PSYP13 - Rcode for home assignment - Emilia Backman, 7th December 2018

```
############################ Assignment 1 #################################

data_sample_1 =
read.csv("https://raw.githubusercontent.com/kekecsz/PSYP13_Data_analysis_clas
s-2018/master/home_sample_1.csv")

View(data_sample_1)

#Load Packages

require(psych)

require(lm.beta)

require(car)

require(ggplot2)

require(rgl)


#removing ID-28, 112, 146

data_sample_2 <- data_sample_1[-c(28, 112, 146), ]

describe(data_sample_2)

View(data_sample_2)


#checking variables

hist(data_sample_2$age, breaks = 30)

hist(data_sample_2$pain, breaks = 30)

hist(data_sample_2$STAI_trait, breaks = 30)

hist(data_sample_2$pain_cat, breaks = 30)

hist(data_sample_2$cortisol_serum, breaks = 30)

hist(data_sample_2$cortisol_saliva, breaks = 30)

hist(data_sample_2$mindfulness, breaks = 30)

hist(data_sample_2$weight, breaks = 30)


#####Regression

mod_pain <- lm(pain ~ age + sex, data = data_sample_2)

print(mod_pain)

summary( mod_pain )

# in the sex variable, female becomes default because it is first in the
alphabet.


mod_pain_plus = lm(pain ~ age + sex + STAI_trait + pain_cat + cortisol_serum
+ cortisol_saliva + mindfulness, data = data_sample_2)

print(mod_pain_plus)

summary(mod_pain_plus)
```

```
##### Checking assumptions from Navarro p-474
windows()
plot(x = mod_pain_plus, which = 2) #etc.

# outliers seem to be 71. 24, 100, 44, 51 ?????? Check.
residualPlots(model = mod_pain_plus)

#multicollinearity
require(car)
require(lmtest)
vif( mod = mod_pain_plus )

### we want to remove one of the cortisol things, saliva
#Regression without cortisol saliva

mod_pain_plus1 = lm(pain ~ age + sex + STAI_trait + pain_cat + cortisol_serum
+ mindfulness, data = data_sample_2)

##### Checking assumptions from Navarro p-474
window()
plot(x = mod_pain_plus)
window()
plot(x = mod_pain_plus, which = 4)
window()
plot(x = mod_pain_plus, which = 3)

# outliers seem to be 19,44,51 from cooks distance,100,47,24 from fitted
values ?????? Check.

windows()
residualPlots(model = mod_pain_plus1)

#multicollinearity
require(car)
require(lmtest)
Vif( mod = mod_pain_plus1)
# VIF okay
```

```
hatvalues( model = mod_pain_plus1 )
windows()
hist( x = residuals( mod_pain_plus1 ), xlab = "Value of residual", main =
"residuals",breaks = 20)


yhat.2 <- fitted.values( object = mod_pain_plus1 )
windows()
plot( x = yhat.2, y = data_sample_2$pain, xlab = "Fitted Values", ylab =
"Observed Values")


######COMPARING MODELS
summary(mod_pain)
summary(mod_pain_plus1)
lm.beta(mod_pain)
lm.beta(mod_pain_plus1)
confint(pod_pain)
confint(mod_pain_plus1)


#summary(mod_pain)$adj.r.squared
#summary(mod_pain_plus1)$adj.r.squared


anova(mod_pain, mod_pain_plus1)


# AIC
AIC(mod_pain)
AIC(mod_pain_plus1)



############################### Assignment 2 ################################


data_sample_1 =
read.csv("https://raw.githubusercontent.com/kekecsz/PSYP13_Data_analysis_clas
s-2018/master/home_sample_1.csv")
View(data_sample_1)


require(psych)
require(lm.beta)
require(car)
require(ggplot2)
require(rgl)

#removing ID-28, 112, 146

data_sample_2 <- data_sample_1[-c(28, 112, 146), ]

describe(data_sample_2)
View(data_sample_2)

hist(data_sample_2$weight, breaks = 30)

#Regression

mod_pain_plus2 = lm(pain ~ age + sex + STAI_trait + pain_cat + cortisol_serum
+ weight + mindfulness, data = data_sample_2)

summary(mod_pain_plus2)

summary(mod_pain_plus2)$adj.r.squared

lm.beta(mod_pain_plus2)
confint(mod_pain_plus2)
confint.lm(mod_pain_plus2)



#### Checking assumptions from Navarro p-474

plot(x = mod_pain_plus2, which=5) etc#

# outliers seem to be 19, 44, 104, 100, 47, 24,  ?????? Check.

residualPlots(model = mod_pain_plus2)

#multicollinearity
```

PSYP13 - Rcode for home assignment - Emilia Backman, 7th December 2018

```
require(car)

require(lmtest)


vif( mod = mod_pain_plus2 )


#### note you can run the analysis with and without the outliers and report
if there is a difference in one line and justify why you decided to keep them
all in.


#####stepwise


mod_pain_back = step(mod_pain_plus2, direction = "backward")


########

backwardmodel = lm(pain ~ age + sex+ pain_cat + cortisol_serum + mindfulness,
data = data_sample_2)

theorybasedmodel = lm(pain ~ age + sex + STAI_trait + pain_cat +
cortisol_serum + mindfulness, data = data_sample_2)


summary(backwardmodel)


AIC(mod_pain_plus2)

AIC(backwardmodel)

anova(mod_pain_plus2, backwardmodel)


AIC(theorybasedmodel)

summary(theorybasedmodel)


summary(backwardmodel)$adj.r.squared

summary(theorybasedmodel)$adj.r.squared


confint(backwardmodel)

lm.beta(backwardmodel)


anova(theorybasedmodel, backwardmodel)


#### AIC

AIC(theorybasedmodel)
AIC(backwardmodel)


vif( mod = mod_pain_back1)


plot()


########################################


data1 =
read.csv("https://raw.githubusercontent.com/kekecsz/PSYP13_Data_analysis_clas
s-2018/master/home_sample_2.csv")


View(data1)
summary(data1)


data2 <- data1[-c(123, 113),]


describe(data2)
View(data2)


#Calculate predicted values


pred_test <- predict(theorybasedmodel, data2)
pred_test_back <- predict(backwardmodel, data2)


pred_test <- predict(theorybasedmodel, data2)
pred_test_back <- predict(backwardmodel, data2)


View(pred_test)
View(pred_test_back)


# calculate sum of squared residuals


RSS_test = sum((data2[, "pain"] - pred_test)^2)
RSS_test_back = sum((data2[, "pain"] - pred_test_back)^2)
RSS_test
RSS_test_back
```

```
plot(RSS_test)

############################## Assignment 3 ################################
library(psych) # for describe
library(reshape2) # for melt function
library(ggplot2) # for ggplot
library(cAIC4) # for cAIC
library(r2glmm) # for r2beta
library(psych) # for describe
library(ggplot2) # for ggplot
library(cAIC4) # for cAIC
library(r2glmm) # for r2beta
library(lme4) # for lmer
library(lmerTest) # to get singificance test in lmer
library(psych) # for pairs.panels
library(ggplot2) # for ggplot
library(reshape2) # for melt function
library(influence.ME) # for influence (this will also load the lme4 package)
library(lattice) # for qqmath

setwd("~/LUND")

#######################
#2.2 Custom functions
#######################

# This is a function to extract standardized beta coefficients from linear
mixed models.
#This function was adapted from:
https://stackoverflow.com/questions/25142901/standardized-coefficients-for-
lmer-model
##############
stdCoef.merMod <- function(object) {
  sdy <- sd(getME(object, "y"))
  sdx <- apply(getME(object, "X"), 2, sd)
  sc <- fixef(object) * sdx/sdy
  se.fixef <- coef(summary(object))[, "Std. Error"]
  se <- se.fixef * sdx/sdy
  return(data.frame(stdcoef = sc, stdse = se))
}


data_sample_1 =
read.csv("https://raw.githubusercontent.com/kekecsz/PSYP13_Data_analysis_clas
s-2018/master/home_sample_3.csv")
#assigning time and ID as a grouping factors
######################################################
#data_sample_1$time = factor(data_sample_1$time)
#data_sample_1$ID = factor(data_sample_1$ID)
####################################################


View(data_sample_1)
summary(data_sample_1)


# descriptives
describe(data_sample_1)
table(data_sample_1[, "sex"])
# histograms
hist(data_sample_1$pain1)
hist(data_sample_1$pain2)
hist(data_sample_1$pain3)
hist(data_sample_1$pain4)
hist(data_sample_1$age)
hist(data_sample_1$STAI_trait)
hist(data_sample_1$pain_cat)
hist(data_sample_1$cortisol_serum)
hist(data_sample_1$mindfulness)
hist(data_sample_1$weight)


# Maybe due to small sample but variables dont seem to be normally
distributed, apparently this does not matter since there is still a lot of
data points.



# designate which are the repeated varibales
```

```
repeated_variables = c("pain1", "pain2", "pain3", "pain4")

View(repeated_variables)


# correlation of repeated variables

cor(data_sample_1[, repeated_variables])


#as expected, the pain measures are highly correlated, especially the first
one with the second and third, the second with the thrid and the third with
the fourth. So those variables close in time have similar values.

#data is clustered within participants, different ratings of pain are not
independent of each other.


###############################

#from wide to long format

##################################


data_sample_long = melt(data_sample_1, measure.vars = repeated_variables,
variable.name = "time", value.name = "pain")


# order data frame by participant ID(not necessary, just makes the dataframe
look more intuitive)

data_sample_long = data_sample_long[order(data_sample_long[,"ID"]),]


#data_sample_long$time = factor(data_sample_long$time) #


# change the time variable to a numerical vector (only if time is set as a
factor)

data_sample_long$time = as.numeric(data_sample_long$time)


View(data_sample_long)


#fit a model including the fixed effects of age, sex, STAI, pain
catastrophizing, mindfulness, serum cortisol, time (day of pain report), and
a random intercept for participant ID (variable called'ID' in the dataset).
Now run a new model containing the same fixed effects, but also including a
random slope over time for each participant (this model should also contian a
random intercept).


#########################################################

########## Building the models #########################

#########################################################
```

```
#Note that the random intercept model means that we suspect that the each
participant is different in their overall pain rating (or baseline pain
rating), but that the effect of the fixed effect predictors is the same for
each participant. On the other hand, the random slope model not only baseline
pain will be different across participants, but also that the fixed effects
will be different from participant to participant as well.


mod_pain_int = lmer(pain ~ age + sex + STAI_trait + pain_cat + mindfulness +
cortisol_serum + time + (1 | ID), data = data_sample_long)


mod_pain_slope = lmer(pain ~ age + sex + STAI_trait + pain_cat + mindfulness
+ cortisol_serum + time + (time | ID), data = data_sample_long)

####OBS error term "singluar fit" comes up###


#First, let's visualize the predictions. For this we will have to save the
predicted values into new variables, then, we can visualize the predicted
values and the actual observations for each participant separately for both
the random intercept and the random slope model. (We create a new copy of the
data object so that our long format data can remain unharmed.)


data_sample_long_withpreds = data_sample_long

data_sample_long_withpreds$pred_int = predict(mod_pain_int)

data_sample_long_withpreds$pred_slope = predict(mod_pain_slope)


# random intercept model

ggplot(data_sample_long_withpreds, aes(y = pain, x = time, group = ID)) +
geom_point(size = 3) + geom_line(color = "red", aes(y = pred_int, x = time))
+ facet_wrap(~ID, ncol = 5)


# random slope and intercept model

ggplot(data_sample_long_withpreds, aes(y = pain, x = time, group = ID)) +
geom_point(size = 3) + geom_line(color = "red", aes(y = pred_slope, x =
time)) + facet_wrap(~ID, ncol = 5)


#Furthermore, we can compare the cAIC of the two models and use the
likelihood ratio test with the anova()function to get further information
about the model fit of the two models in comparison to each other.


cAIC(mod_pain_int)$caic


cAIC(mod_pain_slope)$caic


anova(mod_pain_int, mod_pain_slope)
```

```
######OBS based on ANOVA should we be using the slope model in the quadratic
model?


##########################

# Adding a quadratic term

#########################

#Let's add the quadratic term of time to the model random intercept model to
account for this non-linear relationship.


mod_pain_int_quad = lmer(pain ~ age + sex + STAI_trait + pain_cat +
mindfulness + cortisol_serum + time + I(time^2) + (1 | ID), data =
data_sample_long)


mod_pain_slope_quad = lmer(pain ~ age + sex + STAI_trait + pain_cat +
mindfulness + cortisol_serum + time + I(time^2) + (time | ID), data =
data_sample_long)


#And add the predictions to the new dataframe containing the other predicted
values as well.

data_sample_long_withpreds$pred_int_quad = predict(mod_pain_int_quad)

data_sample_long_withpreds$pred_slope_quad = predict(mod_pain_slope_quad)


#Now we can compare the model fit of the random intercept model containing
the quadratic effect of time with the random intercept model without it. As
usual, we use visualization, cAIC and the likelihood ratio test.


plot_quad1 = ggplot(data_sample_long_withpreds, aes(y = pain, x = time, group
= ID)) + geom_point(size = 3) + geom_line(color = "red", aes(y =
pred_int_quad, x = time)) + facet_wrap(~ID, ncol = 5)

plot_quad1


plot_quad = ggplot(data_sample_long_withpreds, aes(y = pain, x = time, group
= ID)) + geom_point(size = 3) + geom_line(color = "red", aes(y =
pred_slope_quad, x = time)) + facet_wrap(~ID, ncol = 5)

plot_quad


cAIC(mod_pain_int)$caic

cAIC(mod_pain_slope)$caic




cAIC(mod_pain_int_quad)$caic
cAIC(mod_pain_slope_quad)$caic




anova(mod_pain_int, mod_pain_int_quad)

anova(mod_pain_slope, mod_pain_slope_quad)


anova(mod_pain_slope_quad, mod_pain_int_quad)


#### Adding quadratic function seems to make the model fit better, i.e taking
into account the nonlinear relationship between pain and time

#Since we entered time's quadratic term into the model, we can expect
problems with multicollinearity. As seen in the exercise on model
diagnostics, we can avoid this problem by centering the variable time, this
way removing the correlation of time and time^2.Let's do this now and refit
our model with the centered time and its quadratic term as predictors.


data_sample_long_centered_time = data_sample_long


data_sample_long_centered_time$time_centered =
data_sample_long_centered_time$time -
mean(data_sample_long_centered_time$time)


mod_pain_int_quad = lmer(pain ~ age + sex + STAI_trait + pain_cat +
mindfulness + cortisol_serum + time_centered + I(time_centered^2) + (1 | ID),
data = data_sample_long_centered_time)

mod_pain_slope_quad = lmer(pain ~ age + sex + STAI_trait + pain_cat +
mindfulness + cortisol_serum + time_centered + I(time_centered^2) +
(time_centered | ID), data = data_sample_long_centered_time)




# Now we can request the reportable results the same way we did in the
previous exercise.

# Marginal R squared

r2beta(mod_pain_int_quad, method = "nsj", data =
data_sample_long_centered_time)

r2beta(mod_pain_slope_quad, method = "nsj", data =
data_sample_long_centered_time)




# Conditional AIC

cAIC(mod_pain_int_quad)$caic

cAIC(mod_pain_slope_quad)$caic
```

```
# Model coefficients
summary(mod_pain_int_quad)
summary(mod_pain_slope_quad)

# Confidence intervals for the coefficients
confint(mod_pain_int_quad)
confint(mod_pain_slope_quad)

# standardized Betas
stdCoef.merMod(mod_pain_int_quad)
stdCoef.merMod(mod_pain_slope_quad)

#####################
#Running a model diagnostic
#####################

# Best model is
# mod_pain_slope_quad = lmer(pain ~ age + sex + STAI_trait + pain_cat +
mindfulness + cortisol_serum + time_centered + I(time_centered^2) +
(time_centered | ID), data = data_sample_long_centered_time)

data_sample_long_with_resid = data_sample_long_centered_time
data_sample_long_with_resid$resid = residuals(mod_pain_slope_quad)

##Influential outliers
# Checking if model changes with leaving any participant out

influence_observation = influence(mod_pain_slope_quad, obs = T)$alt.fixed #
this can take a minute or so
influence_group = influence(mod_pain_slope_quad, group = "ID")$alt.fixed

boxplot(influence_observation[, "time_centered"])

boxplot(influence_observation[, "cortisol_serum"])
boxplot(influence_observation[, "mindfulness"])

boxplot(influence_observation[, "pain_cat"])

boxplot(influence_observation[, "STAI_trait"])

boxplot(influence_observation[, "age"])

#pred_names = colnames(influence_group)

#source("GraphPlots.R")
#openGraph()
#par(mfrow = c(1, length(pred_names)))
#for (i in 1:length(pred_names)) {
#  boxplot(influence_observation[, pred_names[i]], main = pred_names[i])
#}

#par(mfrow = c(1, 1))

#openGraph()
#for (i in 1:length(pred_names)) {
#  boxplot(influence_group[, pred_names[i]], main = pred_names[i])
#}

#par(mfrow = c(1, 1))

## Normality
openGraph()
qqmath(mod_pain_slope_quad, id = 0.05)

openGraph()
qqmath(ranef(mod_pain_slope_quad))

## Linearity

openGraph()
```

```
plot(mod_pain_slope_quad, arg = "pearson")


openGraph()

plot(resid ~ time_centered, data = data_sample_long_with_resid)


# same as zoltans, may be of use to add a cubic term of time?


openGraph()

plot(resid ~ cortisol_serum, data = data_sample_long_with_resid)


openGraph()

plot(resid ~ mindfulness, data = data_sample_long_with_resid)


# can continue for all variables i guess


## Homoscedasticity

openGraph()

plot(mod_pain_slope_quad, arg = "pearson")


homosced_mod = lm(data_sample_long_with_resid$resid^2 ~
data_sample_long_with_resid$ID)

summary(homosced_mod)


# calculate interquartile range within each cluster

IQR_of_residuals_by_participant = sapply(split(data_sample_long_with_resid, f
= data_sample_long_with_resid$ID), function(x) IQR(x$resid))


# rank ordering them

rank = rank(IQR_of_residuals_by_participant)


# adding rank to the dataframe containing the residuals

data_sample_long_with_resid$rank = rep(rank, each =
length(repeated_variables))

# creating a vector of participant IDs ordered based on the

# rank, this will be used as labels

IDforplot =
unique(data_sample_long_with_resid$ID[order(data_sample_long_with_resid$rank)
])
```

```
# create the plot

openGraph()

ggplot(data_sample_long_with_resid, aes(y = resid, x = factor(rank),labels =
ID)) + geom_boxplot() + scale_x_discrete(labels = IDforplot) + coord_flip()


# Multicollinearity

openGraph()

pairs.panels(data_sample_long_centered_time[, c("time_centered", "age",
"sex", "STAI_trait", "pain_cat", "cortisol_serum", "mindfulness")], col =
"red", lm = T)
```