LECTURE 1: From t-test to Regression

Analysis of property assessment tax data

We first want to know if the age of the property affect its property tax assessment value.

Our null hypothesis is that the expected value of houses is the same among 3 age periods defined (contemporary, modern, and old).

Formally,

$$H_0 : \mu_C = \mu_M = \mu_O$$

These parameters are the expected value of 3 random variables that define the tax value of a house in each age period:

$$Y_C,\ Y_M,\ Y_O$$

To address this question, we take a random sample from each group:

$$Y_{i1},\ Y_{i2}, \ldots, Y_{in_i}$$

and analyze the data observed.

Data

```
library(dplyr)
library(ggplot2)
library(cowplot)
library(tidyverse)
library(broom)

theme_set(theme_bw())
```

```
dat<-read.csv("dataPropTaxAssess.csv")

# Add age of property
dat$age<-2017-as.numeric(dat$YEAR_BUILT)
dat$age_factor <- cut(dat$age,c(0,17,37,120),labels=c("C","M","O"))
dat <- dat %>% mutate(assessment_k = ASSESSMENT / 1000)
dat %>% select(age_factor) %>% summary
```

```
##  age_factor
##  C: 8785
##  M: 8570
##  O:10388
```

```
dat %>% group_by(age_factor) %>% summarize(avg_assessment=mean(ASSESSMENT))
```

```
## # A tibble: 3 x 2
##   age_factor avg_assessment
##       <fctr>          <dbl>
## 1          C       574978.9
## 2          M       515779.8
```

```
## 3          O       413212.9
```

```r
#Small dataframe, unbalanced, and a few properties per group for illustration

dat.small.C<-dat %>% filter(age_factor=="C")
dat.small.M<-dat %>% filter(age_factor=="M")
dat.small.O<-dat %>% filter(age_factor=="O")

set.seed(123)
dat.small<-rbind(dat.small.C[sample(1:nrow(dat.small.C),5),],
                 dat.small.M[sample(1:nrow(dat.small.M),15),],
                 dat.small.O[sample(1:nrow(dat.small.O),50),])


#Some summaries
dat.small %>% group_by(age_factor) %>% summarize(avg_assessment=mean(assessment_k))
```

```
## # A tibble: 3 x 2
##   age_factor avg_assessment
##       <fctr>          <dbl>
## 1          C         521.00
## 2          M         428.80
## 3          O         435.14
```

```r
dat %>% select(age_factor) %>% summary
```

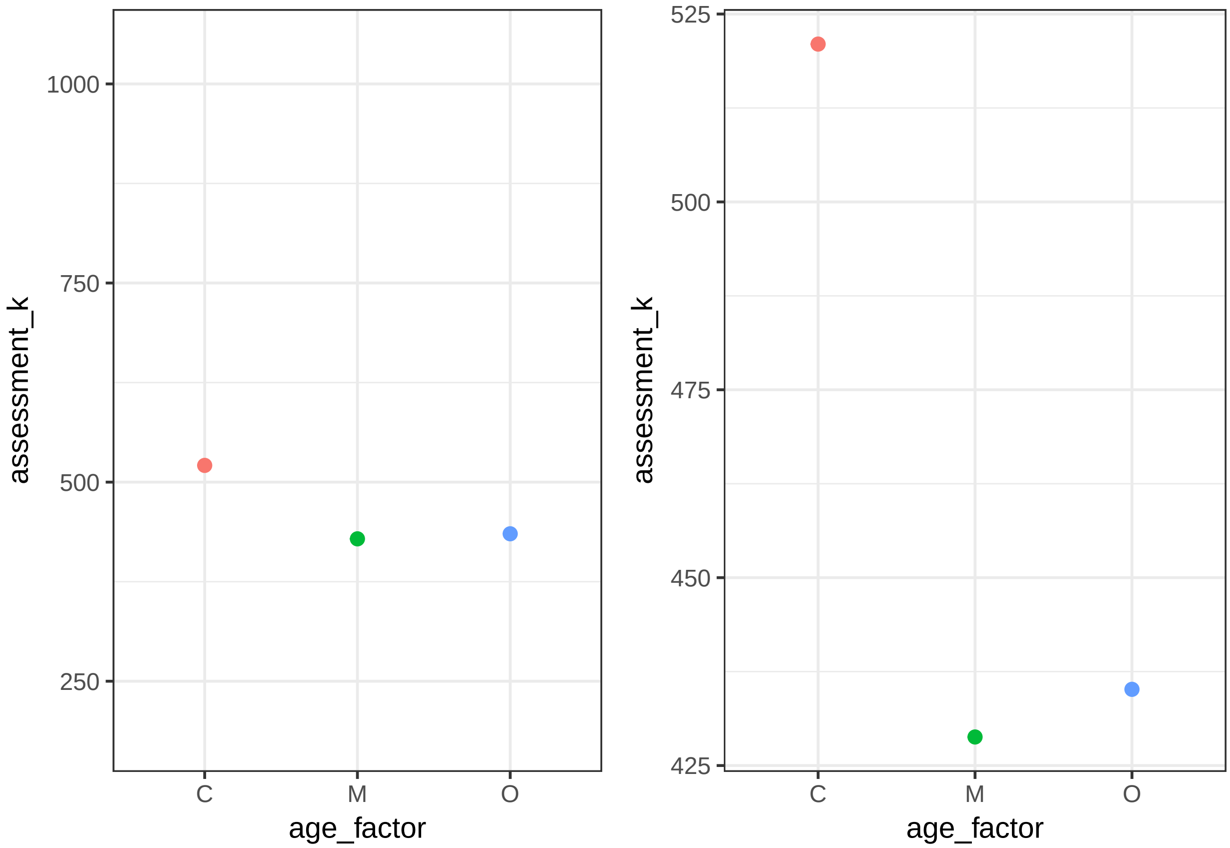
```
##  age_factor
##  C: 8785
##  M: 8570
##  O:10388
```

```r
dat %>% group_by(age_factor) %>% summarize(avg_assessment=mean(assessment_k))
```

```
## # A tibble: 3 x 2
##   age_factor avg_assessment
##       <fctr>          <dbl>
## 1          C       574.9789
## 2          M       515.7798
## 3          O       413.2129
```

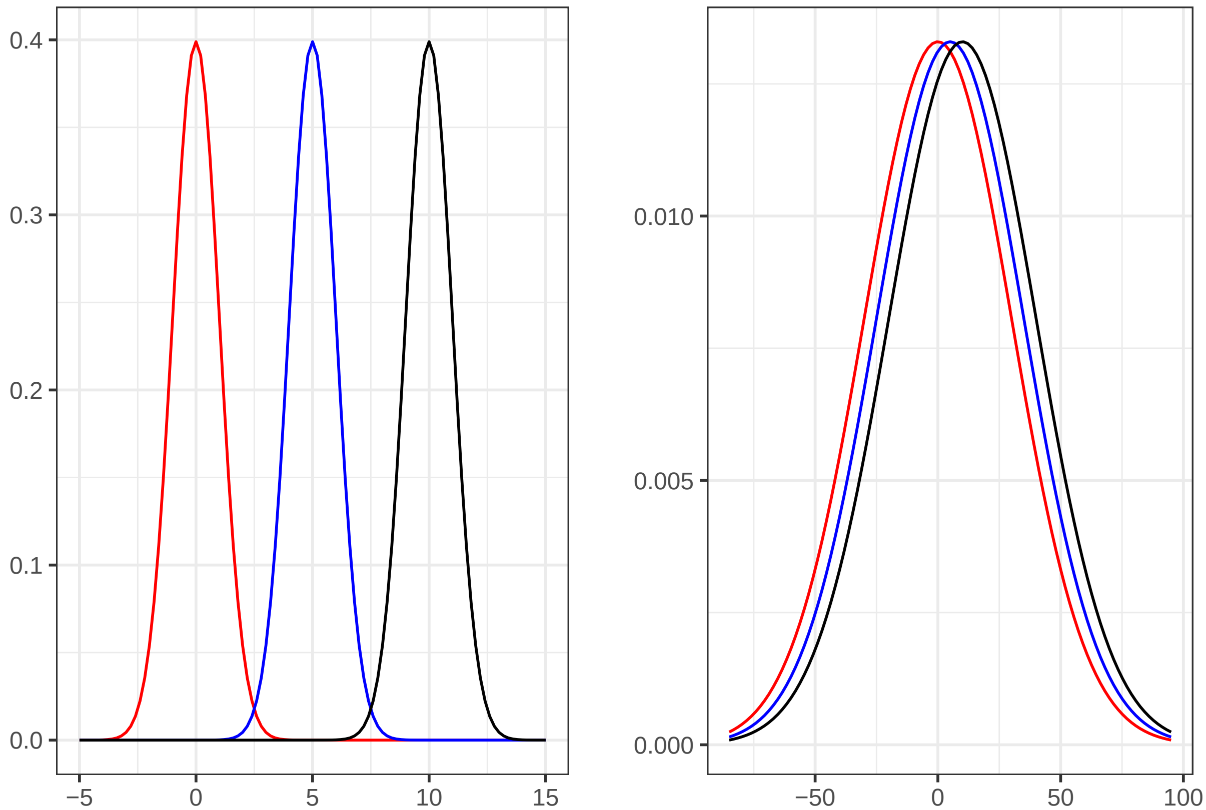
Visualizing summaries of the data

```r
#Plot of means (check the differences in the y-axes)
g1<-ggplot(data=dat.small,aes(age_factor, assessment_k, color = age_factor)) +
  stat_summary(fun.y = "mean", size = 2, geom = "point",aes(color = age_factor))+
  coord_cartesian(ylim = c(180, 1050))+
  theme(legend.position = "none")

g2<-ggplot(data=dat.small,aes(age_factor, assessment_k, color = age_factor)) +
  stat_summary(fun.y = "mean", size = 2, geom = "point",aes(color = age_factor))+
  #coord_cartesian(ylim = c(180, 1050))+
  theme(legend.position = "none")

plot_grid(g1, g2, align = "h")
```



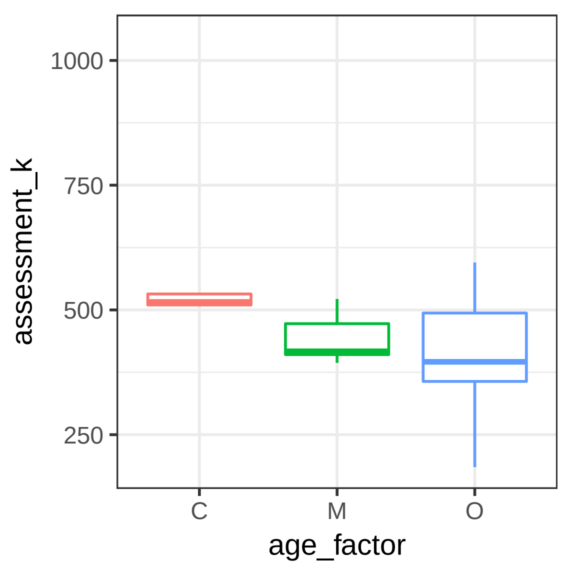
To test differences among population means it is not enough to look at the differences among the sample means. The following plots illustrate the importance of analyzing differences in sample means in relation to the variation observed.

```
#Comparison of densities
#a) small variance
g_sv<-ggplot(data.frame(x = c(-5, 15)), aes(x)) +
  stat_function(fun = dnorm,color="red")+
  stat_function(fun = dnorm,args = list(mean = 5),color="blue")+
  stat_function(fun = dnorm,args = list(mean = 10))+labs(x = "",y="")

#b) large variance
g_lv<-ggplot(data.frame(x = c(-85, 95)), aes(x)) +
  stat_function(fun = dnorm,args = list(mean = 0,sd=30),color="red")+
  stat_function(fun = dnorm,args = list(mean = 5,sd=30),color="blue")+
  stat_function(fun = dnorm,args = list(mean = 10,sd=30))+labs(x = "",y="")
plot_grid(g_sv, g_lv, align = "h")
```



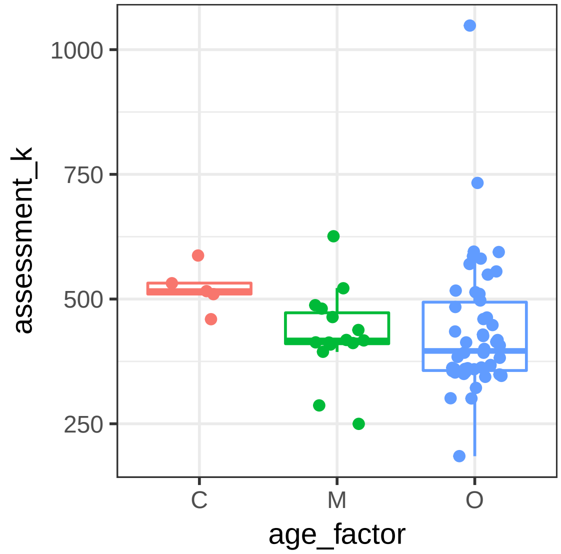
Thus, we further examine the variation observed in each age period.

```
#Boxplots

g<-ggplot(data=dat.small,aes(age_factor, assessment_k, color = age_factor)) +
  geom_boxplot(outlier.shape = NA)+ theme(legend.position = "none")

g
```

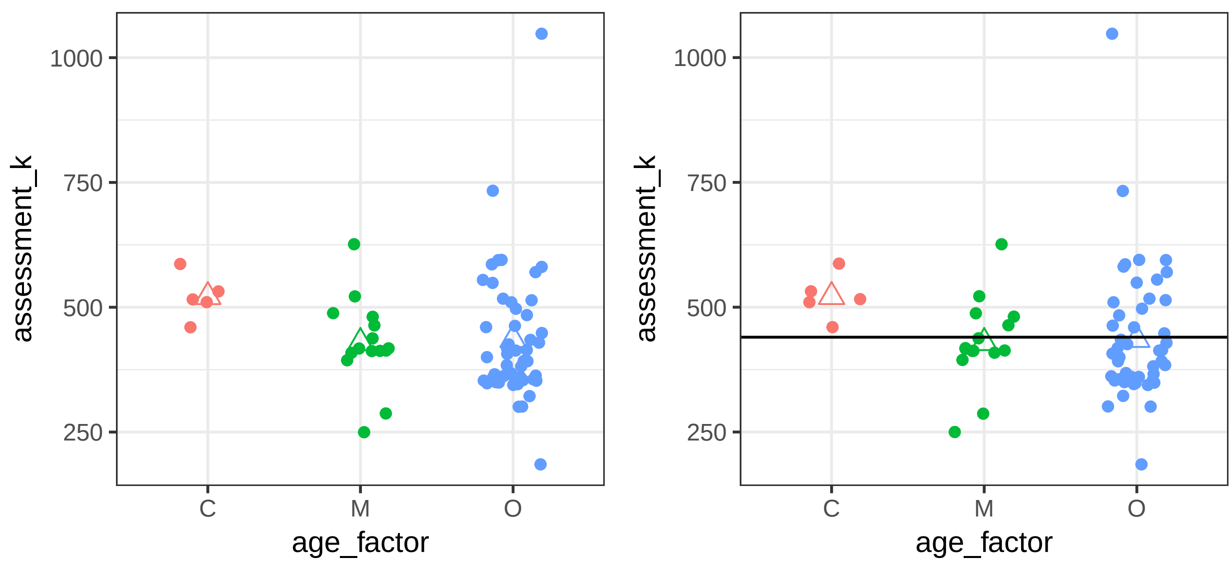


We note that this information is still not sufficient to test our hypothesis since we don't know how many data points were collected. Sample size is important!

```r
#Add data points
g+geom_jitter(position = position_jitter(width  = .2),aes(color=age_factor))+
    theme(legend.position = "none")
```




```r
#Data with mean
g1<-ggplot(data=dat.small,aes(age_factor, assessment_k, color = age_factor)) +
  geom_jitter(position = position_jitter(width  = .2),aes(color=age_factor))+
  stat_summary(fun.y = "mean", size = 3, shape=2,geom = "point",aes(color = age_factor))+
  theme(legend.position = "none")

#Add overall mean
g2<-g1+geom_hline(aes(yintercept = mean(dat.small$assessment_k)),)
plot_grid(g1, g2, align = "h")
```



t-test and ANOVA summary tables in R

Two-groups comparisons (of their population means) can be done with a 2-sample t-test or, equivalently, with an ANOVA. As mentioned in class the squared of the t-statistic equals the F-statistic in a 1-way ANOVA with 2 levels (i.e., $t^2 = F$).

```
#t-test vs ANOVA in a 2-groups analysis
#responses within each group
tax.M <-dat.small %>% subset(age_factor =="M", select=assessment_k)
tax.C <-dat.small %>% subset(age_factor =="C", select=assessment_k)

tt.2<-t.test(tax.M,tax.C,var.equal=T)
tt.2
```

```
##
##  Two Sample t-test
##
## data:  tax.M and tax.C
## t = -2.2034, df = 18, p-value = 0.04083
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -180.111457   -4.288543
## sample estimates:
## mean of x mean of y
##     428.8     521.0
```

```
#subset of 2 age periods
summary(aov(assessment_k~age_factor,data=subset(dat.small,age_factor %in% c("M","C"))))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## age_factor   1  31878   31878   4.855 0.0408 *
## Residuals   18 118188    6566
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

To compare more than 2 groups we can use ANOVA. A 1-way ANOVA can be performed using the `aov` or the `anova` functions.

```
#ANOVA table for a 3-groups analysis
summary(aov(assessment_k~age_factor,data=dat.small))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## age_factor   2  35867   17934   1.226    0.3
## Residuals   67 980328   14632
```

```
# NOTE that t^2=F
tt.2$statistic^2
```

```
##        t
## 4.855017
```

Try to re-compute the numbers in these tables! Do you know what each of them are?

Using the lm() function

The R function aov internally calls the lm function. These two functions differ in the way they output the results. Understanding which hypotesis are tested in these analyses is essential to perform a meaningful analysis of the data!

So, can we perform an ANOVA using lm? Let's start looking again at a 2-group comparison: 1 factor with 2 levels

```
#Linear regression

#LM with 2 age periods
summary(lm(assessment_k~age_factor,data=subset(dat.small,age_factor %in% c("M","C"))))
```

```
##
## Call:
## lm(formula = assessment_k ~ age_factor, data = subset(dat.small,
##     age_factor %in% c("M", "C")))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -178.80  -17.55  -10.90   39.45  197.20
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   521.00      36.24  14.377 2.62e-11 ***
## age_factorM   -92.20      41.84  -2.203   0.0408 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 81.03 on 18 degrees of freedom
## Multiple R-squared:  0.2124, Adjusted R-squared:  0.1687
## F-statistic: 4.855 on 1 and 18 DF,  p-value: 0.04083
```

From the previous output tables (t.test and aov), you may recognize many of the results in the output of lm. We have seen in class that when calling the function lm with one categorical explanatory variable, R creates a design matrix X to estimate the parameters in the model. The form of the X matrix depends on the parametrization chosen. By default, R uses the reference-treatment parametrization. The same is true for one categorical variable with more than 2 groups (see output below).

Note: it is not surprising that aov provides equivalent results to those given by the lm function since behind scences aov calls lm (by definition they will give the same results!). However, you can show that the summary of aov provides the sum of squares (SS) decomposition of an ANOVA table, thus showing that a 1-way ANOVA is a particular case of regression.

```r
#More than 2 groups
#ANOVA with 3 age periods
summary(aov(assessment_k~age_factor,data=dat.small))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## age_factor   2  35867   17934   1.226    0.3
## Residuals   67 980328   14632
```

```r
#LM with 3 age periods
summary(lm(assessment_k~age_factor,data=dat.small))
```

```
##
## Call:
## lm(formula = assessment_k ~ age_factor, data = dat.small)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -250.14  -74.89  -16.97   51.36  612.86
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   521.00      54.10   9.631 2.87e-14 ***
## age_factorM   -92.20      62.46  -1.476    0.145
## age_factorO   -85.86      56.74  -1.513    0.135
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 121 on 67 degrees of freedom
## Multiple R-squared:  0.0353, Adjusted R-squared:  0.006498
## F-statistic: 1.226 on 2 and 67 DF,  p-value: 0.3001
```

LECTURE 2: Review of matrix operations

```r
#Multiplication
A <- matrix(c(1, 2 , 1,4 , 0,5),3,2,byrow=T)
B <- matrix(c(-1, -2 , -2,1),2,2,byrow=T)

A%*%B
```

```
##      [,1] [,2]
## [1,]   -5    0
## [2,]   -9    2
## [3,]  -10    5
```

```r
#Note: A%*%B is not the same as A*B
```

```r
#Inverse
A <- matrix(c(7, 2 , 1,0 , 3 ,-1, -3, 4 , -2),3,3,byrow=T)
solve(A)
```

```
##      [,1] [,2] [,3]
## [1,]   -2    8   -5
## [2,]    3  -11    7
## [3,]    9  -34   21
```

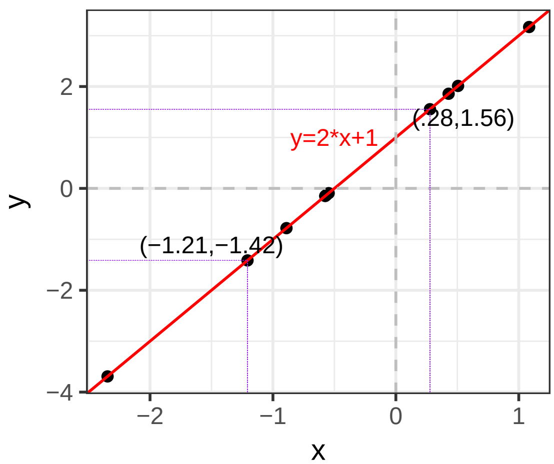
```r
#Projection
A <- matrix(c(1,2,1,4,0,5),3,2,byrow=T)
Px <- matrix(c(rep(0,3),1),2,2)
A%*%Px
```

```
##      [,1] [,2]
## [1,]    0    2
## [2,]    0    4
## [3,]    0    5
```

```r
Py <- matrix(c(1,rep(0,3)),2,2)
A%*%Py
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    1    0
## [3,]    0    0
```

```r
#A line
set.seed(1234)
x<-rnorm(10)
y<-2*x+1

example.line<-data.frame(x=x,y=y)
g<-ggplot(example.line, aes(x,y))+geom_point()+geom_abline(slope=2, intercept = 1, color="red")+
  geom_vline(xintercept = 0,color="grey",linetype = "dashed")+
  geom_hline(yintercept = 0,color="grey",linetype = "dashed")

g+geom_segment(x=example.line[1,1],xend=example.line[1,1],y=example.line[1,2],yend=-4,
               color="purple",linetype = "dashed",size=0.05)+
  geom_segment(x=example.line[2,1],xend=example.line[2,1],y=example.line[2,2],yend=-4,
               color="purple",linetype = "dashed",size=0.05)+
  geom_segment(x=-3,xend=example.line[1,1],y=example.line[1,2],yend=example.line[1,2],
               color="purple",linetype = "dashed",size=0.05)+
  geom_segment(x=-3,xend=example.line[2,1],y=example.line[2,2],yend=example.line[2,2],
               color="purple",linetype = "dashed",size=0.05)+
  annotate("text", x = 0.55, y = 1.4, size=3,label = "(.28,1.56)")+
  annotate("text", x = -1.5, y = -1.1,size=3, label = "(-1.21,-1.42)")+
  annotate("text", x = -.5, y = 1,size=3, label = "y=2*x+1",color="red")
```



LECTURE 3: Linear regression with categorical covariates

The general framework of linear regression allows the use of different type and number of explanatory (or independent) variables. We start analyzing some other special cases in detail.

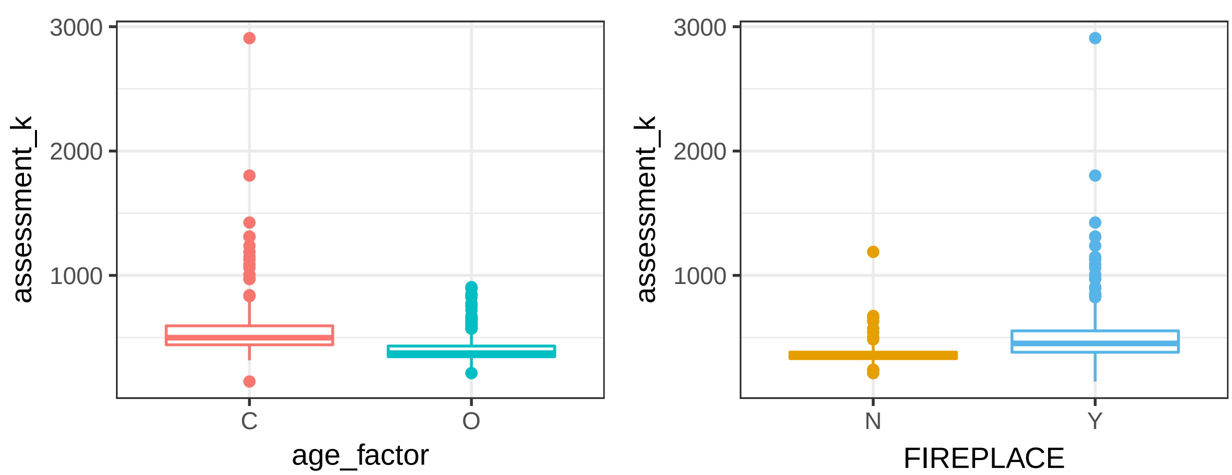
To avoid empty categories, let's extend our analysis to a random sample of size 500 (you can play with the size of the sample and see how results change!). For simplicity, let's first focus on the comparison of "C vs O" residencies and add one more categorical variable: 2 categorical variables, each with 2 levels

More than one categorical variable

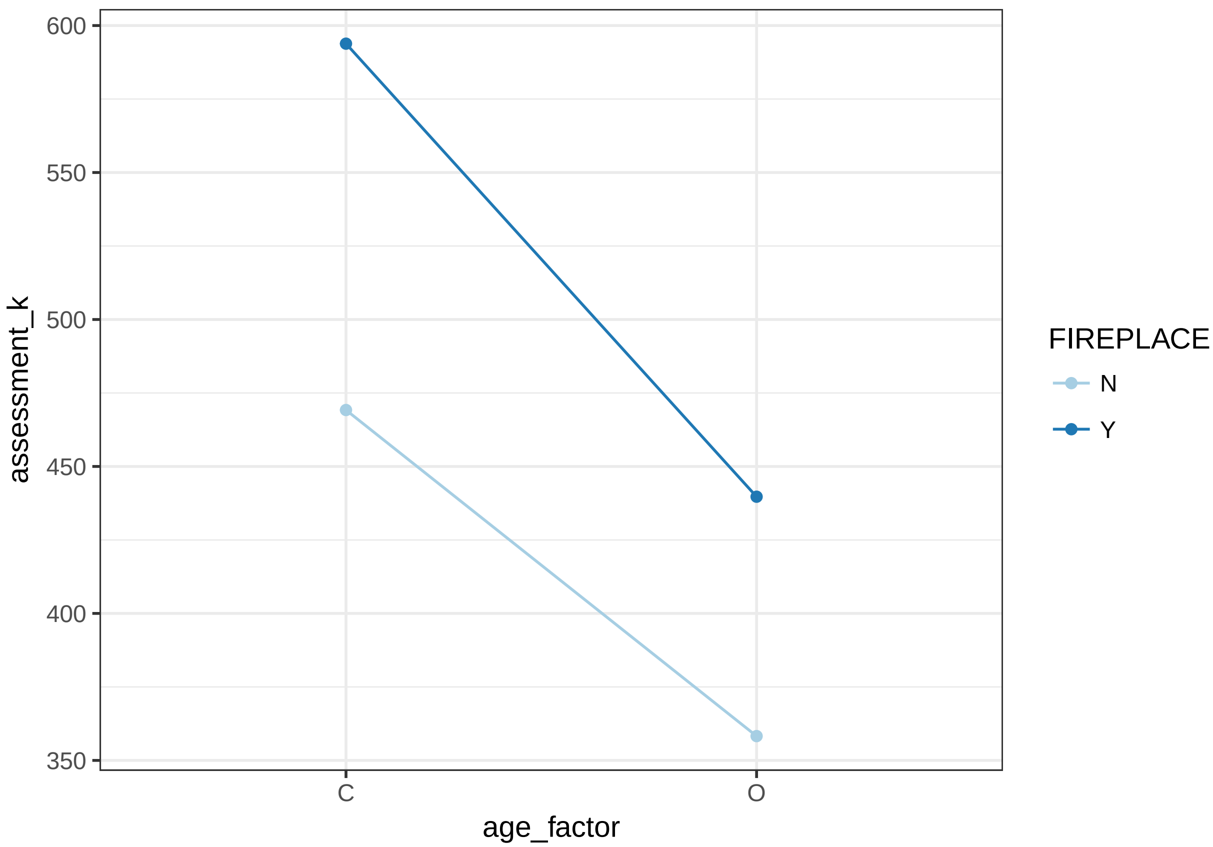
```r
#Use a random sample of 500 residencies in the dataset, 2 age periods
set.seed(12134)
dat.rs<-dat[sample(1:nrow(dat),500),]
dat.2<-dat.rs %>% filter(age_factor%in%c("C","O"))
dat.2 %>% count(age_factor,FIREPLACE)
```

```
## # A tibble: 4 x 3
##   age_factor FIREPLACE     n
##       <fctr>    <fctr> <int>
## 1          C         N    15
## 2          C         Y   136
## 3          O         N    72
## 4          O         Y   145
```

```r
#Include "FIREPLACE" (="Y" if the property has a fireplace) in the model
g_fire<-ggplot(data=dat.2,aes(FIREPLACE, assessment_k, color = FIREPLACE)) +
  geom_boxplot()+
  scale_color_manual(values=c("#E69F00", "#56B4E9"))+
  theme(legend.position = "none")

g_age<-ggplot(data=dat.2,aes(age_factor, assessment_k, color = age_factor)) +
  geom_boxplot()+
  theme(legend.position = "none")

plot_grid(g_age, g_fire, align = "h")
```



```r
# Interaction plot
g<-ggplot(data = dat.2,
       aes(age_factor, assessment_k, color = FIREPLACE, group=FIREPLACE)) +
  stat_summary(fun.y=mean, geom="point")+scale_colour_brewer(palette="Paired")+
  stat_summary(fun.y=mean, geom="line")

g
```



A 2-way ANOVA is used to study the effect of 2 factors (say A and B) on a quantitative response (Y). In a 2-way ANOVA, the total sum of squares (SST) can be decomposed to examine the variation explained by each factor and the interaction term: SST=SS(A) + SS(B) + SS(AB) + SSE. If there is an equal number of observations within each group (aka balanced or orthogonal design) the analysis is simpler, thus we start with that case.

Balanced designs

In the balanced design, the average of groups averages is equal to marginal averages:

$$\hat{\mu}_C = \frac{\hat{\mu}_{CY} + \hat{\mu}_{CN}}{2}$$

Thus, the SS(A) can be used to test the null hypothesis that the population means from the levels of one factor are equal, on average over the other factor:

$$H_0 : \mu_C = \mu_O$$

Let's see this results in R:

```r
#create a balanced dataset:
dat.CY<-dat %>% filter(age_factor=="C"&FIREPLACE=="Y")
dat.CN<-dat %>% filter(age_factor=="C"&FIREPLACE=="N")
dat.OY<-dat %>% filter(age_factor=="O"&FIREPLACE=="Y")
dat.ON<-dat %>% filter(age_factor=="O"&FIREPLACE=="N")

set.seed(123)
dat.balanced<-rbind(dat.CY[sample(1:nrow(dat.CY),100),],
                    dat.CN[sample(1:nrow(dat.CN),100),],
                    dat.OY[sample(1:nrow(dat.OY),100),],
                    dat.ON[sample(1:nrow(dat.ON),100),])

#Two-way ANOVA table
summary(aov(assessment_k~age_factor*FIREPLACE,data=dat.balanced))
```

```
##                       Df   Sum Sq Mean Sq F value   Pr(>F)
## age_factor             1   854608  854608  32.378 2.47e-08 ***
## FIREPLACE              1  1944212 1944212  73.660  < 2e-16 ***
## age_factor:FIREPLACE   1    87942   87942   3.332   0.0687 .
## Residuals            396 10452187   26394
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#order of factors does NOT matter in THIS case
summary(aov(assessment_k~FIREPLACE*age_factor,data=dat.balanced))
```

```
##                       Df   Sum Sq Mean Sq F value   Pr(>F)
## FIREPLACE              1  1944212 1944212  73.660  < 2e-16 ***
## age_factor             1   854608  854608  32.378 2.47e-08 ***
## FIREPLACE:age_factor   1    87942   87942   3.332   0.0687 .
## Residuals            396 10452187   26394
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Let's compare the 2-way and the 1-way ANOVA tables

```r
summary(aov(assessment_k~age_factor,data=dat.balanced))
```

```
##              Df   Sum Sq Mean Sq F value   Pr(>F)
## age_factor    1   854608  854608   27.25 2.89e-07 ***
## Residuals   398 12484340   31368
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#summary(aov(assessment_k~FIREPLACE,data=dat.balanced))
```

As I mentioned above, for the balanced design, the SS(A) are the same in 1-way and in 2-way ANOVA!! Factor B is ignored in SS(A)! Note that the residual sum of squares (SSE, Residuals in table) are not the same since the 2-way ANOVA explains more of the total variance.

The SS(A) are those defined in Lect01 (with just some additional indeces):

$$SS(A) = bM \sum_{i}^{a} (\bar{Y}_{i..} - \bar{Y}_{...})^2$$

, where a, b are the number of levels of the factors A and B, and M is the number of observation in each group (which are all equal in a balanced design).

```
#marginal means of A
means_A<-dat.balanced %>% group_by(age_factor)%>%
  summarize(avg_assessment=mean(assessment_k)) %>%
  select(avg_assessment)%>% unlist %>% as.numeric

#marginal means of B
means_B<-dat.balanced %>% group_by(FIREPLACE)%>%
  summarize(avg_assessment=mean(assessment_k)) %>%
  select(avg_assessment)%>% unlist %>% as.numeric

#overall mean
mean_all<-mean(dat.balanced$assessment_k)
mean_all
```

```
## [1] 457.4025
```

```
#means of marginal means equals the overall mean! this makes the magic!
mean(means_A)
```

```
## [1] 457.4025
```

```
mean(means_B)
```

```
## [1] 457.4025
```

```
#number of levels of each group
a<-2 #(2 age levels)
b<-2 #(FIREPLACE Y vs N)

#number of observations per group
M<-100

#SS(A): compare with ANOVA table
b*M*sum((means_A-mean_all)^2)
```

```
## [1] 854607.8
```

```
#SS(B): compare with ANOVA table
a*M*sum((means_B-mean_all)^2)
```

```
## [1] 1944212
```

And as expected, the same hypotheses can be tested using the `lm` function but this is not so obvious from the default output. By default, `lm` uses the reference-treatment parametrization and reports hypothesis tests at the reference levels, e.g., $H_0 : \mu_{CY} = \mu_{CN}$ (called "FIREPLACEY").

```
summary(lm(assessment_k~age_factor*FIREPLACE,data=dat.balanced))
```

```
##
## Call:
## lm(formula = assessment_k ~ age_factor * FIREPLACE, data = dat.balanced)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -256.17  -76.33  -24.08   19.92 1589.93
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                419.08      16.25  25.795  < 2e-16 ***
## age_factorO                -62.79      22.98  -2.733  0.00656 **
## FIREPLACEY                 169.09      22.98   7.359 1.08e-12 ***
## age_factorO:FIREPLACEY     -59.31      32.49  -1.825  0.06870 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 162.5 on 396 degrees of freedom
## Multiple R-squared:  0.2164, Adjusted R-squared:  0.2105
## F-statistic: 36.46 on 3 and 396 DF,  p-value: < 2.2e-16
```

```
#order doesn't matter (uncomment the second line to check)
#summary(lm(assessment_k~FIREPLACE*age_factor,data=dat.balanced))
```

In other words, the main effect tests do not appearing in the summary table. However, this does not mean that we can not test the main effects! Linear combinations of the model parameters are called *contrasts*. The package `lsmeans` is very useful to set ant test contrasts of interest!! It also gives neat summaries of the data. (Note: apparently this package is being deprecated and users are being encouraged to switch to 'emmeans'. I leave that as a HW for you).

```
library(lsmeans)
lm_AB<-lm(assessment_k~age_factor*FIREPLACE,data=dat.balanced)

means.ab <- lsmeans(lm_AB, specs = c("age_factor", "FIREPLACE"))
means.a <- lsmeans(lm_AB, specs = c("age_factor"))
means.b <- lsmeans(lm_AB, specs = c("FIREPLACE"))
```

Let's estimate 2 contrasts of interest to understand better the `lm` and the `aov` outputs. Using `lsmeans` contrasts can be set and tested easily!! For example, let's tests the null hypothesis $H_0 : \mu_{CN} = \mu_{ON}$. To create a contrast to set this hypothesis, you need to create a vector with entries summing to 0 which gives you the comparison of interest when multiplied by the vector of parameters $(\mu_{CN}, \mu_{ON}, \mu_{CY}, \mu_{OY})$ (the order needs to match that of the model, see `means.ab`):

$$c(1, -1, 0, 0) * \begin{bmatrix} \mu_{CN} \\ \mu_{ON} \\ \mu_{CY} \\ \mu_{OY} \end{bmatrix} = \mu_{CN} - \mu_{ON}$$

Similarly, we can test that the levels of A, on average over the levels of B! That is: $H_0 : \frac{\mu_{CN}+\mu_{CY}}{2} = \frac{\mu_{ON}+\mu_{OY}}{2}$

$$c(.5,-.5,.5,-.5)*\begin{bmatrix}\mu_{CN}\\ \mu_{ON}\\ \mu_{CY}\\ \mu_{OY}\end{bmatrix}=\frac{\mu_{CN}+\mu_{CY}}{2}-\frac{\mu_{ON}+\mu_{OY}}{2}$$

```
contrast(means.ab, list(CvOforN=c(1,-1,0,0),CvO = c(.5,-.5,.5,-.5)))
```

```
##  contrast estimate       SE  df t.ratio p.value
##  CvOforN    62.790 22.97582 396   2.733  0.0066
##  CvO        92.445 16.24636 396   5.690  <.0001
```

Note that the first contrast is the one given by `lm` in its summary table!! and the second contrast is the one given by `aov`. Three important messages out of this analysis:

- it is important to understand that these contrasts are different!! The first one compares the levels of A at a particular level of B. The second one compares the levels of A, on average over the levels of B!
- you can use lsmeans::contrast to test any other contrast you are intersted in, for example, test $H_O : \mu_{CY} = \mu OY$.

So far so good.... however, things get really complicated in *unbalanced* designs!

Unbalanced designs

In unbalanced design, the average of groups averages is *no longer* equal to the marginal average:

$$\hat{\mu}_C \neq \frac{\hat{\mu}_{CY}+\hat{\mu}_{CN}}{2}$$

This complicates the calculation of the sum of squres needed to test different null hypothesis. Thus, it is important to understand what each function in R computes and if you can use it to test your hypothesis of interest. In particular, you need to understand what you want to test. For example: do you want to compare the expected values of C vs O residencies, regardless of having a fireplace? or do you want to compare the expected values of C vs O residencies after controlling for the existance of a fireplace? (perhaps having a fireplace is what drives the value, not the age!) If the sizes of each group are not equal (unbalanced designs) ignoring the fireplace-effect can have a hugh impact on your analysis!!

There are many discussions about different types of sum of squares (type I SS vs type II SS vs type III SS) and how to compute them in R. However, the most important point is "what is your hypothesis of interest?" and "how do you test it?".

Let's the analysis we've done before on an unbalanced dataset.

```
#create an unbalanced dataset with 2 factors:
set.seed(12134)
dat.rs<-dat[sample(1:nrow(dat),400),]
dat.2<-dat.rs %>% filter(age_factor%in%c("C","O"))
```

First complication: order matters in `aov`... (why??)

```r
summary(aov(assessment_k~age_factor*FIREPLACE,data=dat.2))
```

```
##                       Df   Sum Sq Mean Sq F value   Pr(>F)    
## age_factor             1  2106503 2106503  44.077 1.52e-10 ***
## FIREPLACE              1   507230  507230  10.613  0.00125 ** 
## age_factor:FIREPLACE   1   127711  127711   2.672  0.10318    
## Residuals            294 14050842   47792                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#order DOES matter in THIS case (except for the interaction term)
summary(aov(assessment_k~FIREPLACE*age_factor,data=dat.2))
```

```
##                       Df   Sum Sq Mean Sq F value   Pr(>F)    
## FIREPLACE              1  1179624 1179624  24.682 1.15e-06 ***
## age_factor             1  1434110 1434110  30.007 9.25e-08 ***
## FIREPLACE:age_factor   1   127711  127711   2.672    0.103    
## Residuals            294 14050842   47792                     
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The first important point we need to undestand is that `aov` reports *sequencial* sum of squares (aka type I SS). For a model y~A*B

- SS(A): sum of squares for factor A (only factor in the model at this point, factor B is ignored, compare with the SS from a 1-way ANOVA)
- SS(B|A): sum of squares for factor B, controling for A (B is added to a model that contains A already)
- SS(AB|A,B): sum of squares for the interaction AB, controling for A and B (A and B are already in the model)

Since the sum of squares are computed sequentially, after each factor is added in the model, the order matters!! For a model y~B*A:

- SS(B): sum of squares for factor B (only factor in the model at this point, factor A is ignored, compare with the SS from a 1-way ANOVA)
- SS(A|B): sum of squares for factor A, controling for B (A is added to a model that contains B already)
- SS(AB|B,A): sum of squares for the interaction AB, controling for B and A (A and B are already in the model)

Note why results are different! when group sizes are not equal: SS(B) $\neq$ SS(B|A)! also note why the SS for the interaction are the same!!

So, at this point an important question is *are these SS useful?*. Which hypotheses can we test with those? (which hypotheses are `aov` testing?).

```r
#Compare the SS(A) and SS(B) of the 2-way and those of the 1-way ANOVA tables
summary(aov(assessment_k~age_factor,data=dat.2))
```

```
##              Df   Sum Sq Mean Sq F value  Pr(>F)    
## age_factor    1  2106503 2106503   42.46 3.1e-10 ***
## Residuals   296 14685783   49614                    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
summary(lm(assessment_k~FIREPLACE,data=dat.2))
```

```
##
## Call:
## lm(formula = assessment_k ~ FIREPLACE, data = dat.2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -370.69 -122.44  -40.15   20.83 2390.31
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   363.60      28.93  12.566  < 2e-16 ***
## FIREPLACEY    154.09      32.58   4.729 3.49e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 229.7 on 296 degrees of freedom
## Multiple R-squared:  0.07025,    Adjusted R-squared:  0.06711
## F-statistic: 22.36 on 1 and 296 DF,  p-value: 3.495e-06
```

Similarly,

$$SS(A) = \sum_{i=1}^{a}\sum_{k=1}^{n_{i.}}(\bar{Y}_{i..} - \bar{Y}_{...})^2$$

```r
#marginal means of A
means_A<-dat.2 %>% group_by(age_factor)%>%
  summarize(avg_assessment=mean(assessment_k)) %>%
  select(avg_assessment)%>% unlist %>% as.numeric

#marginal means of B
means_B<-dat.2 %>% group_by(FIREPLACE)%>%
  summarize(avg_assessment=mean(assessment_k)) %>%
  select(avg_assessment)%>% unlist %>% as.numeric

#overall mean
mean_all<-mean(dat.2$assessment_k)
mean_all
```

```
## [1] 485.1174
```

```r
#means of marginal means are now NOT equals the overall mean!
mean(means_A)
```

```
## [1] 498.2531
```

```r
mean(means_B)
```

```
## [1] 440.6484
```

```r
#number of levels of each group
a<-2 #(2 age levels)
b<-2 #(FIREPLACE Y vs N)

#number of observations per levels of factors
n_C.<-count(dat.2 %>% filter(age_factor=="C"))
```

```r
n_O.<-count(dat.2 %>% filter(age_factor=="O"))
size_A<-as.numeric(c(n_C.,n_O.))

n_.Y<-count(dat.2 %>% filter(FIREPLACE=="Y"))
n_.N<-count(dat.2 %>% filter(FIREPLACE=="N"))
size_B<-as.numeric(c(n_.N,n_.Y))

#SS(A): compare with ANOVA table(s)
sum(((means_A-mean_all)^2)*size_A)
```

```
## [1] 2106503
```

```r
#SS(B): compare with ANOVA table(s)
sum(((means_B-mean_all)^2)*size_B)
```

```
## [1] 1179624
```

Let's now look at the summary table of the `lm` function.

```r
summary(lm(assessment_k~age_factor*FIREPLACE,data=dat.2))
```

```
##
## Call:
## lm(formula = assessment_k ~ age_factor * FIREPLACE, data = dat.2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -453.05  -91.32  -45.41   14.79 2307.95
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)             389.60      69.13   5.636 4.09e-08 ***
## age_factorO             -30.90      75.37  -0.410  0.68211
## FIREPLACEY              210.45      72.05   2.921  0.00376 **
## age_factorO:FIREPLACEY -131.74      80.59  -1.635  0.10318
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 218.6 on 294 degrees of freedom
## Multiple R-squared:  0.1633, Adjusted R-squared:  0.1547
## F-statistic: 19.12 on 3 and 294 DF,  p-value: 2.356e-11
```

```r
#order doesn't matter here (uncomment the second line to check)
#summary(lm(assessment_k~FIREPLACE*age_factor,data=dat.balanced))
```

Not surprisingly, the order in which we enter the factors does not affect the output of `lm` since the default tests are differences of means of one factor at a particular level of the second factor (and the reference levels do not depend on order).

As before, the main effect tests do not appearing in the summary table but we can formulate other tests using *contrasts*. Using `lsmeans` again:

```
library(lsmeans)
lm_AB<-lm(assessment_k~age_factor*FIREPLACE,data=dat.2)

means.ab <- lsmeans(lm_AB, specs = c("age_factor", "FIREPLACE"))
means.a <- lsmeans(lm_AB, specs = c("age_factor"))
means.b <- lsmeans(lm_AB, specs = c("FIREPLACE"))
```

Let's estimate and test the same contrasts:

$$H_0 : \mu_{CN} = \mu_{ON}$$

$$(\mu_{CN}, \mu_{ON}, \mu_{CY}, \mu_{OY})$$

```
contrast(means.ab, list(CvOforN=c(1,-1,0,0),CvO = c(.5,-.5,.5,-.5)))
```

```
##  contrast estimate       SE  df t.ratio p.value
##  CvOforN  30.90189 75.37197 294   0.410  0.6821
##  CvO      96.77092 40.29437 294   2.402  0.0169
```

As expected, the first contrast is the same as that tested by `lm` in the default output. However, the second contrast (average of means) is no longer the one tested by `aov`. The reason for this discrepancy is that, unlike for the balance design, the average of the cell means is no longer the marginal mean. Since the sample sizes of each group differ, what is being tested by SS(A) is a *weighted* average of the means of the levels of A. The weights are given by the size of each group:

```
#sample sizes of each group
ss<-with(dat.2,table(age_factor,FIREPLACE))

n.C<-as.numeric(ss[1,])
n.O<-as.numeric(ss[3,])

#let's create weights that sum to 1
w.c<-n.C/sum(n.C)
w.o<-n.O/sum(n.O)
```

In the summary of `aov`, the SS(A) is used to test

$$H_0 = \frac{n_{CY}}{n_C}\mu_{CY} + \frac{n_{CN}}{n_C}\mu_{CN} = \frac{n_{OY}}{n_O}\mu_{OY} + \frac{n_{ON}}{n_O}\mu_{ON}$$

where $n_C = n_{CY} + n_{CN}$ and $n_O = n_{OY} + n_{ON}$

Let's check this with our data. Remember that the entries of the contrast vector must sum to 0!

```
#I've entered numbers by hand to make the code easier to follow but you can also use w.c and w.o define
contrast(means.ab, list(wCvO = c(10/126,-53/172,116/126,-119/172)))
```

```
##  contrast estimate       SE  df t.ratio p.value
##  wCvO     170.1922 25.63517 294   6.639  <.0001
```

```
#contrast(means.ab, list(wCvO = c(w.c[1],-w.o[1],w.c[2],-w.o[2])))

#Note that the t.ratio^2 = the first row of the summary of aov
6.639^2
```

```
## [1] 44.07632
```

```
ssa<-summary(aov(assessment_k~age_factor*FIREPLACE,data=dat.2))
ssa[[1]]$`F value`[1]
```

```
## [1] 44.07651
```

However, the null hypothesis should not depend on the size of the sample you get. This shows that although you can use SS(A) to test a null hypothesis, that hypothesis may not be one of your interest! (same for SS(B)).

By this time, you may be aware that testing hypotheses in 2-way ANOVA is more tricky than in 1-way ANOVA. Usually, type II and type III sum of squares are more useful, but again, it all depends on which hypotheses you want to test. For completion, I now present these alternative SS. However, the main focus of this course is that you know how to use `lm` to test all your hypothesis (by default or using contrasts).

Type II SS These are given by:

- SS(A|B)
- SS(B|A)
- SS(AB|A,B)

Thus, you can extract them from the second and third rows of the summaries of `aov` (fitting the model twice in reverse order), or using car::Anova. Note that the SS(A)≠SS(A|B) and visceversa.

```
library(car)
#By default, Anova() reports type II SS
Anova(lm_AB)
```

```
## Anova Table (Type II tests)
##
## Response: assessment_k
##                         Sum Sq  Df F value    Pr(>F)
## age_factor             1434110   1 30.0073 9.251e-08 ***
## FIREPLACE               507230   1 10.6133  0.001254 **
## age_factor:FIREPLACE    127711   1  2.6722  0.103183
## Residuals             14050842 294
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#equivalent to summary(aov()), sequential (type I) SS
anova(lm_AB)
```

```
## Analysis of Variance Table
##
## Response: assessment_k
##                       Df   Sum Sq Mean Sq F value    Pr(>F)
## age_factor             1  2106503 2106503 44.0765 1.522e-10 ***
## FIREPLACE              1   507230  507230 10.6133  0.001254 **
## age_factor:FIREPLACE   1   127711  127711  2.6722  0.103183
## Residuals            294 14050842   47792
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#Order does not matter!!
# These sums are conditional on the other factor being present in the model.
lm_BA<-lm(assessment_k~FIREPLACE*age_factor,data=dat.2)
Anova(lm_BA)
```

```
## Anova Table (Type II tests)
##
## Response: assessment_k
##                        Sum Sq  Df F value    Pr(>F)
## FIREPLACE              507230   1 10.6133  0.001254 **
## age_factor            1434110   1 30.0073 9.251e-08 ***
## FIREPLACE:age_factor   127711   1  2.6722  0.103183
## Residuals            14050842 294
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(lm_BA)
```

```
## Analysis of Variance Table
##
## Response: assessment_k
##                       Df   Sum Sq Mean Sq F value    Pr(>F)
## FIREPLACE              1  1179624 1179624 24.6825 1.149e-06 ***
## age_factor             1  1434110 1434110 30.0073 9.251e-08 ***
## FIREPLACE:age_factor   1   127711  127711  2.6722    0.1032
## Residuals            294 14050842   47792
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Type III SS These are:

- SS(A|B,AB)
- SS(B|A,AB)
- SS(AB|A,B)

If the interaction term is not significant, the type II SS give more powerfull tests. However, if the interaction term is significant, the main effects should not be meaningfull. Thus, some people argue that type II SS are the most useful ones. Note, that here also, the order does not matter.

Technical note: the parametrization needs to be orthogonal, which is not satisfied by the reference-treatment, thus it needs to be changed using the "contrasts" argument of `Anova`.

```r
Anova.3ss<-Anova(lm(assessment_k~age_factor*FIREPLACE,
          contrasts=list(age_factor='contr.sum',FIREPLACE='contr.sum'),data=dat.2),
      type='III')
Anova.3ss
```

```
## Anova Table (Type III tests)
##
## Response: assessment_k
##                        Sum Sq  Df  F value  Pr(>F)
## (Intercept)          23466761   1 491.0188 < 2e-16 ***
## age_factor             275649   1   5.7677 0.01694 *
## FIREPLACE              615317   1  12.8749 0.00039 ***
## age_factor:FIREPLACE   127711   1   2.6722 0.10318
## Residuals            14050842 294
## ---
```

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

Tests of interest

Knowing that you can compute different types of SS is important. But most important is that you know how to test contrasts of interest and how to compare different nested models. You may find that some of these tests are based on the different types of SS discussed before.

Test for interaction

In a 2-way ANOVA, the easiest test is that of the interaction effect since you have seen that regardless of the order in which we enter the factors or the type of SS used, we would always get the same t-statistic.

```r
#Test for interaction using nested models

#Full model: lm_AB
#Reduced model: lm_AB_add (additive model, no interaction)
lm_AB_add<-lm(assessment_k~age_factor+FIREPLACE,data=dat.2)

#Use anova() to compare the reduced vs the full model
anova(lm_AB_add,lm_AB)
```

```
## Analysis of Variance Table
##
## Model 1: assessment_k ~ age_factor + FIREPLACE
## Model 2: assessment_k ~ age_factor * FIREPLACE
##   Res.Df      RSS Df Sum of Sq      F Pr(>F)
## 1    295 14178553
## 2    294 14050842  1    127711 2.6722 0.1032
```

```r
#Compare the F-statistic with that of `aov` based on type I SS (3rd row)
summary(aov(lm_AB))
```

```
##                       Df   Sum Sq Mean Sq F value   Pr(>F)
## age_factor             1  2106503 2106503  44.077 1.52e-10 ***
## FIREPLACE              1   507230  507230  10.613  0.00125 **
## age_factor:FIREPLACE   1   127711  127711   2.672  0.10318
## Residuals            294 14050842   47792
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#Compare the F-statistic with that of `Anova` based on type II SS (3rd row)
Anova(lm_AB)
```

```
## Anova Table (Type II tests)
##
## Response: assessment_k
##                        Sum Sq  Df F value    Pr(>F)
## age_factor            1434110   1 30.0073 9.251e-08 ***
## FIREPLACE              507230   1 10.6133  0.001254 **
## age_factor:FIREPLACE   127711   1  2.6722  0.103183
## Residuals            14050842 294
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#Compare the F-statistic with that of `Anova` based on type III SS (3rd row)
Anova.3ss
```

```
## Anova Table (Type III tests)
##
## Response: assessment_k
##                        Sum Sq  Df  F value  Pr(>F)
## (Intercept)          23466761   1 491.0188 < 2e-16 ***
## age_factor             275649   1   5.7677 0.01694 *
## FIREPLACE              615317   1  12.8749 0.00039 ***
## age_factor:FIREPLACE   127711   1   2.6722 0.10318
## Residuals            14050842 294
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Test of main effects

The test of the main effects is more tricky! For example, do you want to test the effect of one factor ignoring the second factor? or after controling for it? what is your hypothesis of interest?

The type III SS for A are SS(A|B,AB)) and compares:

Full: main effect A + main effect B + interaction Reduced: main effect B + interaction

The reduced model corresponds to the null hypothesis:

$$H_0 : \mu_{CN} = \mu_{ON} \quad \text{and} \quad \mu_{CY} = \mu_{OY}$$

```r
contrast(means.ab, list(CvO = c(1,-1,1,-1)))
```

```
##  contrast estimate       SE  df t.ratio p.value
##  CvO      193.5418 80.58875 294   2.402  0.0169
```

```r
#Compare the square of the t.test with the F from the type III table (second row)
2.402^2
```

```
## [1] 5.769604
```

```r
Anova.3ss
```


```
## Anova Table (Type III tests)
##
## Response: assessment_k
##                        Sum Sq  Df  F value  Pr(>F)
## (Intercept)          23466761   1 491.0188 < 2e-16 ***
## age_factor             275649   1   5.7677 0.01694 *
## FIREPLACE              615317   1  12.8749 0.00039 ***
## age_factor:FIREPLACE   127711   1   2.6722 0.10318
## Residuals            14050842 294
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The null hypothesis matching the test given by the type II SS is more complicated. Instead, let's examine the test of the main effect in an additive model (no interaction).

```
lm_B <-lm(assessment_k~FIREPLACE,data=dat.2)
anova(lm_B,lm_AB_add)
```

```
## Analysis of Variance Table
##
## Model 1: assessment_k ~ FIREPLACE
## Model 2: assessment_k ~ age_factor + FIREPLACE
##   Res.Df      RSS Df Sum of Sq      F    Pr(>F)
## 1    296 15612663
## 2    295 14178553  1   1434110 29.838 9.989e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#compare this test with that of a type II table for the additive model:
Anova(lm_AB_add)
```

```
## Anova Table (Type II tests)
##
## Response: assessment_k
##              Sum Sq  Df F value    Pr(>F)
## age_factor  1434110   1  29.838 9.989e-08 ***
## FIREPLACE    507230   1  10.553  0.001294 **
## Residuals  14178553 295
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#or with the result of a contrast
means.ab.add <- lsmeans(lm_AB_add, specs = c("age_factor", "FIREPLACE"))
contrast(means.ab.add, list(Cv0 = c(1,-1,1,-1)))
```

```
##  contrast estimate       SE  df t.ratio p.value
##  Cv0      292.2726 53.50592 295   5.462  <.0001
```

Conclusion

To conclude, I want to emphasize that it is important that you understand the hypothesis you are testing in order to make a meaningful interpretation of your results.

Be careful when fitting an unbalanced design! In particular, the `aov` function may not give you useful results. The `lm` function can be used to fit the model of interest and contrasts can be further estimated to test other hypothesis of interest.

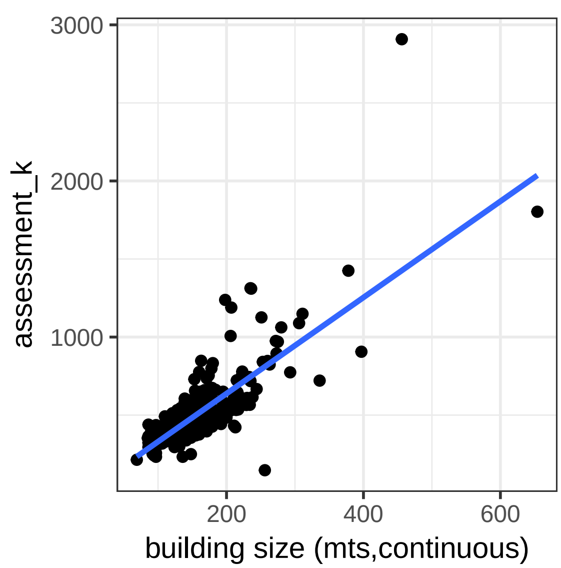
Although not fully covered in this course, it is useful to know that there are different types of SS.

LECTURE 4: Linear regression with continuous covariates

```
#Note that there are 2 different datasets called dat.2 in this document
set.seed(12134)
dat.rs<-dat[sample(1:nrow(dat),500),]
dat.2<-dat.rs %>% filter(age_factor%in%c("C","O")) %>%
  droplevels()

nobs<-nrow(dat.2)

str(dat.2)
```

```
## 'data.frame':    368 obs. of  15 variables:
##  $ TAX_YEAR    : int  2015 2015 2015 2015 2015 2015 2015 2015 2015 2015 ...
##  $ YEAR_BUILT  : int  1966 2005 1958 2007 2002 2009 1970 1970 1970 1969 ...
##  $ BLDG_DESC   : Factor w/ 15 levels "1 1/2 Storey & Basement",..: 4 8 4 8 4 8 14 12 4 4 ...
##  $ BLDG_METRE  : int  97 166 108 217 145 171 106 160 99 104 ...
##  $ BLDG_FEET   : int  1039 1784 1168 2334 1557 1837 1142 1718 1062 1119 ...
##  $ GARAGE      : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 2 2 2 2 ...
##  $ FIREPLACE   : Factor w/ 2 levels "N","Y": 2 2 1 2 2 2 2 2 1 1 ...
##  $ BASEMENT    : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 1 2 2 2 ...
##  $ BSMTDEVL    : Factor w/ 2 levels "N","Y": 2 2 2 2 2 1 1 2 2 2 ...
##  $ ASSESSMENT  : int  354000 449000 383000 536000 595000 449000 363000 776000 349000 371000 ...
##  $ LATITUDE    : num  53.5 53.5 53.5 53.6 53.5 ...
##  $ LONGITUDE   : num  -113 -113 -113 -113 -113 ...
##  $ age         : num  51 12 59 10 15 8 47 47 47 48 ...
##  $ age_factor  : Factor w/ 2 levels "C","O": 2 1 2 1 1 1 2 2 2 2 ...
##  $ assessment_k: num  354 449 383 536 595 449 363 776 349 371 ...
```

```
g<-ggplot(data=dat.2,aes(BLDG_METRE, assessment_k))+geom_point()+
  geom_smooth(method="lm",se=FALSE)+xlab("building size (mts,continuous)")

g
```




```
lm_ageBLDG<-lm(assessment_k~BLDG_METRE*age_factor,data=dat.2)
```

```
#fit a linear model to each subset data and compare the results!
lm_BLDG_C<-lm(assessment_k~BLDG_METRE,data=subset(dat.2,age_factor=="C"))
lm_BLDG_O<-lm(assessment_k~BLDG_METRE,data=subset(dat.2,age_factor=="O"))

#compare coefficients tables:
tidy(lm_ageBLDG)
```

```
##                     term   estimate  std.error statistic      p.value
## 1            (Intercept) -52.365371 31.1763051 -1.679653 9.388291e-02
## 2             BLDG_METRE   3.544841  0.1635329 21.676620 1.677771e-67
## 3            age_factorO 186.824749 42.1280284  4.434690 1.222759e-05
## 4 BLDG_METRE:age_factorO  -1.385608  0.2649399 -5.229898 2.867503e-07
```

```
tidy(lm_BLDG_C)
```

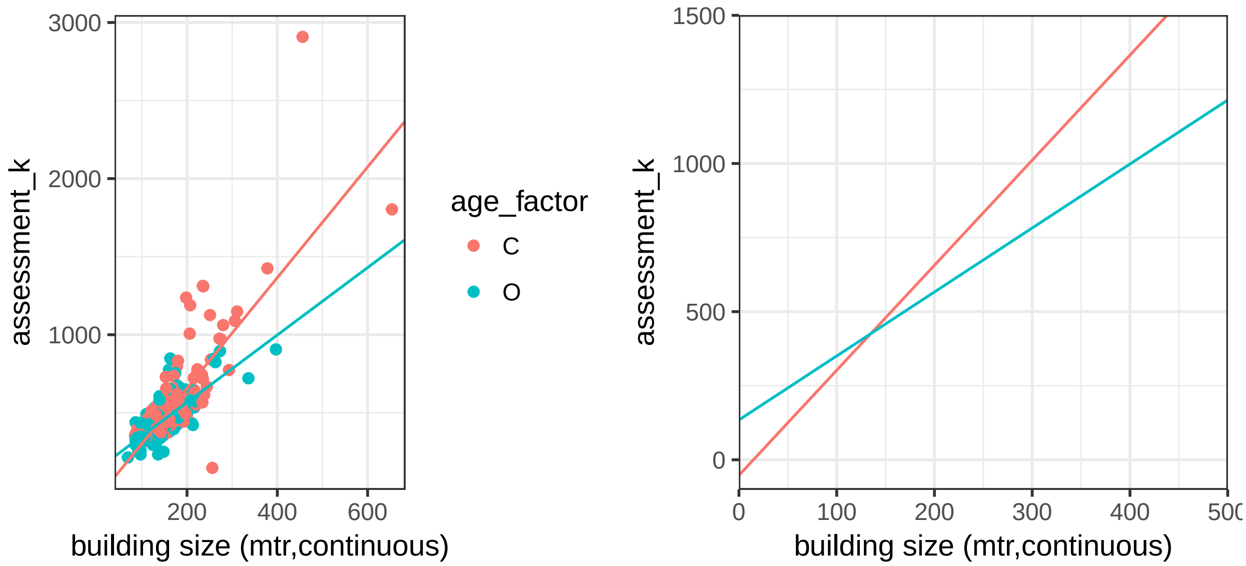
```
##          term   estimate  std.error statistic      p.value
## 1 (Intercept) -52.365371 44.5311375 -1.175927 2.414995e-01
## 2  BLDG_METRE   3.544841  0.2335847 15.175829 4.884061e-32
```

```
tidy(lm_BLDG_O)
```

```
##          term   estimate  std.error statistic      p.value
## 1 (Intercept) 134.459378 14.9687421  8.982677 1.362957e-16
## 2  BLDG_METRE   2.159233  0.1101221 19.607619 9.090155e-50
```

```
#Reference level
b1_C<-tidy(lm_ageBLDG) %>%
  filter(term == "BLDG_METRE") %>%
  select(estimate)%>% as.numeric

b0_C<-tidy(lm_ageBLDG) %>%
  filter(term == "(Intercept)") %>%
  select(estimate) %>% as.numeric

#other level
b1_O<-as.numeric(tidy(lm_ageBLDG) %>%
  filter(term == "BLDG_METRE:age_factorO") %>%
  select(estimate)%>% as.numeric)+b1_C

b0_O<-as.numeric(tidy(lm_ageBLDG) %>%
  filter(term == "age_factorO") %>%
  select(estimate)%>% as.numeric)+b0_C

#plot with age_factor colors

#Note: the lines added by geom_smooth when color=age_factor are not
# same as those estimated with an interaction term in `lm` (???)

g<-ggplot(data=dat.2,aes(BLDG_METRE, assessment_k))+geom_point(data=dat.2,aes(BLDG_METRE, assessment_k,
   geom_abline(slope=b1_C,intercept = b0_C,col="#F8766D")+
  geom_abline(slope=b1_O,intercept =b0_O,col="#00BFC4")+
  xlab("building size (mtr,continuous)")

g_zoom<-ggplot(data=dat.2,aes(BLDG_METRE, assessment_k))+
  geom_abline(slope=b1_C,intercept = b0_C,col="#F8766D")+
  geom_abline(slope=b1_O,intercept =b0_O,col="#00BFC4")+
```

```
    scale_x_continuous(expand=c(0,0), limits=c(0,500))+
    scale_y_continuous(expand=c(0,0), limits=c(-100,1500))+
  xlab("building size (mtr,continuous)")

plot_grid(g, g_zoom, align = "h")
```




```
lm_ageBLDG_add<-lm(assessment_k~BLDG_METRE+age_factor,data=dat.2)
summary(lm_ageBLDG_add)
```

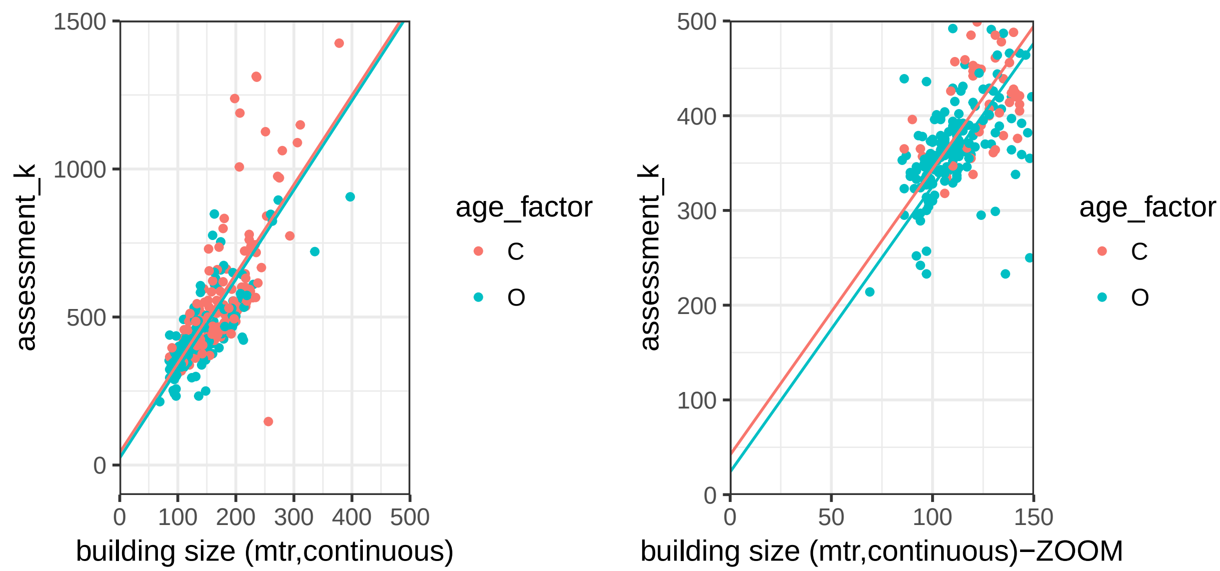
```
##
## Call:
## lm(formula = assessment_k ~ BLDG_METRE + age_factor, data = dat.2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -667.36  -68.91    1.14   38.84 1490.25
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  42.0249    26.3226   1.597    0.111
## BLDG_METRE    3.0169     0.1332  22.645   <2e-16 ***
## age_factorO -18.0826    16.0313  -1.128    0.260
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 137.6 on 365 degrees of freedom
## Multiple R-squared:  0.6392, Adjusted R-squared:  0.6372
## F-statistic: 323.4 on 2 and 365 DF,  p-value: < 2.2e-16
```

```
#One slope
b1<-tidy(lm_ageBLDG_add) %>%
  filter(term == "BLDG_METRE") %>%
  select(estimate)%>% as.numeric

#intercept of reference
b0_C<-tidy(lm_ageBLDG_add) %>%
```

```r
  filter(term == "(Intercept)") %>%
  select(estimate) %>% as.numeric

#intercept of other level
b0_O<-as.numeric(tidy(lm_ageBLDG_add) %>%
  filter(term == "age_factorO") %>%
  select(estimate)%>% as.numeric)+b0_C

g_add<-ggplot(data=dat.2,aes(BLDG_METRE, assessment_k))+
  geom_point(aes(color=age_factor),size=1)+
  geom_abline(slope=b1,intercept = b0_C,col="#F8766D")+
  geom_abline(slope=b1,intercept =b0_O,col="#00BFC4")+
    scale_x_continuous(expand=c(0,0), limits=c(0,500))+
    scale_y_continuous(expand=c(0,0), limits=c(-100,1500))+
  xlab("building size (mtr,continuous)")

g_add_zoom<-ggplot(data=dat.2,aes(BLDG_METRE, assessment_k))+
  geom_point(aes(color=age_factor),size=1)+
  geom_abline(slope=b1,intercept = b0_C,col="#F8766D")+
  geom_abline(slope=b1,intercept =b0_O,col="#00BFC4")+
    scale_x_continuous(expand=c(0,0), limits=c(0,150))+
    scale_y_continuous(expand=c(0,0), limits=c(0,500))+
  xlab("building size (mtr,continuous)-ZOOM")

plot_grid(g_add, g_add_zoom, align = "h")
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```

```
## Warning: Removed 157 rows containing missing values (geom_point).
```



LECTURE 5: Estimation and inference

```
lm_BLDG<-lm(assessment_k~BLDG_METRE,data=dat.2)
tidy(lm_BLDG)
```

```
##          term  estimate  std.error statistic      p.value
## 1 (Intercept) 22.046047 19.4790234  1.131784 2.584662e-01
## 2  BLDG_METRE  3.079313  0.1212517 25.396038 9.282329e-83
```

```
summary(lm_BLDG)
```

```
##
## Call:
## lm(formula = assessment_k ~ BLDG_METRE, data = dat.2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -663.35  -62.18   -2.37   39.15 1481.79
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  22.0460    19.4790   1.132    0.258
## BLDG_METRE    3.0793     0.1213  25.396   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 137.7 on 366 degrees of freedom
## Multiple R-squared:  0.638,  Adjusted R-squared:  0.637
## F-statistic:   645 on 1 and 366 DF,  p-value: < 2.2e-16
```

Let's calculate the slope of the regression line by hand:

```
(beta_1 <- cor(x = dat.2$assessment_k, y = dat.2$BLDG_METRE) * sd(dat.2$assessment_k) / sd(dat.2$BLDG_MI
```

```
## [1] 3.079313
```

We can also calculate the intercept by hand as well:

```
(beta_0 <- mean(dat.2$assessment_k) - beta_1 * mean(dat.2$BLDG_METRE))
```

```
## [1] 22.04605
```

Let's take a look at the residuals: observed `assessment_k` values minus the predicted/fitted values from the *estimated* regression line.

```
vals_n_errors <- augment(lm_BLDG) %>%
  select(assessment_k, BLDG_METRE, .fitted, .resid)
head(vals_n_errors)
```

```
##   assessment_k BLDG_METRE  .fitted     .resid
## 1          354         97 320.7394   33.26057
## 2          449        166 533.2120  -84.21204
## 3          383        108 354.6119   28.38813
## 4          536        217 690.2570 -154.25702
## 5          595        145 468.5465  126.45354
## 6          449        171 548.6086  -99.60861
```

Let's just look at the first row, and prove that `.resid` values are equal to the actual `assessment_k` values minus the predicted/fitted values:

```
(first_observation <- vals_n_errors[1, ])
```

```
##   assessment_k BLDG_METRE  .fitted   .resid
## 1          354         97 320.7394 33.26057
```

```
# Does rpg - .fitted == .resid
all.equal((first_observation$assessment_k - first_observation$.fitted),
          first_observation$.resid)
```

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

Let's compute by hand the SD of the residuals

```
sd(vals_n_errors$.resid)*sqrt((nobs-1)/(nobs-2))
```

```
## [1] 137.677
```

```
with(dat.2,sqrt(sum((assessment_k-vals_n_errors$.fitted)^2)/(nobs-2)))
```

```
## [1] 137.677
```

We can also calculate 95% confidence intervals by using the `confint()` function in R, or using the mathematical formula:

```
# using confint
(beta_1_ci <- confint(lm_BLDG, 'BLDG_METRE', level = 0.95))
```

```
##               2.5 %   97.5 %
## BLDG_METRE 2.840876 3.317751
```

```
# using formula
b1<-tidy(lm_BLDG) %>%
  filter(term == "BLDG_METRE") %>%
  select(estimate) %>% as.numeric

b1_se<-tidy(lm_BLDG) %>%
  filter(term == "BLDG_METRE") %>%
  select(std.error) %>% as.numeric

b1 + c(-1,1) * qt(0.975, nobs - 2) * b1_se
```

```
## [1] 2.840876 3.317751
```

Because our intercept and slope are estimates, there is some uncertainty associated with the "predicted" values we calculate. When we make predictions about a new observation, we can create prediction interval and/or confidence interval for the predicted values (not the same thing!).

```
sigma_hat<-sd(vals_n_errors$.resid)*sqrt((nobs-1)/(nobs-2))
predicted_y_se <- sigma_hat * sqrt(1 + (1/nobs) + (dat.2$BLDG_METRE - mean(dat.2$BLDG_METRE))^2 / ((nobs

upr <- lm_BLDG$fitted.values + qt(0.975, nobs - 2) * predicted_y_se
lwr <- lm_BLDG$fitted.values- qt(0.975, nobs - 2) * predicted_y_se

head(prediction_int <- data.frame(lwr, upr))
```

```
##         lwr      upr
## 1  49.34739 592.1315
## 2 262.07810 804.3460
## 3  83.32781 625.8959
```

```
## 4 418.67268 961.8414
## 5 197.43962 739.6533
## 6 277.45459 819.7626
```

We can use R's `predict.lm()` function to get these as well!

```r
head(predict.lm(lm_BLDG, interval = "prediction"))
```

```
## Warning in predict.lm(lm_BLDG, interval = "prediction"): predictions on current data refer to _future
```

```
##        fit       lwr      upr
## 1 320.7394  49.34739 592.1315
## 2 533.2120 262.07810 804.3460
## 3 354.6119  83.32781 625.8959
## 4 690.2570 418.67268 961.8414
## 5 468.5465 197.43962 739.6533
## 6 548.6086 277.45459 819.7626
```

note - our `predicted_y_se` is NOT the same as what R returns as `se.fit`:

```r
head(predicted_y_se)
```

```
## [1] 138.0100 137.8787 137.9551 138.1078 137.8650 137.8889
```

```r
head(augment(lm_BLDG)$.se.fit)
```

```
## [1]  9.581013  7.455523  8.754609 10.899372  7.196246  7.642027
```

Thus, if you want to calculate the confidence interval for the predicted values, instead of the prediction interval, you need to use the `se.fit` R returns

```r
upr <- lm_BLDG$fitted.values + qt(0.975, nobs - 2) * augment(lm_BLDG)$.se.fit
lwr <- lm_BLDG$fitted.values - qt(0.975, nobs - 2) * augment(lm_BLDG)$.se.fit
head(predicted_ci_int <- data.frame(lwr, upr))
```

```
##        lwr      upr
## 1 301.8987 339.5802
## 2 518.5510 547.8731
## 3 337.3962 371.8275
## 4 668.8238 711.6903
## 5 454.3953 482.6976
## 6 533.5808 563.6364
```

```r
predicted_fits <- data.frame(predict.lm(lm_BLDG, interval = "confidence", se.fit = TRUE)$fit)
head(predicted_fits)
```

```
##        fit      lwr      upr
## 1 320.7394 301.8987 339.5802
## 2 533.2120 518.5510 547.8731
## 3 354.6119 337.3962 371.8275
## 4 690.2570 668.8238 711.6903
## 5 468.5465 454.3953 482.6976
## 6 548.6086 533.5808 563.6364
```

LECTURE 6: Bootstrapping in Regression

In Lecture 5, we derived closed form formulas for the OLS estimators of the regression coefficients and their first two moments (more precisely expected value and variance). In addition, if we assume that the error term in the regression is normally distributed, we derived the distribution of the OLS estimator. However, for some other estimators (different from the OLS) closed form formulas may not exist and only some assymptotic results may be available (in some cases, not even assymptotic results are known!). Making inference about estimators of these kind is not so easy and we need additional machinary.

Bootstrapping can be used to estimate the sampling distribution of the regression estimator without assuming any population distribution for the error term. Having an estimated sampling distribution allows you to perform (non-parametric) hypothesis testing about the regression coefficients and to estimate the expected value and variance of the regression estimators. The sampling distribution of the regression estimates is obtained sampling (with replacement) from the *data*.

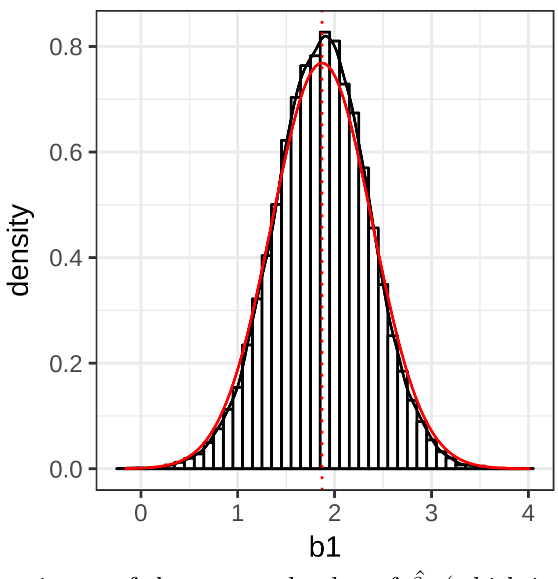
Let's first generate data from a Normal distribution since in that case we *know* that the OLS estimators are normally distributed.

```
nobs<-100
#true regression coefficients
b0<-1
b1<-2

set.seed(1234)
x<-rnorm(nobs,0,1)
#I generate more noisy data so that the p-values are not 0
# and we can compare different methods
y<-b0+b1*x+rnorm(nobs,0,5)

dat.reg<-data.frame(y=y,x=x)
lm.original<-lm(y~x,data=dat.reg)
b1_hat_original<-lm.original$coefficients[2]
se_b1_original<-summary(lm.original)$coef[2,2]
summary(lm.original)
```

```
##
## Call:
## lm(formula = y ~ x, data = dat.reg)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.4313  -3.0700   0.0118   2.9322  14.9387
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.1858     0.5249   2.259 0.026095 *
## x             1.8696     0.5189   3.603 0.000496 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.186 on 98 degrees of freedom
## Multiple R-squared:  0.117,  Adjusted R-squared:  0.108
## F-statistic: 12.98 on 1 and 98 DF,  p-value: 0.0004961
```

```
#Draw bootstrap samples
B <- 50000
```

```
boot_estimates<-data.frame(b1=c(),s1=c())

set.seed(12345)
for(i in 1:B){
#sample rows of dat.reg with replacement (z=(y,x))
boot_obs<-sample(1:nobs,size = nobs, replace = TRUE)
boot_sample <-dat.reg[boot_obs,]
#compute the OLS slope in each bootstrap sample
lm_b<-summary(lm(y~x,data=boot_sample))$coef
boot_estimates[i,"b1"] <-lm_b[2,1]
#note that in this test I rely on knowing the SE of beta_hat_1
boot_estimates[i,"s1"] <-lm_b[2,2]
}

#Sampling distribution of of beta_hat_1: histogram
g<-ggplot(boot_estimates, aes(x=b1)) +
  geom_histogram(aes(y=..density..),
                 binwidth=.1,colour="black", fill="white") +
  # Overlay estimated density
  geom_density(alpha=.2)+
  # Overlay normal density with (overall) sample estimates as parameteres
  stat_function(fun=dnorm,color="red",
                args=list(mean=b1_hat_original,sd=se_b1_original))+
  geom_vline(xintercept=b1_hat_original,colour="red",linetype="dotted")

g
```



The average of all bootstrap estimates ($\frac{1}{B}\sum_{b=1}^{B}\hat{\beta}_1^{\ b}$ is an estimate of the expected value of $\hat{\beta}_1$ (which in some cases may be unknown). Similarly, in the absence of a closed form or assymptotic standard error of the estimator, we can use the standard deviation of the bootstrap estimates as an estimate.

```
#Estimating the mean of beta_hat_1
mean(boot_estimates$b1)
```

```
## [1] 1.868602
```

```r
#bootstrap bias
(bootBias<-mean(boot_estimates$b1)-b1_hat_original)
```

```
##             x
## -0.0009735867
```

```r
#Estimating the SE of beta_hat_1
(bootStdErr <- sqrt(var(boot_estimates$b1)))
```

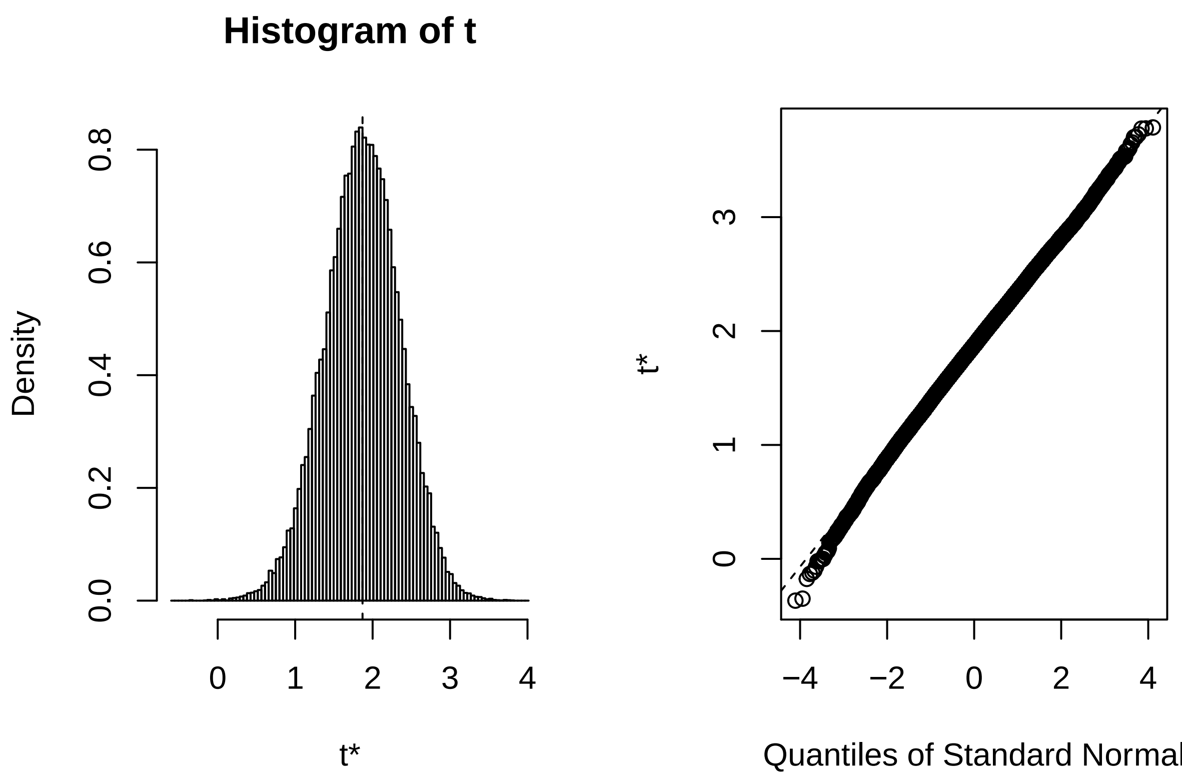
```
## [1] 0.4882007
```

We can also use the sampling distribution or the estimates to get confidence intervals for $\hat{\beta}_1$ and to test hypothesis of interest. Although we can continue with the results obtained before, I now show hot to use R packages for bootstrapping.

```r
#Using `boot` package
library(boot)

# create a function that retains estimates of interest
# note that I'm storing only the slope and it's SE
boot.lm <- function(data, indices){
 data <- data[indices,] # select obs. in bootstrap sample
 lm_boot <- lm(y ~ x, data=data)
 # return estimated slope and SE
 summary(lm_boot)$coef[2,1:2]}

boot_res <- boot(dat.reg, boot.lm, B)

#Note that t1 corresponds to the slope and t2 to its SE
boot_res
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = dat.reg, statistic = boot.lm, R = B)
##
##
## Bootstrap Statistics :
##      original        bias    std. error
## t1* 1.8695753 -3.462972e-04  0.48449469
## t2* 0.5188795 -7.802421e-05  0.05855692
```

```r
#sampling distribution of the estiamted slope
plot(boot_res)
```




```r
#compare with the bias in `boot_res` object
mean(boot_res$t[,1])-boot_res$t0[1]
```

```
##      Estimate
## -0.0003462972
```

```r
#Estimating the SE of the slope: compare with the std.error in `boot_res` object
sqrt(var(boot_res$t[,1]))
```

```
## [1] 0.4844947
```

There are different type of confidence intervals depending on the assumptions you can make: for example, you can use the estimated bootstrap standard error to create normal-theory intervals or you can use the quantiles of the sampling distribution to create percentile intervals.

```r
#normal-theory confidence intervals for beta_hat_1
boot.ci(boot_res,index=1,type="norm")
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 50000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_res, type = "norm", index = 1)
##
## Intervals :
## Level      Normal
## 95%   ( 0.92,  2.82 )
## Calculations and Intervals on Original Scale
```

```r
#percentile confidence intervals for beta_hat_1
boot.ci(boot_res,index=1,type="perc")
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
```

```
## Based on 50000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_res, type = "perc", index = 1)
##
## Intervals :
## Level     Percentile
## 95%   ( 0.902,  2.799 )
## Calculations and Intervals on Original Scale
```

These confidence intervals can be used the null hypothesis $H_0 : \beta_1 = 0$ at an α significance level checking if 0 is in the interval. Otherwise, we can calculate (non-parametrically) the p-value of a test-statistic to test this hypothesis based on the following pivotal test statistic:

$$Z^b = \frac{\hat{\beta}_1^b - \hat{\beta}_1}{SE(\hat{\beta}_1^b)}$$

Note that the sample acts as the "population" to center the estimated coefficients.

To calculate the p-value we need to count how many times the absolute value of this Z-statistic is larger than one obtained from the original sample, centered at the null value 0.

Note: there are many ways of calculating bootstrapping p-values. If the test is not based on a pivotal test statistics, the data needs to be generated under the null hypothesis before bootstrapping (see Davison, A. C. & D. V. Hinkley. 1997. Bootstrap Methods and their Application. Cambridge: Cambridge University Press)

```
#pval
#create a pivotal test statistic from each bootstrapped sample:
z_boot<-(boot_res$t[,1]-boot_res$t0[1])/boot_res$t[,2]

(1 + sum(abs(z_boot) > abs(boot_res$t0[1]/boot_res$t0[2])))/ ( B + 1)
```

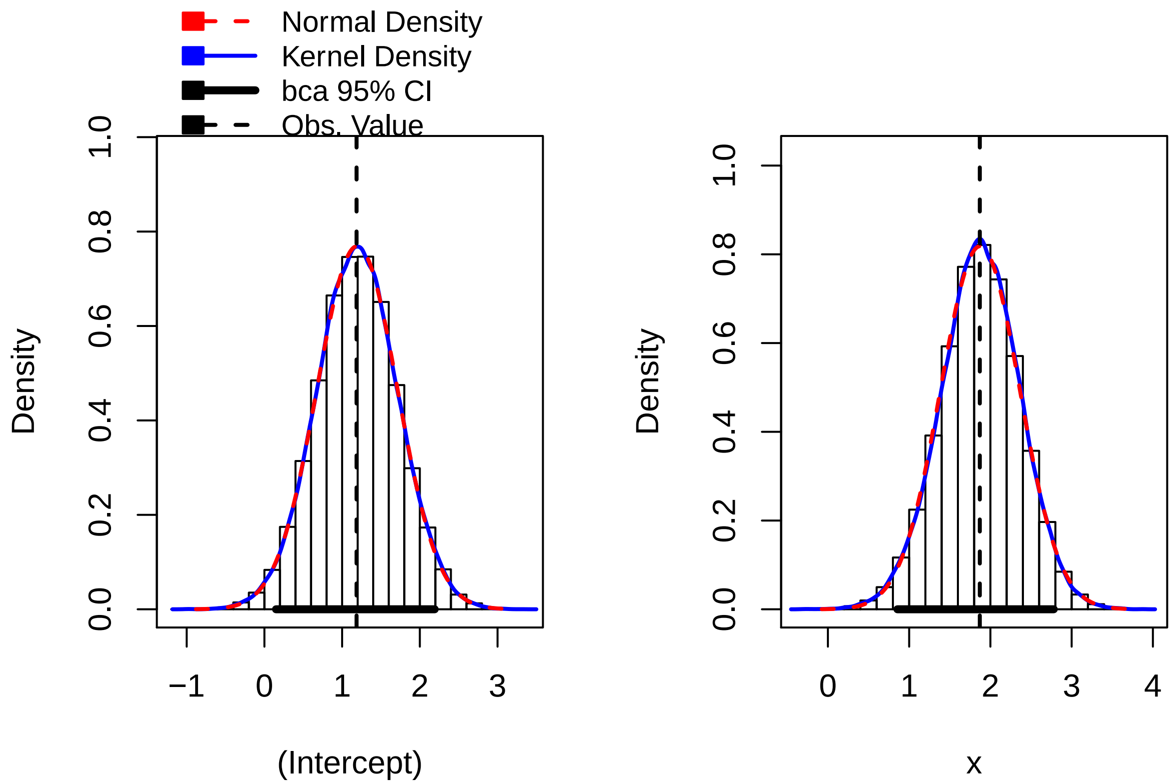
```
## [1] 0.0003399932
```

The function `Boot` from the package `car` is also very useful and provides good summary of the results.

```
library(car)
Boot_res <- Boot(lm.original, R=B)
summary(Boot_res)
```

```
##                 R original  bootBias  bootSE bootMed
## (Intercept) 50000   1.1858 0.0082808 0.51872  1.1934
## x           50000   1.8696 0.0042896 0.48747  1.8782
```

```
hist(Boot_res)
```




```
#Confidence Intervals
confint(Boot_res, level=.95, type="norm")
```

```
## Bootstrap quantiles, type =  normal
##
##                  2.5 %   97.5 %
## (Intercept) 0.1608074 2.194172
## x           0.9098685 2.820703
```

```
confint(Boot_res, level=.95, type="perc")
```

```
## Bootstrap quantiles, type =  percent
##
##                  2.5 %   97.5 %
## (Intercept) 0.1729451 2.212408
## x           0.8949920 2.814357
```

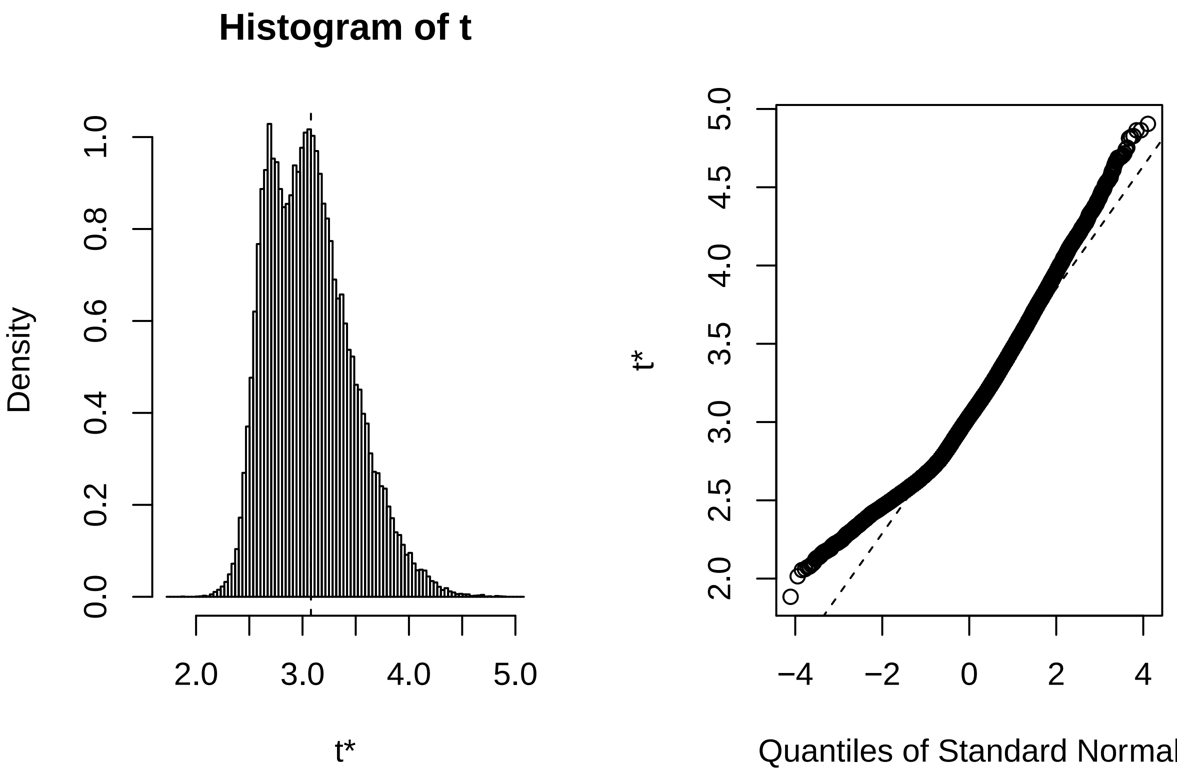
Let's now go back to the Property Assessment Tax data. We can sample (many times with replacement) from the *data* to obtain the sampling distribution of the OLS estimator.

```
set.seed(12134)
dat.rs<-dat[sample(1:nrow(dat),500),]
dat.2<-dat.rs %>% filter(age_factor%in%c("C","O")) %>%
  droplevels()

nobs<-nrow(dat.2) #368
lm_BLDG_original<-lm(assessment_k~BLDG_METRE,data=dat.2)

#Using `boot`
boot.lm <- function(data, indices){
 data <- data[indices,] # select obs. in bootstrap sample
```

```
 lm_BLDG<-lm(assessment_k~BLDG_METRE,data)
 summary(lm_BLDG)$coef[2,1:2]}

boot_tax <- boot(dat.2, boot.lm, B)

boot_tax
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = dat.2, statistic = boot.lm, R = B)
##
##
## Bootstrap Statistics :
##      original        bias    std. error
## t1* 3.0793132 -0.008641481  0.39098165
## t2* 0.1212517 -0.001987248  0.01914459
```

```
plot(boot_tax,index=1)
```




```
#a glance at the bootstrap-estimates
head(boot_tax$t)
```

```
##          [,1]       [,2]
## [1,] 3.541241 0.13186664
## [2,] 3.594908 0.15253397
## [3,] 2.814972 0.12756366
## [4,] 3.359333 0.12139881
```

```
## [5,] 3.524629 0.11834688
## [6,] 2.599836 0.08377013
```

```r
#hypothesis test
#H0:b1=0

z_boot_tax<-(boot_tax$t[,1]-boot_tax$t0[1])/boot_tax$t[,2]

(1 + sum(abs(z_boot_tax) > abs(boot_tax$t0[1]/boot_tax$t0[2])))/ ( B + 1)
```

```
## [1] 1.99996e-05
```

LECTURE 8: Diagnostics

```r
set.seed(12134)
dat.rs<-dat[sample(1:nrow(dat),500),]
```

Regression with different type of variables

In this section we will consider `age` as a continuous variable

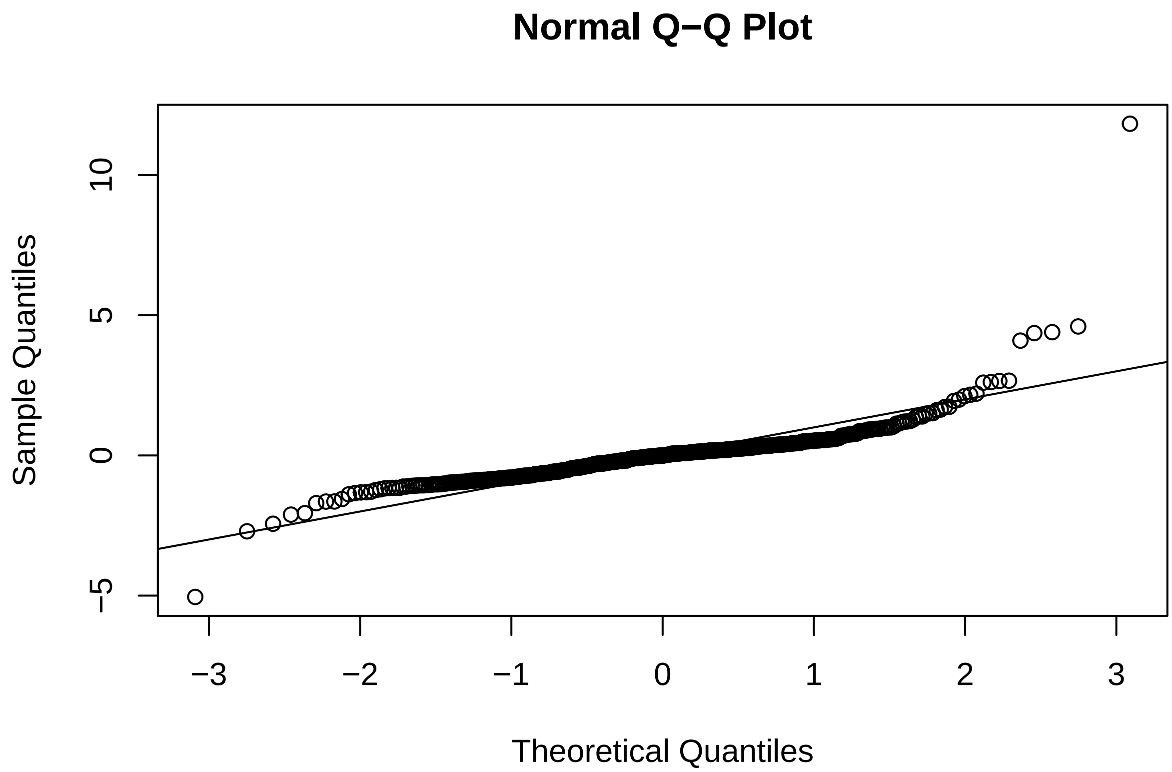
```r
summary(lm(assessment_k~BLDG_METRE+age+FIREPLACE,data=dat.rs))
```

```
##
## Call:
## lm(formula = assessment_k ~ BLDG_METRE + age + FIREPLACE, data = dat.rs)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -657.65  -71.57   -0.52   44.30 1541.48
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  80.5134    27.0066   2.981  0.00301 **
## BLDG_METRE    2.8243     0.1126  25.093  < 2e-16 ***
## age          -0.7482     0.4023  -1.860  0.06347 .
## FIREPLACEY    3.3583    15.5389   0.216  0.82898
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 130.7 on 496 degrees of freedom
## Multiple R-squared:  0.6304, Adjusted R-squared:  0.6282
## F-statistic:   282 on 3 and 496 DF,  p-value: < 2.2e-16
```

Some initial diagnostic plots

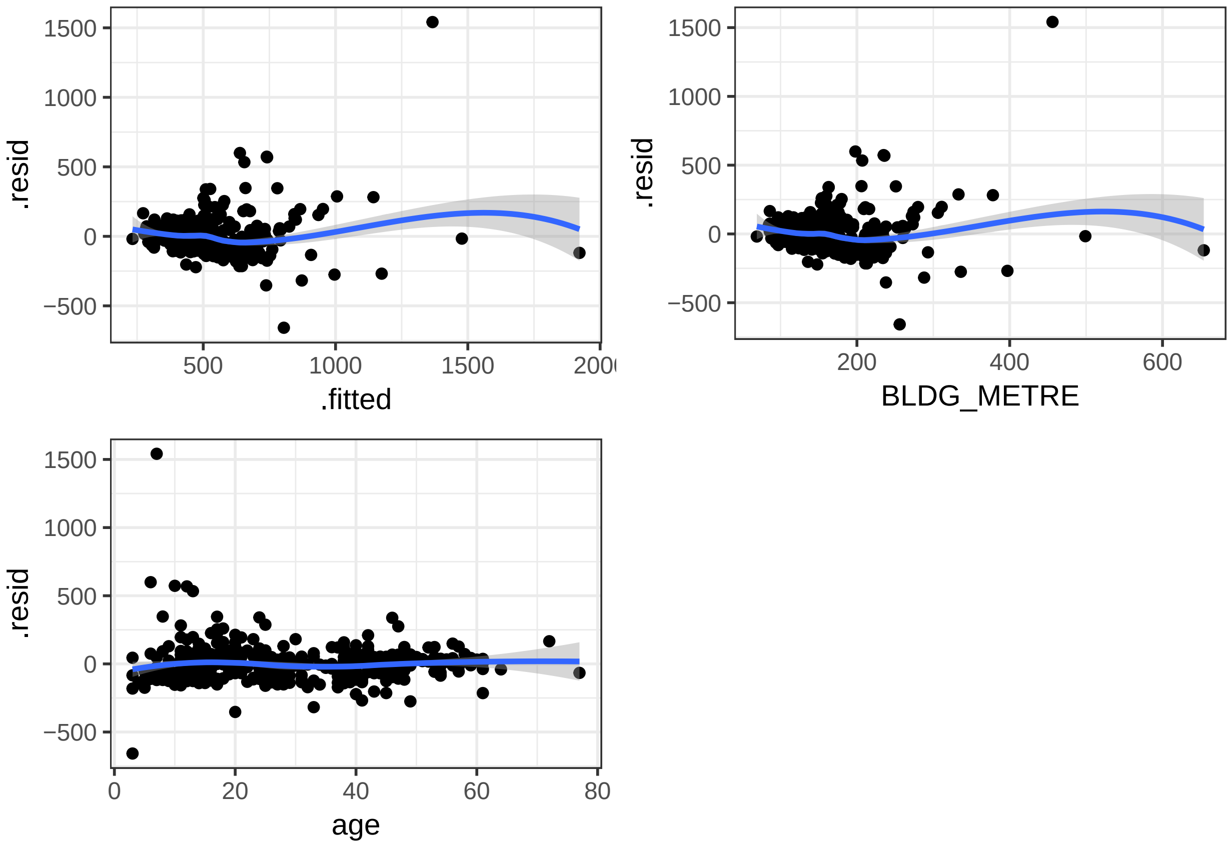
The Normal Q-Q plot suggests some large deviation from the normality assumption.

```
fit <- dat.rs %>%
do(augment(lm(assessment_k~BLDG_METRE+age+FIREPLACE,data=.), data=.))
qqnorm(scale(fit$.resid))
abline(0,1)
```



The plot of the predicted values (`.fitted`) against the residuals also shows an outlying residence (underestimation when the predicted values is high). Perhaps a transformation of the size of the building can improve the fit.

```
g1 <- fit  %>% ggplot(aes(.fitted,.resid)) + geom_point() + geom_smooth()
g2 <- fit  %>% ggplot(aes(BLDG_METRE,.resid)) + geom_point() + geom_smooth()
g3 <- fit  %>% ggplot(aes(age,.resid)) + geom_point() + geom_smooth()
plot_grid(g1, g2,g3)
```


.resid
.fitted
.resid
BLDG_METRE
.resid
age