 female White (not Hispanic)
183 male Middle-Eastern, Arabic
184 female Black (not Hispanic)
185 female Middle-Eastern, Arabic
186 male Middle-Eastern, Arabic
187 male Middle-Eastern, Arabic
188 male White (not Hispanic)
189 male White (not Hispanic)
190 male White (not Hispanic)
191 male White (not Hispanic)
192 male White (not Hispanic)
193 male White (not Hispanic)
194 female Middle-Eastern, Arabic
195 male White (not Hispanic)
196 male Middle-Eastern, Arabic
197 female Hispanic
198 male White (not Hispanic)
199 male White (not Hispanic)
200 male White (not Hispanic)

```

There are specific functions for picking columns or rows, especially useful in pipes.

```
rawdata %>% select(PatID:Ethnicity, `sysRR (mmHg)` :HR)
```

```
# A tibble: 200 x 6
  PatID Sex   Ethnicity      `sysRR (mmHg)` diaRR    HR
  <chr> <chr> <fct>          <dbl>     <dbl> <dbl>
1 168   male  White (not Hispanic) 120       80    100
2 169   male  White (not Hispanic) 125       85    105
3 170   male  White (not Hispanic) 130       90    110
4 171   male  American Indian or Native Alaskan 135       95    115
5 172   male  White (not Hispanic) 140       100   120
6 173   male  White (not Hispanic) 145       105   125
7 174   female White (not Hispanic) 150       110   130
8 175   male  White (not Hispanic) 155       115   135
9 176   male  Black (not Hispanic) 160       120   140
10 177   male  Hispanic 165       125   145
11 178   male  White (not Hispanic) 170       130   150
12 179   male  White (not Hispanic) 175       135   155
13 180   male  White (not Hispanic) 180       140   160
14 181   male  White (not Hispanic) 185       145   165
15 182   female White (not Hispanic) 190       150   170
16 183   male  Middle-Eastern, Arabic 195       155   175
17 184   female Black (not Hispanic) 200       160   180
18 185   female Middle-Eastern, Arabic 205       165   185
19 186   male  Middle-Eastern, Arabic 210       170   190
20 187   male  Middle-Eastern, Arabic 215       175   195
21 188   male  White (not Hispanic) 220       180   200
22 189   male  White (not Hispanic) 225       185   205
23 190   male  White (not Hispanic) 230       190   210
24 191   male  White (not Hispanic) 235       195   215
25 192   male  White (not Hispanic) 240       200   220
26 193   male  White (not Hispanic) 245       205   225
27 194   female Middle-Eastern, Arabic 250       210   230
28 195   male  White (not Hispanic) 255       215   235
29 196   male  Middle-Eastern, Arabic 260       220   240
30 197   female Hispanic 265       225   245
31 198   male  White (not Hispanic) 270       230   250
32 199   male  White (not Hispanic) 275       235   255
33 200   male  White (not Hispanic) 280       240   260

```

```

1 P 1   male   White (not Hispanic)          99    87    84
2 P 2   male   White (not Hispanic)         134    79    98
3 P 3   female White (not Hispanic)        129    68    81
4 P 4   female White (not Hispanic)        144    92    97
5 P 5   male   White (not Hispanic)        154    69    82
6 P 6   male   White (not Hispanic)        123    60    88
7 P 7   male   White (not Hispanic)        149    84    92
8 P 8   male   White (not Hispanic)        145    88    78
9 P 9   female Middle-Eastern, Arabic    118    62    93
10 P 10 male   White (not Hispanic)       110    59    98
# i 190 more rows

```

```
rawdata %>% select(PatID:Ethnicity, `sysRR (mmHg)` :HR) %>% slice(1:5)
```

```
# A tibble: 5 x 6
  PatID Sex    Ethnicity      `sysRR (mmHg)` diaRR    HR
  <chr> <chr>  <fct>           <dbl>     <dbl>    <dbl>
1 P 1   male   White (not Hispanic)      99       87     84
2 P 2   male   White (not Hispanic)    134      79     98
3 P 3   female White (not Hispanic)   129      68     81
4 P 4   female White (not Hispanic)   144      92     97
5 P 5   male   White (not Hispanic)   154      69     82
```

```
rawdata %>% select(contains('RR', ignore.case = F))
```

```
# A tibble: 200 x 2
`sysRR (mmHg)` diaRR
<dbl>    <dbl>
1          99     87
2         134     79
3         129     68
4         144     92
5         154     69
6         123     60
7         149     84
8         145     88
9         118     62
10        110     59
# i 190 more rows
```

```
rawdata %>% select(ends_with('r'))
```

```
# A tibble: 200 x 2
  diaRR     HR
  <dbl> <dbl>
1    87     84
2    79     98
3    68     81
4    92     97
5    69     82
6    60     88
7    84     92
8    88     78
9    62     93
10   59     98
# i 190 more rows
```

```
rawdata %>% select(-contains('name'))
```

```
# A tibble: 200 x 7
  PatID Sex   Ethnicity      Treatment `sysRR (mmHg)` diaRR     HR
  <chr> <chr> <fct>        <chr>           <dbl> <dbl> <dbl>
1 P 1   male  White (not Hispanic) Verum            99    87    84
2 P 2   male  White (not Hispanic) Verum           134    79    98
3 P 3   female White (not Hispanic) Placebo         129    68    81
4 P 4   female White (not Hispanic) Placebo         144    92    97
5 P 5   male  White (not Hispanic) Placebo         154    69    82
6 P 6   male  White (not Hispanic) Verum           123    60    88
7 P 7   male  White (not Hispanic) Placebo         149    84    92
8 P 8   male  White (not Hispanic) Placebo         145    88    78
9 P 9   female Middle-Eastern, Arabic Verum          118    62    93
10 P 10  male  White (not Hispanic) Verum          110    59    98
# i 190 more rows
```

```
rawdata %>% select(`sysRR (mmHg)`)
```

```
# A tibble: 200 x 1
`sysRR (mmHg)`
```

```

<dbl>
1      99
2     134
3     129
4     144
5     154
6     123
7     149
8     145
9     118
10    110
# i 190 more rows

rawdata %>% select(contains('r'), -contains('rr'))

# A tibble: 200 x 2
  Treatment     HR
  <chr>      <dbl>
1 Verum        84
2 Verum        98
3 Placebo      81
4 Placebo      97
5 Placebo      82
6 Verum        88
7 Placebo      92
8 Placebo      78
9 Verum        93
10 Verum       98
# i 190 more rows

rawdata %>% pull(`sysRR (mmHg)`)

[1]  99 134 129 144 154 123 149 145 118 110 139 137 135 136 148 144 136 126
[19] 134 132 143 120 107 130 132 144 150 110 121 138 126 144 139 137 127 121
[37] 148 125 142 129 157 154 131 160 126 139 128 138 137 141 112 145 142 116
[55] 117 128 123 123 134 120 123 119 105 139 136 143 128 119 131 138 113 133
[73] 141 140 129 109 136 145 132 140 128 149 155 120 137 147 141 135 137 129
[91] 117 152 140 114 132 125 129 114 111 133 143 130 133 157 111 126 138 138
[109] 135 152 131 123 129 122 125 126 130 155 131 126 135 124 126 129 152 145

```

```
[127] 138 133 133 147 120 121 140 148 129 135 126 114 116 127 118 136 129 117
[145] 141 148 128 133 131 133 131 144 129 126 121 121 118 148 118 136 128 117
[163] 123 131 111 150 145 122 119 122 114 142 150 139 131 128 108 140 133 145
[181] 115 145 99 132 126 133 128 114 118 110 131 113 133 132 130 143 147 138
[199] 123 142
```

2.5.4 List

While

```
shopping<-list(beverages=c('beer','water',
                             'gin(not Gordons!!)','tonic'),
                 snacks=c('chips','pretzels'),
                 nonfood=c('DVDs','Akku'),
                 mengen=1:10,
                 volumen=rnorm(50,100,2))

shopping

$beverages
[1] "beer"           "water"          "gin(not Gordons!!)"
[4] "tonic"

$snacks
[1] "chips"         "pretzels"

$nonfood
[1] "DVDs"          "Akku"

$mengen
[1] 1 2 3 4 5 6 7 8 9 10

$volumen
[1] 99.01911 102.26546 98.34278 101.51579 101.52151 98.32359 102.32438
[8] 101.08534 97.90134 101.19480 97.65527 100.26568 99.85359 100.54450
[15] 98.15791 98.20536 98.42395 100.43631 98.20100 99.50755 99.83395
[22] 101.77545 97.83997 100.63909 97.03825 98.15553 98.51111 102.49644
[29] 98.24594 101.82904 102.71481 98.13227 102.08017 99.79152 101.17492
[36] 100.53233 99.92241 98.79890 97.43198 98.63006 101.10832 98.62621
[43] 101.71247 100.88292 98.99049 102.91137 101.39126 98.41433 98.86586
[50] 98.37110
```

```
shopping$snacks

[1] "chips"      "pretzels"

shopping[1]      #returns a list

$beverages
[1] "beer"           "water"          "gin(not Gordons!!)"
[4] "tonic"

shopping[[1]]    #returns a vector

[1] "beer"           "water"          "gin(not Gordons!!)"
[4] "tonic"

str(shopping[1])

List of 1
$ beverages: chr [1:4] "beer" "water" "gin(not Gordons!!)" "tonic"

str(shopping[[1]])

chr [1:4] "beer" "water" "gin(not Gordons!!)" "tonic"

str(shopping$beverages)

chr [1:4] "beer" "water" "gin(not Gordons!!)" "tonic"

shopping[1][2]

$<NA>
NULL
```

```
shopping[[1]][2]
```

```
[1] "water"
```

```
shopping$beverages[2]
```

```
[1] "water"
```

```
t_out <- t.test(x = rnorm(n = 20, mean = 10, sd = 1),  
                  y = rnorm(20, 12, 1))  
str(t_out)
```

```
List of 10
```

```
$ statistic : Named num -4.55  
..- attr(*, "names")= chr "t"  
$ parameter : Named num 37.6  
..- attr(*, "names")= chr "df"  
$ p.value   : num 5.46e-05  
$ conf.int  : num [1:2] -2.173 -0.834  
..- attr(*, "conf.level")= num 0.95  
$ estimate  : Named num [1:2] 10.3 11.8  
..- attr(*, "names")= chr [1:2] "mean of x" "mean of y"  
$ null.value: Named num 0  
..- attr(*, "names")= chr "difference in means"  
$ stderr    : num 0.331  
$ alternative: chr "two.sided"  
$ method    : chr "Welch Two Sample t-test"  
$ data.name  : chr "rnorm(n = 20, mean = 10, sd = 1) and rnorm(20, 12, 1)"  
- attr(*, "class")= chr "htest"
```

```
t_out$p.value
```

```
[1] 5.458335e-05
```

```
#package rlist
```

```

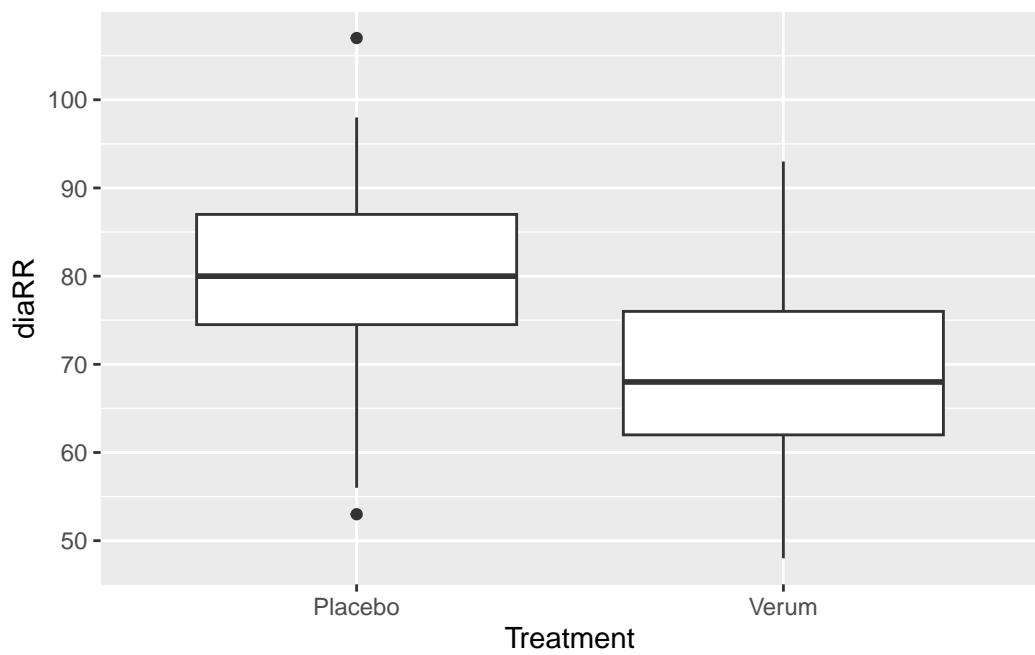
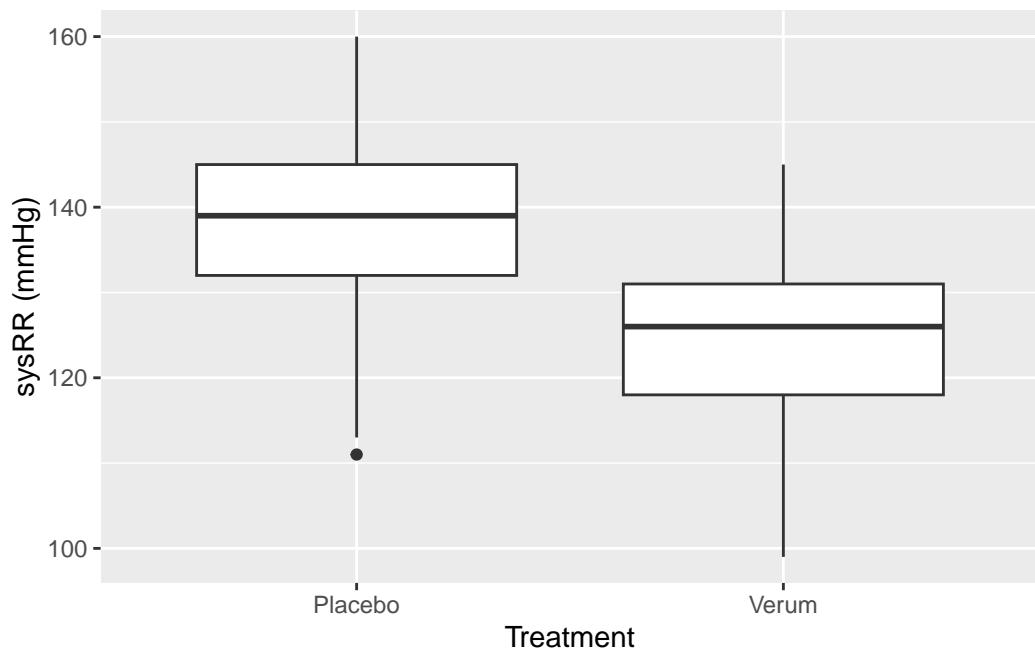
# tibble #####
testvars <- FindVars(varnames = c('Eth','Sex','R'))
rawdata %>% select(testvars$index)

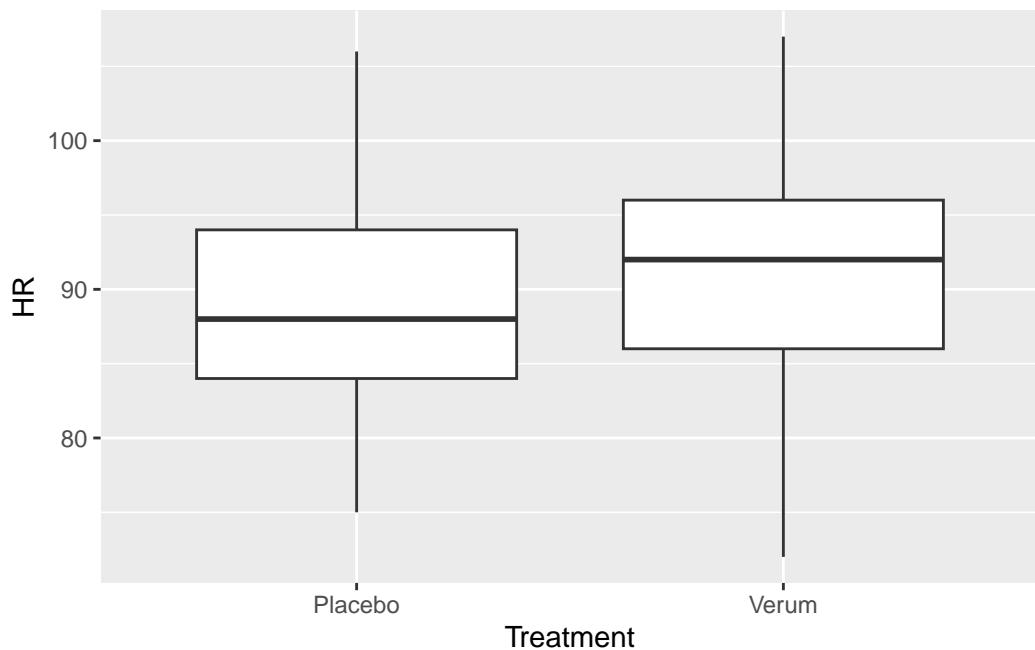
# A tibble: 200 x 5
  Sex    Ethnicity      `sysRR (mmHg)` diaRR     HR
  <chr>  <fct>          <dbl>   <dbl>   <dbl>
1 male   White (not Hispanic)      99     87     84
2 male   White (not Hispanic)     134     79     98
3 female White (not Hispanic)     129     68     81
4 female White (not Hispanic)     144     92     97
5 male   White (not Hispanic)     154     69     82
6 male   White (not Hispanic)     123     60     88
7 male   White (not Hispanic)     149     84     92
8 male   White (not Hispanic)     145     88     78
9 female Middle-Eastern, Arabic  118     62     93
10 male  White (not Hispanic)     110     59     98
# i 190 more rows

normvars <- FindVars(varnames = c('R'))
for(var_i in 3:5){
  print(ggplot(rawdata,aes_string(x = 'Treatment',
                                    y = testvars$bticked[var_i]))+
    geom_boxplot())
}

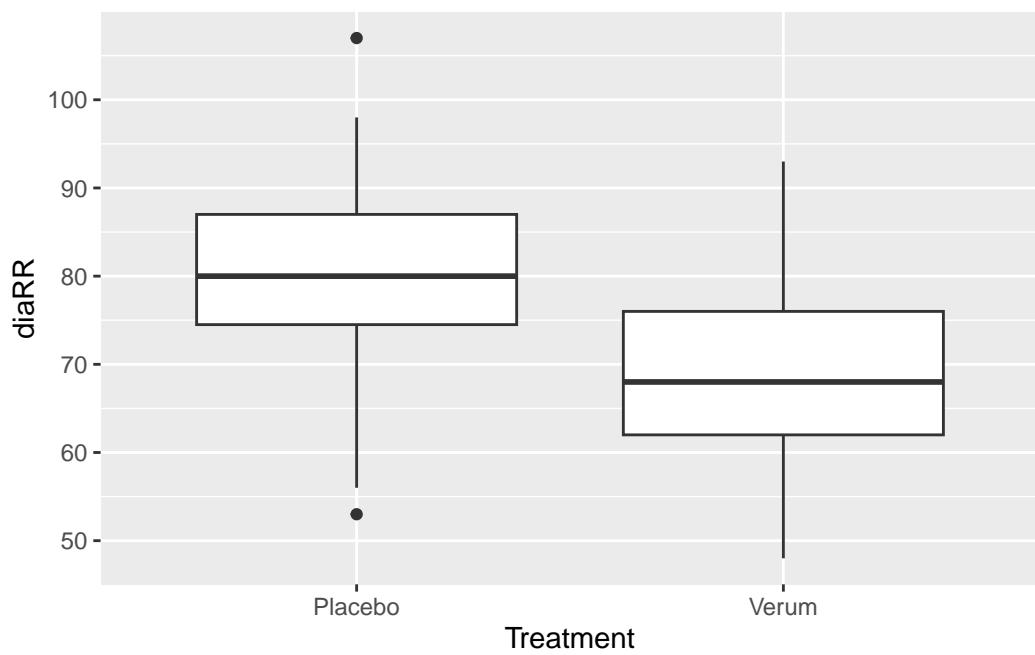
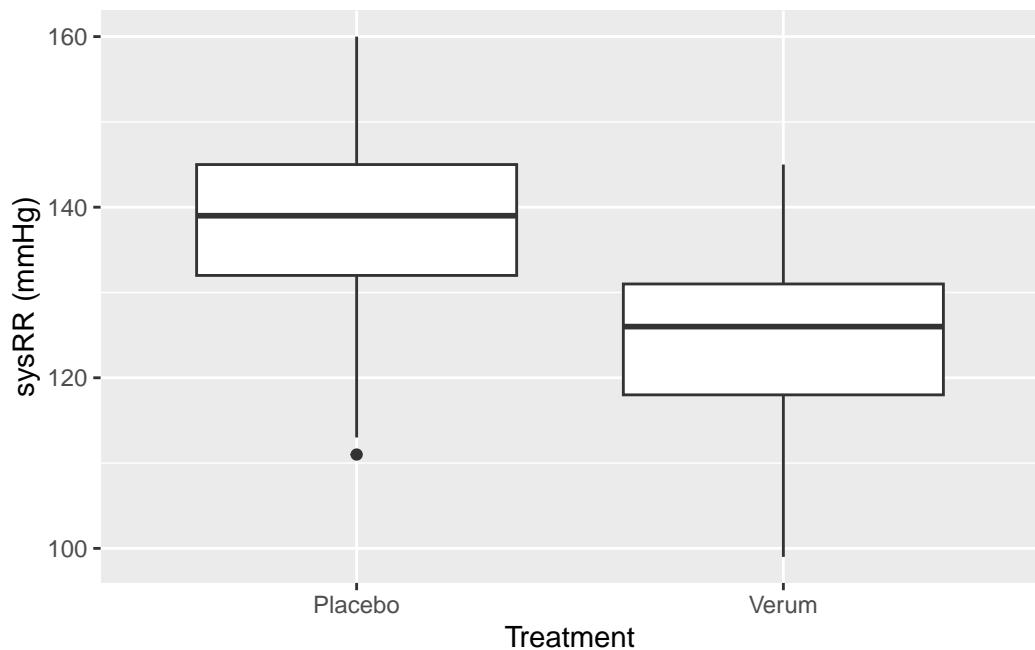
```

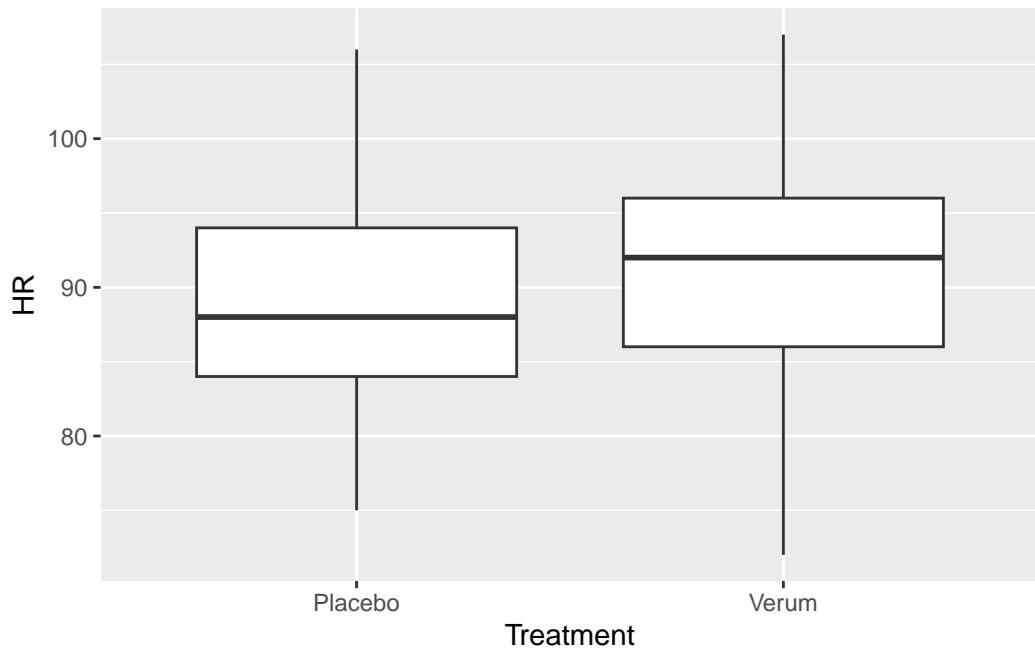
Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
 i Please use tidy evaluation idioms with `aes()`.
 i See also `vignette("ggplot2-in-packages")` for more information.





```
for(var_i in testvars$bticked[3:5]){
  print(ggplot(rawdata,aes_string(x = 'Treatment',
                                    y = var_i))+  
    geom_boxplot())
}
```





```
# Example cruet_stand / Gewürzmenage
n_elements <- 5*10^3
# tibble menage with columns saltshaker, peppercaster and n_elements each
# saltgrain and pepperflake
# print saltshaker
# print salt
# print 100 saltgrain

### functions ####
#FunctionName<-function(parameters...){definition}
markSign<-function(SignIn) {
  SignIn <- as.numeric(SignIn)
  if(is.na(SignIn)){
    SignOut<-'wrong input, stupido!'
  } else {
    # if (!is.na(SignIn)) {
    SignOut<-'n.s.'
```

```

    if (SignIn<=0.1) {SignOut<-'+'}
    if (SignIn<=0.05) {SignOut<-'*' }
    if (SignIn<=0.01) {SignOut<-'**'}
    if (SignIn<=0.001) {SignOut<-'***'}
}
return(SignOut)
}

markSign(SignIn=0.035)

```

[1] "*"

```
markSign(SignIn="0.35")
```

[1] "n.s."

```
markSign(SignIn = 'p=3,5%')      #wrong parameter
```

Warning in markSign(SignIn = "p=3,5%"): NAs durch Umwandlung erzeugt

[1] "wrong input, stupido!"

```

Mymean<-function(werte)
{
  return(base::mean(werte,na.rm=T))
}
# source('myfunctions.R')
### loops ####
print('### Game of Loops ###')

```

[1] "### Game of Loops ###"

```

for(season_i in 1:3) {
  cat(paste('GoL Season',season_i,'\n'))

```

```
for(episode_i in 1:5) {  
  cat(paste0('  GoL S.',season_i,  
            ' Episode ',episode_i,'\n'))  
}  
cat('\n')  
}
```

```
GoL Season 1  
  GoL S.1 Episode 1  
  GoL S.1 Episode 2  
  GoL S.1 Episode 3  
  GoL S.1 Episode 4  
  GoL S.1 Episode 5
```

```
GoL Season 2  
  GoL S.2 Episode 1  
  GoL S.2 Episode 2  
  GoL S.2 Episode 3  
  GoL S.2 Episode 4  
  GoL S.2 Episode 5
```

```
GoL Season 3  
  GoL S.3 Episode 1  
  GoL S.3 Episode 2  
  GoL S.3 Episode 3  
  GoL S.3 Episode 4  
  GoL S.3 Episode 5
```

```
for(col_i in colnames(rawdata)){  
  print(col_i)  
}
```

```
[1] "PatID"  
[1] "Sex"  
[1] "Ethnicity"  
[1] "Given name"  
[1] "Family name"  
[1] "Treatment"  
[1] "sysRR (mmHg)"  
[1] "diaRR"
```

```

[1] "HR"

for(col_i in shopping){
  print(col_i)
}

[1] "beer"           "water"          "gin(not Gordons!!)"
[4] "tonic"
[1] "chips"         "pretzels"
[1] "DVDs"          "Akku"
[1] 1 2 3 4 5 6 7 8 9 10
[1] 99.01911 102.26546 98.34278 101.51579 101.52151 98.32359 102.32438
[8] 101.08534 97.90134 101.19480 97.65527 100.26568 99.85359 100.54450
[15] 98.15791 98.20536 98.42395 100.43631 98.20100 99.50755 99.83395
[22] 101.77545 97.83997 100.63909 97.03825 98.15553 98.51111 102.49644
[29] 98.24594 101.82904 102.71481 98.13227 102.08017 99.79152 101.17492
[36] 100.53233 99.92241 98.79890 97.43198 98.63006 101.10832 98.62621
[43] 101.71247 100.88292 98.99049 102.91137 101.39126 98.41433 98.86586
[50] 98.37110

for(col_i in 1:length(colnames(rawdata))){
  print(colnames(rawdata)[col_i])
}

[1] "PatID"
[1] "Sex"
[1] "Ethnicity"
[1] "Given name"
[1] "Family name"
[1] "Treatment"
[1] "sysRR (mmHg)"
[1] "diaRR"
[1] "HR"

for(col_i in seq_along(colnames(rawdata))){
  print(colnames(rawdata)[col_i])
}

```

```

[1] "PatID"
[1] "Sex"
[1] "Ethnicity"
[1] "Given name"
[1] "Family name"
[1] "Treatment"
[1] "sysRR (mmHg)"
[1] "diaRR"
[1] "HR"

for(col_i in seq_len(0)){
  print(colnames(rawdata)[col_i])
}

test <- 0
while(test<100){
  print(test)
  test  <- test + 1 #test <- test+1
}

[1] 0
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
[1] 7
[1] 8
[1] 9
[1] 10
[1] 11
[1] 12
[1] 13
[1] 14
[1] 15
[1] 16
[1] 17
[1] 18
[1] 19

```

```
[1] 20  
[1] 21  
[1] 22  
[1] 23  
[1] 24  
[1] 25  
[1] 26  
[1] 27  
[1] 28  
[1] 29  
[1] 30  
[1] 31  
[1] 32  
[1] 33  
[1] 34  
[1] 35  
[1] 36  
[1] 37  
[1] 38  
[1] 39  
[1] 40  
[1] 41  
[1] 42  
[1] 43  
[1] 44  
[1] 45  
[1] 46  
[1] 47  
[1] 48  
[1] 49  
[1] 50  
[1] 51  
[1] 52  
[1] 53  
[1] 54  
[1] 55  
[1] 56  
[1] 57  
[1] 58  
[1] 59  
[1] 60  
[1] 61  
[1] 62
```

```
[1] 63
[1] 64
[1] 65
[1] 66
[1] 67
[1] 68
[1] 69
[1] 70
[1] 71
[1] 72
[1] 73
[1] 74
[1] 75
[1] 76
[1] 77
[1] 78
[1] 79
[1] 80
[1] 81
[1] 82
[1] 83
[1] 84
[1] 85
[1] 86
[1] 87
[1] 88
[1] 89
[1] 90
[1] 91
[1] 92
[1] 93
[1] 94
[1] 95
[1] 96
[1] 97
[1] 98
[1] 99

### conditions -----
sex<-'male'
if (sex=='male') {
```

```
    print('Male')
} else {
    print('Female')
}
```

```
[1] "Male"
```

```
if (sex=='male') {
    print('Male')
}
```

```
[1] "Male"
```

```
if (sex!='male'){
    print('Female')
}
```

```
if (sex!='male') {
    print('not male')
} else {
    print('male')
}
```

```
[1] "male"
```

```
if (!sex=='male') {
    print('not male')
} else {
    print('male')
}
```

```
[1] "male"
```

```
print(ifelse(test = sex=='male',
            yes = 'is male',
            no = 'is female'))
```



```
[1] "is male"
```



```
p <- .0012
paste0('That is ',
       ifelse(test = p<=.05, yes = '', no = 'not '),
       'significant')
```



```
[1] "That is significant"
```

```
testvar <- 3
if(testvar %in% c(1,3,5)){
  print('uneven')
}
```

```
[1] "uneven"
```

```
TRUE&FALSE
```

```
[1] FALSE
```

```
(1>10)&(1<5)
```

```
[1] FALSE
```

```
TRUE|FALSE
```

```
[1] TRUE
```

```
(1>10) | (1<5)
```

```
[1] TRUE
```

```
#case_when  
#### useful tools -----  
?any  
?'%in%'  
2%in%1:5
```

```
[1] TRUE
```

```
2%in%c(1,3,4,5)
```

```
[1] FALSE
```

3 Importing data

```
pacman::p_load(tidyverse, wrappedtools, readxl)
```

3.1 Import from text files (.txt, .csv)

3.2 Import from Excel

3.2.1 Tidy Excel files

```
rawdata <- read_excel('data/Medtest_e.xlsx')
rawdata <- rename(rawdata,
  `Size (cm)` = size,
  `Weight (kg)` = weight) |>
  select(-Sex_m)
saveRDS(rawdata, file = "data/rawdata.rds")
```

3.2.2 Dirty Excel files

3.3 Import from SPSS

3.4 Import from SAS

4 Grouping of variables by type / distribution / use

```
pacman::p_load(wrappedtools)
```

4.1 Test for normal distribution

4.2 Picking column names and positions

```
rawdata <- readRDS('data/rawdata.rds')
quantvars <- ColSeeker(data = rawdata,
                        varclass = "numeric",
                        exclude = "code")
gaussvars <- ColSeeker(namepattern = c("si","we","BMI","BP","mri"))

ordvars <- ColSeeker(namepattern = c("age","lab"))

factvars <- ColSeeker(namepattern = c("sex","med","NYHA"),
                      returnclass = TRUE)
save(rawdata,list = ls(pattern = "vars"),file = "data/bookdata1.RData")
```

5 Visualize data with ggplot

```
pacman::p_load(tidyverse,#plotrix,
                grid,gridExtra,car,
                ggsci,ggsignif, ggthemes, ggridges,
                ganimate,ggforce,
                # survival, survminer,
                ggdendro, ggbeeswarm,
                rpart,rpart.plot,
                gapminder,
                wrappedtools)

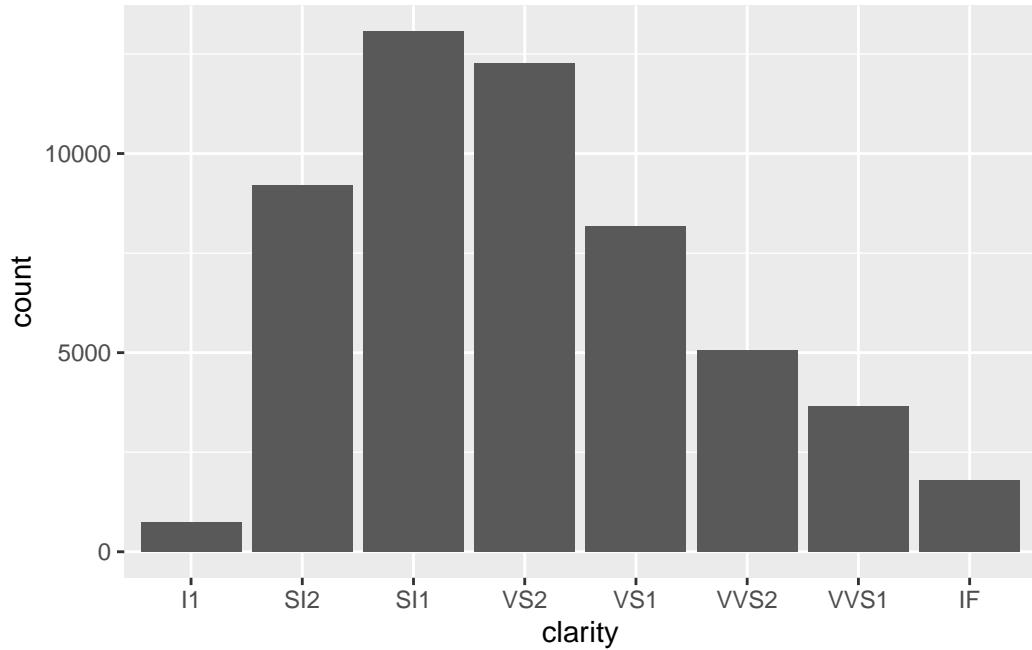
#sample data
head(diamonds)

# A tibble: 6 x 10
  carat cut      color clarity depth table price     x     y     z
  <dbl> <ord>    <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
1 0.23 Ideal    E      SI2      61.5   55    326  3.95  3.98  2.43
2 0.21 Premium  E      SI1      59.8   61    326  3.89  3.84  2.31
3 0.23 Good     E      VS1      56.9   65    327  4.05  4.07  2.31
4 0.29 Premium  I      VS2      62.4   58    334  4.2   4.23  2.63
5 0.31 Good     J      SI2      63.3   58    335  4.34  4.35  2.75
6 0.24 Very Good J      VVS2     62.8   57    336  3.94  3.96  2.48

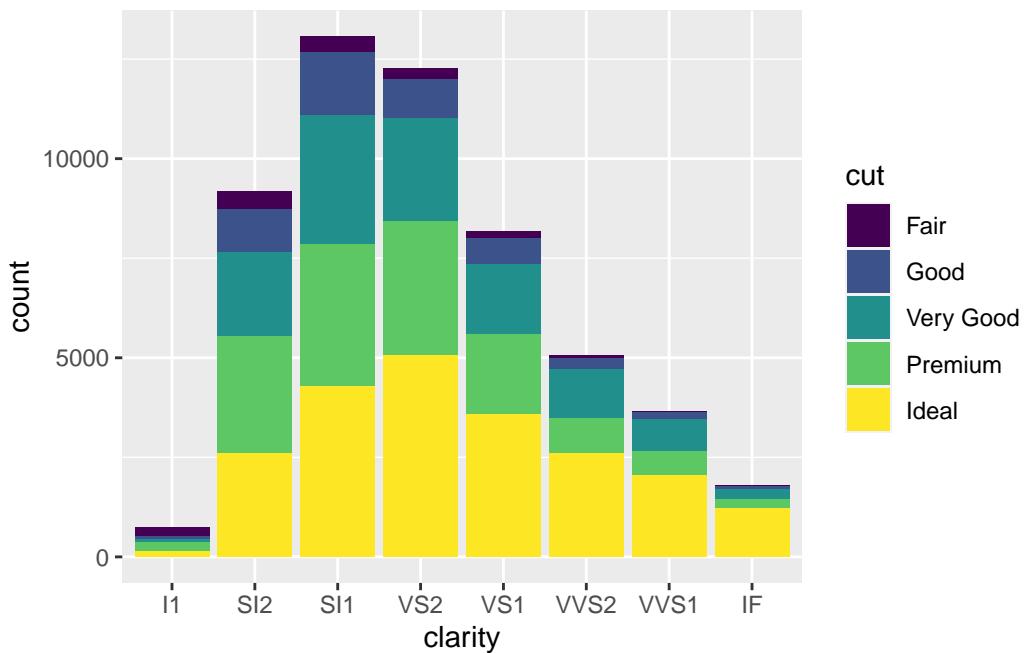
head(mtcars)

          mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4     21.0   6 160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag 21.0   6 160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710    22.8   4 108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive 21.4   6 258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8 360 175 3.15 3.440 17.02  0  0    3    2
Valiant       18.1   6 225 105 2.76 3.460 20.22  1  0    3    1
```

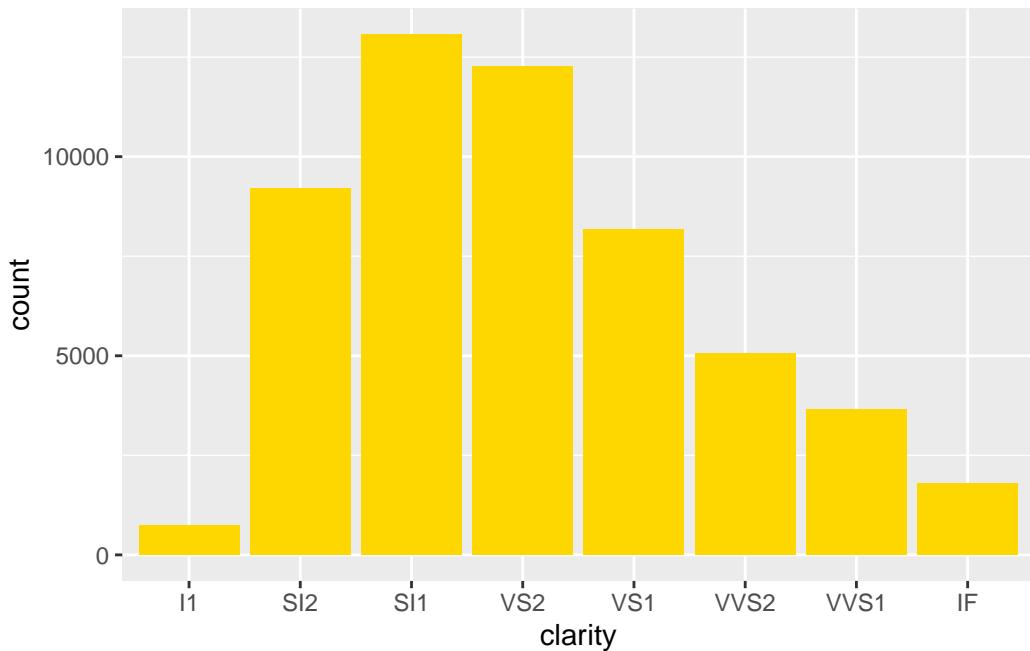
```
ggplot(data=diamonds,mapping = aes(x=clarity))+  
  geom_bar()
```



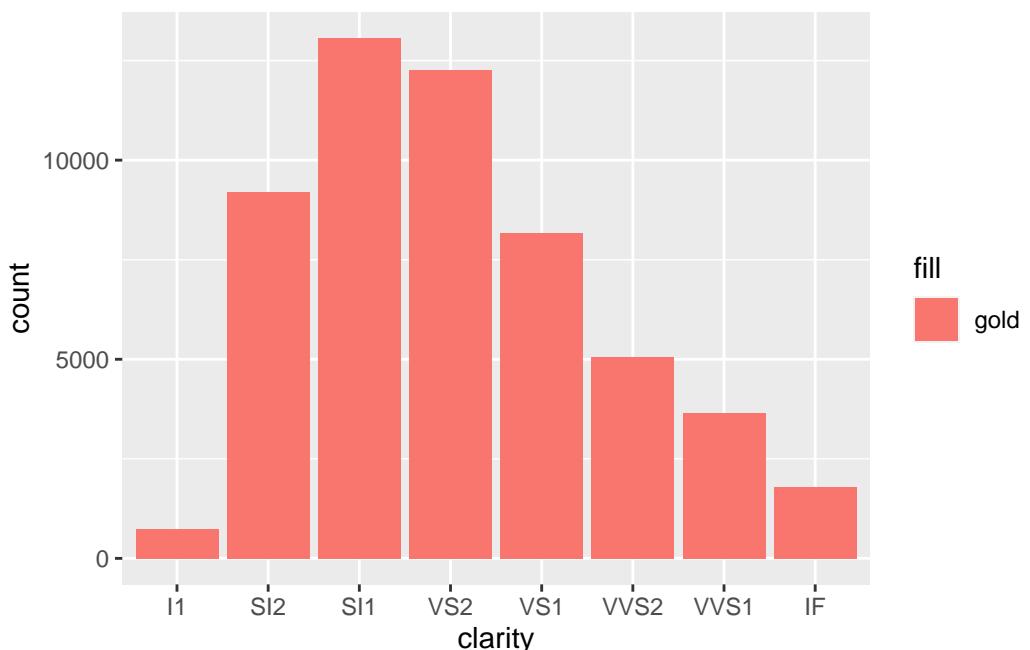
```
#define aesthetics  
ggplot(data=diamonds,aes(x=clarity,fill=cut))+  
  geom_bar()
```



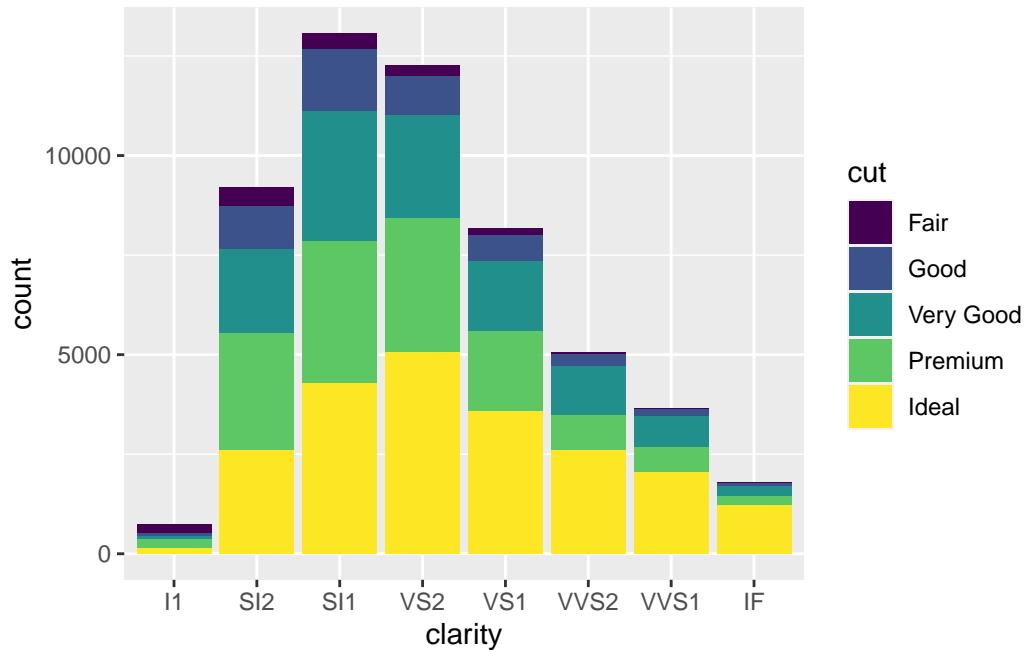
```
#aesthetics outside aes
ggplot(data=diamonds,aes(x=clarity))+
  geom_bar(fill='gold')
```



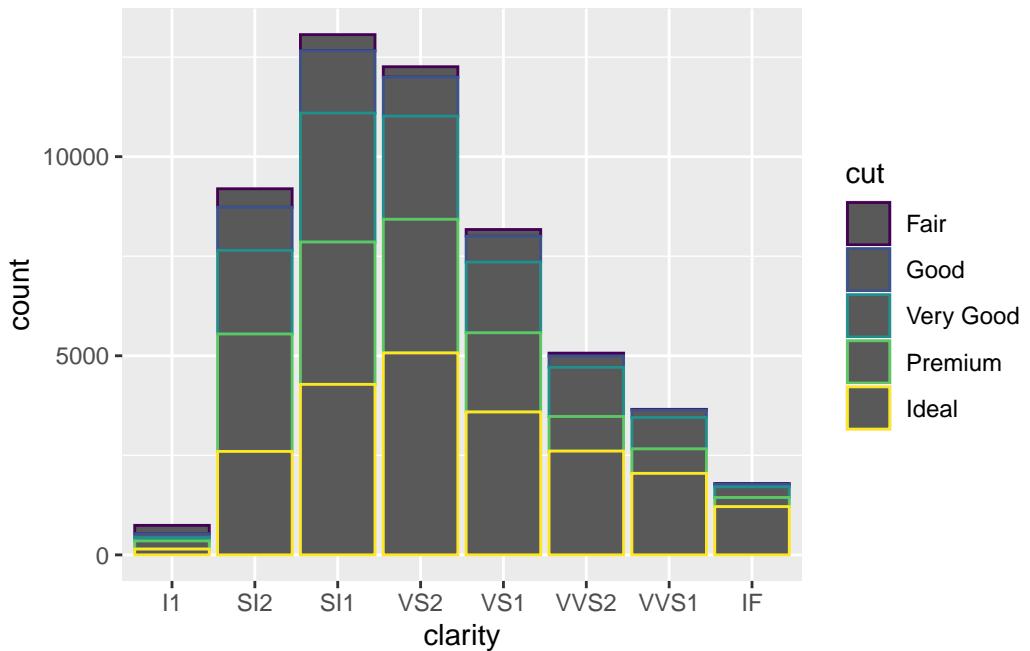
```
ggplot(data=diamonds,aes(x=clarity))+  
  geom_bar(aes(fill='gold')) #should be outside aes!
```



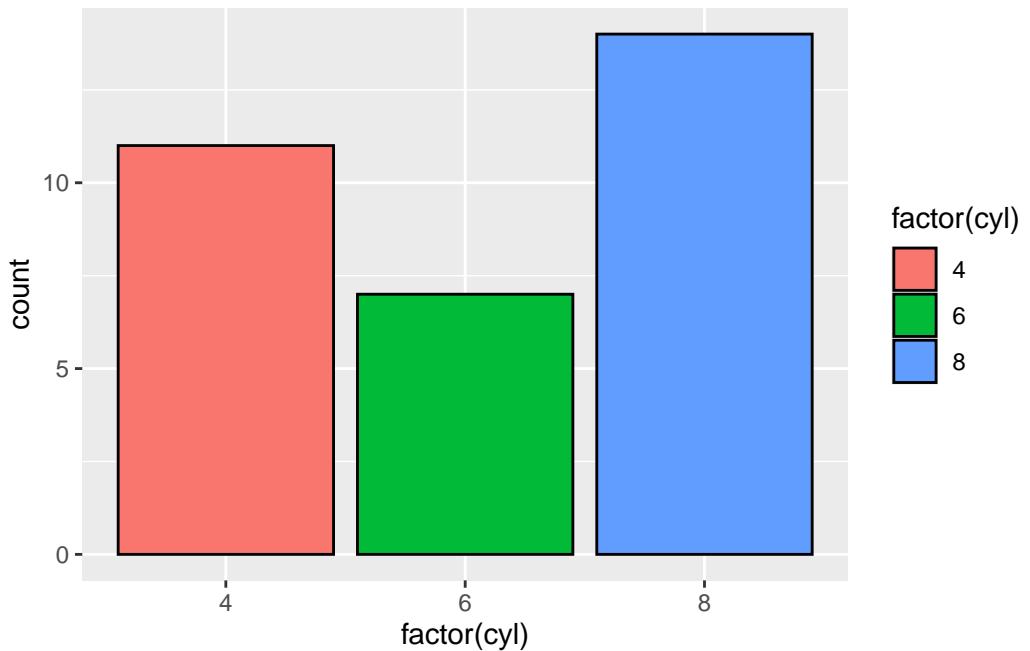
```
ggplot(data=diamonds,aes(x=clarity))+  
  geom_bar(aes(fill=cut))
```



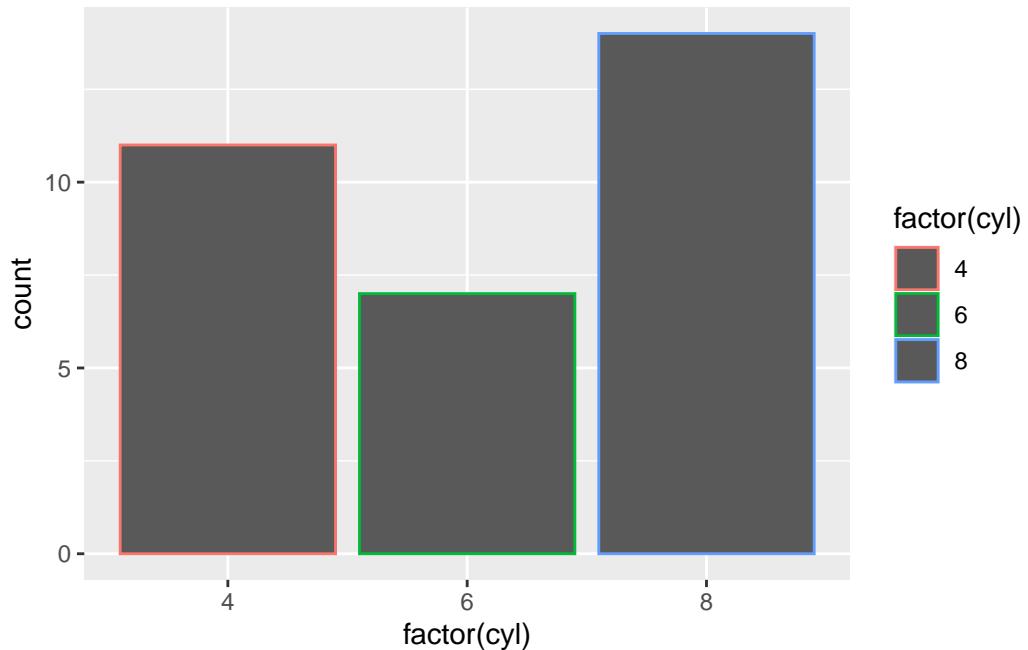
```
#fill/color  
ggplot(data=diamonds,aes(x=clarity,color=cut))+  
  geom_bar()
```



```
ggplot(data=mtcars,aes(factor(cyl),fill=factor(cyl)))+
  geom_bar(color='black')
```



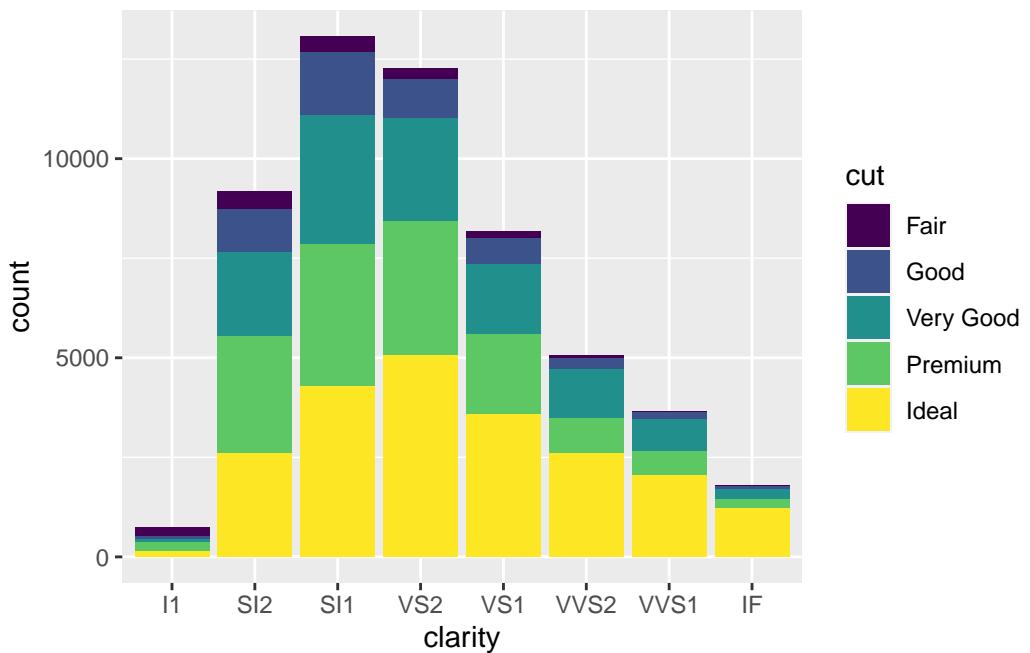
```
ggplot(data=mtcars,aes(factor(cyl),color=factor(cyl)))+  
  geom_bar()
```



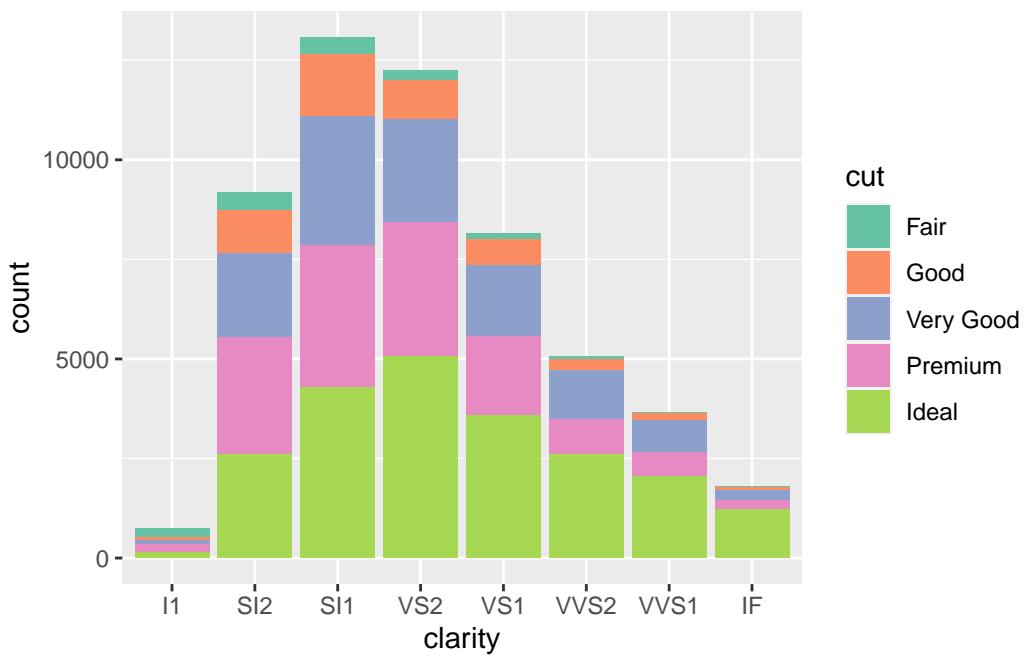
```
help(points)
```

starte den http Server für die Hilfe fertig

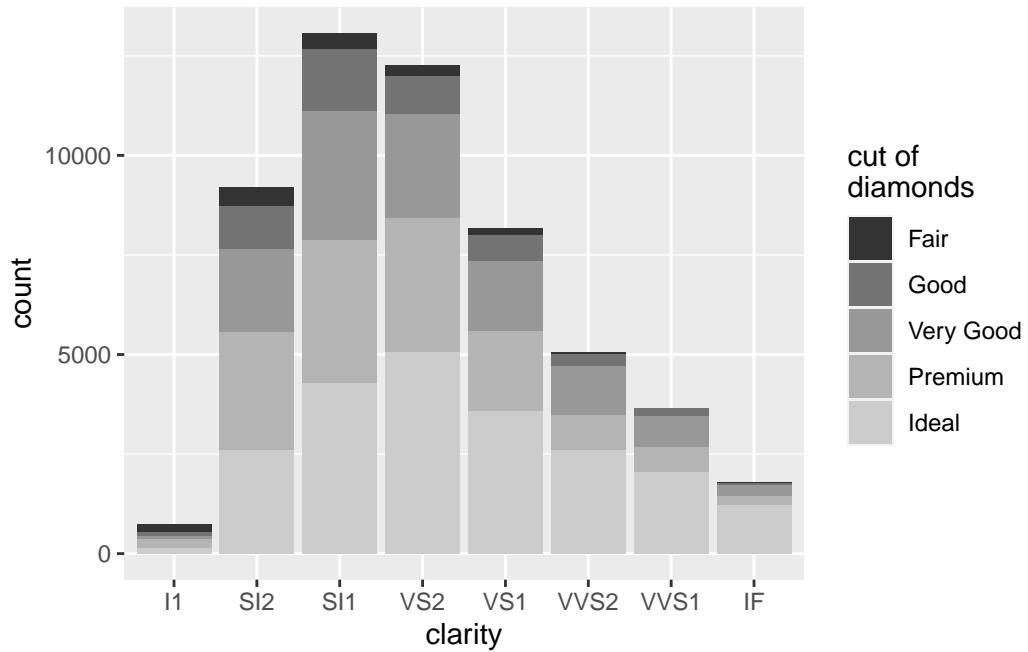
```
#other color systems  
(plottemp <- ggplot(data=diamonds,aes(x=clarity,fill=cut))+  
  geom_bar())
```



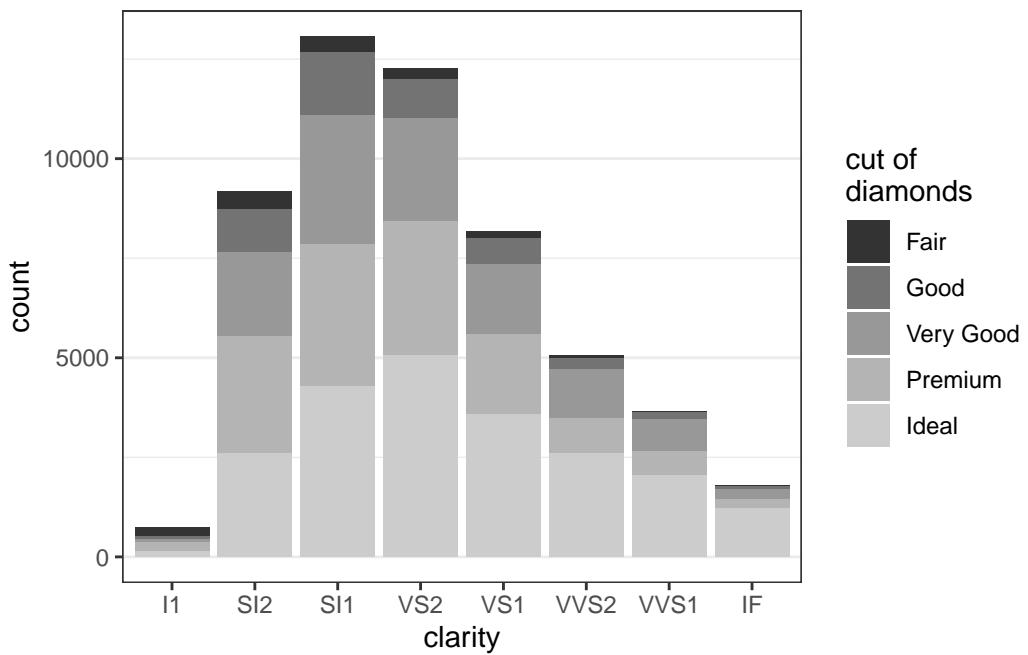
```
plottemp + scale_fill_brewer(palette='Set2') #in-built
```



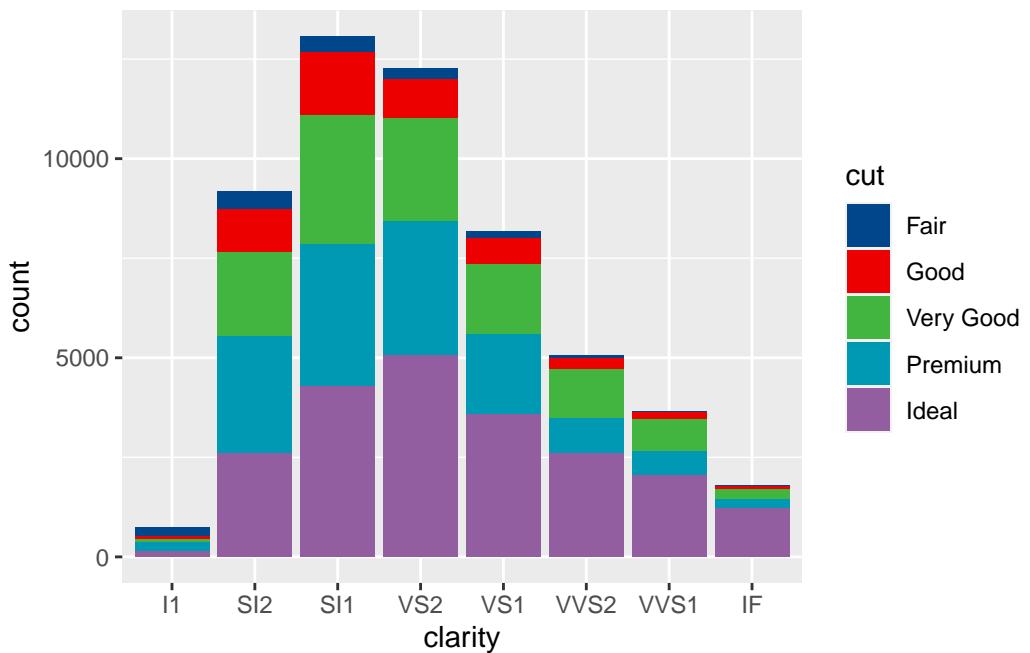
```
plottemp + scale_fill_grey(name = "cut of\ndiamonds")
```



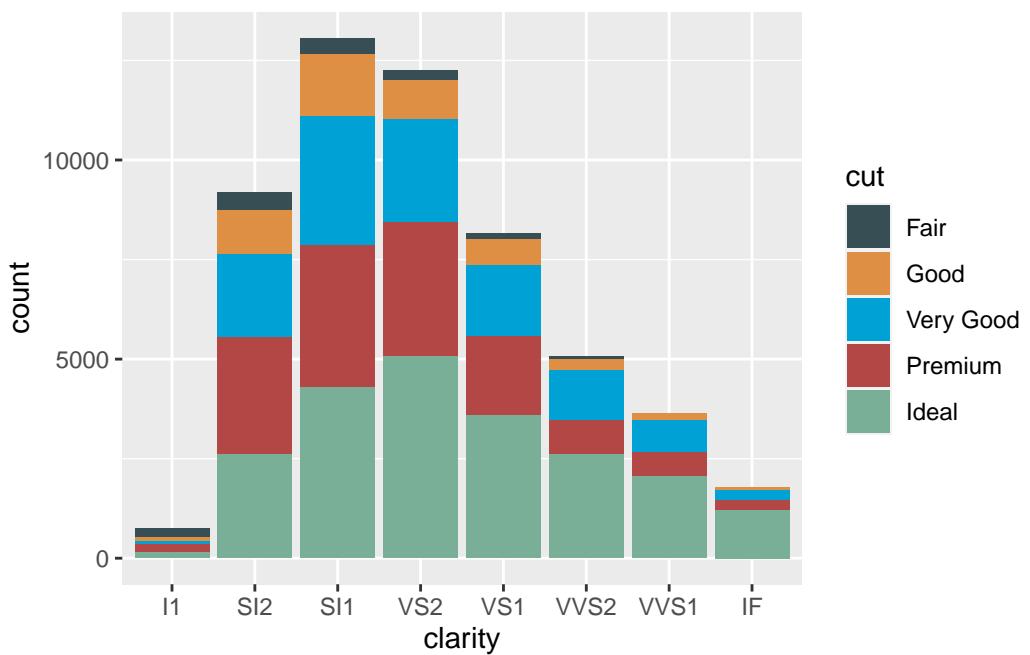
```
plottemp + scale_fill_grey(name = "cut of\ndiamonds") +  
  theme_bw() +  
  theme(panel.grid.minor.x=element_blank(),  
        panel.grid.major.x = element_blank())#+
```



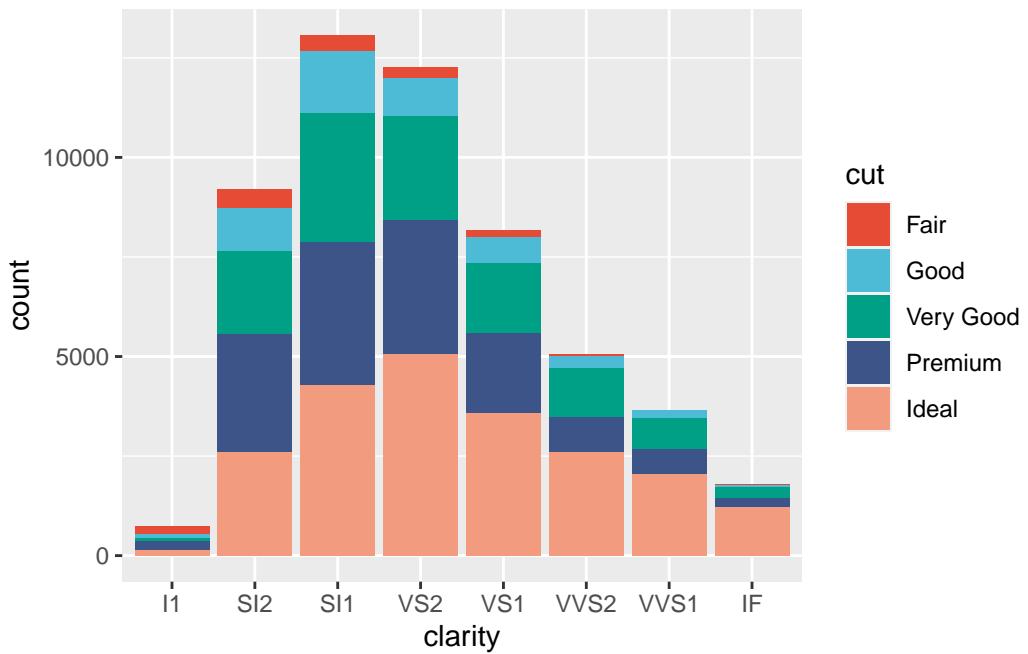
```
# scale_y_continuous(expand = expansion(mult = c(0,.5)))  
# ggsci #####  
plottemp+scale_fill_lancet()
```



```
plottemp+scale_fill_jama()
```



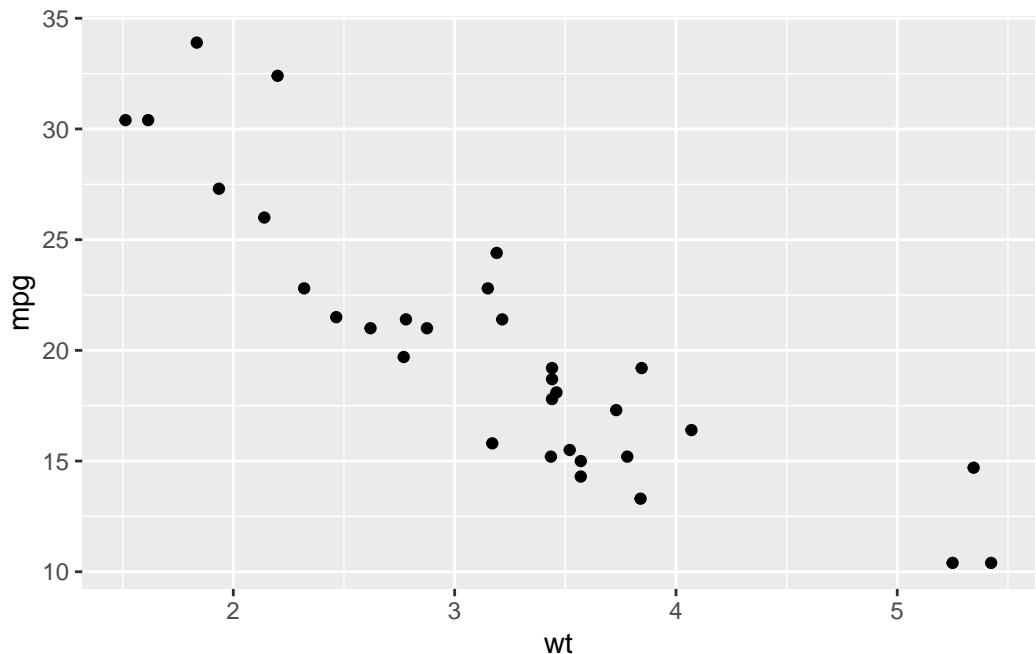
```
plottemp+scale_fill_npg()
```



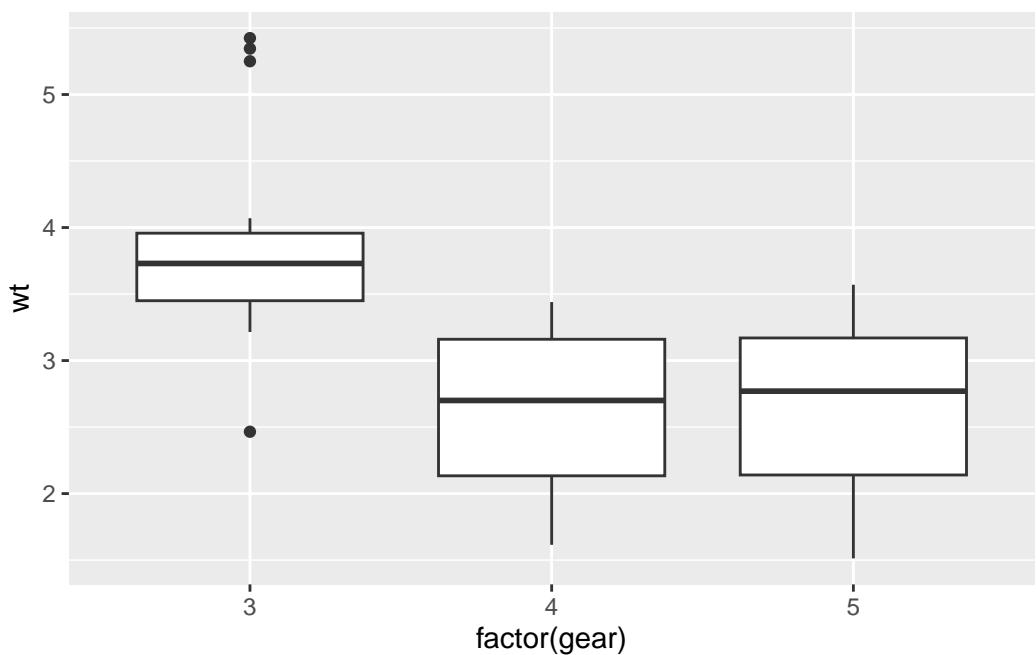
```
printplot <- plottemp+scale_fill_startrek()
```

```
#save ggplots
# ggsave(filename = 'Graphs/ggtestplot.png',
#        plot = printplot,
#        width=20,height=20,
#        units='cm',dpi=150)
# alternative:
# png(filename = 'Graphs/ggtestplot2.png',
#      width = 20,height = 20,units = 'cm',res = 150)
# plottemp
# dev.off()
# scatterplot

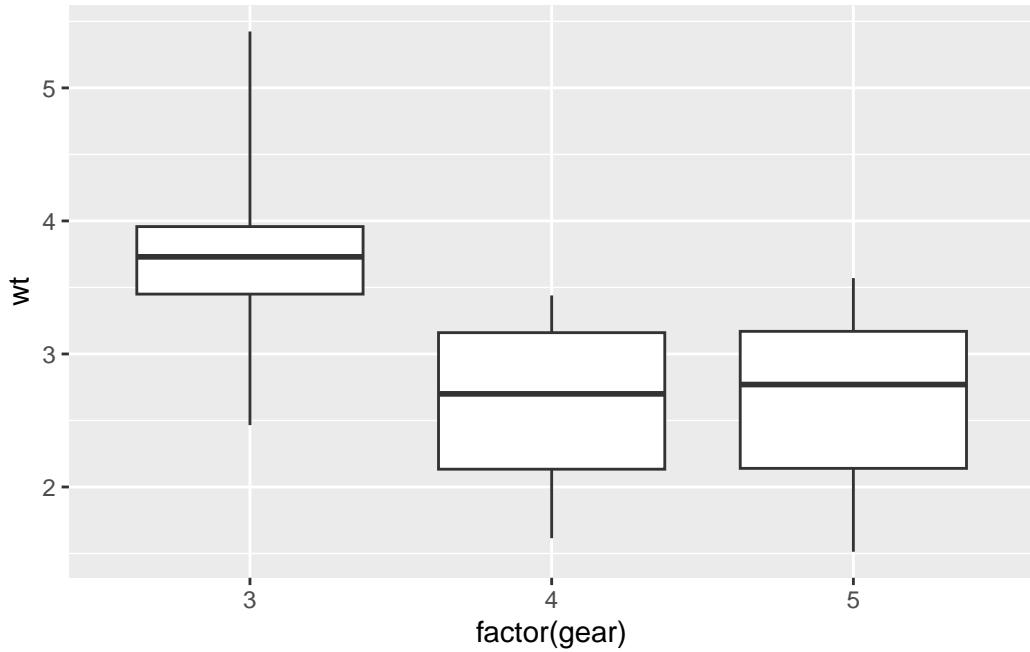
#other geoms
ggplot(data=mtcars,aes(x = wt,y = mpg))+  
  geom_point()
```



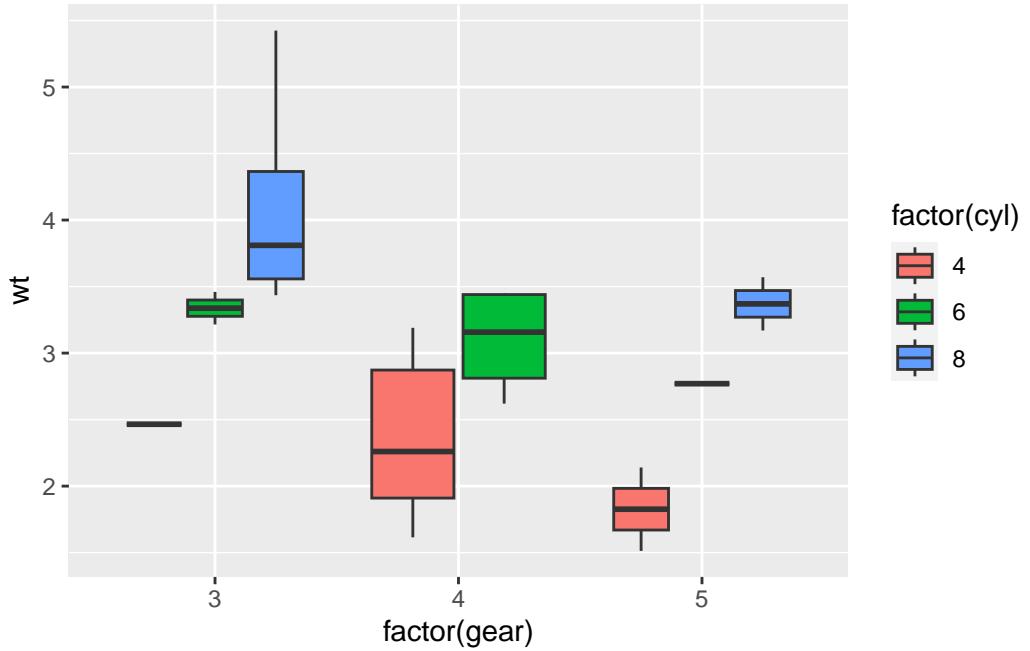
```
ggplot(mtcars,aes(x = factor(gear),y = wt))+  
  geom_boxplot() #default 1.5 IQR
```



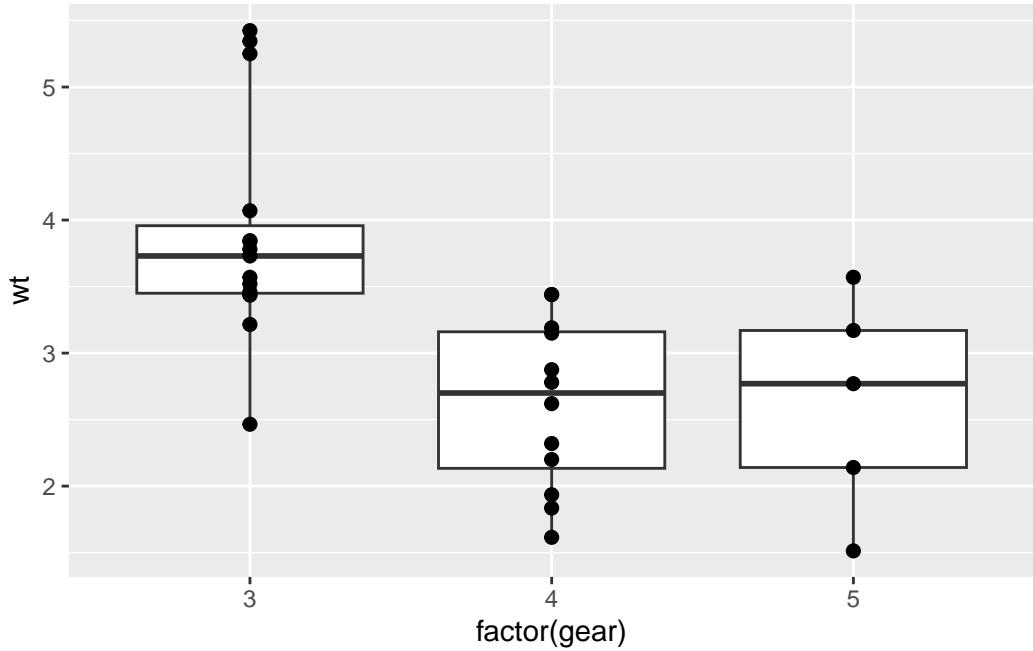
```
ggplot(mtcars,aes(x = factor(gear),y = wt))+  
  geom_boxplot(coef=3)
```



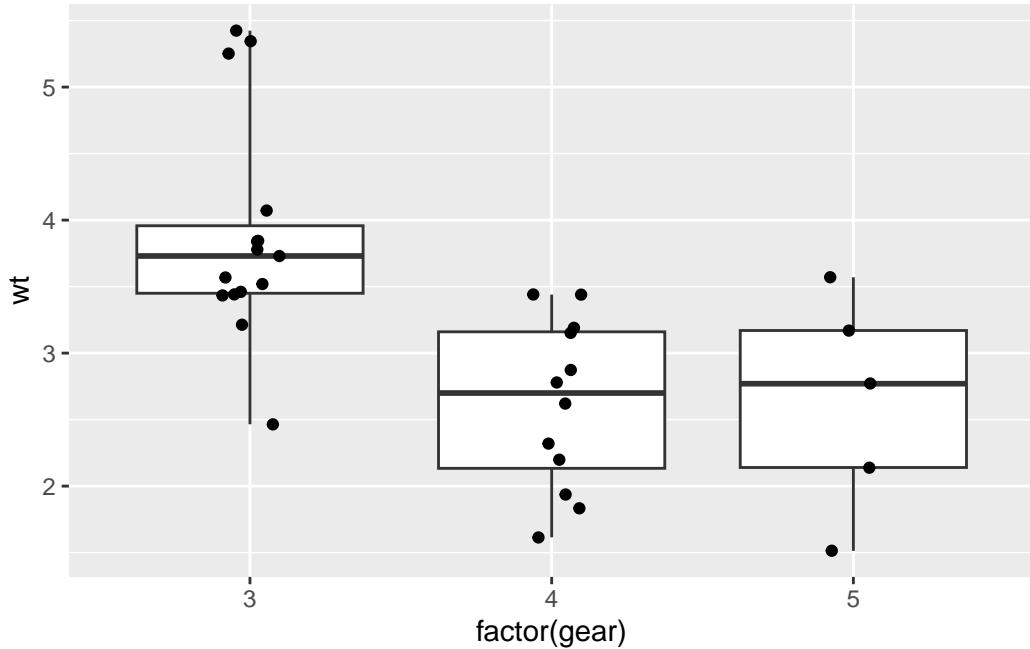
```
ggplot(mtcars,aes(x = factor(gear),y = wt,  
                   fill=factor(cyl)))+  
  geom_boxplot(coef=3)
```



```
ggplot(mtcars,aes(x = factor(gear),y = wt))+  
  geom_boxplot(coef=3)+  
  geom_point(size=2)
```

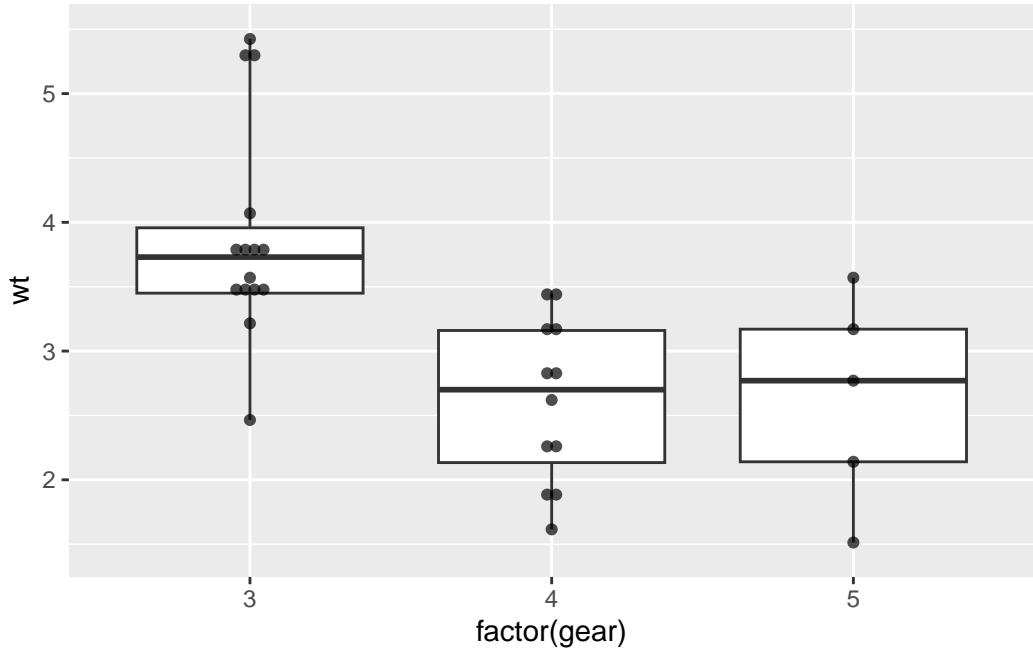


```
ggplot(mtcars,aes(x = factor(gear),y = wt))+  
  geom_boxplot(coef=3)+  
  geom_point(position = position_jitter(width = .1))
```

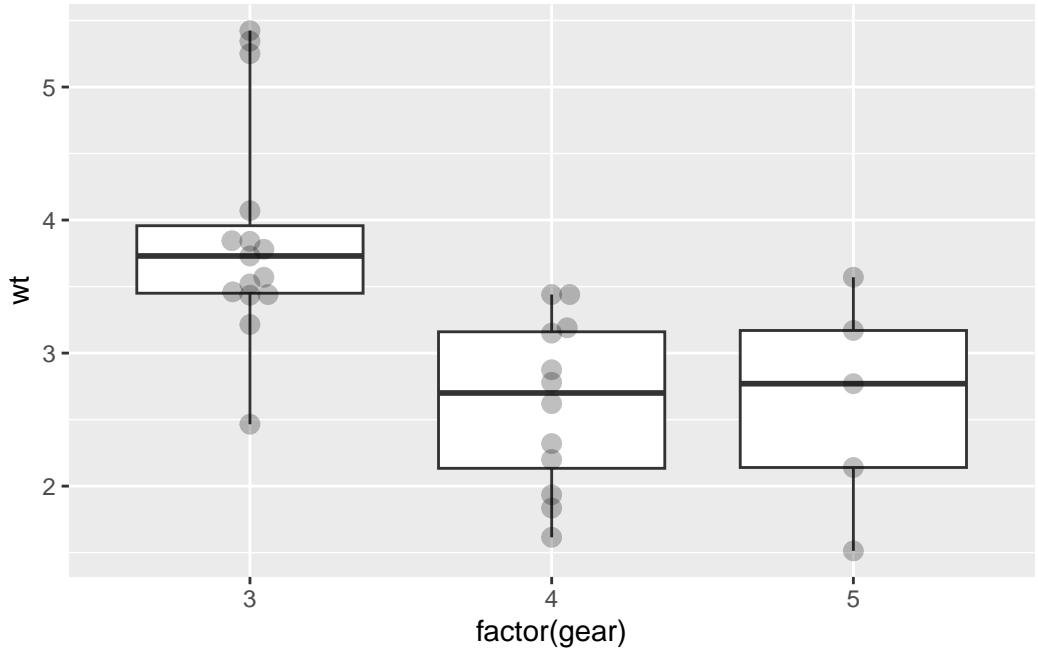


```
ggplot(mtcars,aes(x = factor(gear),y = wt))+  
  geom_boxplot(coef=3)+  
  geom_dotplot(alpha=.7,  
              binaxis = 'y',stackdir = 'center',  
              stackratio = .9,dotsize = .6)
```

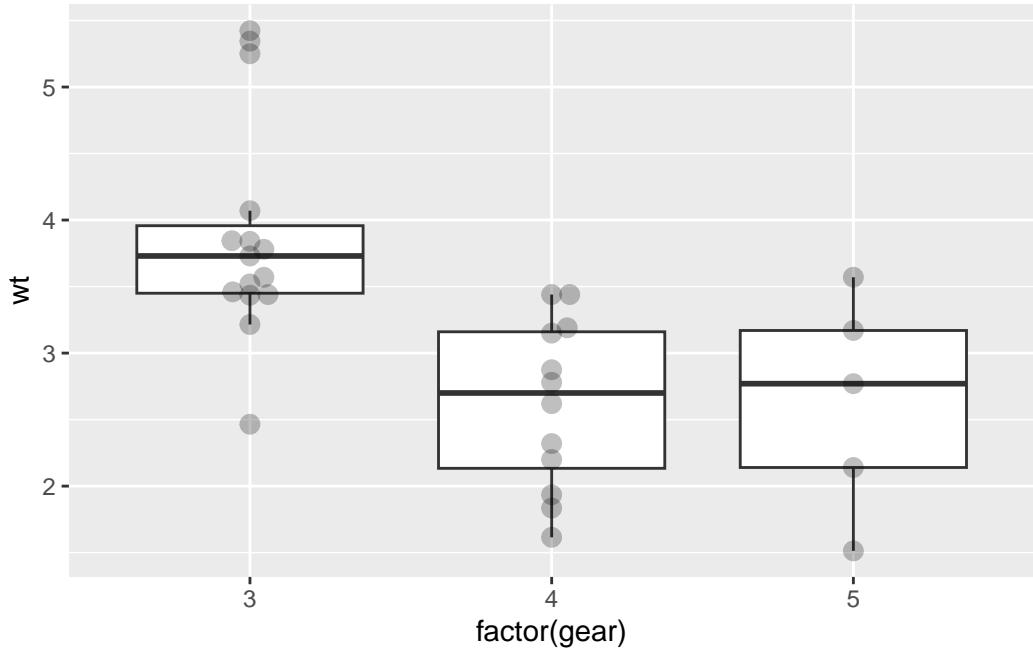
Bin width defaults to 1/30 of the range of the data. Pick better value with `binwidth`.



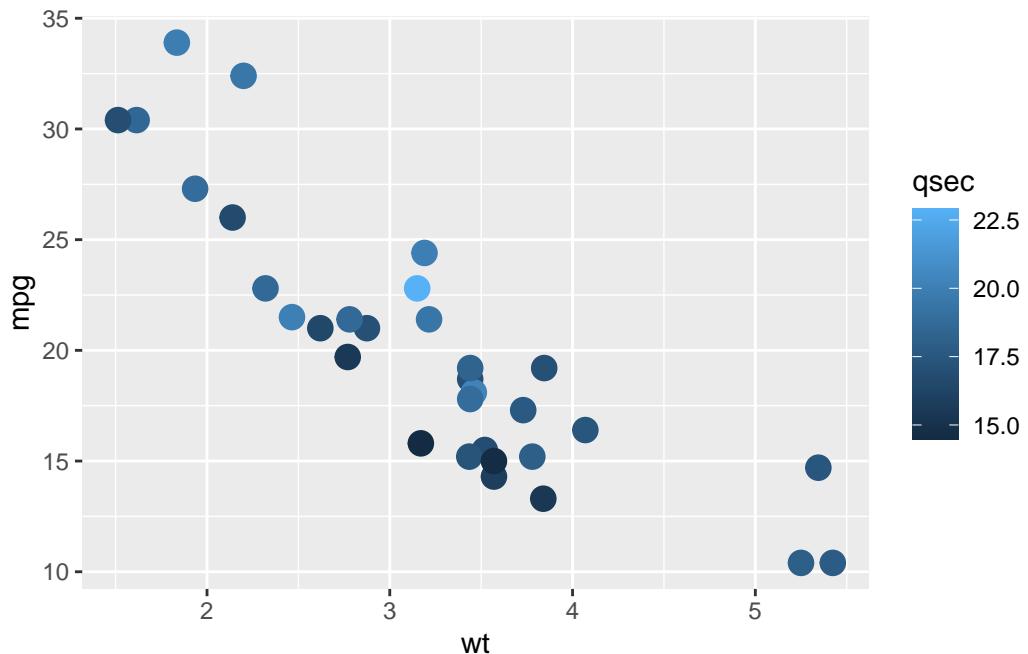
```
ggplot(mtcars,aes(x = factor(gear),y = wt))+  
  geom_boxplot(coef=3)+  
  ggbeeswarm::geom_beeswarm(cex = 2,size=3,alpha=.25)
```



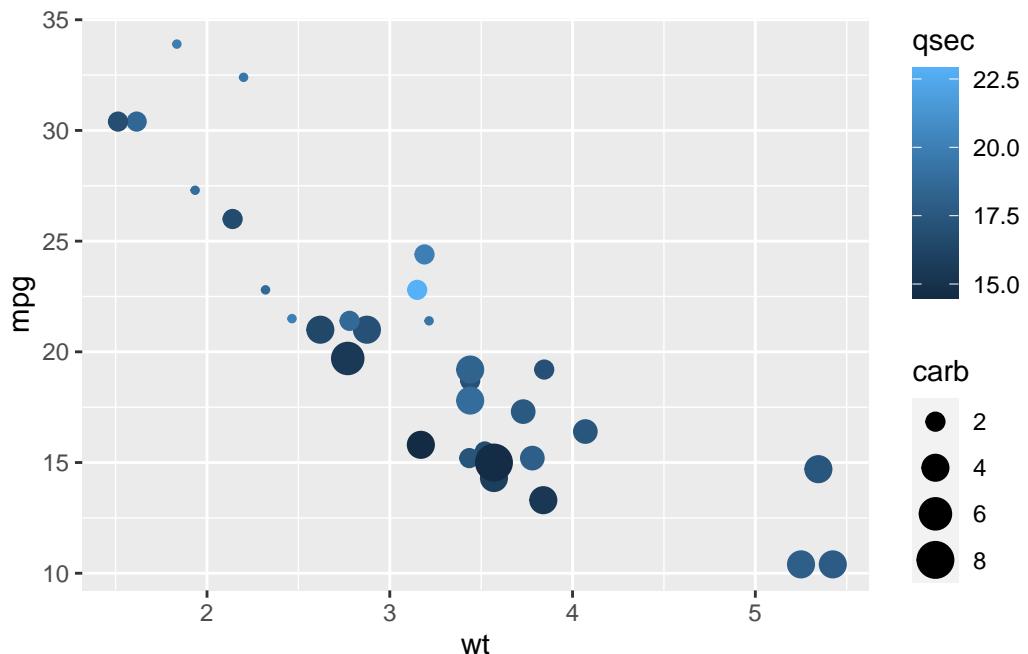
```
ggplot(mtcars,aes(x = factor(gear),y = wt))+  
  geom_boxplot(outlier.alpha = 0)+  
  geom_beeswarm(cex = 2,size=3,alpha=.25)
```



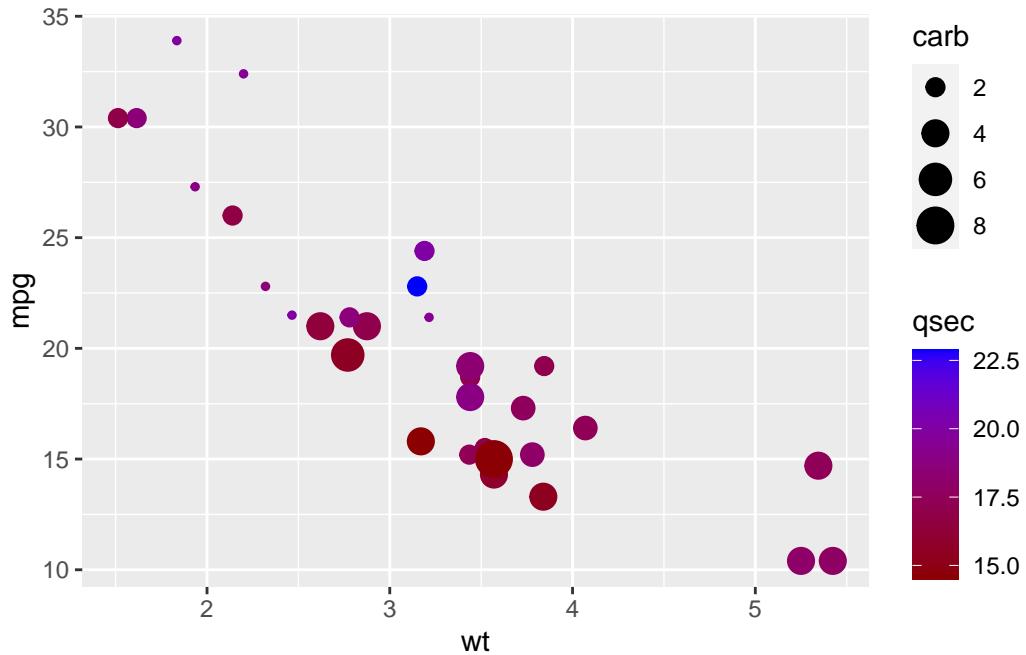
```
#aesthetics again (finetuning)
ggplot(data=mtcars,aes(wt, mpg,color=qsec))+  
  geom_point(size=4) #outside aes!
```



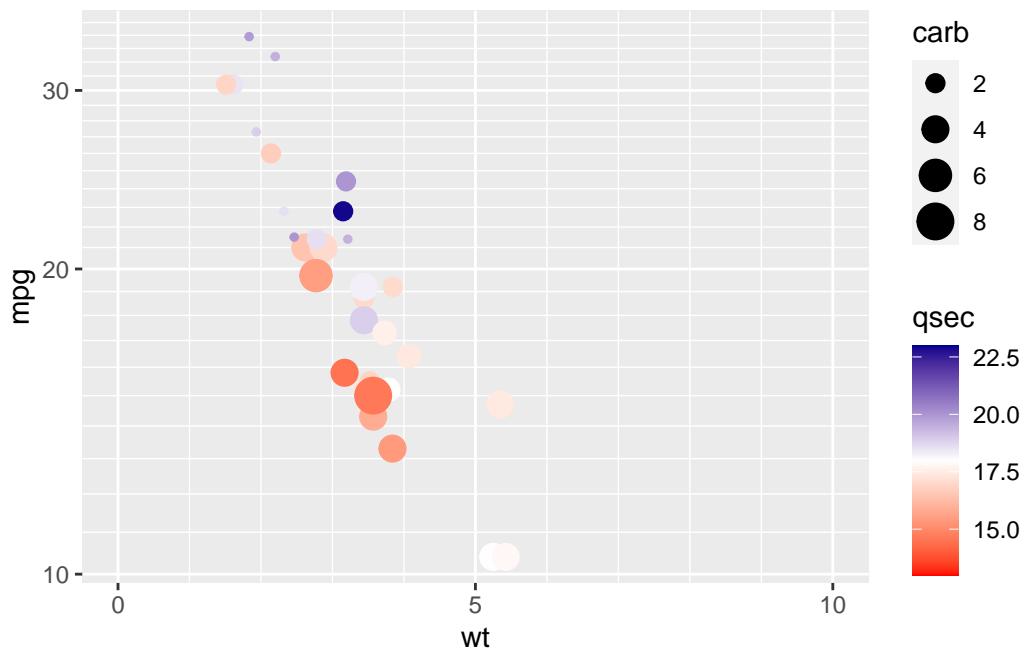
```
ggplot(data=mtcars,aes(wt, mpg,color=qsec, size=carb))+  
  geom_point()
```



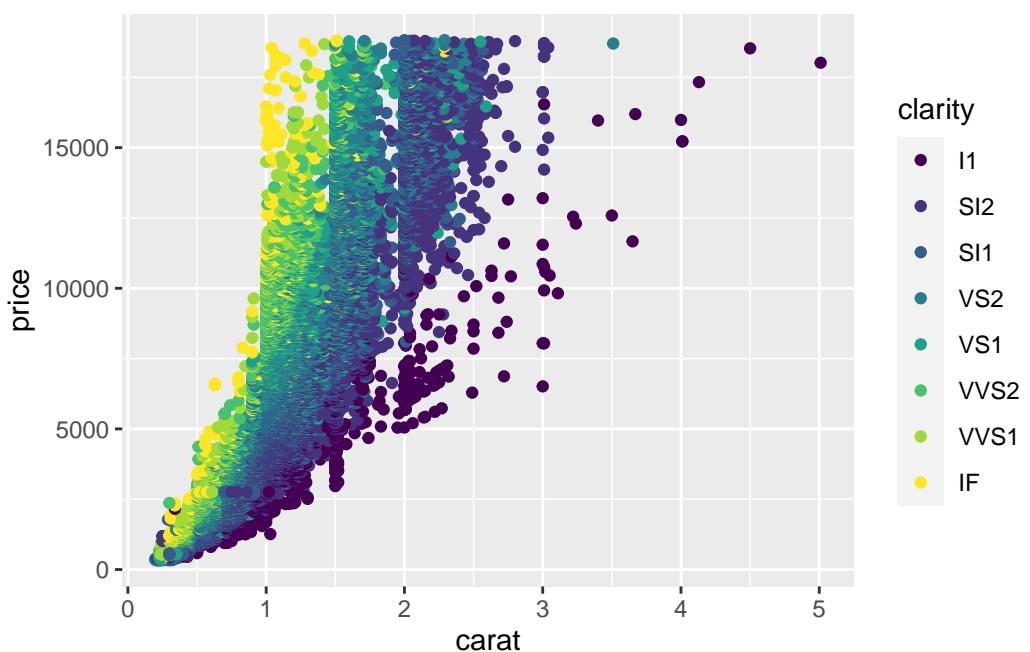
```
ggplot(data=mtcars,aes(wt, mpg,color=qsec, size=carb))+  
  scale_color_gradient(low='darkred',high='blue')+  
  geom_point()
```



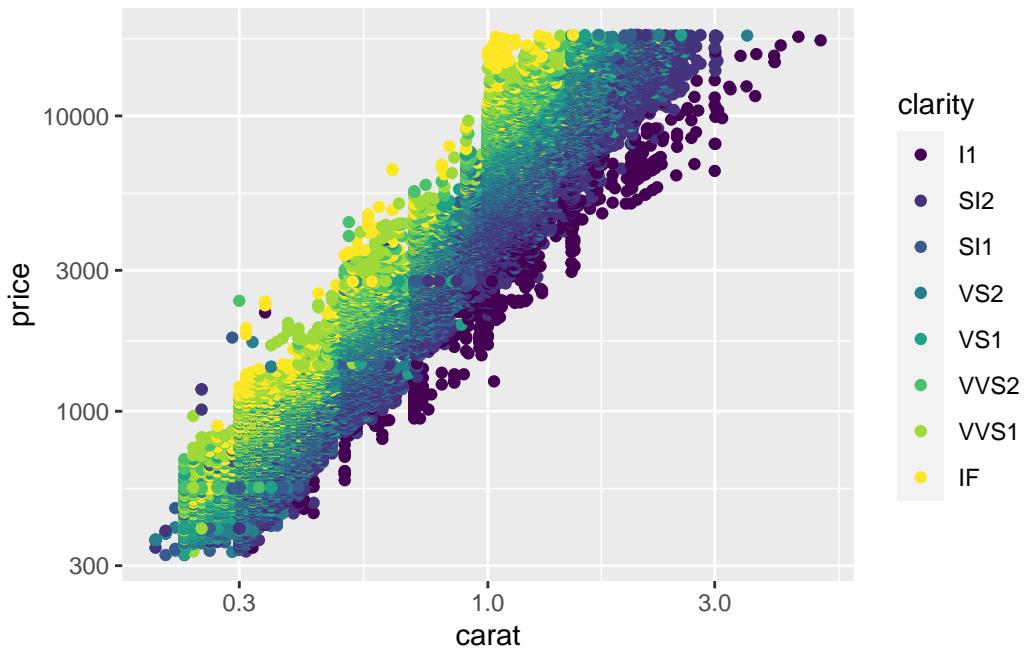
```
ggplot(data=mtcars,aes(wt, mpg,color=qsec, size=carb))+  
  scale_color_gradient2(low='red',high='darkblue',  
    mid='white',  
    limits=c(13,23),midpoint=18)+  
  geom_point() +  
  scale_x_continuous(breaks = seq(0,100,5),  
    minor_breaks=seq(0,100,1),  
    limits = c(0,10))+  
  scale_y_log10(minor_breaks=seq(0,100,1))
```



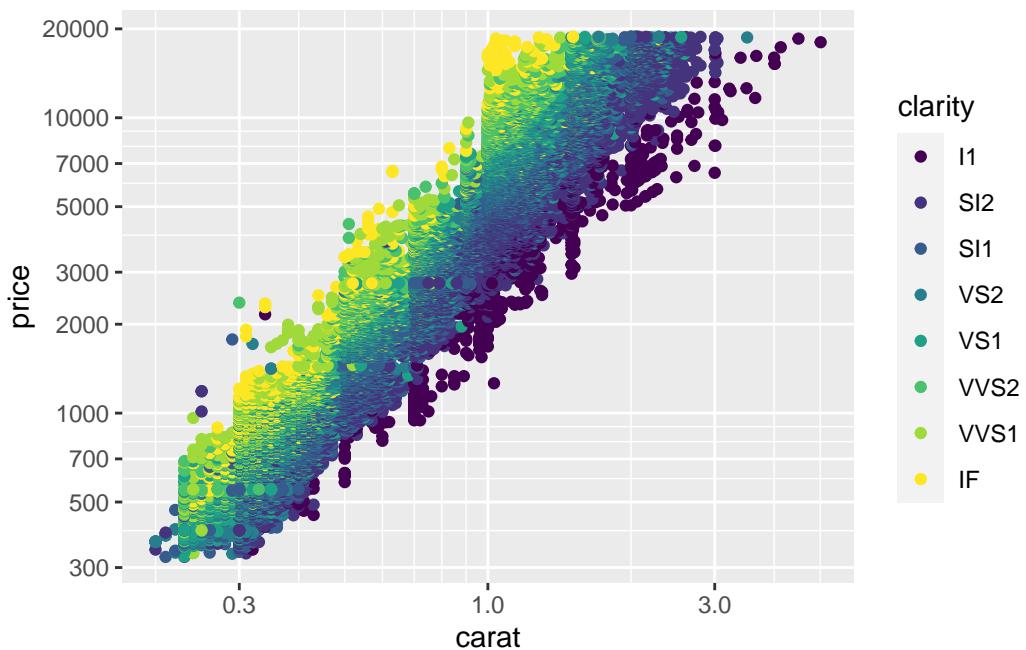
```
ggplot(diamonds,aes(carat,price,color=clarity))+  
  geom_point()
```



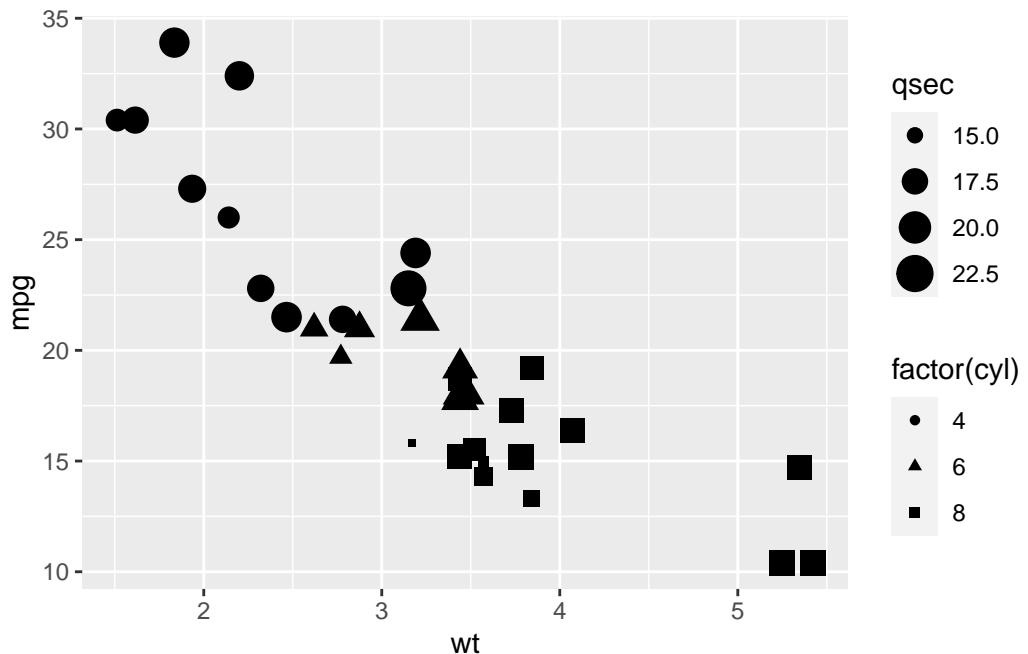
```
ggplot(diamonds,aes(carat,price,color=clarity))+  
  geom_point() +  
  scale_x_log10() +  
  scale_y_log10()
```



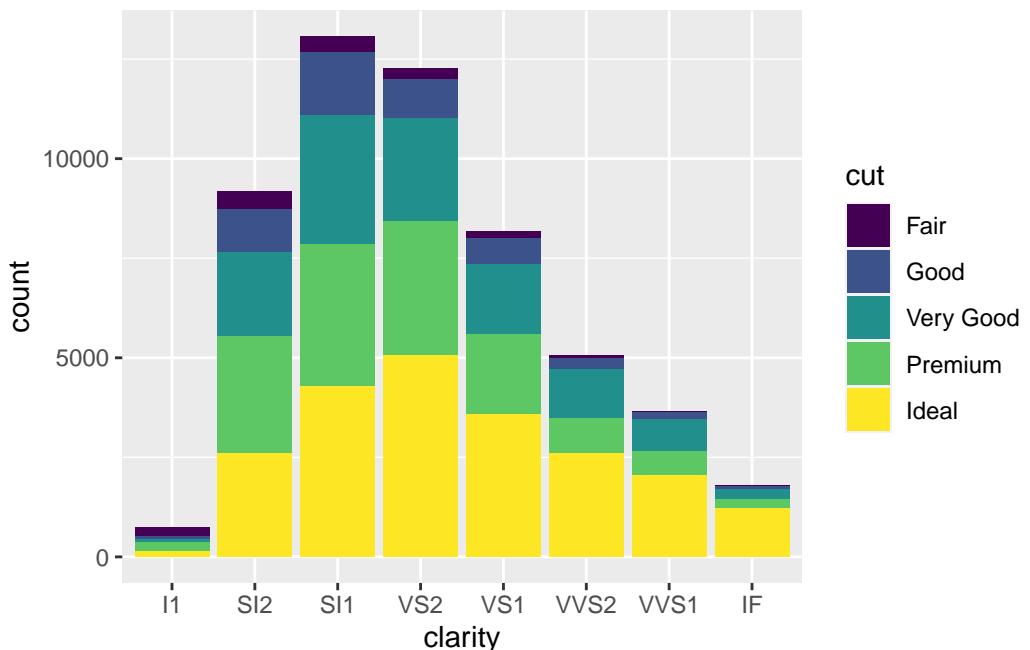
```
ggplot(diamonds,aes(carat,price,color=clarity))+  
  geom_point() +  
  scale_x_log10(minor_breaks=logrange_123456789) +  
  scale_y_log10(  
    breaks=logrange_12357,  
    minor_breaks=logrange_123456789)
```



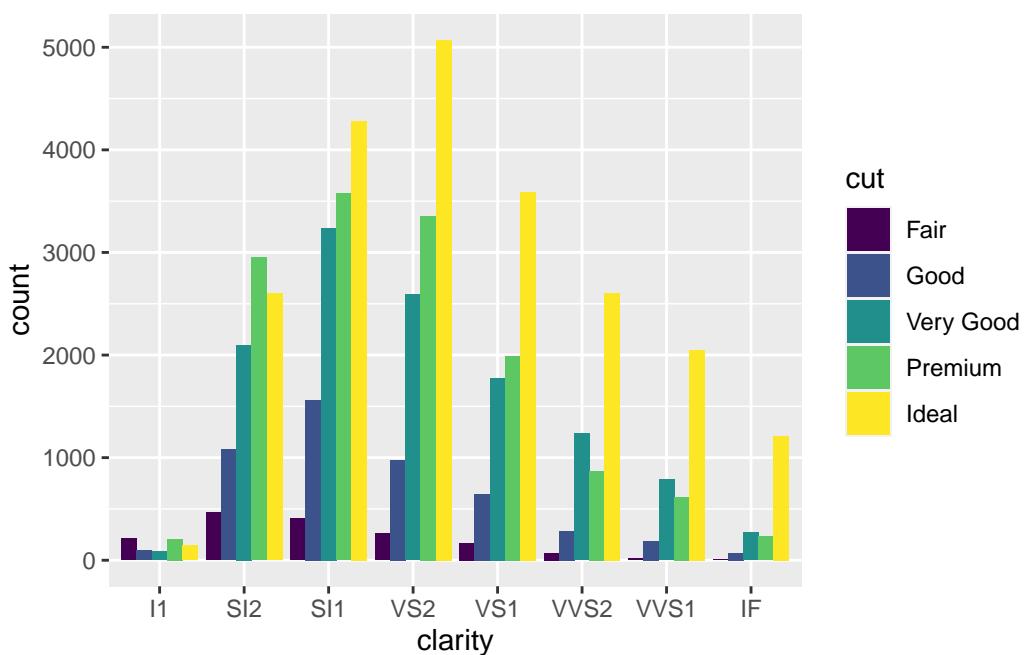
```
# use different aesthetic mappings
ggplot(data=mtcars,
        aes(wt, mpg, size=qsec, shape=factor(cyl)))+
  geom_point()
```



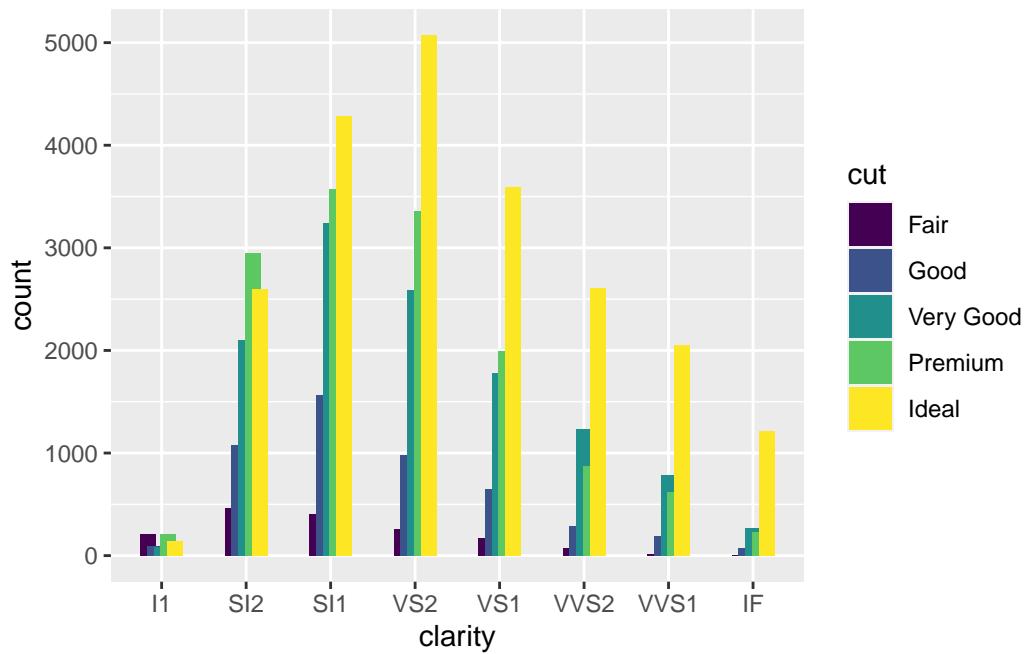
```
# ggplot(data=mtcars,aes(carb, mpg,color=qsec, size=wt))+  
#   geom_point(shape='\u2642')  
  
#location, location,position  
  
p<-ggplot(data=diamonds,aes(clarity,fill=cut))  
p+geom_bar(position="stack")
```



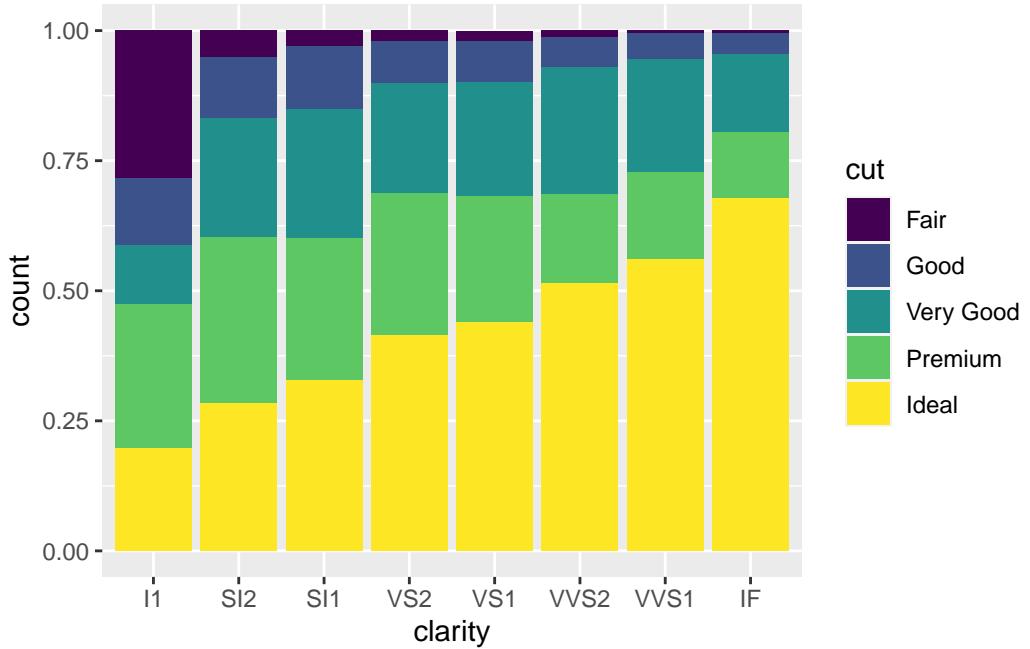
```
p+geom_bar(position="dodge")
```



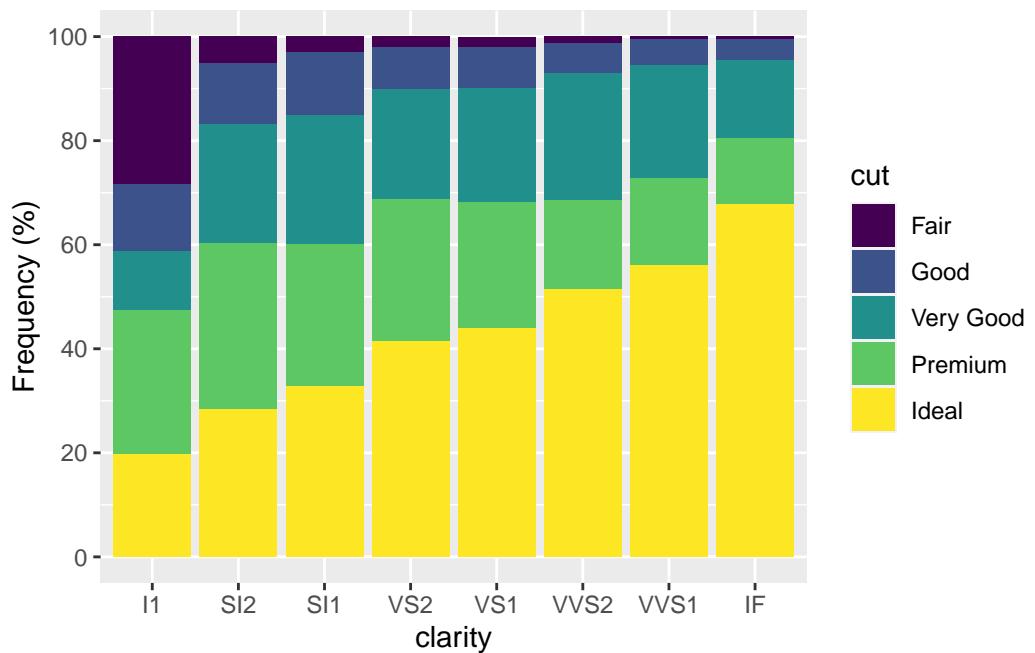
```
p+geom_bar(position=position_dodge(width = .4))
```



```
p+geom_bar(position="fill")
```

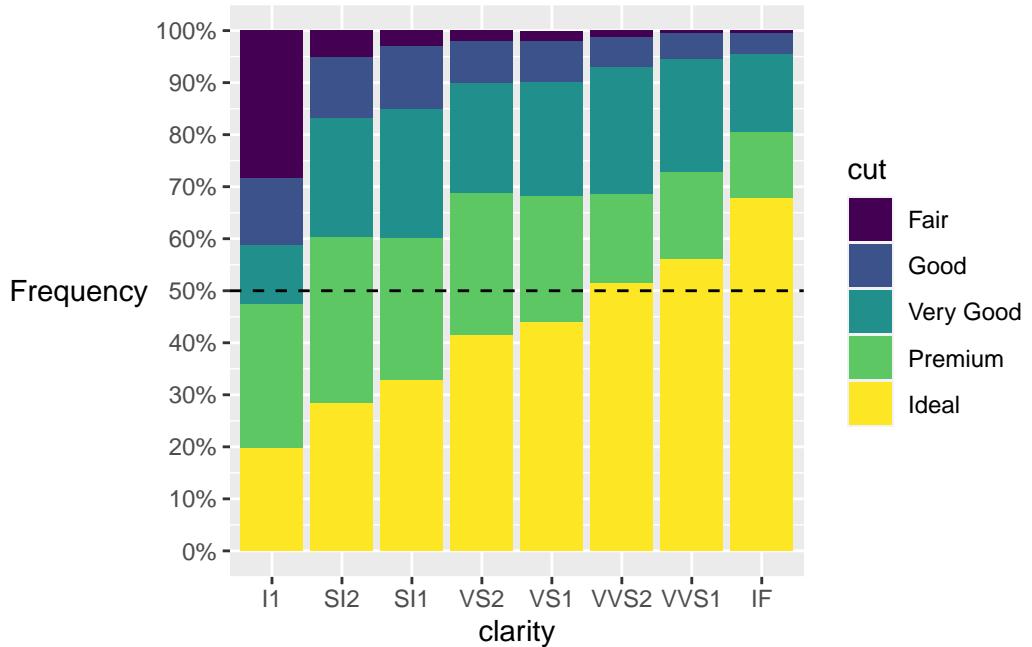


```
p+geom_bar(position="fill")+
  scale_y_continuous('Frequency (%)',
                     breaks=seq(0,1,.2),
                     labels=seq(0,100,20))
```

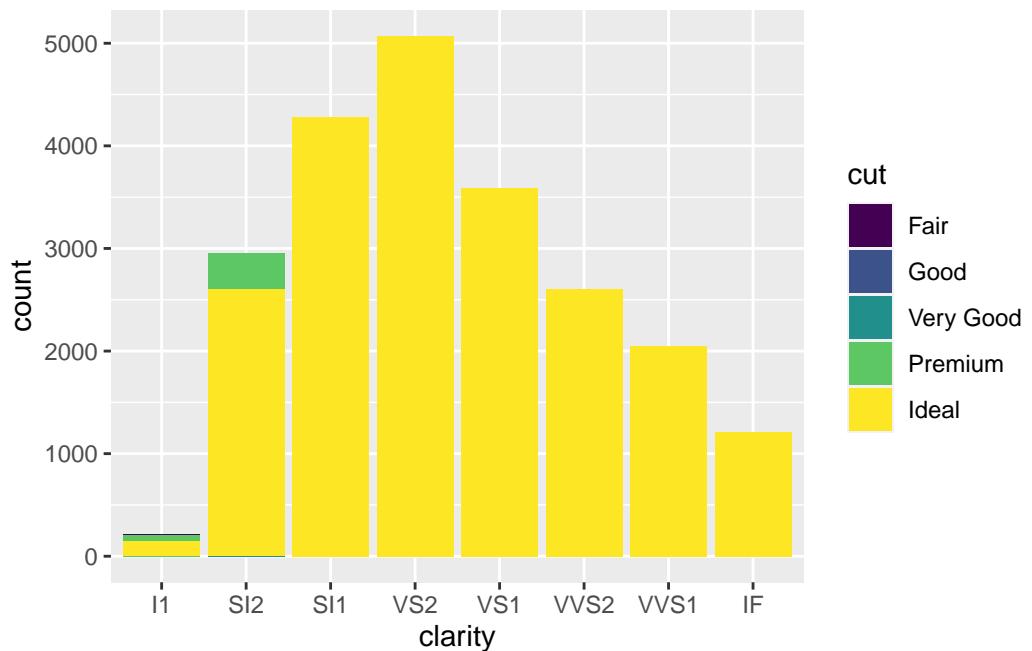


```
p+geom_bar(position="fill")+
  scale_y_continuous('Frequency',
                     breaks=seq(0,1,.1),
                     labels=scales::percent)+
  theme(axis.title.y = element_text(angle = 0,
                                    vjust = .5))+
```

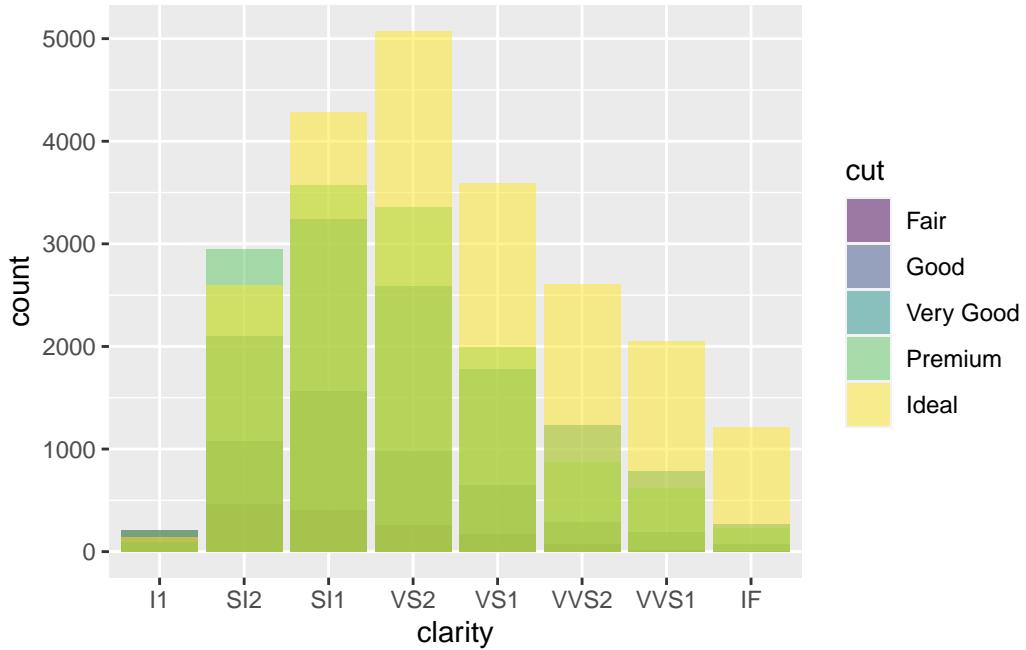
geom_hline(yintercept = .5, linetype=2)



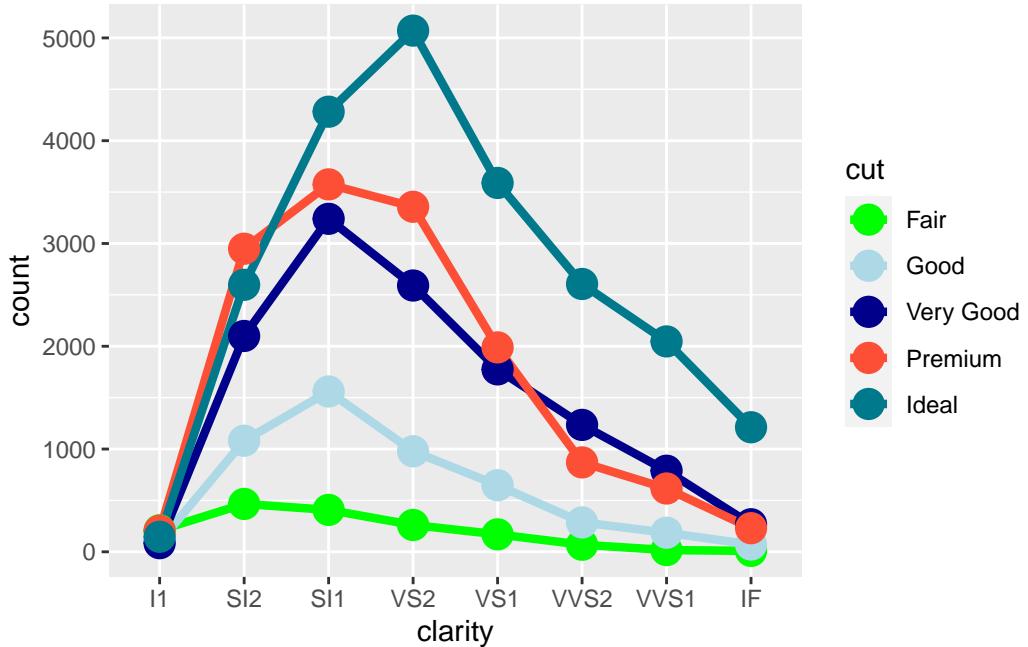
```
p+geom_bar(position="identity")
```



```
p+geom_bar(position="identity",alpha=.5)
```



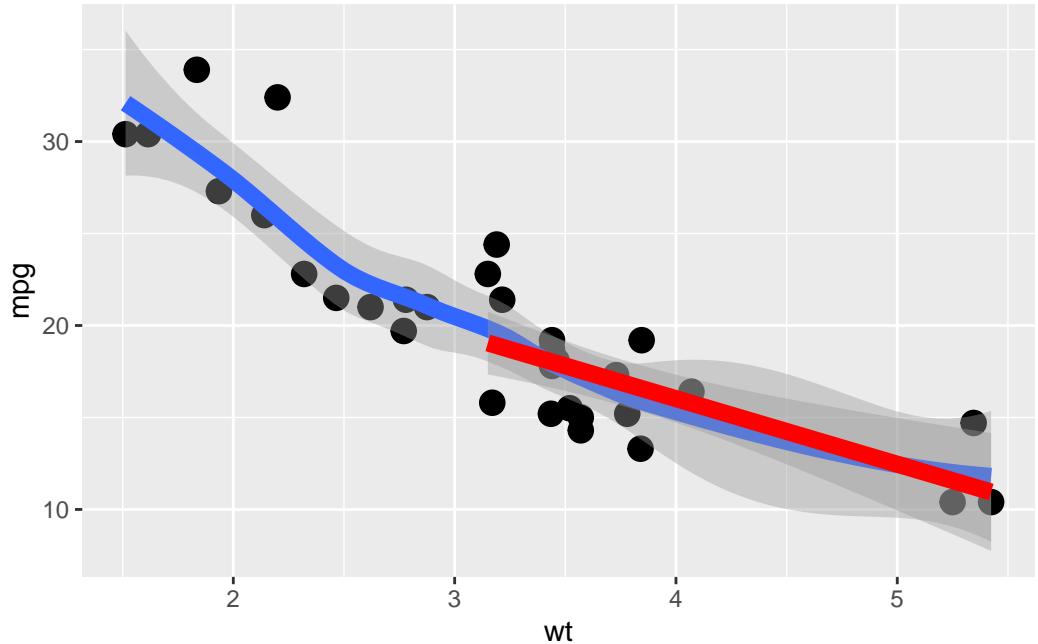
```
ggplot(data=diamonds,aes(clarity,color=cut, group=cut))+  
  geom_freqpoly(stat='count',position="identity",lwd=1.5)+  
  geom_point(stat='count',size=5)+  
  scale_color_manual(values = c('green','lightblue',  
    'darkblue',  
    'rgb(253,79,54,maxColorValue = 255)',  
    '#00798d'))
```



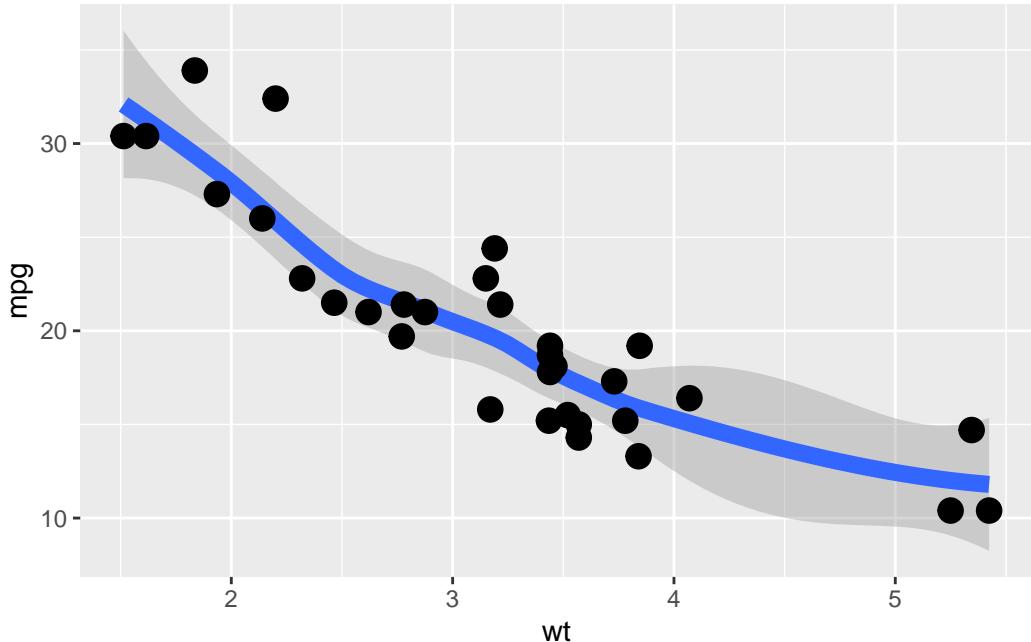
```
#layer/order / computed geoms
ggplot(data=mtcars,aes(wt, mpg))+
  geom_point(size=4)+
  geom_smooth(size=3)+
  geom_smooth(data=mtcars %>% filter(wt>3),
              method='lm',size=3, color='red')
```

Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.

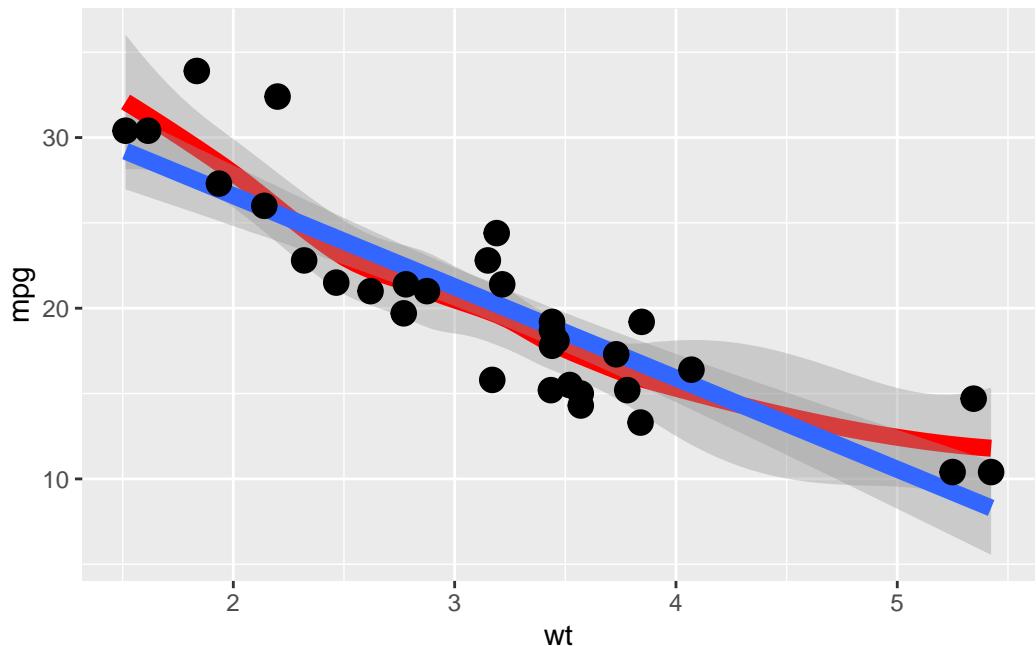
```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'  
`geom_smooth()` using formula = 'y ~ x'
```



```
ggplot(data=mtcars,aes(wt, mpg))+  
  geom_smooth(size=3)+  
  geom_point(size=4)  
  
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

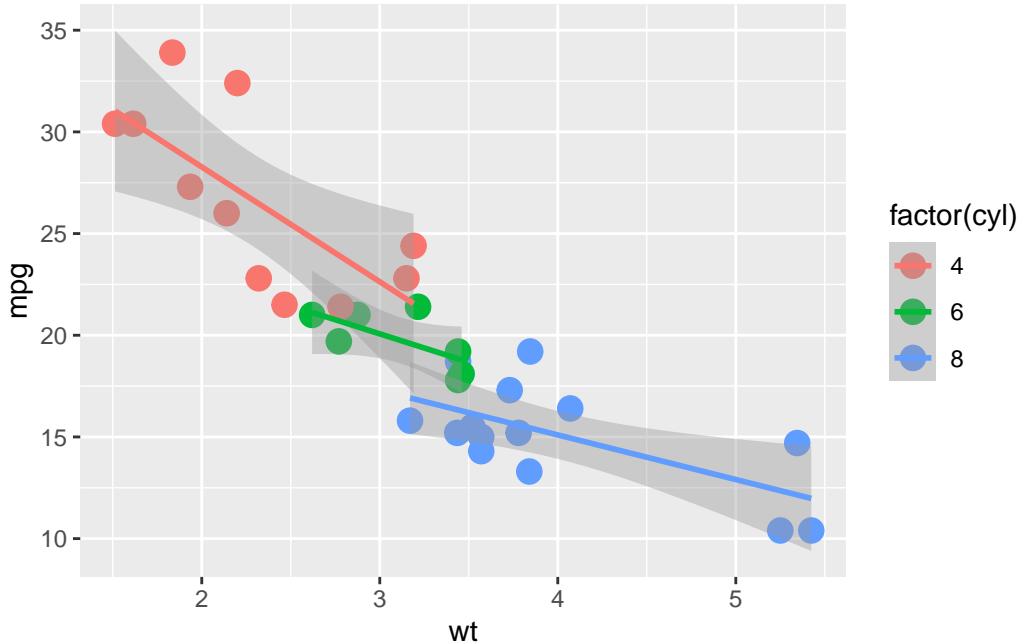


```
ggplot(data=mtcars,aes(wt, mpg))+  
  geom_smooth(size=3,color='red')+  
  geom_smooth(method='lm',size=3)+  
  geom_point(size=4)  
  
'geom_smooth()' using method = 'loess' and formula = 'y ~ x'  
'geom_smooth()' using formula = 'y ~ x'
```



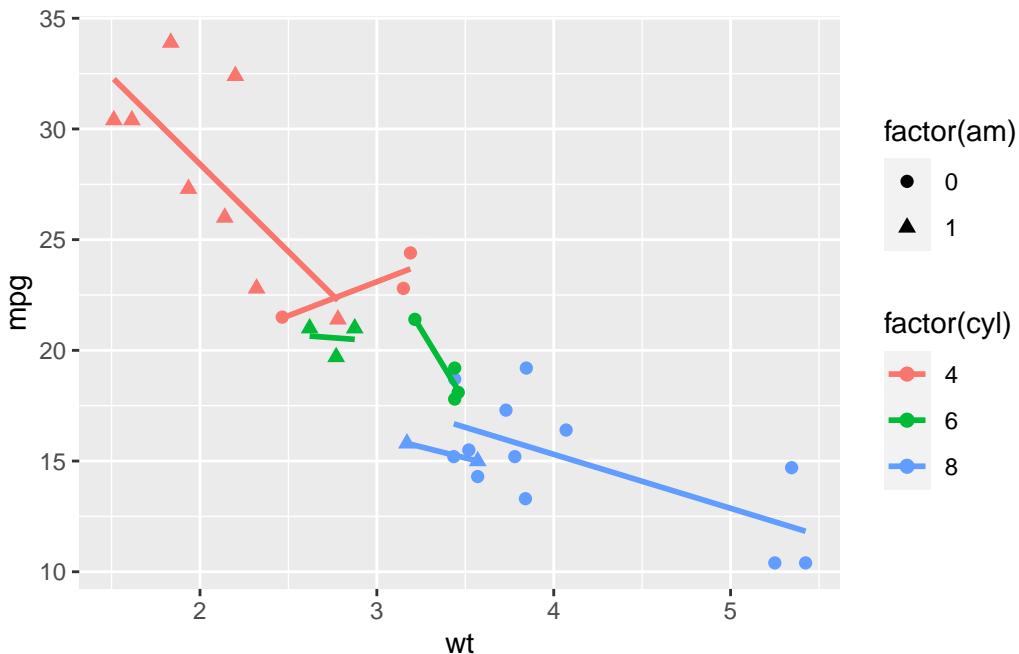
```
ggplot(data=mtcars,aes(wt, mpg,
                        color=factor(cyl)))+
  geom_point(size=4)+
  geom_smooth(method='lm',size=1)

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



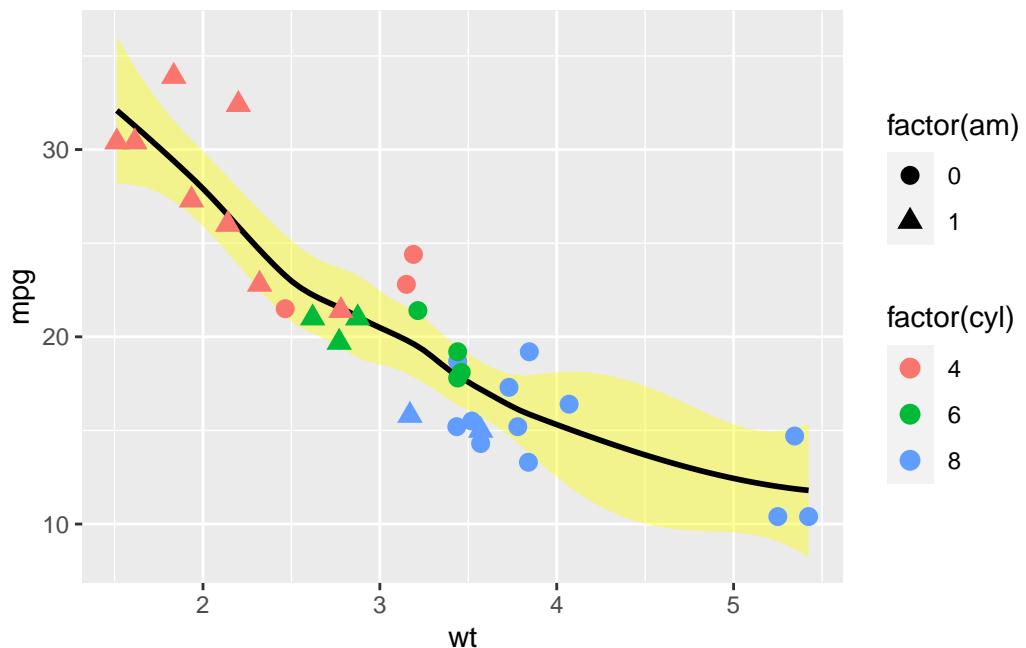
```
ggplot(data=mtcars, aes(wt, mpg,
                        color=factor(cyl),
                        shape=factor(am)))+
  geom_point(size=2)+
  geom_smooth(method='lm', size=1, se=F)
```

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

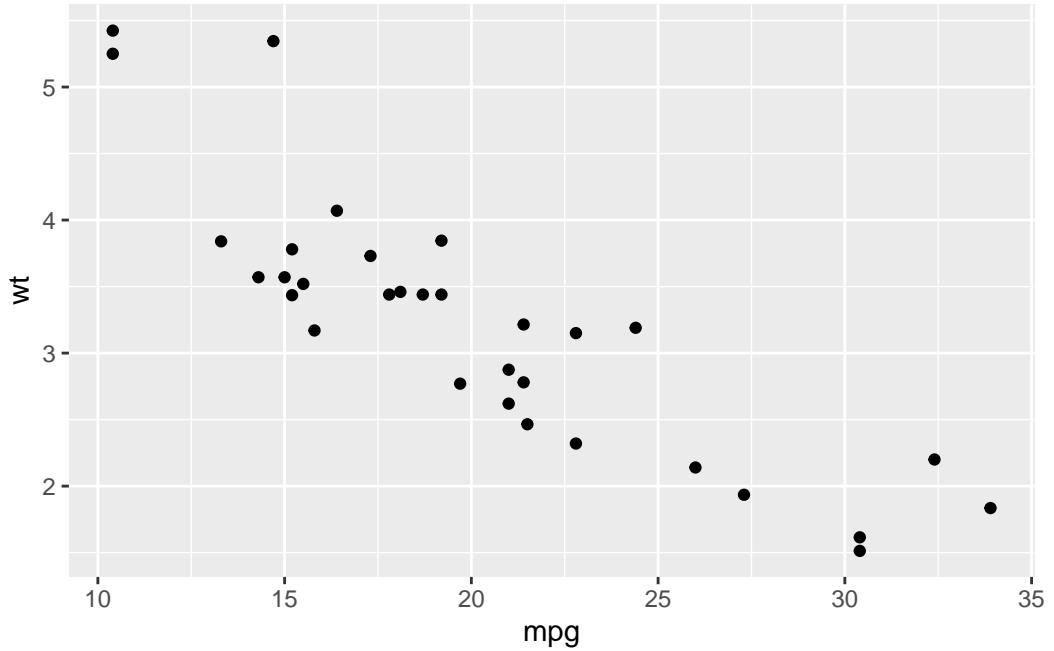


```
#? lm for all?
ggplot(data=mtcars,aes(wt, mpg))+
  geom_smooth(size=1,color='black',fill='yellow')+
  geom_point(size=3,aes(color=factor(cyl),shape=factor(am))) #aes for geom only

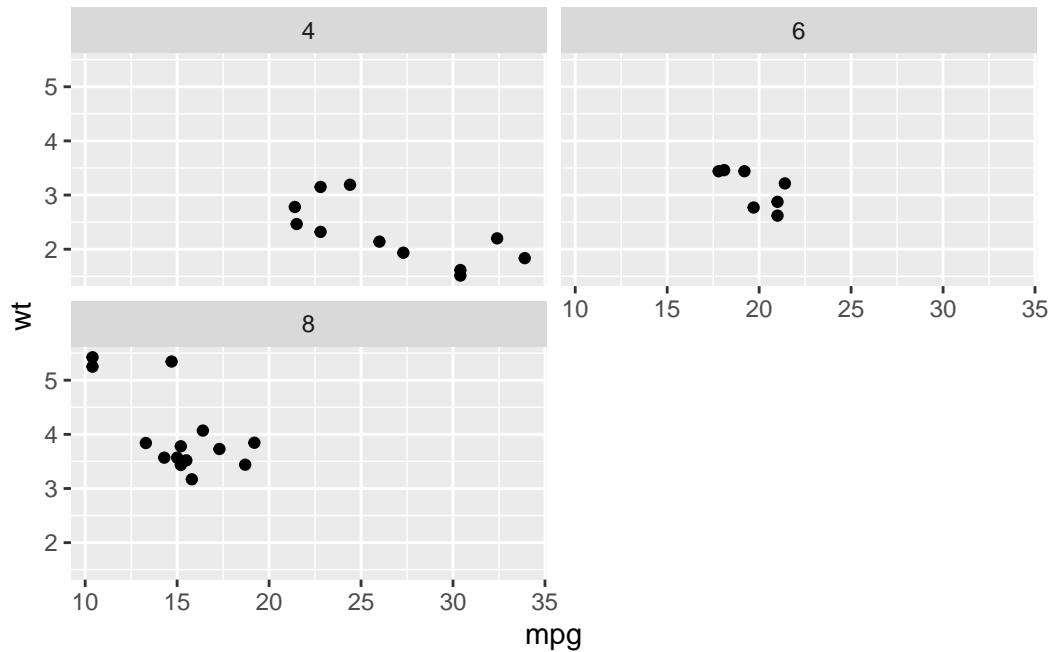
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```



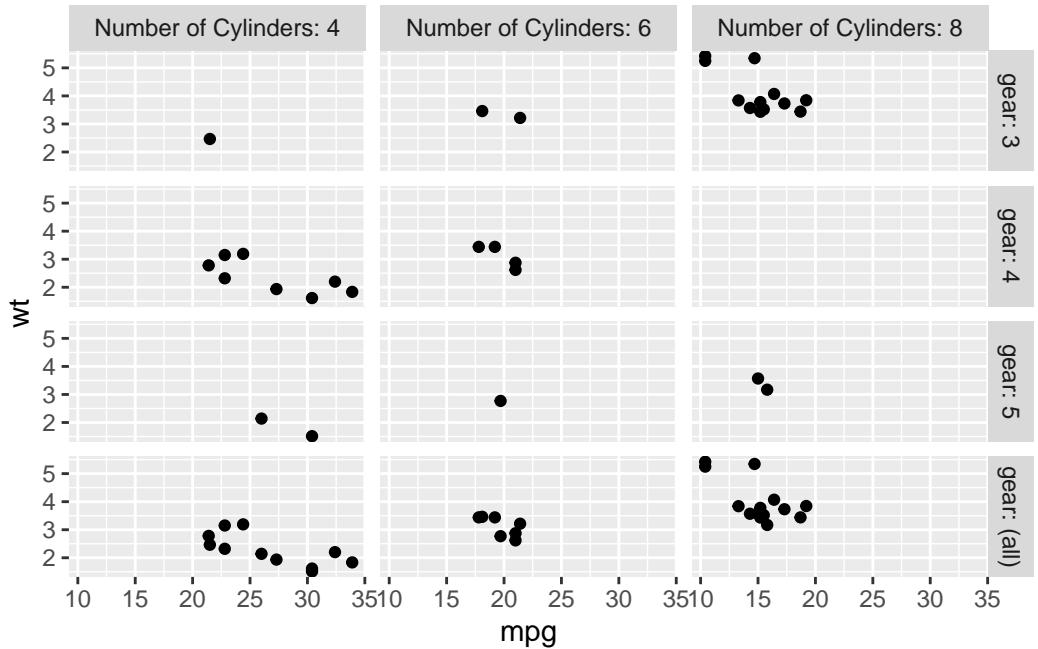
```
# facet_wrap / facet_grid  
(p.tmp <- ggplot(mtcars, aes(mpg, wt)) +  
  geom_point())
```



```
p.tmp + facet_wrap(facets = vars(cyl), ncol=2)
```



```
#p.tmp + facet_wrap(~cyl, ncol=2)
p.tmp + facet_grid(rows = vars(gear),
                    cols = vars(`Number of Cylinders` =
                                cyl),
                    labeller=label_both, margins='gear')
```

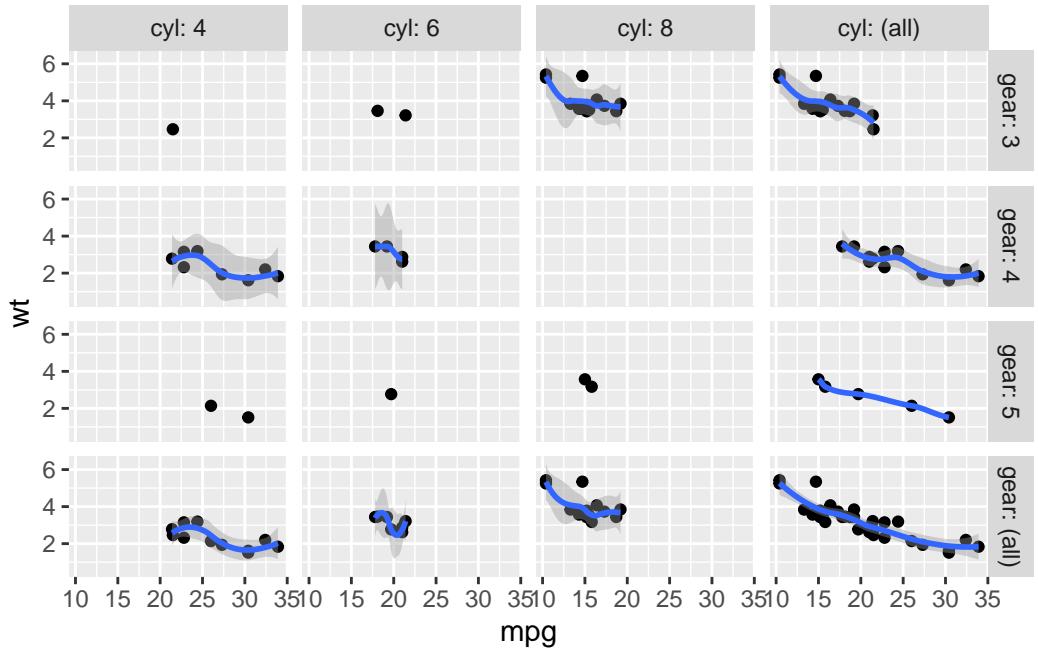


```

options(warn=-1)
p.tmp + geom_smooth()+
  facet_grid(gear~cyl, labeller=label_both,margins=T)

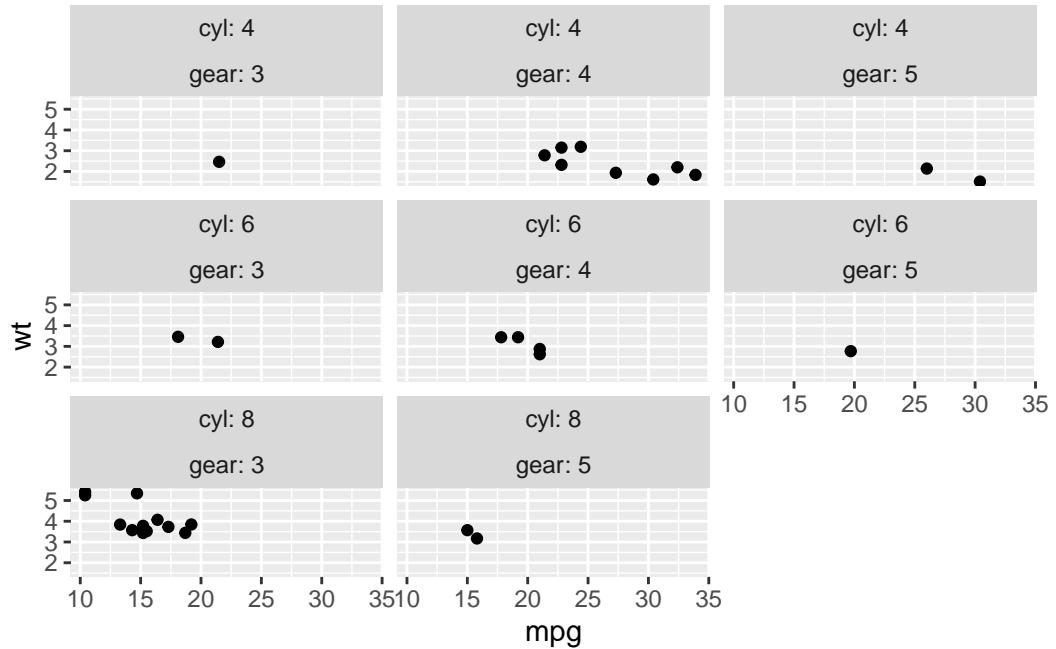
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'

```

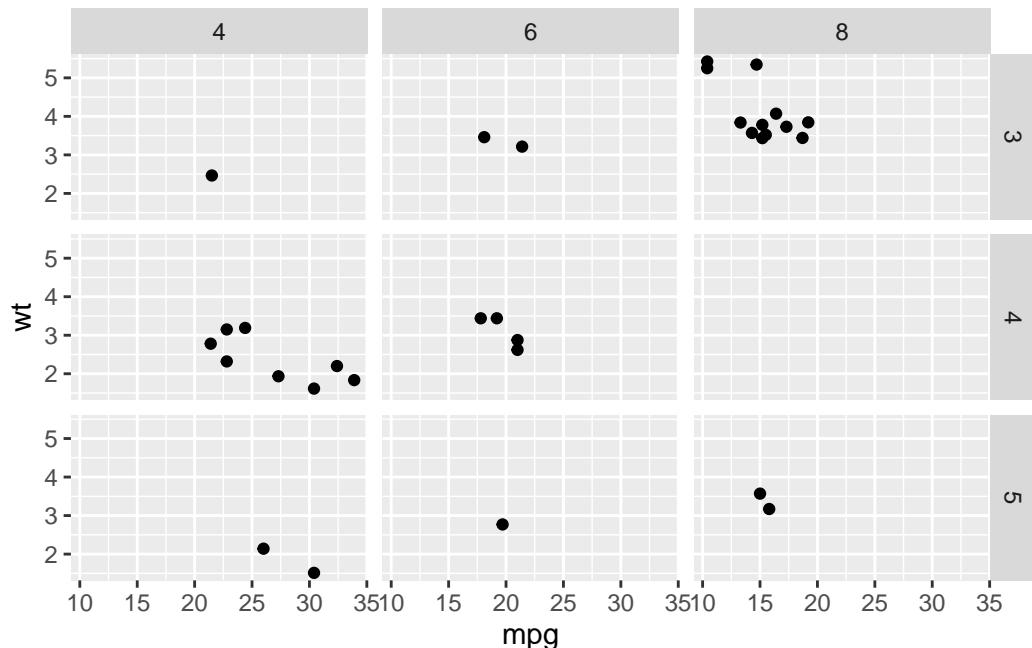


```
options(warn=0)
```

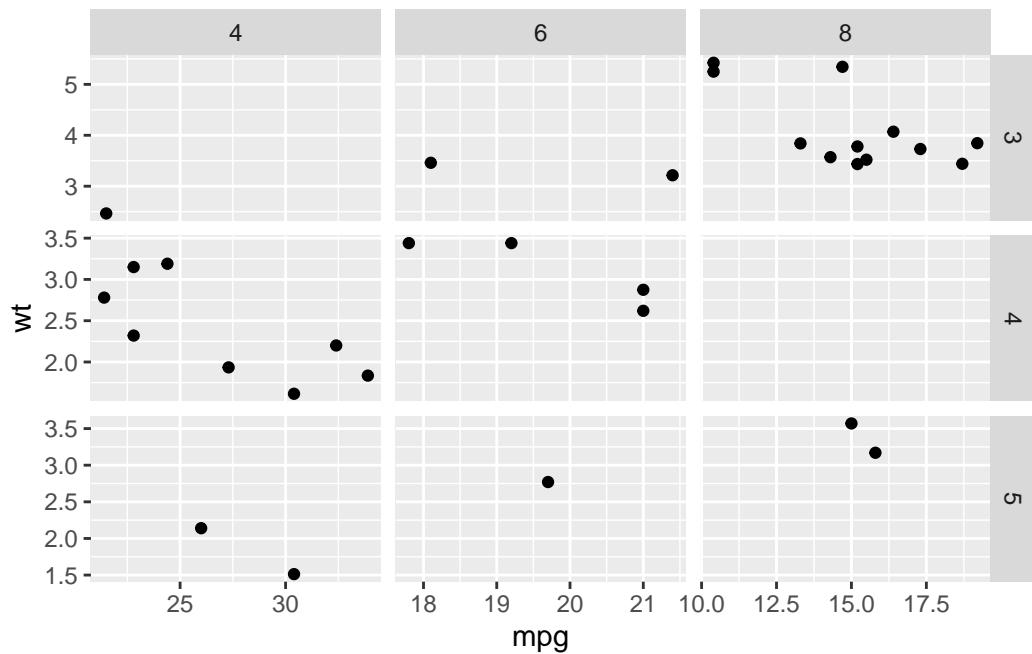
```
p.tmp + facet_wrap(~cyl+gear, labeller=label_both) # empty combination is dropped
```



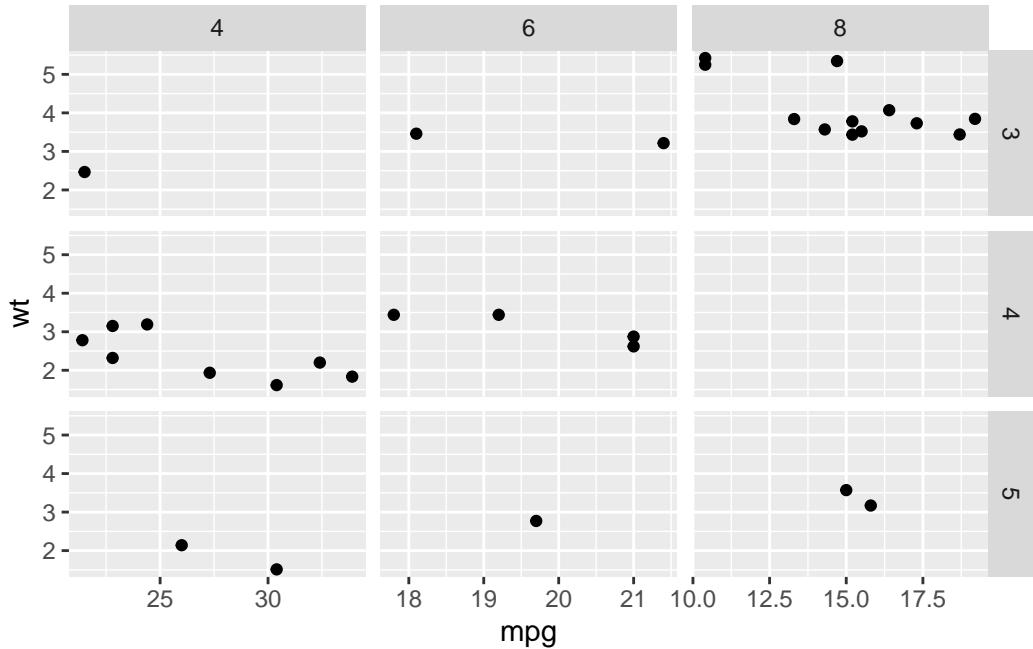
```
# controlling scales in facets (default: scales="fixed")
p.tmp + facet_grid(gear~cyl, scales="fixed")
```



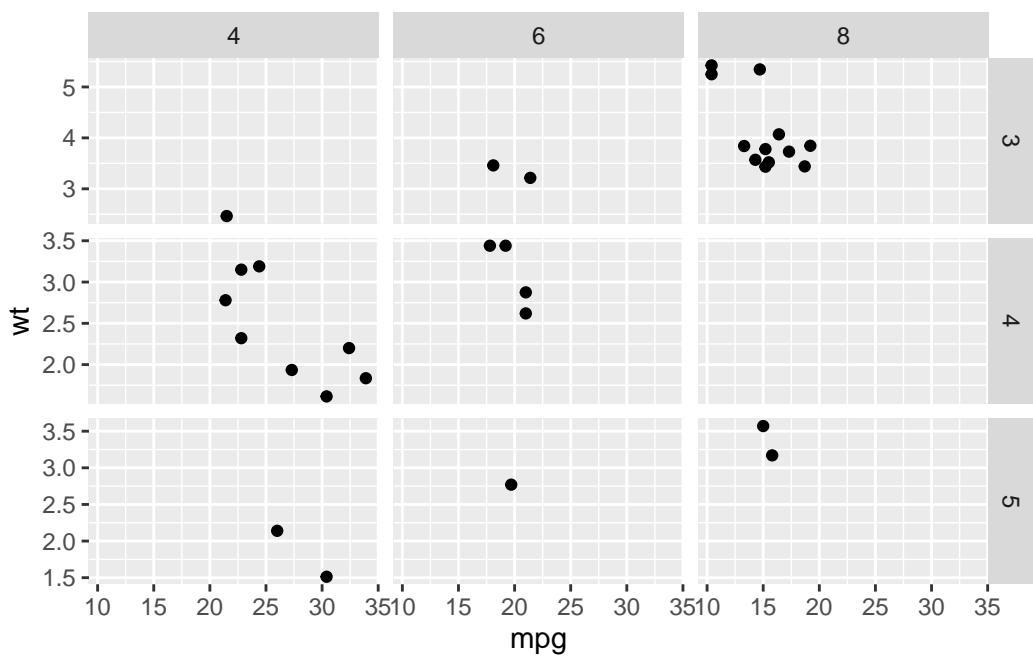
```
p.tmp + facet_grid(gear~cyl, scales="free")
```



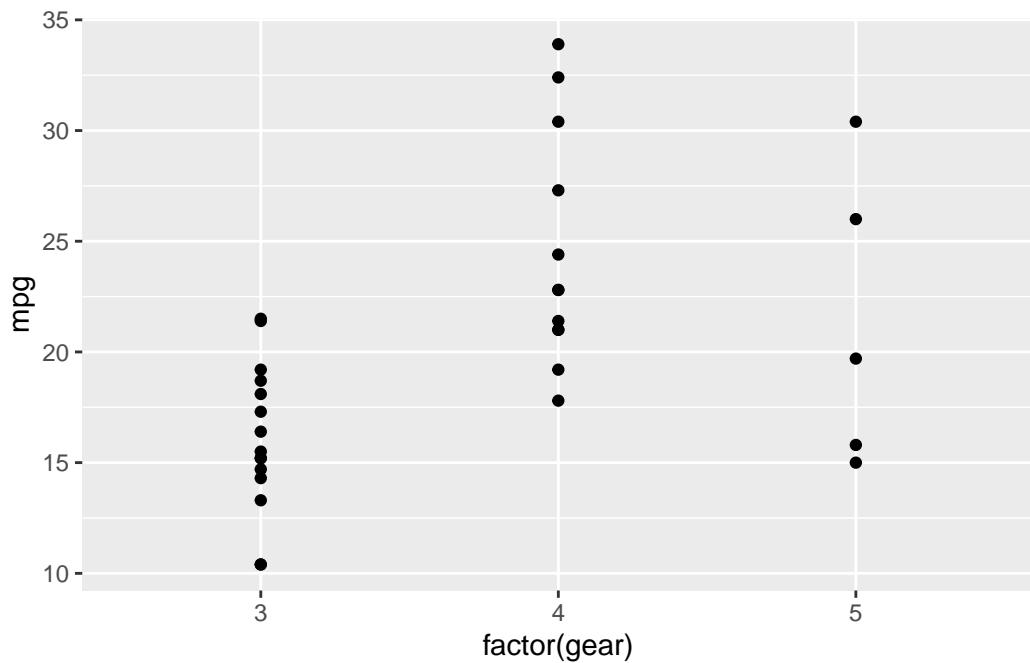
```
p.tmp + facet_grid(gear~cyl, scales="free_x")
```



```
p.tmp + facet_grid(gear~cyl, scales="free_y")
```

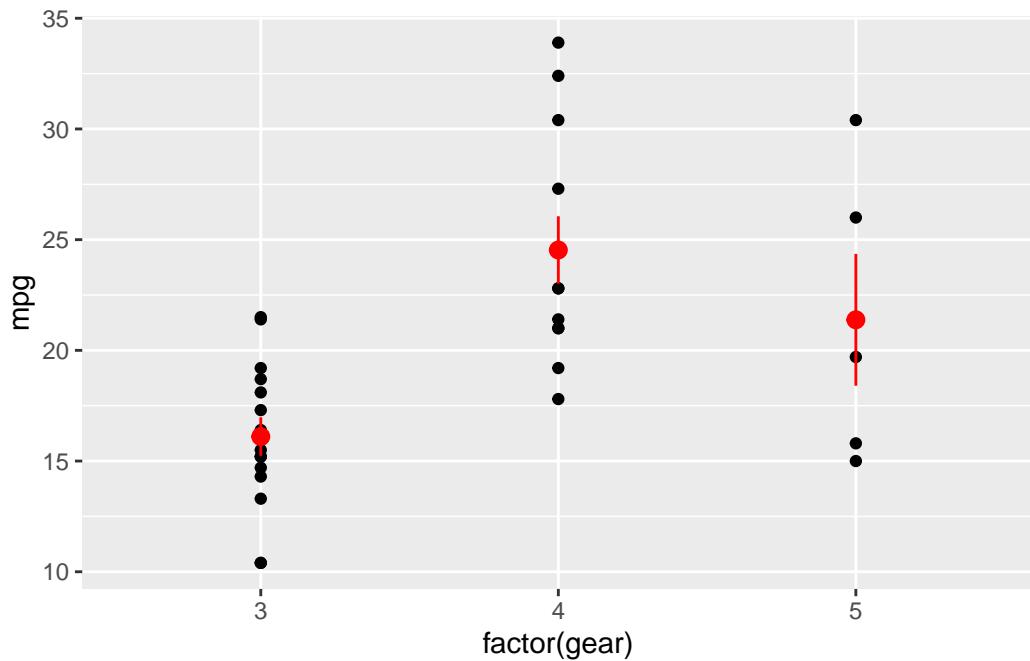


```
# summaries ####
(plottemp <- ggplot(mtcars,aes(factor(gear),mpg))+  
  geom_point())
```

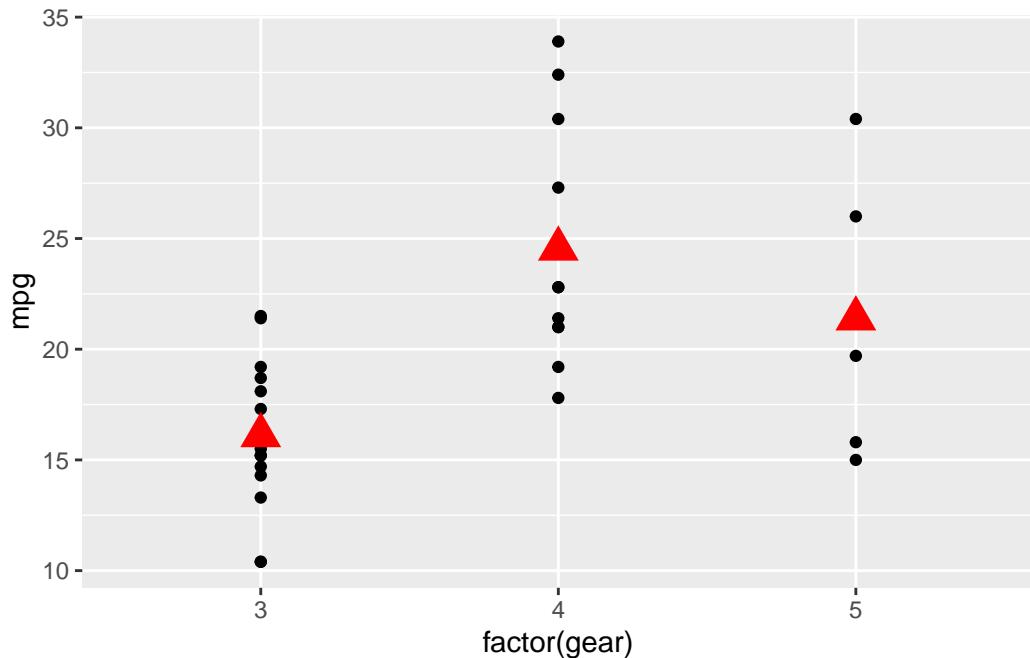


```
plottemp+stat_summary(color='red')
```

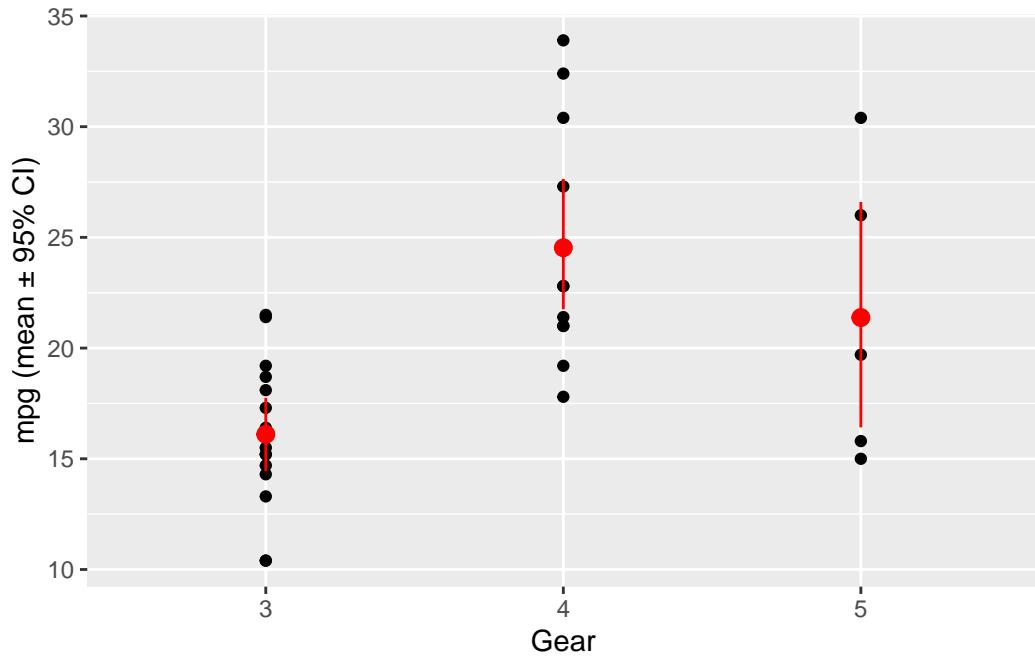
No summary function supplied, defaulting to `mean_se()`



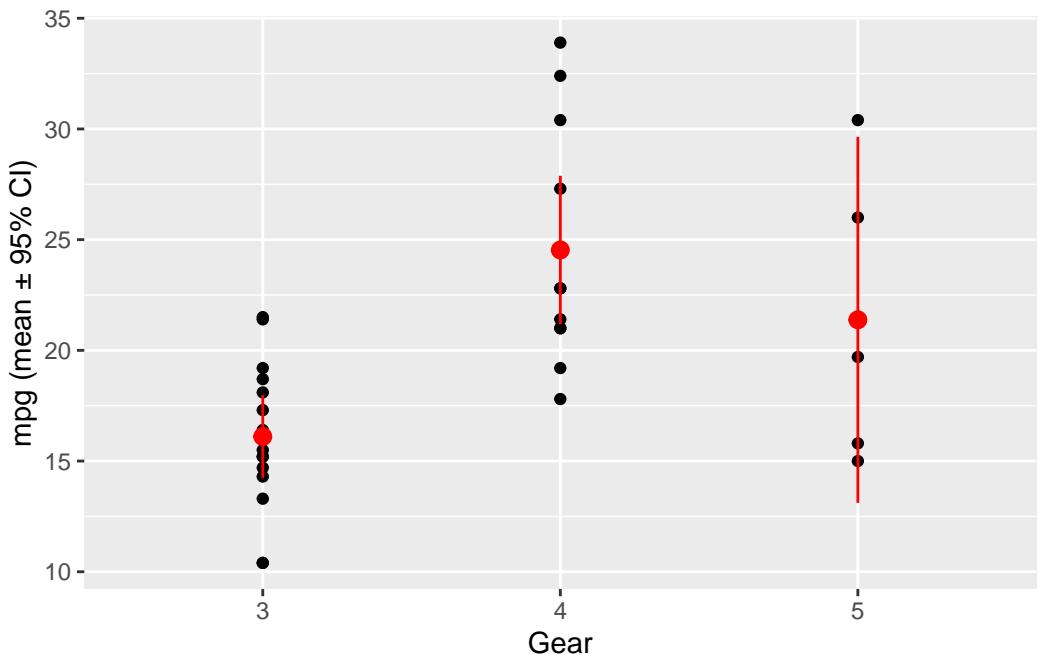
```
plottemp+stat_summary(geom = 'point', shape=17, size=5,  
                      fun = 'mean', color='red')
```



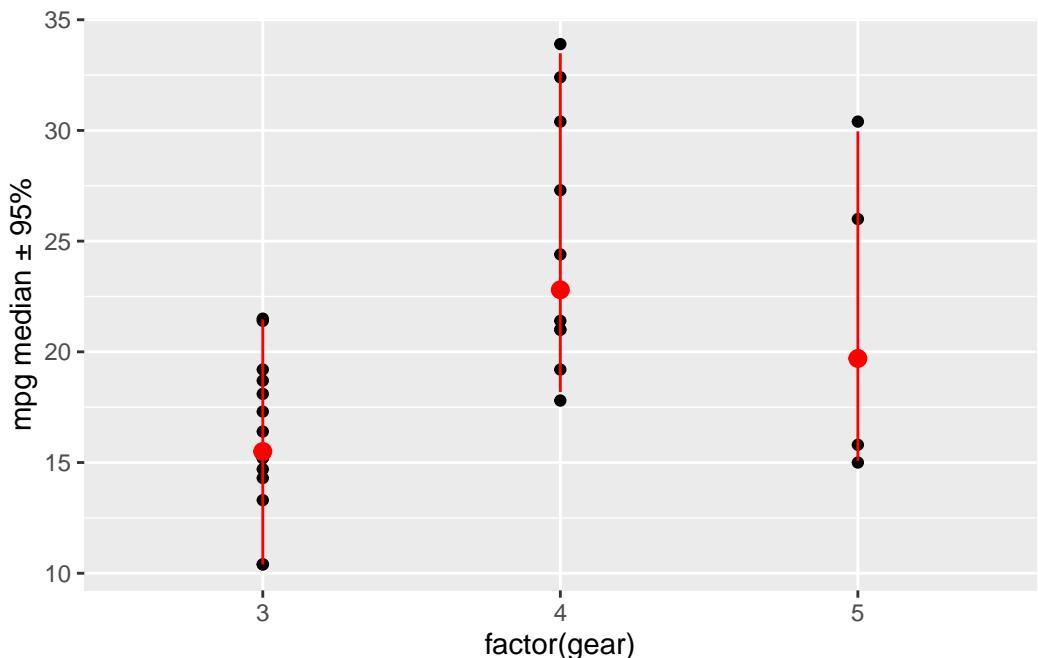
```
plottemp+stat_summary(fun.data='mean_cl_boot',color='red')+  
  ylab('mpg (mean \u00b1 95% CI)')+  
  xlab('Gear')
```



```
plottemp+stat_summary(fun.data='mean_cl_normal',color='red')+  
  ylab('mpg (mean \u00b1 95% CI)')+  
  xlab('Gear')
```



```
plottemp+stat_summary(fun.data='median_hilow',color='red')+  
ylab('mpg median \u00b1 95%)
```

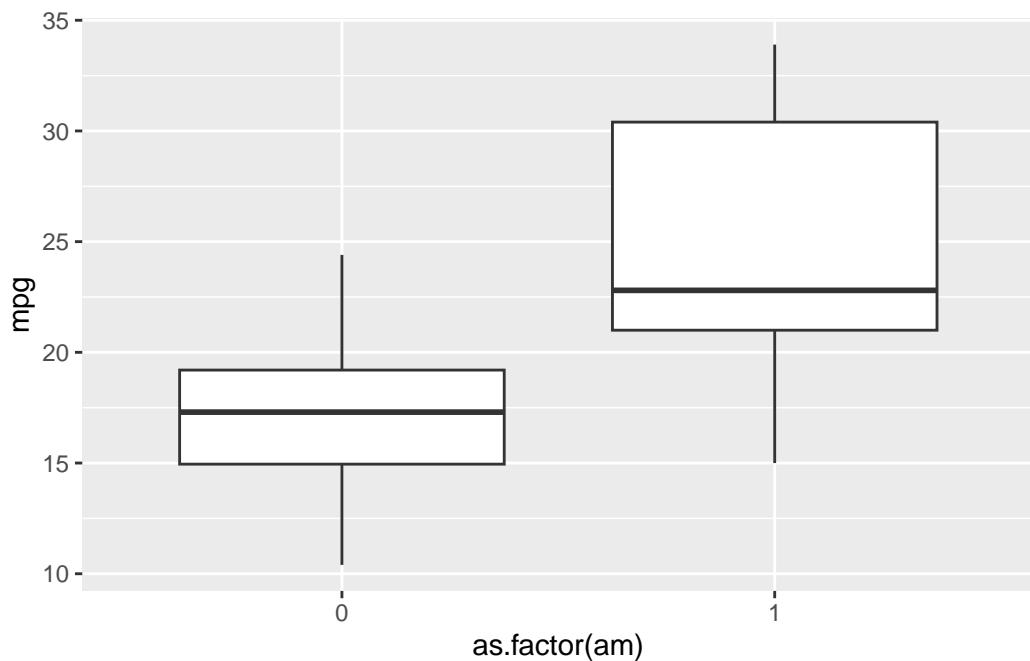


```
# geom_pointrange()

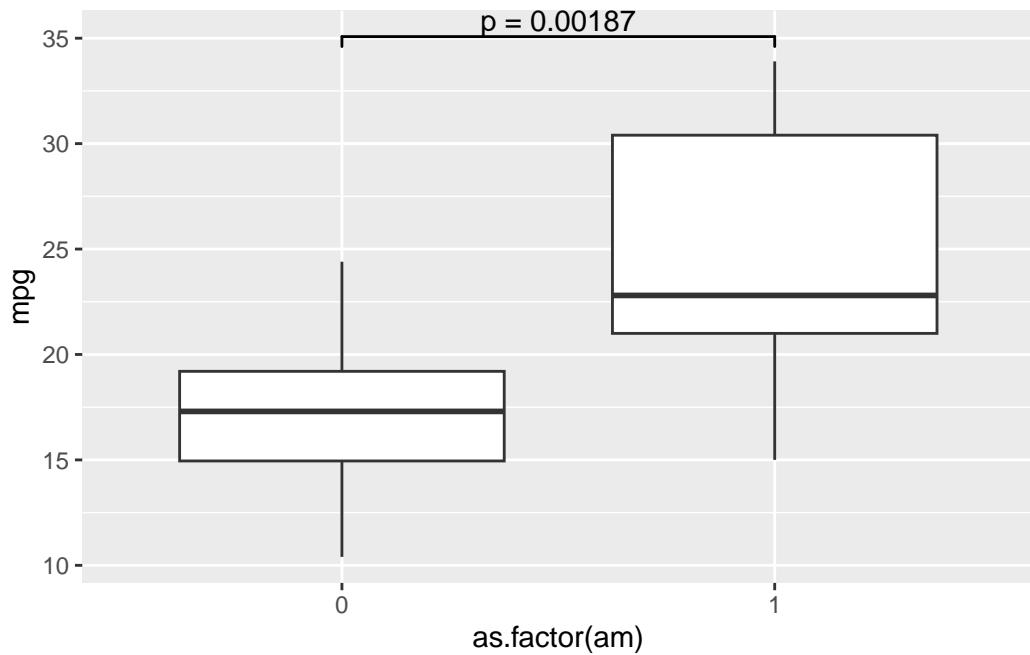
# ggsignif #####
p <- round(wilcox.test(mtcars$mpg~mtcars$am)$p.value,5)
```

Warning in wilcox.test.default(x = DATA[[1L]], y = DATA[[2L]], ...): kann bei Bindungen keinen exakten p-Wert Berechnen

```
(plottemp <- ggplot(mtcars,aes(as.factor(am),mpg))+  
  geom_boxplot())
```

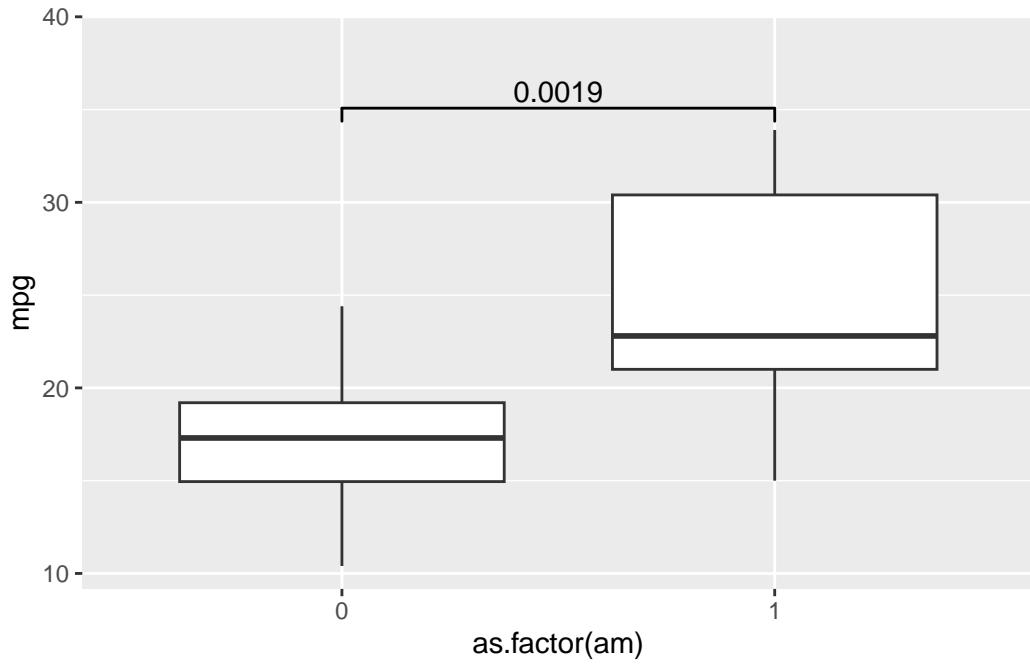


```
plottemp+geom_signif(  
  comparisons=list(c(1,2)),  
  # aes(y=0),  
  # textsize = rel(3), vjust = .0,  
  #y_position=max(mtcars$mpg+3),  
  annotations=paste0('p = ', p),  
  # annotations=p,  
  tip_length=.02)
```

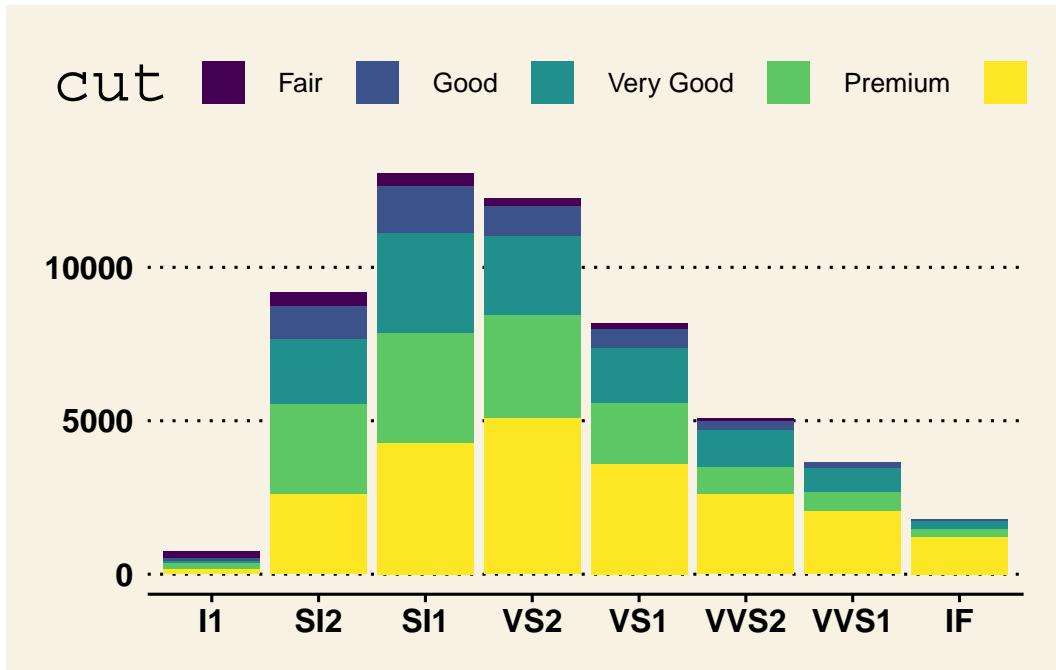


```
plottemp + geom_signif(  
  comparisons=list(1:2))+  
  scale_y_continuous(expand = expansion(mult=c(0.05,.2)))
```

Warning in wilcox.test.default(c(21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19.2, :
kann bei Bindungen keinen exakten p-Wert Berechnen



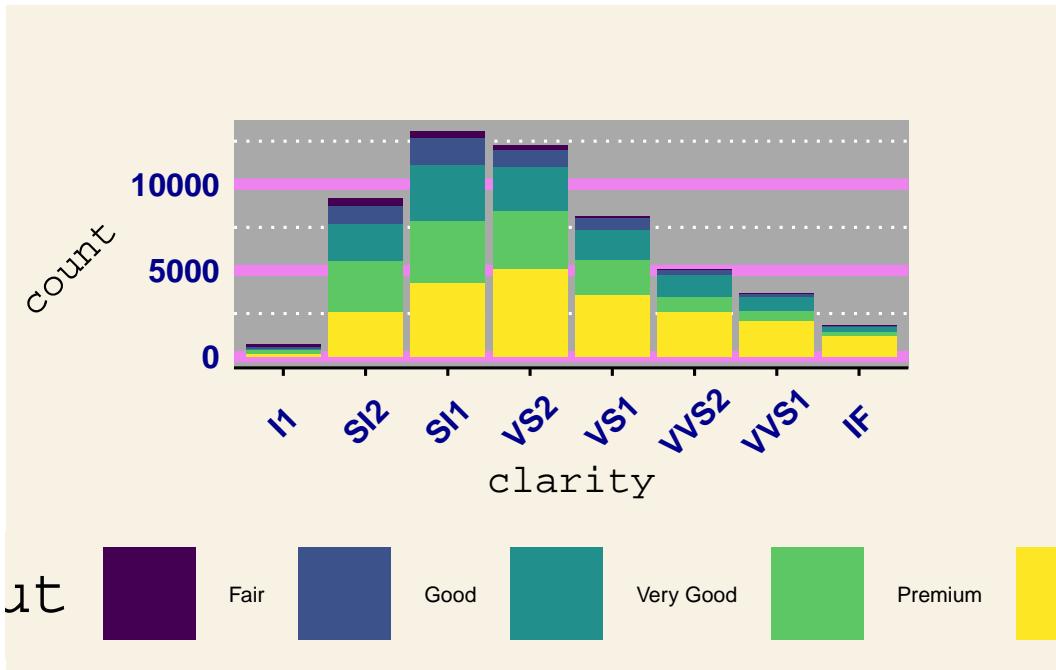
```
old <- theme_set(theme_wsj())
# theme_set(theme_wsj())
ggplot(data=diamonds,aes(x=clarity,fill=cut))+  
  geom_bar()
```



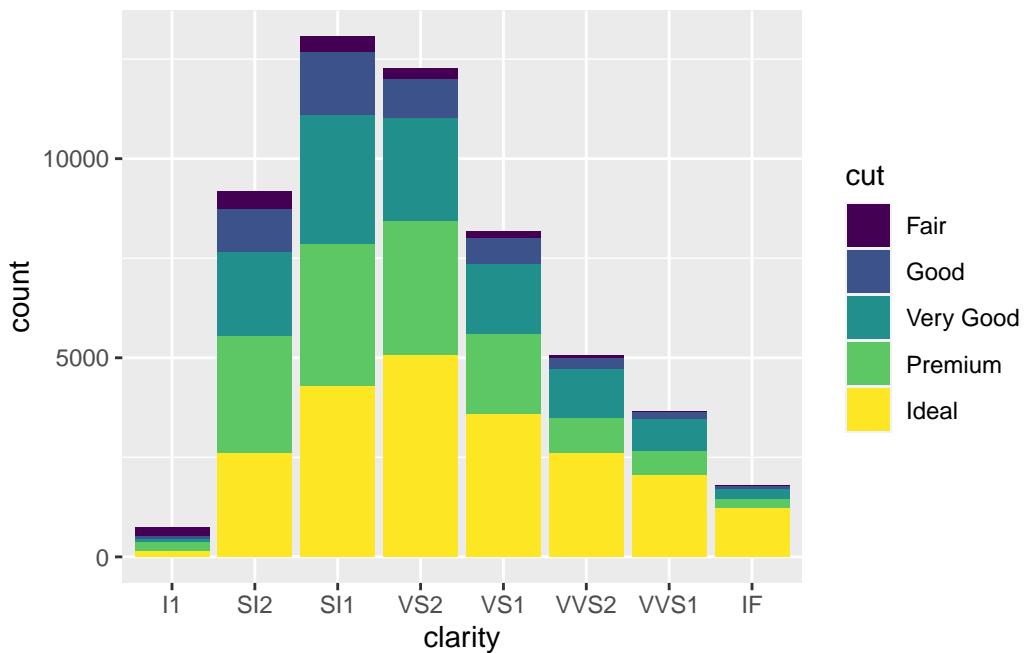
```
theme_update(legend.position="bottom",
            axis.text=element_text(colour = "darkblue",
                                   size=12),
            axis.text.x=element_text(vjust=0.5,angle=45),
            axis.title=element_text(size=15),
            plot.margin=unit(c(3,4,.5,.3),'lines'),      #N,E,S,W
            axis.title.y=element_text(vjust=0.4,angle=45),
            legend.key.size=unit(2.5,'lines'),
            panel.background=element_rect(fill='darkgrey'),
            panel.grid.minor = element_line(colour='white'),
            panel.grid.major = element_line(
              linetype=1,
              color='violet',size = 2),
            legend.text = element_text(size = 8))
```

Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
i Please use the `linewidth` argument instead.

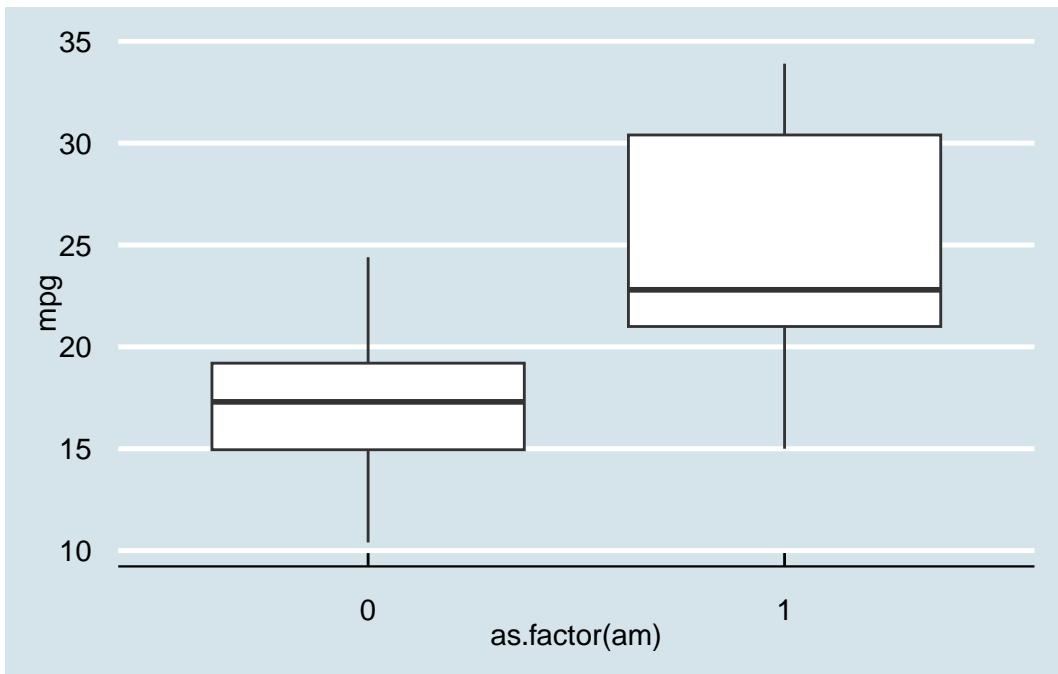
```
ggplot(data=diamonds,aes(x=clarity,fill=cut))+  
  geom_bar()
```



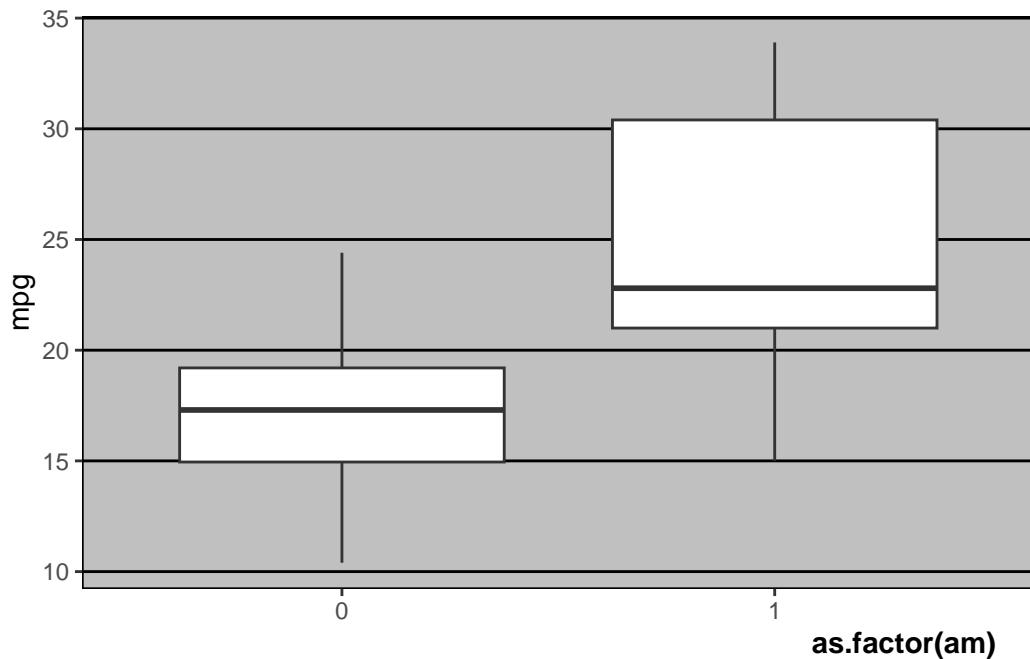
```
theme_set(theme_grey())
#theme_set(old)
ggplot(data=diamonds,aes(x=clarity,fill=cut))+  
  geom_bar()
```



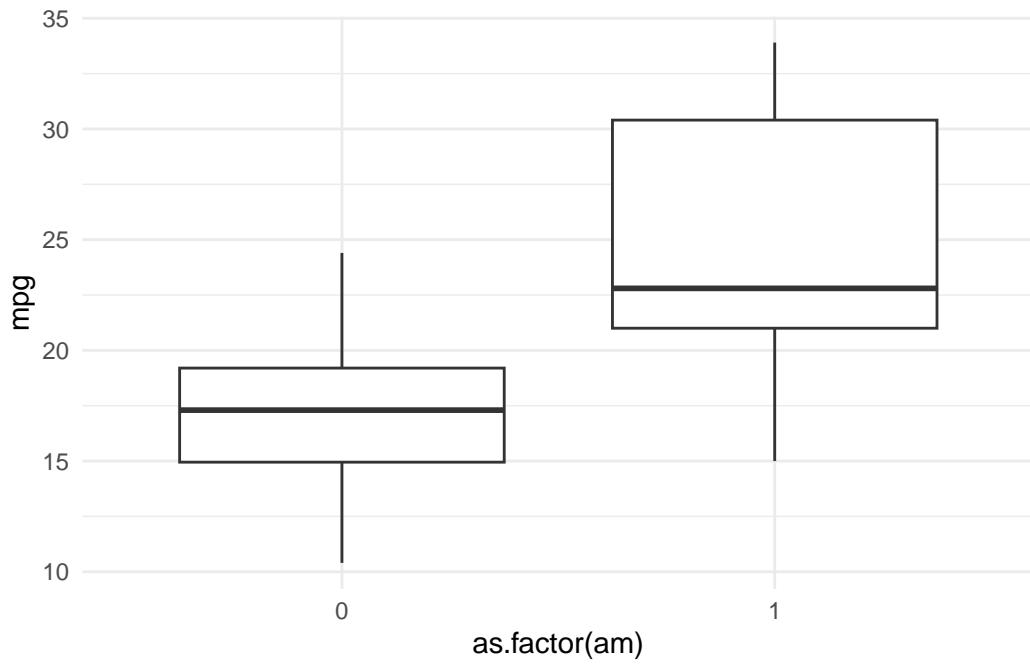
```
# ggthemes #####
plottemp+theme_economist()
```



```
plottemp+theme_excel()+
  theme(axis.title.x = element_text(face='bold', hjust=0.95))
```



```
plottemp+theme_minimal()
```



```
# https://github.com/thomasp85/patchwork  
  
# https://www.data-imaginist.com/2019/a-flurry-of-facets/  
# https://github.com/thomasp85/gganimate
```

6 Descriptive statistics

```
pacman::p_load(tidyverse,wrappedtools)
#dir('Data/')
load('data/bookdata1.RData')

ggplot(rawdata,aes(sysBP_V0, diaBP_V0))+
  geom_point()+
  geom_smooth(se=F)+
  geom_smooth(method='lm',color='red',
              fill='gold', alpha=.15)

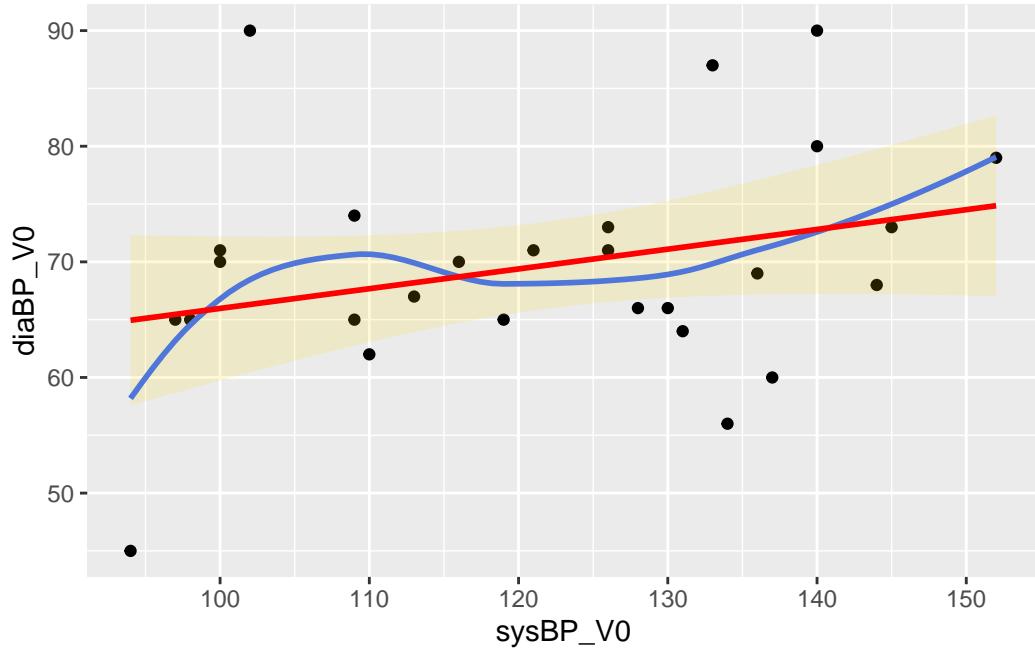
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'

Warning: Removed 1 rows containing non-finite values (`stat_smooth()`).

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

Warning: Removed 1 rows containing non-finite values (`stat_smooth()`).

Warning: Removed 1 rows containing missing values (`geom_point()`).
```



```

# descriptives #####
mean_size <- mean(rawdata$`Size (cm)` )
sd_size <- sd(rawdata$`Size (cm)` )
min(rawdata$`Size (cm)` )

[1] 160

round(mean_size,digits = 2)

[1] 174.11

roundR(mean_size,level = 2)

[1] "174"

meansd(rawdata$`Size (cm)` , roundDig = 4,range = T,add_n = T)

[1] "174.1 ± 7.8 [160.0 -> 193.0] [n=28]"

```

```
meansd(rawdata$sysBP_V0, roundDig = 4, range = T,  
       add_n = T, .german = T)
```

```
[1] "121,9 ± 16,9 [94,0 -> 152,0] [n=27]"
```

```
median(rawdata$`Size (cm)`)
```

```
[1] 173.5
```

```
quantile(rawdata$`Size (cm)`, probs = c(.25, .75))
```

```
25%      75%  
168.00 178.25
```

```
median_quart(rawdata$`Size (cm)`)
```

```
[1] "174 (168/179)"
```

```
table(rawdata$sex, useNA = 'a')
```

```
f      m <NA>  
4     24     0
```

```
sex_count <- table(rawdata$sex, useNA = 'ifany')  
table(rawdata$NYHA_V2, useNA = 'always')
```

```
0      1      2      3 <NA>  
1     6      2      2     17
```

```

table(rawdata$NYHA_V2,useNA = 'i')

0      1      2      3 <NA>
1     6     2     2    17

randomize <- table(rawdata$sex, rawdata$testmedication)
prop.table(sex_count)

f          m
0.1428571 0.8571429

prop.table(randomize,margin = 2)*100

0          1
f 14.28571 14.28571
m 85.71429 85.71429

cat_desc_stats(rawdata$NYHA_V2)

$level
# A tibble: 4 x 1
  value
  <chr>
1 0
2 1
3 2
4 3

$freq
# A tibble: 4 x 1
  desc
  <chr>
1 1 (9%)

```

```

2 6 (55%)
3 2 (18%)
4 2 (18%)

cat_desc_stats(rawdata$sex, singleline = T)

$level
[1] "f m"

$freq
# A tibble: 1 x 1
  desc
  <glue>
1 4 (14%) 24 (86%)


rawdata |>
  group_by(sex,testmedication) |>
  summarise(WeightSummary=meansd(`Weight (kg)`))

`summarise()` has grouped output by 'sex'. You can override using the ` `.groups` argument.

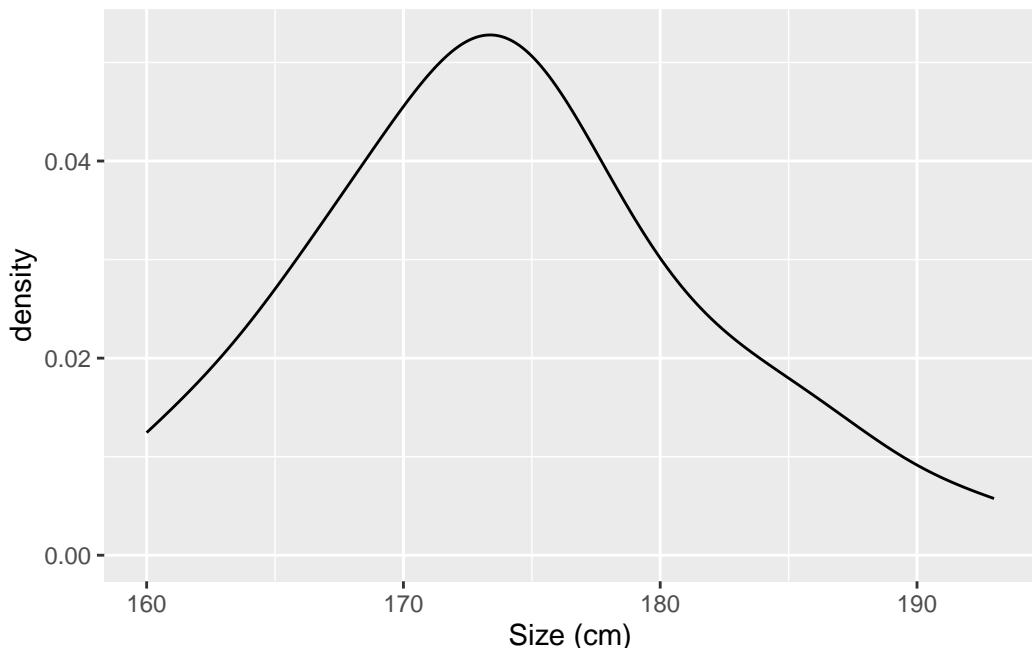
# A tibble: 4 x 3
# Groups:   sex [2]
  sex    testmedication WeightSummary
  <chr>      <dbl> <chr>
1 f            0 86 ± 9
2 f            1 66 ± 3
3 m            0 91 ± 14
4 m            1 90 ± 14


compare2numvars(rawdata,
                 dep_vars = c( "Size (cm)", "Weight (kg)",
                               "sysBP_V0", "diaBP_V0"),
                 indep_var = 'sex',
                 gaussian = F)

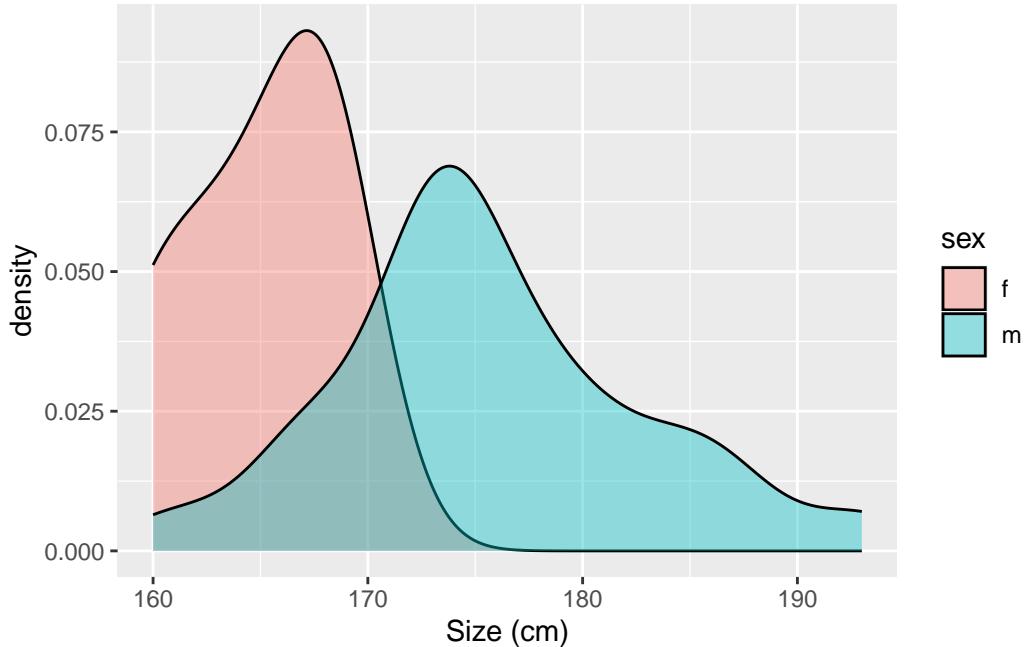
```

```
# A tibble: 4 x 5
  Variable desc_all `sex f` `sex m` p
  <fct>    <chr>     <chr>     <chr>   <chr>
1 Size (cm) 174 (168/179) 166 (162/168) 174 (172/180) 0.011
2 Weight (kg) 84 (77/96) 74 (66/88) 86 (80/104) 0.107
3 sysBP_V0 126 (109/136) 119 (105/129) 126 (109/137) 0.609
4 diaBP_V0 69 (65/73) 66 (64/80) 70 (65/73) 0.784
```

```
ggplot(rawdata,aes(`Size (cm)`))+
  geom_density()
```



```
ggplot(rawdata,aes(`Size (cm)` , fill=sex))+  
  geom_density(alpha=.4)
```



```
ks.test(rawdata$`Size (cm)`,
       pnorm) #WRONG!!!!!
```

Warning in ks.test.default(rawdata\$`Size (cm)` , pnorm): für den Komogorov-Smirnov-Test sollten keine Bindungen vorhanden sein

Asymptotic one-sample Kolmogorov-Smirnov test

```
data: rawdata$`Size (cm)`
D = 1, p-value < 2.2e-16
alternative hypothesis: two-sided
```

```
ks.test(rawdata$`Size (cm)` ,
       pnorm,
       mean=mean(rawdata$`Size (cm)`),
       sd=sd(rawdata$`Size (cm)`))
```

Warning in ks.test.default(rawdata\$`Size (cm)` , pnorm, mean =

```
mean(rawdata$`Size (cm)`), : für den Komogorov-Smirnov-Test sollten keine  
Bindungen vorhanden sein
```

```
Asymptotic one-sample Kolmogorov-Smirnov test
```

```
data: rawdata$`Size (cm)`  
D = 0.13284, p-value = 0.7064  
alternative hypothesis: two-sided
```

```
ksnormal(rawdata$`Size (cm)`)
```

```
[1] 0.7063825
```

```
shapiro.test(rawdata$`Size (cm)`)
```

```
Shapiro-Wilk normality test
```

```
data: rawdata$`Size (cm)`  
W = 0.9766, p-value = 0.7627
```

7 Better tables with `flextable`

8 Simple test statistics

```
pacman::p_load(plotrix,tidyverse, wrappedtools,
                coin,ggsignif, patchwork, ggbeeswarm)
#conflicted)
# conflict_prefer("filter", "dplyr")
load('data/bookdata1.RData')

##### Test for normal distribution -----
ks.test(x = rawdata$`Size (cm)`,
        'pnorm',
        mean(rawdata$`Size (cm)` ,na.rm=T),
        sd(rawdata$`Size (cm)` ,na.rm=T),
        exact=F)
```

Warning in ks.test.default(x = rawdata\$`Size (cm)`, "pnorm", mean(rawdata\$`Size (cm)` , : für den Komogorov-Smirnov-Test sollten keine Bindungen vorhanden sein

```
Asymptotic one-sample Kolmogorov-Smirnov test

data: rawdata$`Size (cm)`
D = 0.13284, p-value = 0.7064
alternative hypothesis: two-sided

ksnormal(rawdata$`Weight (kg)`)

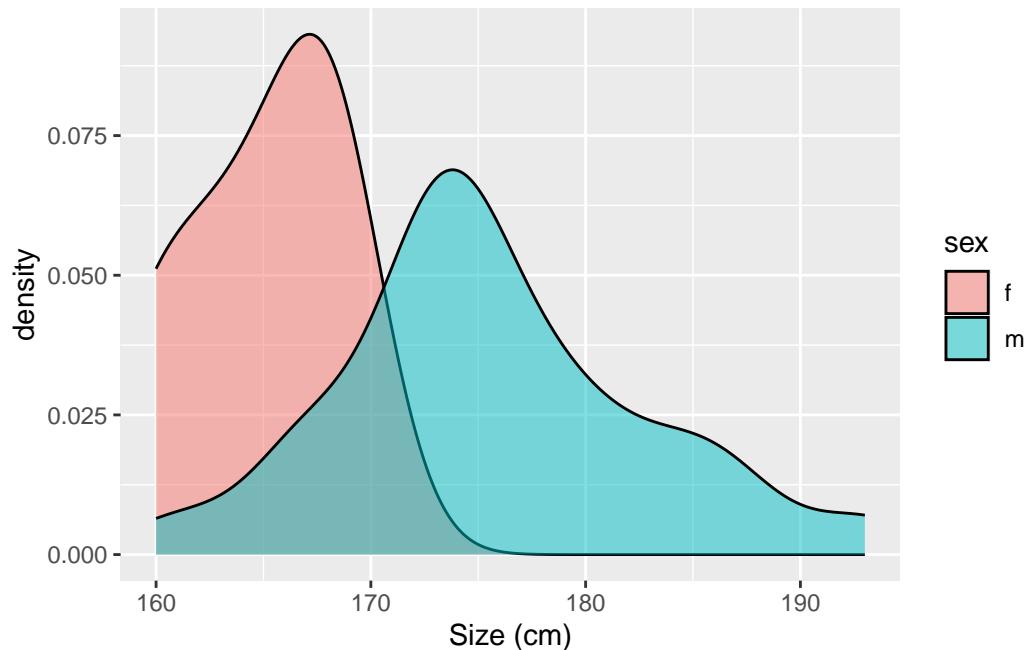
[1] 0.7598605

shapiro.test(rawdata$`Size (cm)`)
```

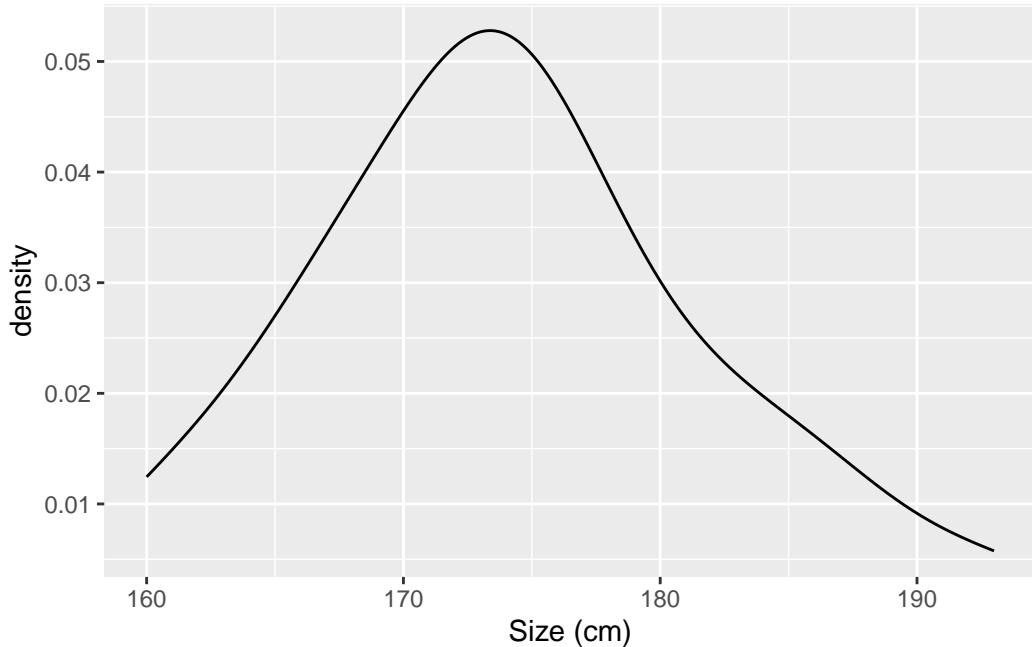
Shapiro-Wilk normality test

```
data: rawdata$`Size (cm)`  
W = 0.9766, p-value = 0.7627
```

```
ggplot(rawdata,aes(x = `Size (cm)`,fill=sex))+  
  geom_density(alpha=.5)
```



```
ggplot(rawdata,aes(x = `Size (cm)`))+  
  geom_line(stat='density')
```



```
(ksout<-ksnormal(x = rawdata$`Size (cm)`))
```

```
[1] 0.7063825
```

```
# test, if log-normal (->then transform)
if (ksnormal(rawdata$`Size (cm)`)<=.05 &
    ksnorma(log(rawdata$`Size (cm)`))>.05) {
  print('lognormal')
  # create new log-transformed variable or just transform...
} else {
  print('not lognormal')
}
```

```
[1] "not lognormal"
```

```
##### quantitative measures with Gaussian distribution -----
ggplot(rawdata,aes(x=sex,y=`Size (cm)`))+  

  geom_beeswarm(size=3)+  

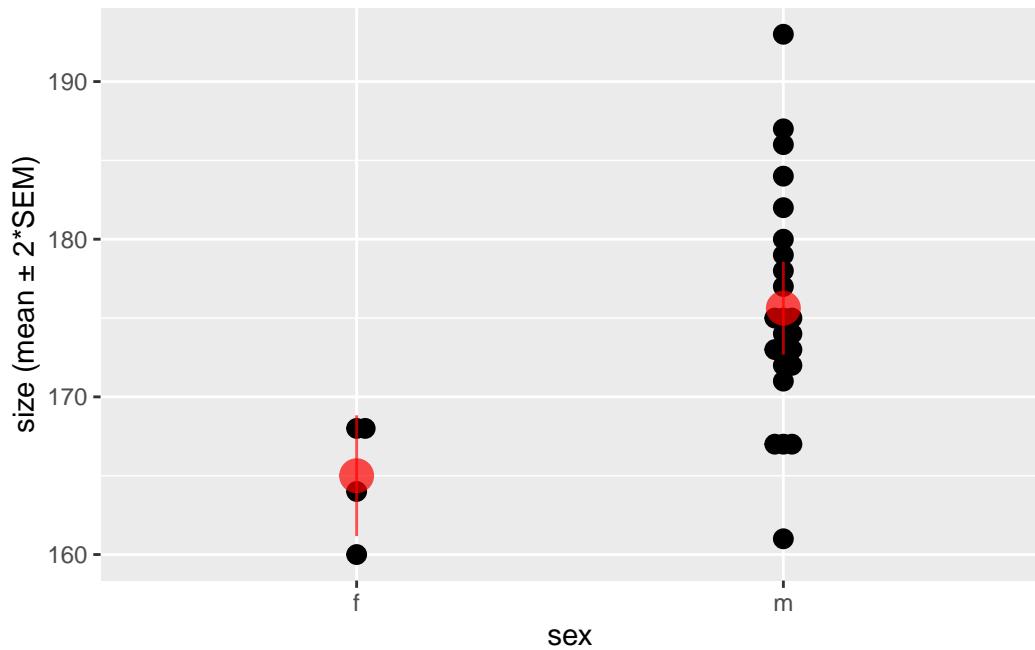
  stat_summary(color='red',size=1.2,alpha=.7,
```

```

    fun.data='mean_se',fun.args=list(mult=2))+  

ylab('size (mean \u00B1 2*SEM)')

```



```

by(data = rawdata$`Size (cm)`, INDICES = rawdata$sex,  

  FUN = meanse)

```

```

rawdata$sex: f  

[1] "165±2"

```

```

rawdata$sex: m  

[1] "176±1"

```

```

apply(X=rawdata[quantvars$index], MARGIN=2,  

  FUN=meansd, roundDig=2)

```

included	finalized	testmedication	Size (cm)
"0.96 ± 0.19"	"0.86 ± 0.36"	"0.50 ± 0.51"	"174 ± 8"
Weight (kg)	sysBP_V0	diaBP_V0	lv_edv_mri
"88 ± 15"	"122 ± 17"	"70 ± 10"	"206 ± 70"

```

lv_esv_mri      lv_ef_mri  lv_ef_biplan_mri      ptt_lab
"110 ± 70"     "50 ± 15"   "49 ± 14"           "32 ± 6"
ferritin_lab    iron_lab   transferrin_lab       sysBP_V2
"281 ± 237"    "85 ± 37"   "258 ± 36"         "119 ± 13"
diaBP_V2        BMI        age                  NYHA_V1
"66 ± 9"        "29 ± 4"    "60 ± 8"           "1.4 ± 0.9"
NYHA_V2         NYHA_V3
"1.5 ± 0.9"    "1.8 ± 0.9"

```

```

rawdata %>% group_by(sex) %>%
  summarize(Mean=mean(`Size (cm)` ,na.rm=T)) %>%
  ungroup()

```

```

# A tibble: 2 x 2
  sex      Mean
  <chr> <dbl>
1 f        165
2 m        176.

```

```

t.test(x = rawdata$`Size (cm)`[which(rawdata$sex=='f')],
       y = rawdata$`Size (cm)`[which(rawdata$sex=='m')])

```

Welch Two Sample t-test

```

data: rawdata$`Size (cm)`[which(rawdata$sex == "f")] and rawdata$`Size (cm)`[which(rawdata$sex == "m")]
t = -4.3967, df = 7.2767, p-value = 0.002887
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-16.295575 -4.954425
sample estimates:
mean of x mean of y
165.000 175.625

```

```
(tOut<-t.test(rawdata$`Size (cm)`~rawdata$sex))
```

```
Welch Two Sample t-test
```

```
data: rawdata$`Size (cm)` by rawdata$sex
t = -4.3967, df = 7.2767, p-value = 0.002887
alternative hypothesis: true difference in means between group f and group m is not equal to
95 percent confidence interval:
-16.295575 -4.954425
sample estimates:
mean in group f mean in group m
165.000          175.625
```

```
tOut$p.value
```

```
[1] 0.00288704
```

```
# equal variances assumption?
(vartestOut<-var.test(rawdata$`Size (cm)`~rawdata$sex))
```

```
F test to compare two variances
```

```
data: rawdata$`Size (cm)` by rawdata$sex
F = 0.2812, num df = 3, denom df = 23, p-value = 0.3232
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
0.07497668 3.97434769
sample estimates:
ratio of variances
0.281199
```

```
(tOut<-t.test(rawdata$`Size (cm)`~rawdata$sex,
               var.equal = vartestOut$p.value>.05))
```

```
Two Sample t-test
```

```
data: rawdata$`Size (cm)` by rawdata$sex
```

```
t = -2.8446, df = 26, p-value = 0.008552
alternative hypothesis: true difference in means between group f and group m is not equal to
95 percent confidence interval:
-18.302594 -2.947406
sample estimates:
mean in group f mean in group m
165.000          175.625

(t0out<-
  t.test(rawdata$`Size (cm)`~rawdata$sex,
         var.equal=var.test(
           rawdata$`Size (cm)`~rawdata$sex)$p.value>.05))
```

Two Sample t-test

```
data: rawdata$`Size (cm)` by rawdata$sex
t = -2.8446, df = 26, p-value = 0.008552
alternative hypothesis: true difference in means between group f and group m is not equal to
95 percent confidence interval:
-18.302594 -2.947406
sample estimates:
mean in group f mean in group m
165.000          175.625

t_var_test(data = rawdata,formula = '`Size (cm)`~sex')
```

Two Sample t-test

```
data: Size (cm) by sex
t = -2.8446, df = 26, p-value = 0.008552
alternative hypothesis: true difference in means between group f and group m is not equal to
95 percent confidence interval:
-18.302594 -2.947406
sample estimates:
mean in group f mean in group m
165.000          175.625
```

```
print(c(mean(rawdata$sysBP_V0,na.rm=T),  
       mean(rawdata$sysBP_V2,na.rm=T)))
```

```
[1] 121.8519 119.4583
```

```
t.test(rawdata$sysBP_V0,  
       rawdata$sysBP_V2,  
       alternative='greater', # x>y  
       paired=T) #pairwise t-test, within subject
```

Paired t-test

```
data: rawdata$sysBP_V0 and rawdata$sysBP_V2  
t = 0.88151, df = 23, p-value = 0.1936  
alternative hypothesis: true mean difference is greater than 0  
95 percent confidence interval:  
-2.793386      Inf  
sample estimates:  
mean difference  
2.958333
```

```
t.test(rawdata$sysBP_V0,  
       rawdata$sysBP_V2,  
       # alternative='greater', # x>y  
       paired=T)$p.value/2 #pairwise t-test, within subject
```

```
[1] 0.1935805
```

```
t.test(rawdata`Size (cm)`,mu = 173)
```

One Sample t-test

```
data: rawdata`Size (cm)`  
t = 0.75384, df = 27, p-value = 0.4575
```

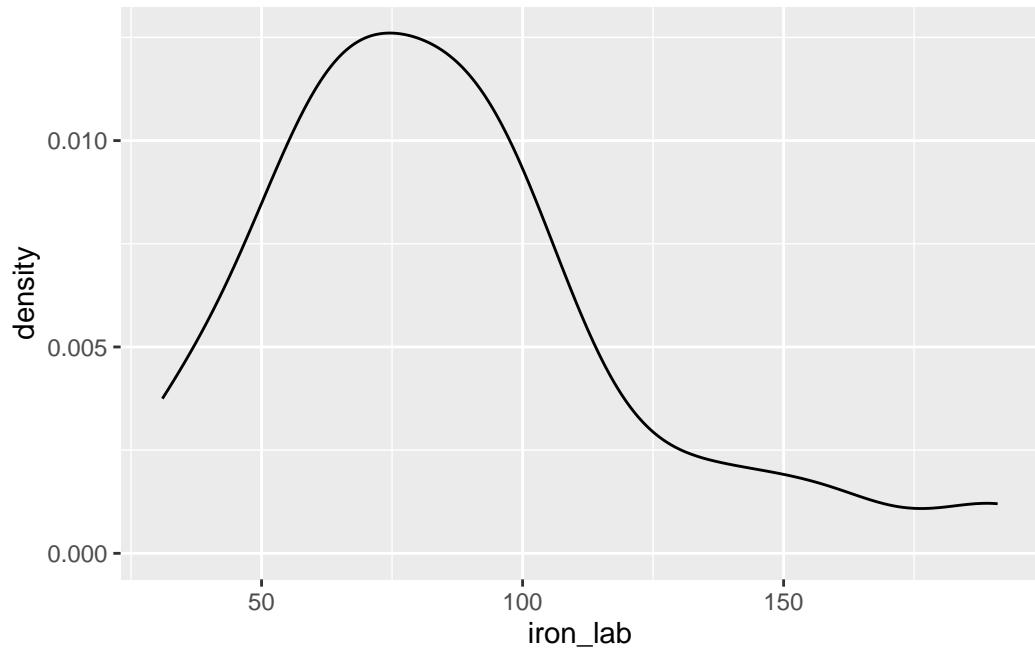
```
alternative hypothesis: true mean is not equal to 173
95 percent confidence interval:
171.0937 177.1206
sample estimates:
mean of x
174.1071
```

```
##### ordinal data -----
ordvars$names
```

```
[1] "ptt_lab"           "ferritin_lab"      "iron_lab"        "transferrin_lab"
[5] "age"
```

```
ggplot(rawdata,aes(iron_lab))+
  geom_density()
```

```
Warning: Removed 1 rows containing non-finite values (`stat_density()`).
```



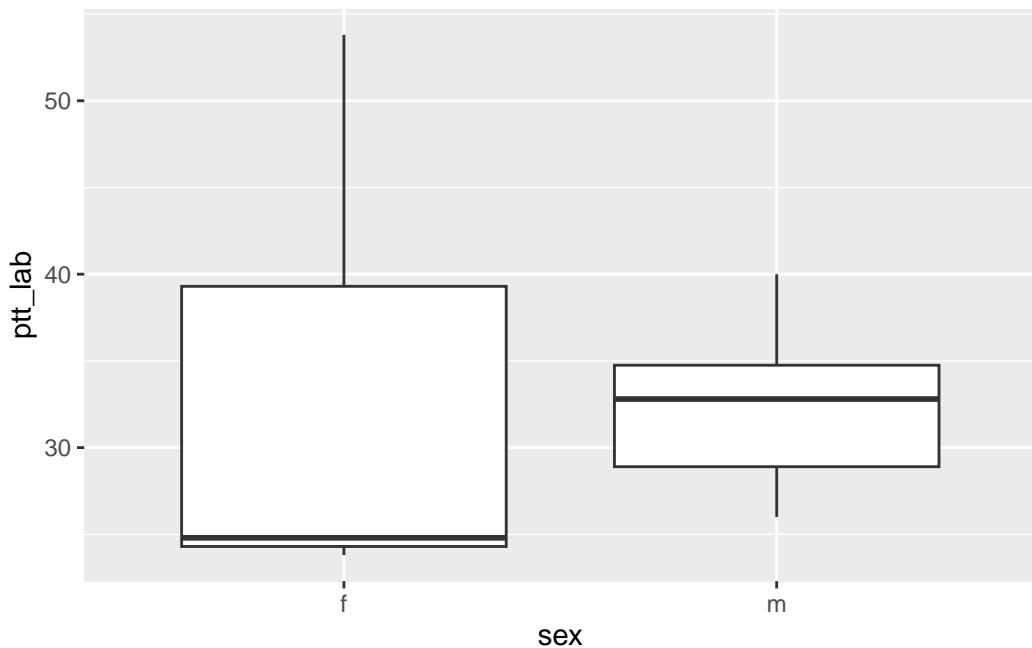
```
by(data = rawdata[[ordvars$index[1]]],  
  INDICES = rawdata$sex, FUN = median_quart)
```

```
rawdata$sex: f  
[1] "25 (24/49)"
```

```
-----  
rawdata$sex: m  
[1] "33 (29/35)"
```

```
ggplot(rawdata, aes(sex, ptt_lab)) +  
  geom_boxplot()
```

Warning: Removed 1 rows containing non-finite values (`stat_boxplot()`).



```
(uOut<-wilcox.test(  
  rawdata[[ordvars$index[1]]] ~ rawdata$sex, exact=F))
```

```

Wilcoxon rank sum test with continuity correction

data: rawdata[[ordvars$index[1]]] by rawdata$sex
W = 24, p-value = 0.3748
alternative hypothesis: true location shift is not equal to 0

uOut$p.value

[1] 0.3748016

# coin::wilcox_test
(uOut2<-wilcox_test(ptt_lab~as.factor(sex),
                     data=rawdata))

Asymptotic Wilcoxon-Mann-Whitney Test

data: ptt_lab by as.factor(sex) (f, m)
Z = -0.9261, p-value = 0.3544
alternative hypothesis: true mu is not equal to 0

pvalue(uOut2) #no list-object, but methods to extract infos like p

[1] 0.3543925

wilcox.test(ptt_lab~sex,exact=F,correct=F,
            data=rawdata)

Wilcoxon rank sum test

data: ptt_lab by sex
W = 24, p-value = 0.3544
alternative hypothesis: true location shift is not equal to 0

```

```
wilcox.test(x=rawdata$sysBP_V0,y=rawdata$sysBP_V2,  
            exact=F,  
            correct=T,paired=T)
```

```
Wilcoxon signed rank test with continuity correction  
  
data: rawdata$sysBP_V0 and rawdata$sysBP_V2  
V = 143, p-value = 0.3478  
alternative hypothesis: true location shift is not equal to 0
```

```
##### categorial data -----  
factvars$names
```

```
[1] "testmedication" "sex"                 "NYHA_V1"           "NYHA_V2"  
[5] "NYHA_V3"
```

```
(crosstab<-table(rawdata$sex,rawdata$testmedication))
```

```
      0   1  
f    2   2  
m 12  12
```

```
chisq.test(crosstab,simulate.p.value=T,B=10^5) #empirical p-value
```

```
Pearson's Chi-squared test with simulated p-value (based on 1e+05  
replicates)
```

```
data: crosstab  
X-squared = 0, df = NA, p-value = 1
```

```
chisq.test(table(rawdata$sex,rawdata$NYHA_V1)) #based on table
```

```
Warning in chisq.test(table(rawdata$sex, rawdata$NYHA_V1)):  
Chi-Quadrat-Approximation kann inkorrekt sein
```

```
Pearson's Chi-squared test
```

```
data: table(rawdata$sex, rawdata$NYHA_V1)  
X-squared = 4.3849, df = 3, p-value = 0.2228
```

```
chisq.test(x=rawdata$sex,y=rawdata$NYHA_V1,  
simulate.p.value=T,B=10^5) #based on rawdata
```

```
Pearson's Chi-squared test with simulated p-value (based on 1e+05  
replicates)
```

```
data: rawdata$sex and rawdata$NYHA_V1  
X-squared = 4.3849, df = NA, p-value = 0.1741
```

```
(crosstab1<-table(rawdata$sex,  
rawdata$`Weight (kg)`<=  
median(rawdata$`Weight (kg)`)))
```

```
      FALSE  TRUE  
f       1     3  
m      13    11
```

```
prop.table(crosstab1)
```

```
      FALSE      TRUE  
f 0.03571429 0.10714286  
m 0.46428571 0.39285714
```

```
round(prop.table(crosstab1)*100)
```

```
    FALSE TRUE  
f      4   11  
m     46   39  
  
round(prop.table(crosstab1,margin=2)*100) #% per column
```

```
    FALSE TRUE  
f      7   21  
m     93   79  
  
round(prop.table(crosstab1,margin=1)*100) #% per row
```

```
    FALSE TRUE  
f     25   75  
m     54   46  
  
(tabletestOut<-chisq.test(crosstab1,simulate.p.value=T,  
                           B=10^5))
```

Pearson's Chi-squared test with simulated p-value (based on 1e+05 replicates)

```
data: crosstab1  
X-squared = 1.1667, df = NA, p-value = 0.5988
```

```
tabletestOut$p.value
```

```
[1] 0.598784
```

```
tabletestOut$expected
```

```

    FALSE TRUE
f      2     2
m     12    12

tabletestOut$observed

    FALSE TRUE
f      1     3
m     13    11

tabletestOut$statistic

X-squared
1.166667

# if minimum(expected<5) then Fishers exact test
if (min(tabletestOut$expected)<5) {
  tabletestOut<-fisher.test(crosstab1)
}
tabletestOut$p.value

[1] 0.5955556

groupvars <- ColSeeker(namepattern = c('sex','test'))

##### report -----
# report_qv %<>%
#   mutate(
#     Variable=factor(Variable, levels=quantvars$names),
#     female='', male='', `p sexdiff`="") %>%
#   arrange(Variable)
# for (var_i in 1:quantvars$count) {
#   report_qv[var_i, c('female','male')] <-
#     meansd(rawdata[[quantvars$index[var_i]]]),

```

```

#         groupvar = rawdata %>% pull(groupvars$names[2]), #$sex,
#         add_n = T, rangesep = ' ' ,
#         range = T) %>%
#         as.list()
# evOut<-var.test((rawdata %>% pull(quantvars)index[var_i]))~
#                     rawdata$sex)
# tOut<-t.test((rawdata %>% pull(quantvars)index[var_i]))~
#                     rawdata$sex,
#                     var.equal=evOut$p.value>.05)
# report_qv$p sexdiff`[var_i] <- formatP(tOut$p.value)
# rm(evOut,tOut)
# }

compare2numvars(data = rawdata,dep_vars = quantvars$names,
                 indep_var = "sex",gaussian = T)

```

```

# A tibble: 22 x 5
  Variable     desc_all   `sex f`   `sex m`      p
  <fct>       <chr>      <chr>      <chr>      <chr>
1 included    0.96 ± 0.19 1.0 ± 0.0 1.0 ± 0.2  0.328
2 finalized   0.86 ± 0.36 0.75 ± 0.50 0.88 ± 0.34 0.526
3 testmedication 0.50 ± 0.51 0.50 ± 0.58 0.50 ± 0.51 1.000
4 Size (cm)   174 ± 8    165 ± 4    176 ± 7    0.009
5 Weight (kg) 88 ± 15    76 ± 13    90 ± 14    0.072
6 sysBP_V0    122 ± 17   118 ± 14   123 ± 18   0.587
7 diaBP_V0    70 ± 10    71 ± 13    69 ± 9    0.780
8 lv_edv_mri  206 ± 70   235 ± 72   203 ± 71   0.559
9 lv_esv_mri  110 ± 70   158 ± 56   105 ± 71   0.322
10 lv_ef_mri  50 ± 15    33 ± 3    52 ± 15   0.095
# i 12 more rows

```

```

compare2numvars(data = rawdata,dep_vars = quantvars$names,
                 indep_var = "testmedication",gaussian = T)

```

```

# A tibble: 21 x 5
  Variable     desc_all   `testmedication 0`   `testmedication 1`      p
  <fct>       <chr>      <chr>                  <chr>      <chr>
1 included    0.96 ± 0.19 1.0 ± 0.0            0.9 ± 0.3  0.336
2 finalized   0.86 ± 0.36 0.79 ± 0.43          0.93 ± 0.27 0.297

```

```

3 Size (cm)      174 ± 8    173 ± 6    175 ± 9    0.653
4 Weight (kg)    88 ± 15    90 ± 14    86 ± 16    0.448
5 sysBP_V0       122 ± 17    122 ± 15    122 ± 19    0.966
6 diaBP_V0       70 ± 10     70 ± 7     70 ± 12     0.943
7 lv_edv_mri    206 ± 70    243 ± 77    173 ± 45    0.019
8 lv_esv_mri    110 ± 70    138 ± 87    86 ± 41     0.108
9 lv_ef_mri     50 ± 15     48 ± 18     52 ± 12     0.532
10 lv_ef_biplan_mri 49 ± 14   47 ± 17     50 ± 13     0.747
# i 11 more rows

for(group_i in seq_len(groupvars$count)){
  resulttmp <- compare2numvars(data = rawdata, dep_vars = quantvars$names,
                                indep_var = groupvars$names[group_i], gaussian = T)
  print(resulttmp)
}

# A tibble: 21 x 5
  Variable      desc_all `testmedication 0` `testmedication 1` p
  <fct>        <chr>      <chr>          <chr>          <chr>
1 included     0.96 ± 0.19 1.0 ± 0.0    0.9 ± 0.3    0.336
2 finalized    0.86 ± 0.36 0.79 ± 0.43   0.93 ± 0.27   0.297
3 Size (cm)    174 ± 8    173 ± 6    175 ± 9    0.653
4 Weight (kg)   88 ± 15    90 ± 14    86 ± 16    0.448
5 sysBP_V0     122 ± 17    122 ± 15    122 ± 19    0.966
6 diaBP_V0     70 ± 10     70 ± 7     70 ± 12     0.943
7 lv_edv_mri   206 ± 70    243 ± 77    173 ± 45    0.019
8 lv_esv_mri   110 ± 70    138 ± 87    86 ± 41     0.108
9 lv_ef_mri    50 ± 15     48 ± 18     52 ± 12     0.532
10 lv_ef_biplan_mri 49 ± 14   47 ± 17     50 ± 13     0.747
# i 11 more rows
# A tibble: 22 x 5
  Variable      desc_all `sex f`   `sex m`   p
  <fct>        <chr>      <chr>      <chr>      <chr>
1 included     0.96 ± 0.19 1.0 ± 0.0  1.0 ± 0.2  0.328
2 finalized    0.86 ± 0.36 0.75 ± 0.50 0.88 ± 0.34 0.526
3 testmedication 0.50 ± 0.51 0.50 ± 0.58 0.50 ± 0.51 1.000
4 Size (cm)    174 ± 8    165 ± 4    176 ± 7    0.009
5 Weight (kg)   88 ± 15    76 ± 13    90 ± 14    0.072
6 sysBP_V0     122 ± 17    118 ± 14    123 ± 18    0.587
7 diaBP_V0     70 ± 10     71 ± 13    69 ± 9     0.780
8 lv_edv_mri   206 ± 70    235 ± 72    203 ± 71    0.559

```

```

9 lv_esv_mri      110 ± 70     158 ± 56     105 ± 71     0.322
10 lv_ef_mri      50 ± 15      33 ± 3       52 ± 15      0.095
# i 12 more rows

# report_ov

# report_ov %<>%
#   mutate(
#     Variable=factor(Variable, levels=ordvars$names),
#     female='', male='', `p sexdiff`="") %>%
#   arrange(Variable)
# for (var_i in seq_len(ordvars$count)) {
#   report_ov[var_i,c('female','male')] <-
#     # by(rawdata[[ordvars$names[var_i]]],
#     #     rawdata$sex,quantile,probs=c(.25,.75),na.rm=T)
#     median_quart(rawdata[[ordvars$index[var_i]]]),
#     groupvar = rawdata$sex,
#     add_n = T,rangesep = '  ',
#     range = T) %>%
#   as.list()
#   wOut<-wilcox.test((rawdata %>% pull(ordvars$index[var_i]))~
#                         rawdata$sex,exact=F)
#   report_ov$`p sexdiff`[var_i] <- formatP(wOut$p.value)
#   plottmp <-
#     rawdata %>% ggplot(aes_string('sex',ordvars$bticked[var_i]))+
#     geom_boxplot()+
#     geom_signif(comparisons = list(c(1,2)),
#                 annotations = paste0('p ',
#                                       formatP(wOut$p.value,
#                                               pretext = T)))+
#     scale_y_continuous(expand = expansion(mult = c(.1,.15)))
#   print(plottmp)
# }

compare2numvars(data = rawdata,dep_vars = ordvars$names,
                 indep_var = "sex",gaussian = F)

# A tibble: 5 x 5
  Variable      desc_all    `sex f`    `sex m`      p
  <fct>        <chr>       <chr>       <chr>       <chr>
1 ptt_lab      33 (28/35)  25 (24/49)  33 (29/35)  0.375

```

```

2 ferritin_lab    222 (162/339) 138 (90/591) 224 (172/316) 0.616
3 iron_lab        80 (61/102)    95 (92/103)   75 (60/99)    0.165
4 transferrin_lab 261 (233/276) 260 (227/299) 262 (235/276) 1.000
5 age              62 (53/67)     66 (58/69)    60 (53/66)    0.250

# report_cat
groupvar <- 'sex'
# report_cat %<>%
#   mutate(
#     Variable=factor(Variable, levels=factvars$names),
#     female='', male='', `p sexdiff`="") %>%
#   arrange(Variable)
#
# for(var_i in seq_along(factvars$names)){
#   # if(factvars$names[var_i]!=groupvar){
#   report_cat[var_i,c('female','male')] <-
#     cat_desc_stats(rawdata[[factvars$index[var_i]]],
#                   groupvar = rawdata$sex,
#                   singleline = T,separator = ' / ',
#                   return_level = F) %>%
#       as.list()
#   test_out <- fisher.test(rawdata[[factvars$index[var_i]]],
#                           rawdata[[groupvar]])
#   report_cat$p sexdiff`[var_i] <- formatP(test_out$p.value)
#
#   p1 <- ggplot(rawdata,aes_string(groupvar,
#                                     fill=factvars$bticked[var_i]))+
#     geom_bar()+
#     geom_signif(
#       aes(y=max(table(sex))+2),
#       comparisons = list(c(1,2)),
#       annotations = paste0('p ',
#                           formatP(test_out$p.value,
#                                   pretext = T)))+
#       scale_y_continuous(expand = expansion(mult = c(.1,.15)))
#   p2 <- ggplot(rawdata,aes_string(groupvar,
#                                     fill=factvars$bticked[var_i]))+
#     geom_bar(position='fill')+
#     geom_signif(
#       aes(y=1.05),
#       comparisons = list(c(1,2)),

```

```

#           annotations = paste0('p ',
#                               formatP(test_out$p.value,
#                                       pretext = T)))+
#           scale_y_continuous(expand = expansion(mult = c(.1,.15)),
#                               labels = scales::percent)
# p3 <- ggplot(rawdata %>%
#               filter(!is.na (!!sym(factvars$names[var_i]))),
#               aes_string(groupvar,
#                          fill=factvars$bticked[var_i]))+
#               geom_bar(position='fill')+
#               geom_signif(
#                 aes(y=1.05),
#                 comparisons = list(c(1,2)),
#                 annotations = paste0('p ',
#                                     formatP(test_out$p.value,
#                                             pretext = T)))+
#               scale_y_continuous(expand = expansion(mult = c(.1,.15)),
#                                   labels = scales::percent)
# print(p1/(p2+p3))
# print(p1/(p2+p3)+plot_layout(guides = "collect") &
#       theme(legend.position = 'bottom'))
#   # }
# }

compare2qualvars(rawdata,dep_vars = factvars$names,
                  indep_var = groupvar,spacer = ' ')

```

# A tibble: 20 x 5	Variable	desc_all	`sex f`	`sex m`	p
	<chr>	<chr>	<chr>	<chr>	<chr>
1	"testmedication"	" "	" "	" "	"1.000"
2	"0"	"14 (50%)"	"2 (50%)"	"12 (50%)"	" "
3	"1"	"14 (50%)"	"2 (50%)"	"12 (50%)"	" "
4	"sex"	" "	" "	" "	"0.001"
5	"f"	"4 (14.29%)"	"4 (100%)"	"0 (0%)"	" "
6	"m"	"24 (85.71%)"	"0 (0%)"	"24 (100%)"	" "
7	"NYHA_V1"	" "	" "	" "	"0.096"
8	"0"	"2 (11.76%)"	"1 (33.33%)"	"1 (7.14%)"	" "
9	"1"	"9 (52.94%)"	"0 (0%)"	"9 (64.29%)"	" "
10	"2"	"3 (17.65%)"	"1 (33.33%)"	"2 (14.29%)"	" "
11	"3"	"3 (17.65%)"	"1 (33.33%)"	"2 (14.29%)"	" "

12	"NYHA_V2"	" "	" "	" "	"1.000"
13	" 0"	"1 (9.09%)"	"0 (0%)"	"1 (12.5%)"	" "
14	" 1"	"6 (54.55%)"	"2 (66.67%)"	"4 (50%)"	" "
15	" 2"	"2 (18.18%)"	"1 (33.33%)"	"1 (12.5%)"	" "
16	" 3"	"2 (18.18%)"	"0 (0%)"	"2 (25%)"	" "
17	"NYHA_V3"	" "	" "	" "	"0.091"
18	" 1"	"6 (50%)"	"0 (0%)"	"6 (66.67%)"	" "
19	" 2"	"3 (25%)"	"1 (33.33%)"	"2 (22.22%)"	" "
20	" 3"	"3 (25%)"	"2 (66.67%)"	"1 (11.11%)"	" "

9 Intro to lm

In this chapter, linear models (including linear regression and ANOVA) will be introduced. Output is not optimized for print, but rather for interactive use.

9.1 Setup

All packages necessary will be invoked by p_load. Packages with only a single function call or potential for name conflicts can be unloaded, this way we still checked for their existence and installed them if need be.

```
pacman::p_load(conflicted,wrappedtools,car,nlme,broom,
                 multcomp,tidyverse,foreign,DescTools, ez,
                 ggbeeswarm,
                 lme4, nlme,merTools,
                 easystats, patchwork,here)#conflicted,
# rayshader,av)
# pacman::p_unload(DescTools, foreign)
# conflict_scout()
conflicts_prefer(dplyr::select,
                  dplyr::filter,
                  modelbased::standardize)

[conflicted] Will prefer dplyr::select over any other package.
[conflicted] Will prefer dplyr::filter over any other package.
[conflicted] Will prefer modelbased::standardize over any other package.

base_dir <- here::here()
```

9.2 Import / Preparation

Data are read from an SPSS file. Numeric column Passage is mutated into a factor as Passage_F, this is necessary for group comparisons in ANOVA. The call to here() expands the path to a file from the project directory to the full system path.

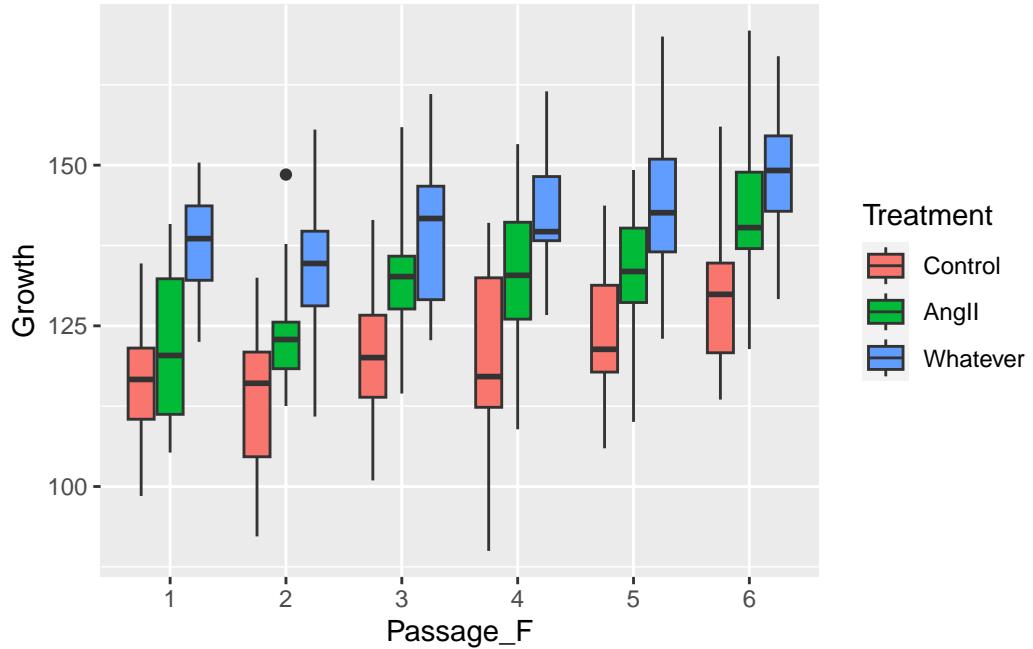
```
rawdata<-foreign::read.spss(file=here('Data/Zellbeads.sav'),  
                           use.value.labels=T,to.data.frame=T) %>%  
  as_tibble() %>%  
  dplyr::select(-ZahlZellen) |>  
  rename(Growth=Wachstum,Treatment=Bedingung) |>  
  mutate(Passage_F=factor(Passage),  
         Treatment=fct_recode(Treatment,  
                               Control="Kontrolle"))
```

re-encoding from CP1252

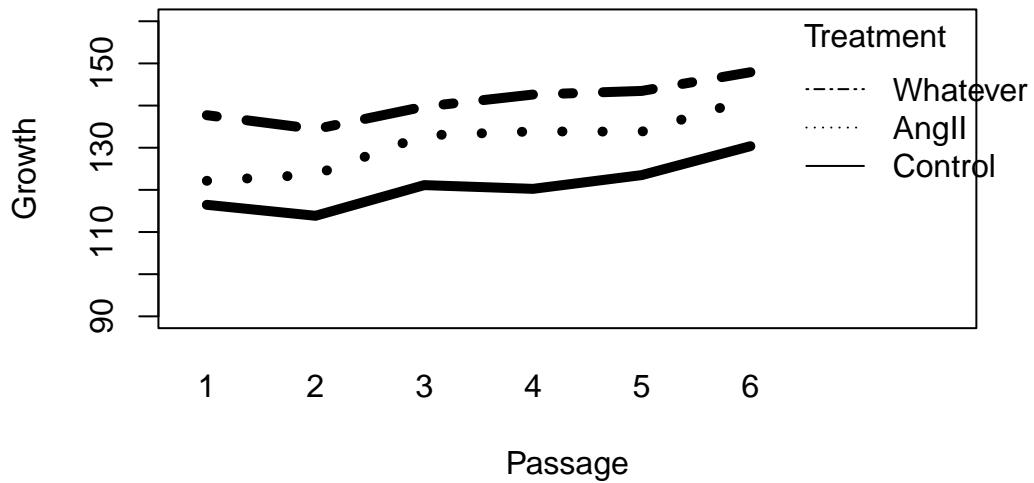
9.3 Graphical exploration

First impression of the data will be attempted by grouped boxplot, followed by interaction plots, both as basic and ggplot with variations.

```
ggplot(rawdata,aes(Passage_F,Growth, fill=Treatment))+  
  geom_boxplot(coef=3)
```



```
with(rawdata, interaction.plot(
  x.factor=Passage, trace.factor=Treatment, response=Growth,
  ylim = c(90, 160), lty = c(1,3,12), lwd = 5,
  ylab = "Growth", xlab = "Passage",
  trace.label = "Treatment"))
```



```

# p1<-ggplot(rawdata,aes(x=Passage,y=Growth))+  

#   stat_summary(geom='line',fun='mean',aes(color=Treatment))+  

#   stat_summary(geom='line',fun='mean')  

p1<-ggplot(rawdata,aes(x=Passage,y=Growth))+  

  stat_summary(geom='line',fun='mean',aes(color=Treatment))+  

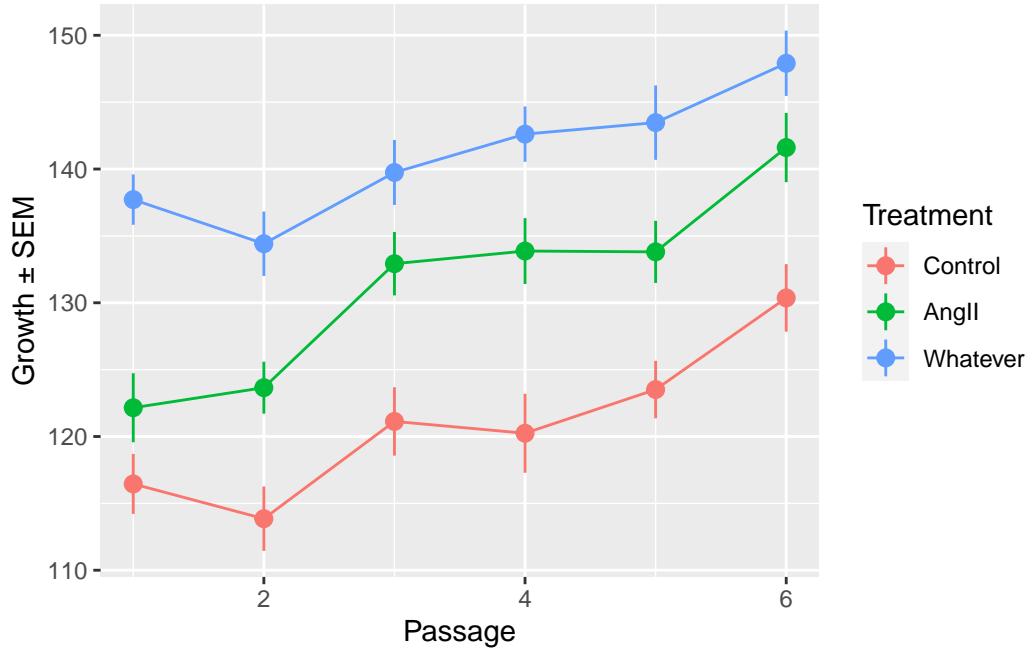
  stat_summary(aes(color=Treatment))+  

  ylab('Growth \u00b1 SEM')  

p1

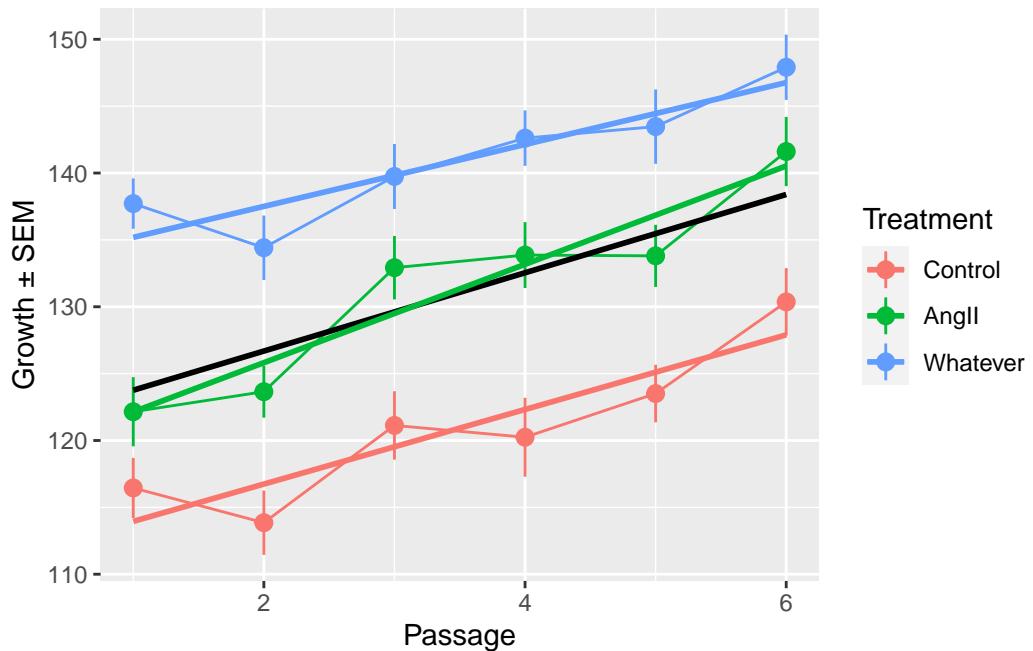
```

No summary function supplied, defaulting to `mean_se()`



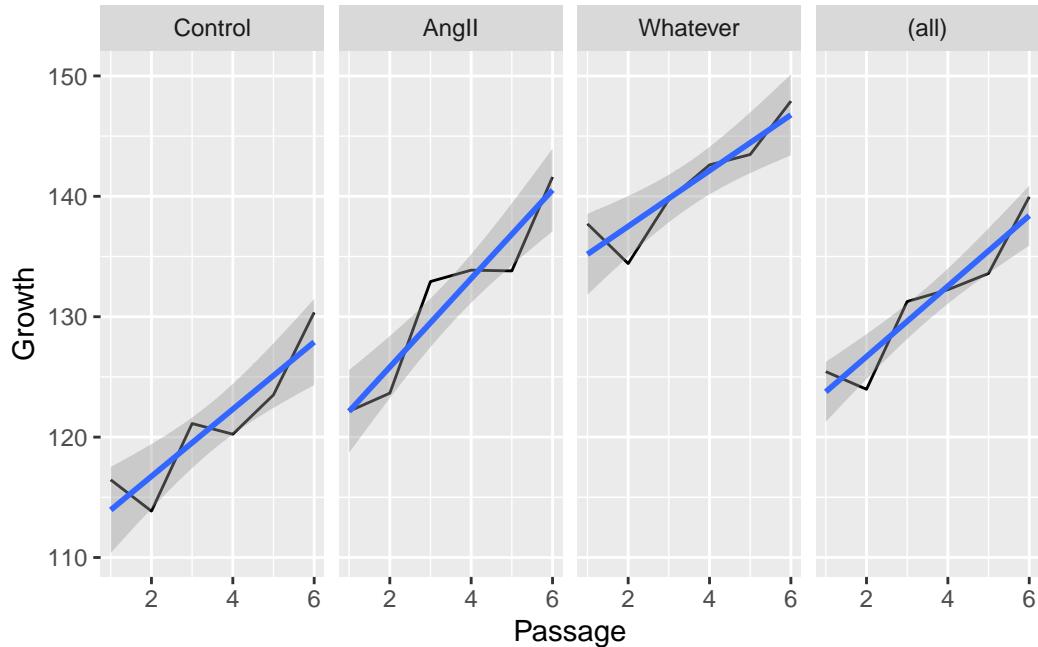
```
p1+geom_smooth(method='lm',color='black',se=F)+  
  geom_smooth(method='lm',aes(color=Treatment),se=F)
```

No summary function supplied, defaulting to `mean_se()`
`geom_smooth()` using formula = 'y ~ x'
`geom_smooth()` using formula = 'y ~ x'



```
ggplot(rawdata,aes(x=Passage,y=Growth))+  
  stat_summary(geom='line',fun='mean')+  
  geom_smooth(method='lm')+  
  facet_grid(cols = vars(Treatment), margins=T)
```

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



9.4 Linear Models

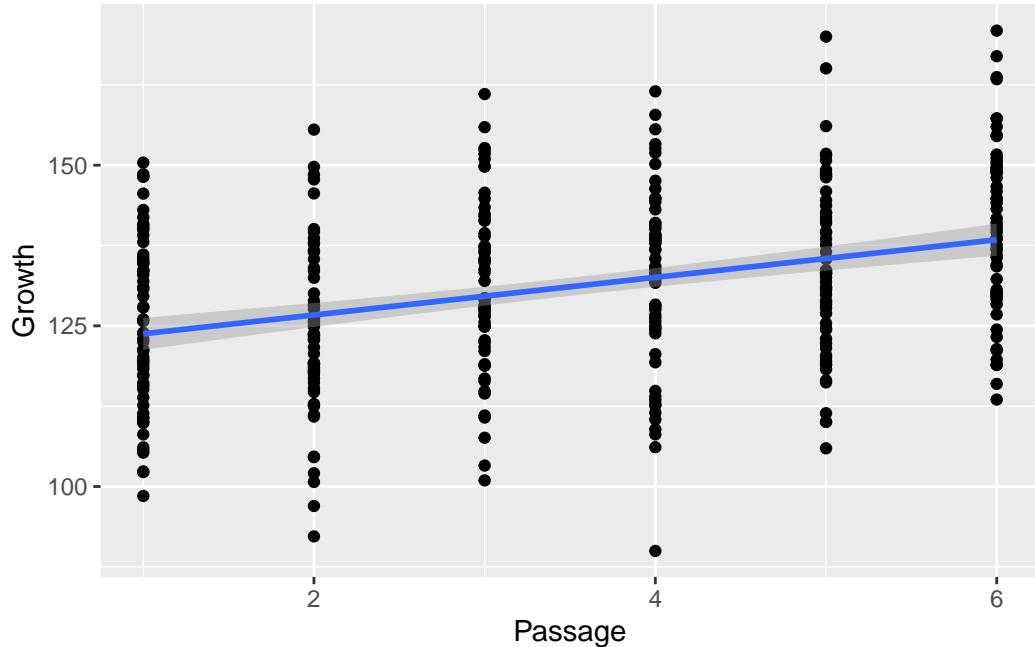
9.4.1 Linear regression

We will analyse the relation between independent variable (IV) Passage and dependent variable (DV) Growth.

9.4.1.1 Graphical exploration

```
ggplot(rawdata,aes(Passage,Growth))+
  geom_point()+
  geom_smooth(method=lm)

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



```

ggplot(rawdata,aes(Passage,Growth))+
  geom_point()+
  scale_x_continuous(breaks=seq(0,10,1))+  

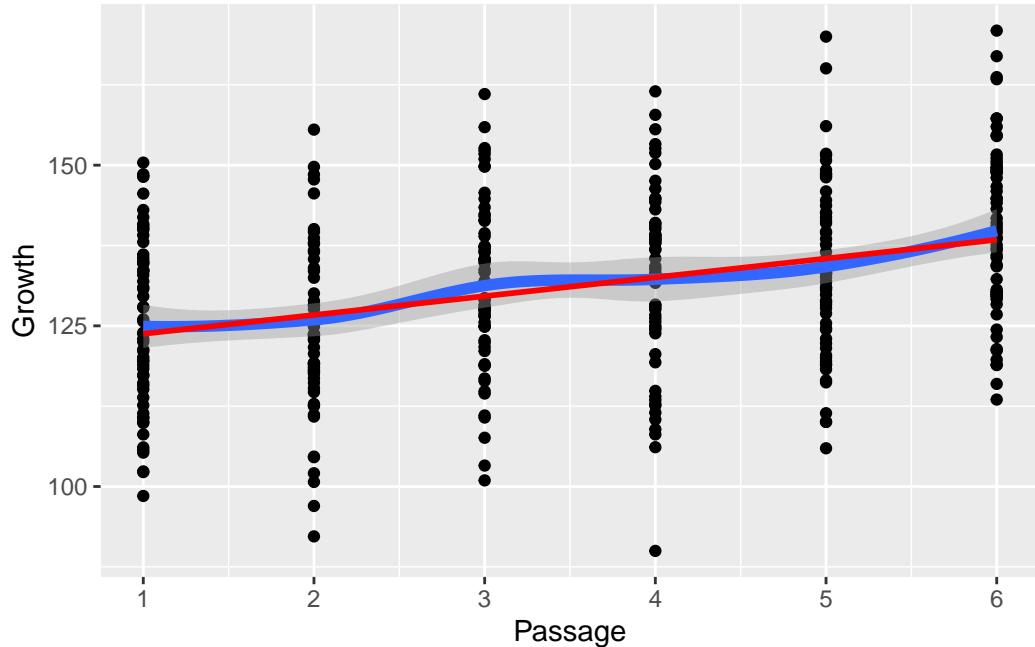
  geom_smooth(linewidth=2)+  

  geom_smooth(method=lm,se=F,color='red')

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

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

```



9.4.1.2 Modelling

This takes 2 steps, building the model and computing p-values.

```
# model
(regressionOut<-lm(Growth~Passage,data=rawdata))
```

```
Call:
lm(formula = Growth ~ Passage, data = rawdata)
```

```
Coefficients:
(Intercept)      Passage
 120.834        2.927
```

```
# model and p.value for slope, not recommended
tidy(regressionOut)
```

```
# A tibble: 2 x 5
```

```

term      estimate std.error statistic p.value
<chr>      <dbl>     <dbl>      <dbl>     <dbl>
1 (Intercept) 121.       1.63      74.2  1.77e-219
2 Passage      2.93      0.418      7.00  1.26e- 11

```

```

# computation of SSQs and p-values, use this!
(anova_out<-anova(regressionOut))

```

Analysis of Variance Table

Response: Growth

	Df	Sum Sq	Mean Sq	F value	Pr(>F)						
Passage	1	8996	8996.2	49.022	1.257e-11 ***						
Residuals	358	65698	183.5								

Signif. codes:	0	'***'	0.001	'**'	0.01	'*'	0.05	'. '	0.1	' '	1

```

anova_out$`Pr(>F)` #|> na.omit()

```

```

[1] 1.257266e-11          NA

```

```

tidy(anova_out)

```

```

# A tibble: 2 x 6
  term      df    sumsq   meansq statistic p.value
  <chr>    <int>  <dbl>    <dbl>      <dbl>     <dbl>
1 Passage     1  8996.  8996.      49.0  1.26e-11
2 Residuals  358 65698. 184.        NA     NA

```

```

# summary(regressionOut)
# str(regressionOut)

```

9.4.1.3 Adjusting

To take out the variance due to Passage effects, we can use the residuals and shift them to the original mean:

```

rawdata <-
  mutate(rawdata,
    growthAdj = regressionOut$residuals+mean(Growth))

summarise(rawdata,
  across(contains('growth'),
  ~meansd(.x,roundDig =4)))

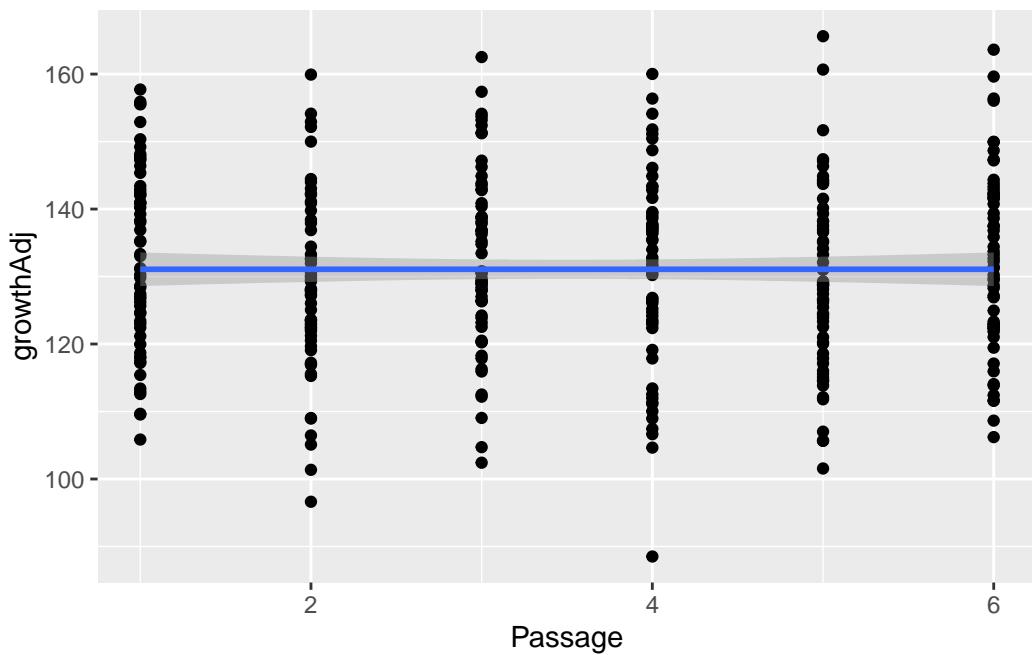
# A tibble: 1 x 2
#>   Growth      growthAdj
#>   <chr>       <chr>
#> 1 131.1 ± 14.4 131.1 ± 13.5

ggplot(rawdata,aes(Passage,growthAdj))+  

  geom_point()+
  geom_smooth(method = 'lm')

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

```



```

lm(growthAdj~Passage, data=rawdata) |> tidy()

# A tibble: 2 x 5
  term      estimate std.error statistic p.value
  <chr>      <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept) 1.31e+ 2     1.63     8.05e+ 1 2.04e-231
2 Passage      6.80e-15    0.418    1.63e-14 1.00e+ 0

```

9.4.2 ANOVA

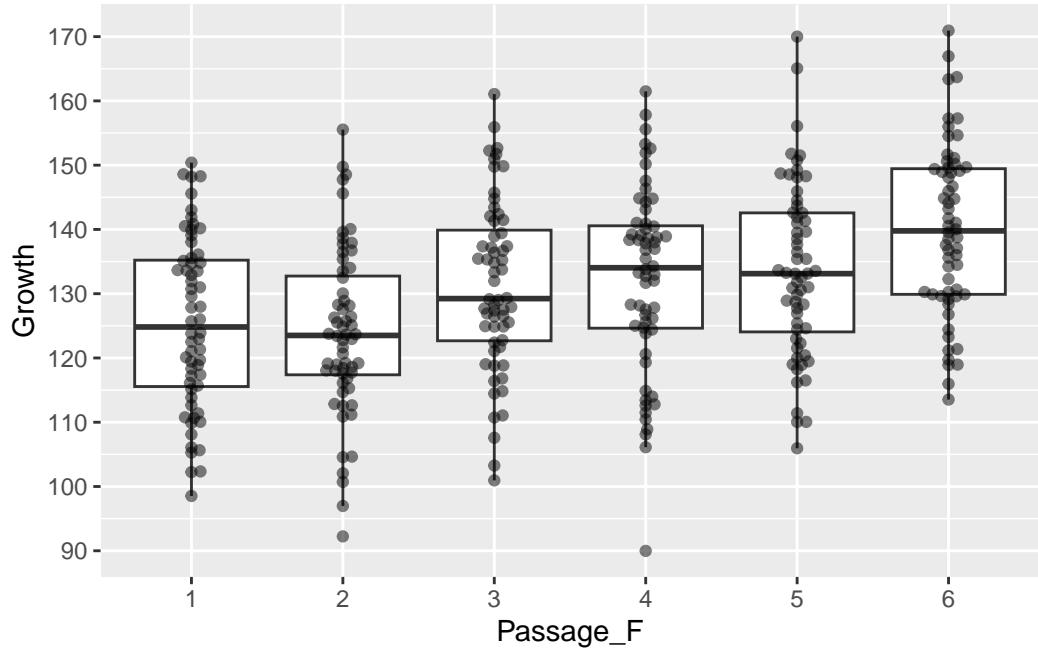
In the linear regression, we had Passage as a continuous IV, estimating a global ‘universal’ effect supposed to be constant. Now we look at Passage_F and model a discrete IV, allowing for specific effects, and thereby comparing means between groups.

9.4.2.1 Graphical exploration

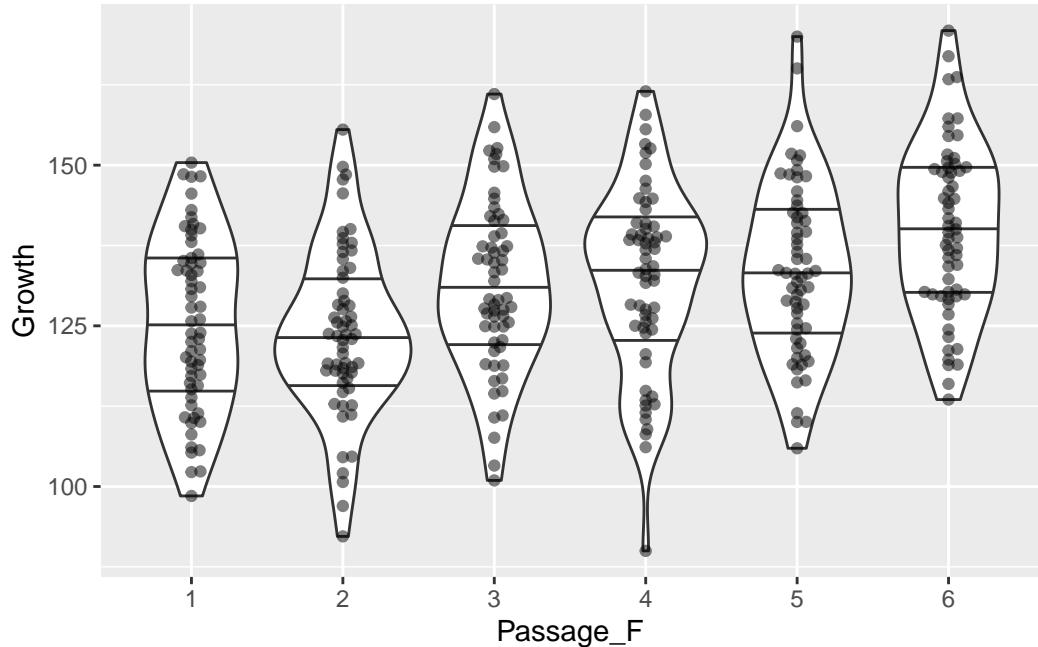
```

ggplot(rawdata, aes(x = Passage_F, y = Growth)) +
  geom_boxplot(outlier.alpha = 0) +
  geom_beeswarm(alpha=.5) +
  scale_y_continuous(breaks=seq(0,1000,10))

```



```
ggplot(rawdata,aes(x = Passage_F, y = Growth))+  
  geom_violin(draw_quantiles = c(.25,.5,.75))+  
  geom_beeswarm(alpha=.5)
```



9.4.2.2 Modelling

```
(AnovaOut<-lm(Growth~Passage_F, data=rawdata))
```

```
Call:
lm(formula = Growth ~ Passage_F, data = rawdata)

Coefficients:
(Intercept)  Passage_F2  Passage_F3  Passage_F4  Passage_F5  Passage_F6
125.440      -1.467       5.824      6.801      8.156     14.520
```

```
tidy(AnovaOut)
```

```
# A tibble: 6 x 5
  term      estimate std.error statistic p.value
  <chr>      <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept) 125.        1.74      71.9   2.20e-213
2 Passage_F2   -1.47      2.47     -0.595  5.52e- 1
```

```

3 Passage_F3      5.82      2.47      2.36  1.87e- 2
4 Passage_F4      6.80      2.47      2.76  6.11e- 3
5 Passage_F5      8.16      2.47      3.31  1.04e- 3
6 Passage_F6     14.5       2.47      5.89  9.03e- 9

```

```

# summary(AnovaOut)
(t <- anova(AnovaOut))

```

Analysis of Variance Table

Response: Growth

	Df	Sum Sq	Mean Sq	F value	Pr(>F)						
Passage_F	5	10134	2026.71	11.113	5.852e-10 ***						
Residuals	354	64561	182.38								

Signif. codes:	0	'***'	0.001	'**'	0.01	'*'	0.05	'. '	0.1	' '	1

```
t$`Pr(>F)`
```

```
[1] 5.851856e-10           NA
```

```
tidy(t)
```

```

# A tibble: 2 x 6
  term        df    sumsq   meansq statistic   p.value
  <chr>     <int>  <dbl>    <dbl>    <dbl>    <dbl>
1 Passage_F      5 10134.   2027.     11.1  5.85e-10
2 Residuals    354 64561.   182.      NA     NA

```

9.4.2.3 Post-hoc analyses

The p-value from our model only tests the global Null hypothesis of no differences between any group (all means are the same / all groups come from the same population). Post-hoc tests are used to figure out which groups are different. Those tests need to take multiple testing into account. Try to limit selection of tests!

```
# possible in a loop, but nominal p
t.test(rawdata$Growth[which(rawdata$Passage==1)],
       rawdata$Growth[which(rawdata$Passage==2)],
       var.equal = T)
```

Two Sample t-test

```
data: rawdata$Growth[which(rawdata$Passage == 1)] and rawdata$Growth[which(rawdata$Passage ==
t = 0.60679, df = 118, p-value = 0.5452
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
-3.321297 6.255936
sample estimates:
mean of x mean of y
125.4396 123.9723
```

```
# all pairwise group combinations
pt_out<-pairwise.t.test(x=rawdata$Growth,
                         g=rawdata$Passage_F,
                         p.adjust.method='none')
pt_out
```

Pairwise comparisons using t tests with pooled SD

```
data: rawdata$Growth and rawdata$Passage_F
```

	1	2	3	4	5
2	0.55215	-	-	-	-
3	0.01871	0.00331	-	-	-
4	0.00611	0.00088	0.69214	-	-
5	0.00104	0.00011	0.34487	0.58296	-
6	9e-09	3e-10	0.00048	0.00189	0.01025

P value adjustment method: none

```
pairwise.t.test(x=rawdata$Growth,g=rawdata$Passage,
                 p.adjust.method='fdr')
```

```
Pairwise comparisons using t tests with pooled SD
```

```
data: rawdata$Growth and rawdata$Passage
```

	1	2	3	4	5
2	0.62460	-	-	-	-
3	0.02552	0.00621	-	-	-
4	0.01018	0.00259	0.69214	-	-
5	0.00259	0.00057	0.43109	0.62460	-
6	6.8e-08	4.5e-09	0.00178	0.00405	0.01538

```
P value adjustment method: fdr
```

```
pairwise.t.test(x=rawdata$Growth,g=rawdata$Passage,  
p.adjust.method='bonferroni')
```

```
Pairwise comparisons using t tests with pooled SD
```

```
data: rawdata$Growth and rawdata$Passage
```

	1	2	3	4	5
2	1.0000	-	-	-	-
3	0.2807	0.0497	-	-	-
4	0.0917	0.0133	1.0000	-	-
5	0.0155	0.0017	1.0000	1.0000	-
6	1.4e-07	4.5e-09	0.0071	0.0283	0.1538

```
P value adjustment method: bonferroni
```

```
# comparison against reference group 1  
pt_out$p.value[,1]
```

	2	3	4	5	6
	5.521460e-01	1.871115e-02	6.110172e-03	1.036173e-03	9.031123e-09

```

# comparison against reference group 6
pt_out$p.value[5,]

      1          2          3          4          5
9.031123e-09 3.001066e-10 4.757018e-04 1.889098e-03 1.025037e-02

# comparison for selection
c(pt_out$p.value[1,1],pt_out$p.value[3,2],
  pt_out$p.value[5,1])

[1] 5.521460e-01 8.842382e-04 9.031123e-09

# comparison against next level
diag(pt_out$p.value)

[1] 0.55214600 0.00331248 0.69214393 0.58295615 0.01025037

# adjusting for multiple testing for selected comparisons
p.adjust(diag(pt_out$p.value),method='fdr')

[1] 0.69214393 0.01656240 0.69214393 0.69214393 0.02562592

formatP(p.adjust(pt_out$p.value[,1],method='fdr'))

[1] "0.552" "0.023" "0.010" "0.003" "0.001"

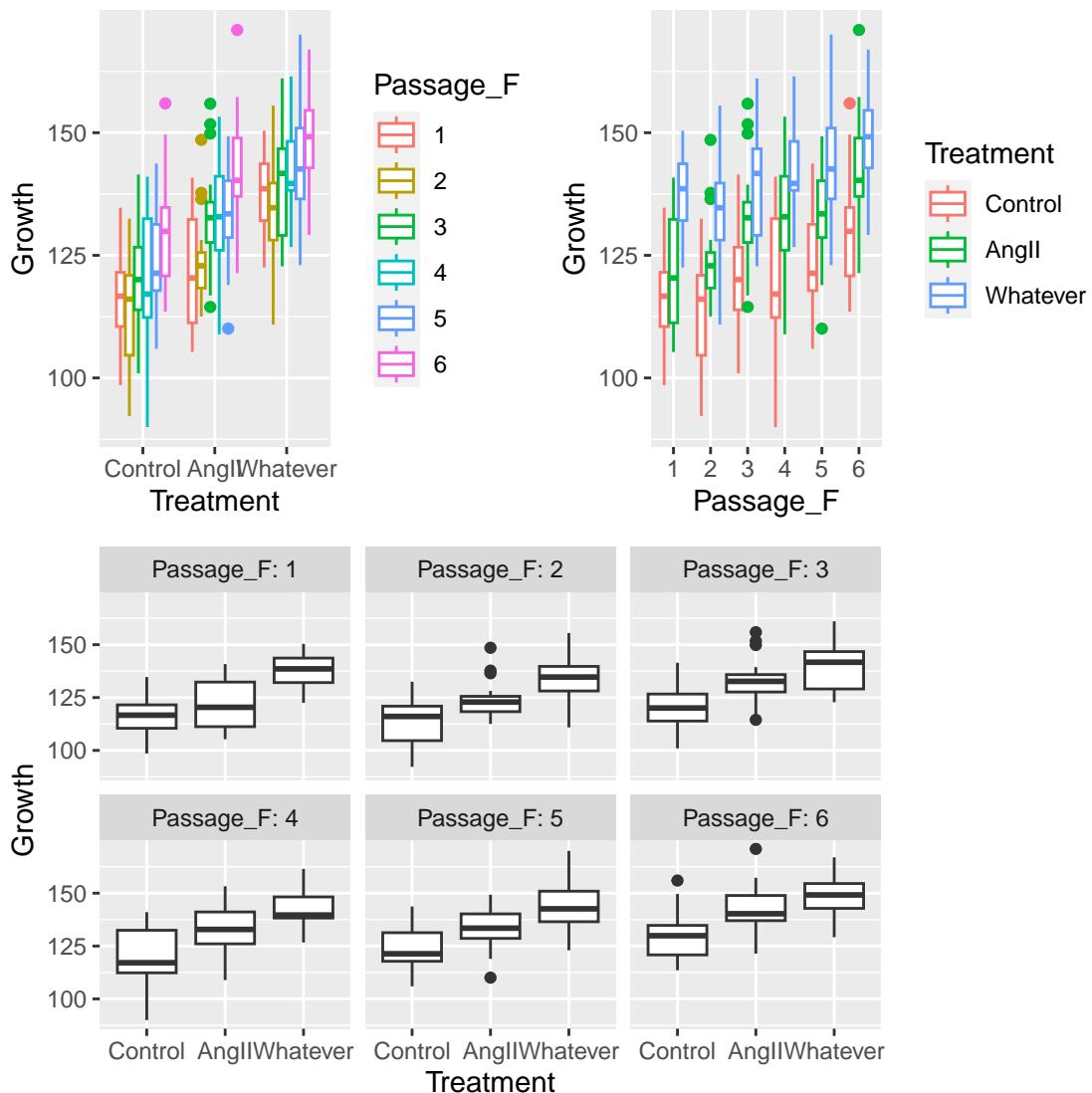
```

9.4.3 LM with continuous AND categorical IV

Traditionally you may think of *regression OR ANOVA*, but they are no different and can be combined. This is called a general linear model. Multivariable models may contain interactions between independent variables V IV1*IV2.

9.4.3.1 Graphical exploration

```
p0 <- ggplot(rawdata,aes(Treatment,Growth))+  
  geom_boxplot()  
p1 <- ggplot(rawdata,aes(Treatment,Growth, color=Passage_F))+  
  geom_boxplot()  
p2 <- ggplot(rawdata,aes(color=Treatment,Growth, x=Passage_F))+  
  geom_boxplot()  
p3 <- ggplot(rawdata,aes(Treatment,Growth))+  
  geom_boxplot()  
  facet_wrap(facets = vars(Passage_F), labeller='label_both')  
# from patchwork  
(p1+p2)/p3
```



9.4.3.2 Modelling

Models with (*) and without (+) interaction are build and tested.

```
lmOut_interaction<-lm(Growth~Passage*Treatment,data=rawdata)
Anova(lmOut_interaction,type = 3)
```

Anova Table (Type III tests)

```

Response: Growth
            Sum Sq Df F value    Pr(>F)
(Intercept) 285160  1 2448.5613 < 2.2e-16 ***
Passage      2723   1  23.3855 1.981e-06 ***
Treatment    5635   2  24.1924 1.419e-10 ***
Passage:Treatment 335   2   1.4376   0.2389
Residuals    41227 354
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```

#
lmOut_additive<-lm(Growth~Passage+Treatment,data=rawdata)
Anova(lmOut_additive,type=2)

```

Anova Table (Type II tests)

```

Response: Growth
            Sum Sq Df F value    Pr(>F)
Passage      8996   1 77.058 < 2.2e-16 ***
Treatment    24137   2 103.372 < 2.2e-16 ***
Residuals    41562 356
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

```

# for comparison, here is the univariable model
lmOut_uni<-lm(Growth~Treatment,data=rawdata)
aOut<-Anova(lmOut_uni,type=3)
a_uni <- anova(lmOut_uni)
a_uni$`Pr(>F)`
```

```
[1] 5.549803e-31          NA
```

9.4.3.3 Post-hoc analyses

For multivariable models, pairwise.t.test() is not appropriate, Dunnet or Tukey tests (depending on hypothesis) are typical solutions.

```
summary(glht(model=lmOut_additive, linfct=mcp(Treatment='Dunnett')))
```

Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Dunnett Contrasts

Fit: lm(formula = Growth ~ Passage + Treatment, data = rawdata)

Linear Hypotheses:

	Estimate	Std. Error	t value	Pr(> t)
AngII - Control == 0	10.409	1.395	7.462	<1e-10 ***
Whatever - Control == 0	20.052	1.395	14.375	<1e-10 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)

```
summary(glht(model=lmOut_additive, linfct=mcp(Treatment='Tukey')))
```

Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts

Fit: lm(formula = Growth ~ Passage + Treatment, data = rawdata)

Linear Hypotheses:

	Estimate	Std. Error	t value	Pr(> t)
AngII - Control == 0	10.409	1.395	7.462	<1e-10 ***
Whatever - Control == 0	20.052	1.395	14.375	<1e-10 ***
Whatever - AngII == 0	9.643	1.395	6.913	<1e-10 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)

```
DescTools::DunnettTest(Growth~Passage_F, data=rawdata)
```

```
Dunnett's test for comparing several treatments with a control :  
95% family-wise confidence level
```

```
$`1`  
    diff      lwr.ci     upr.ci     pval  
2-1 -1.467320 -7.6899251  4.755285  0.9648  
3-1  5.824059 -0.3985468 12.046664  0.0752 .  
4-1  6.801105  0.5784996 13.023710  0.0263 *  
5-1  8.156143  1.9335375 14.378748  0.0048 **  
6-1 14.520106  8.2975011 20.742712 6.9e-08 ***  
  
---  
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
pairwise.t.test(rawdata$Growth, rawdata$Treatment, p.adjust.method = 'n')
```

```
Pairwise comparisons using t tests with pooled SD
```

```
data: rawdata$Growth and rawdata$Treatment
```

```
          Control  AngII  
AngII      5.1e-11 -  
Whatever < 2e-16 1.0e-09
```

```
P value adjustment method: none
```

```
mean(rawdata$Growth[which(rawdata$Passage==1 &  
                           rawdata$Treatment=='Control')])
```

```
[1] 116.4531
```

```
a0ut$'Pr(>F)'
```

```
[1] 2.909117e-279  5.549803e-31           NA
```

```

aOut$`Sum Sq`  

[1] 1754742.53   24136.66   50557.95  

summary(lmOut_additive)  

Call:  

lm(formula = Growth ~ Passage + Treatment, data = rawdata)  

Residuals:  

    Min      1Q  Median      3Q     Max  

-32.407 -7.793 -0.281  7.255 32.283  

Coefficients:  

            Estimate Std. Error t value Pr(>|t|)  

(Intercept) 110.6802    1.5280  72.432 < 2e-16 ***  

Passage      2.9271    0.3334   8.778 < 2e-16 ***  

TreatmentAngII 10.4089    1.3949   7.462 6.59e-13 ***  

TreatmentWhatever 20.0520    1.3949  14.375 < 2e-16 ***  

---  

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  

Residual standard error: 10.8 on 356 degrees of freedom  

Multiple R-squared:  0.4436,    Adjusted R-squared:  0.4389  

F-statistic:  94.6 on 3 and 356 DF,  p-value: < 2.2e-16  

(result<-tibble(predictor=rownames(aOut),  

                 p=formatP(aOut$'Pr(>F)',ndigits=5)))
```

```
# A tibble: 3 x 2  

  predictor   p  

  <chr>       <chr>  

1 (Intercept) "0.00001"  

2 Treatment    "0.00001"  

3 Residuals    "      NA"
```

```
broom:::tidy(aOut)

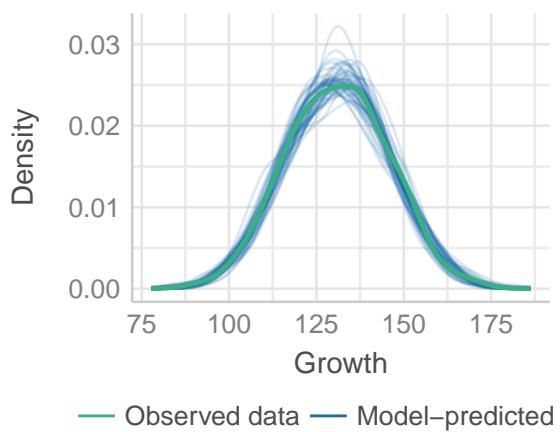
# A tibble: 3 x 5
  term        sumsq    df statistic   p.value
  <chr>      <dbl> <dbl>     <dbl>     <dbl>
1 (Intercept) 1754743.     1    12391.  2.91e-279
2 Treatment    24137.      2      85.2  5.55e- 31
3 Residuals    50558.    357       NA      NA
```

9.4.4 Model exploration with package performance

```
# x11() #interactive only!  
  
# from package performance  
check_model(lmOut_additive)
```

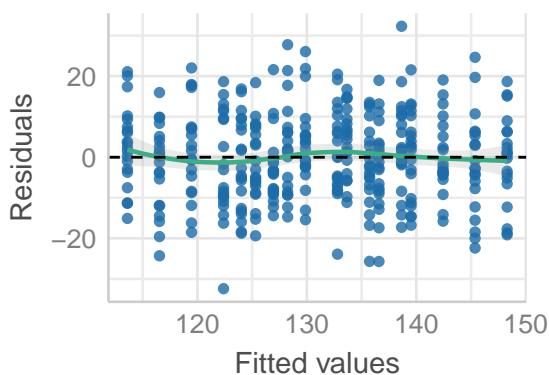
Posterior Predictive Check

Model-predicted lines should resemble observed data



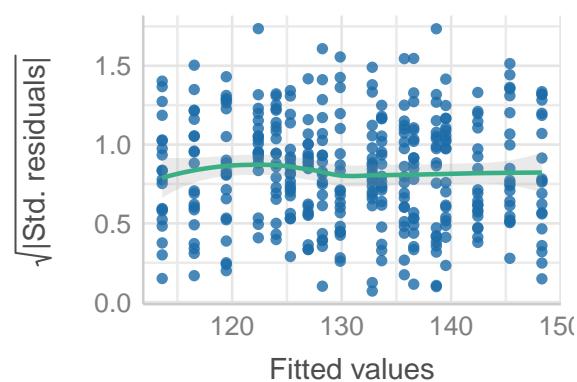
Linearity

Reference line should be flat and horizontal



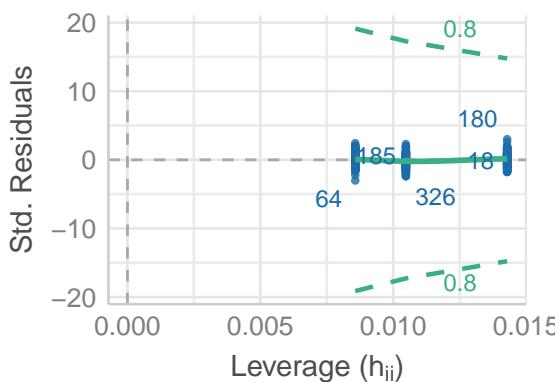
Homogeneity of Variance

Reference line should be flat and horizontal



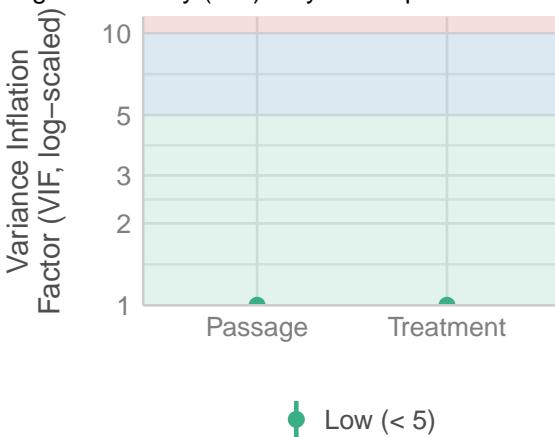
Influential Observations

Points should be inside the contour lines



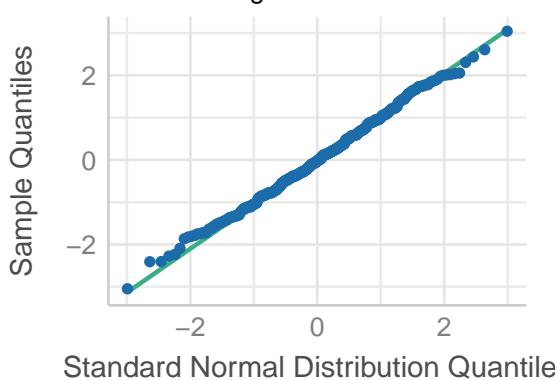
Collinearity

High collinearity (VIF) may inflate parameter uncertainty



Normality of Residuals

Dots should fall along the line



10 Summary

In summary, this book has no content whatsoever.

[1 + 1](#)

[1] 2

References