1. Heatmap

library("gplots")

setwd("U:/hasan/Privat/PCR\_Sus\_Scrofa/R")

x <- read.csv("VitaD\_New.csv")

rownames(x)<-x[,1] # First row is labeled as the name of the row

x<-x[,-1] # To delete the repeated column you will see on console

### Filtering, criteria?? below 5 = NA, tissues with less then 3 observations

x[x < 5] <- NA

### mean, with NA or 0?

tis<-unique(sub("[0-9]","",colnames(x)))

a <- matrix(nrow = dim(x)[1], ncol = length(tis))

colnames(a)<-tis

rownames(a)<-rownames(x)

for (i in tis){

a[,i]<-rowMeans(x[,grep(i,colnames(x))],na.rm=TRUE)

}

alog<-log(a,2)

heatmap.2(as.matrix(alog), main = "Sample", trace = "none")

data.matrix(x)

yb <- colorRampPalette(c("yellow", "blue"))

heatmap.2(log(a,2), col= yb, main = "Sample", trace = "none", margins = c(10,12), cexRow=0.5)

1. Import data to xlsx

install.packages("openxlsx")

library(openxlsx)

write.xlsx(x, “mydata. Xlsx”)

1. Deleting dendogram

heatmap(mtscaled, Colv=NA, Rowv=NA, scale='none')

1. Heatmap (Without dendogram and changing title margin)

iinstall.packages("gplots")

install.packages("openxlsx")

library("openxlsx")

library("gplots")

setwd("Z:/30Deser/user/hasan/Privat/PCR\_Sus\_Scrofa\_V4/R")

x <- read.csv("Z:/30Deser/user/hasan/Privat/PCR\_Sus\_Scrofa\_V4/R/VitaD\_Data\_Finals\_New2.csv")

rownames(x)<-x[,1] # First row is labeled as the name of the row

x<-x[,-1] # To delete the repeated column you will see on console

### Filtering, criteria?? below 5 = NA, tissues with less then 3 observations

x[x < 10] <- NA

### mean, with NA or 0?

tis<-unique(sub("[0-9]","",colnames(x)))

a <- matrix(nrow = dim(x)[1], ncol = length(tis))

colnames(a)<-tis

rownames(a)<-rownames(x)

for (i in tis){

a[,i]<-rowMeans(x[,grep(i,colnames(x))],na.rm=TRUE)

}

alog<-log(a,2)

heatmap.2(as.matrix(alog),col= yb, main = "Gene expression", na.color="white", trace = "none", Rowv=NA, Colv=NA, scale='none', lhei=c(1.1,5), margins = c(9,7))

yb <- colorRampPalette(c("red", "yellow"))

write.xlsx(alog, "Gene\_Expression\_Data.xlsx")

#str (x) is used to see the list on console. You should type it there.

#Normalization of data using housekeeping gene

library(openxlsx)

library(qvalue)

library(lme4)

library(lmerTest)

library(car)

setwd("~/PCR\_Sus\_Scrofa\_V7/R")

outlier<-"n" # y or n

dat<-read.csv("~/PCR\_Sus\_Scrofa\_V7/LightCycler480/qPCR\_Result.csv") #edited file

dat<-dat[,c(1,5,6,7)]

dat$gene<-sub("\_.\*","",dat$Experiment.Name)

dat<-dat[!dat$SampleName=="NC",]

dat<-dat[-grep("std.\*",dat$SampleName),]

dat<-dat[,c(5,2,3,4)]

colnames(dat)<-c("gene","Sample","CP","Conc")

data<-dat[,-3]

data<-data[order(data[,1],data[,2]),]

data1 <- data[seq\_along(data[,1]) %% 2==0,]

data1$ID<-paste0(data1$gene,"x",data1$Sample)

data2 <- data[!seq\_along(data[,1]) %% 2==0,]

data2$ID<-paste0(data2$gene,"x",data2$Sample)

dat.comb<-merge(data1,data2,by="ID")

dat.comb<-dat.comb[,c(3,2,4,7)]

colnames(dat.comb)<-c("Sample","gene","rep1","rep2")

a<-split(dat.comb,dat.comb$gene)

a<-lapply(a, function (x) {colnames(x)[c(3,4)]<-c(paste0(x[1,2],1),paste0(x[1,2],2));x})

a<-lapply(a, function(x) { x["gene"] <- NULL; x })

res<-Reduce(function(x, y) merge(x, y, all=TRUE,by="Sample"), a)

pheno<-cbind.data.frame(Sample=res$Sample,tissue=gsub("[0-9]+","",res$Sample),ID=sub("[A-z]+","",res$Sample))

dat<-merge(pheno,res,by.x="Sample",by.y="Sample")

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

### Edit!!! ###

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

datum <- c("12012021"); #date

housekeeping\_genes <- c("RPL32")

ErstesTranscript <- "CYP24A1"; ### give name of first gene (excluding HK)

norm\_method <- "per\_group" ### "per\_group", or "all"

normgroup <- "tissue" ### if "per\_group" provide the group

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

### reorder columns that HK come first ###

Tieranzahl = dim(dat)[1]; #number of samples

pos\_HK<-grep(housekeeping\_genes[1],colnames(dat))

pos\_gene<-grep(ErstesTranscript[1],colnames(dat))

if (pos\_HK[1]<pos\_gene[1]){

dat<-dat[,c(1:(min(pos\_HK)-1),pos\_HK,(max(pos\_HK)+1):dim(dat)[2])]

} else

dat<-dat[,c(1:(min(pos\_gene)-1),pos\_HK,pos\_gene,(max(pos\_gene)+1):(min(pos\_HK)-1),(max(pos\_HK)+1):dim(dat)[2])]

colnames(dat)<-gsub("[0-9]$","",colnames(dat))

housekeeping\_genes\_out <- c()

if(length(housekeeping\_genes\_out)>0){

dat <- dat[,-grep(housekeeping\_genes\_out, colnames(dat))]

} else

rm(housekeeping\_genes\_out)

pos <- match(housekeeping\_genes[1], colnames(dat))

#outlier test

library(outliers)

library(ggplot2)

grubbs.flag <- function(x) {

outliers <- NULL

test <- x

grubbs.result <- grubbs.test(test)

pv <- grubbs.result$p.value

while(pv < 0.05) {

outliers <- c(outliers,as.numeric(strsplit(grubbs.result$alternative," ")[[1]][3]))

test <- x[!x %in% outliers]

grubbs.result <- grubbs.test(test)

pv <- grubbs.result$p.value

}

return(data.frame(X=x,Outlier=(x %in% outliers)))

}

dat.outtest <- dat[,c(pos:dim(dat)[2])]

dat.outtest2a <- dat.outtest[,seq(1,by=2,ncol(dat.outtest))]

dat.outtest2b <- dat.outtest[,seq(2,by=2,ncol(dat.outtest))]

colnames(dat.outtest2b) <- colnames(dat.outtest2a)

dat.outtest <- rbind(dat.outtest2a, dat.outtest2b)

dat.outtest.list <- apply(dat.outtest, 2, grubbs.flag)

dat.out2 <- data.frame(ID=c(1:Tieranzahl))

for(i in c(1:dim(dat.outtest)[2])) {

truefalse <- as.data.frame(dat.outtest.list[i])[,2]

dat.out2 <- cbind(dat.out2, truefalse)

colnames(dat.out2)[i+1] <- colnames(dat.outtest)[i]

}

dat.out2 <- cbind(dat.out2[1:(nrow(dat.out2)/2),],dat.out2[-(1:(nrow(dat.out2)/2)),])[,c(matrix(data = 1:(ncol(dat.out2)\*2), nrow=2, byrow=TRUE))]

dat.out2 <- dat.out2[,-2]

#dat.out2

#write.table(dat.out2, paste(datum,"qPCR\_outlier.txt",sep="\_"), col.names=TRUE, row.names=FALSE, quote=FALSE,sep='\t')

#### remove outlier #####

# no outlier found for u, we skip this part ...

if (outlier=="y"){ # something is missing here ... outlier is not defined correctly

df2<-cbind("ID"=dat.out2[,1],dat[,c(pos:dim(dat)[2])])

remov<-which(dat.out2=="TRUE",arr.ind=T)

if (dim(remov)[1]>0) {

for (i in 1:dim(remov)[1])

df2[remov[i,1],remov[i,2]]<-NA

dat<-cbind(dat[,c(1:pos-1)],df2[,c(2:dim(df2)[2])])

}else { dat<-dat}

}

######

#Generate means per transcript from replicates

dat2 <- data.frame(ID=c(1:Tieranzahl))

for(i in c(1: c(length(pos:dim(dat)[2])/2) )){

Means <- rowMeans(dat[,(2\*i+pos-2):(2\*i+pos-1)],na.rm=TRUE)

dat2 <- cbind(dat2, Means)

colnames(dat2)[i+1] <- colnames(dat[(2\*i+pos-2)])

}

dat2 <- cbind(dat[,(1:c(pos-1))], dat2[,2:dim(dat2)[2]])

ui<-sapply(dat2,is.nan)

dat2[ui]<-NA

#dat2

#mean per group or over all samples using housekeeping genes

dat3 <- dat2

if(norm\_method == "per\_group") {

for(i in match(housekeeping\_genes, colnames(dat2))){ # Mittelwertberechnungen per Gruppe für housekeeping\_genes

means=sapply(split(dat3[1:Tieranzahl,i], dat3[1:Tieranzahl, match(normgroup, colnames(dat2))]), mean, na.rm=TRUE)

dat3[,i]=means[dat3[,match(normgroup, colnames(dat2))]]

}

} else

for(i in match(housekeeping\_genes, colnames(dat2))){ # Mittelwertberechnungen über alle für housekeeping\_genes

means=mean(dat3[1:Tieranzahl,i],na.rm=TRUE)

dat3[,i]=means

}

#dat3

#geometric mean of averaged values based on housekeeping genes

norm\_fac1 <- dat2[,match(housekeeping\_genes, colnames(dat2))]/dat3[,match(housekeeping\_genes, colnames(dat3))]

norm\_fac2 <- c(1:Tieranzahl)

if(length(housekeeping\_genes)>1){

norm\_fac2=apply(norm\_fac1, 1, function(x) exp(mean(log(x),na.rm=TRUE))) #geometric.mean(norm\_fac[i,]) per row

}else

norm\_fac2=norm\_fac1

#divid target genes by the geometric mean of housekeeping\_genes, rename +combine +transpose colnames/rownames

qPCR <- dat2[,match(ErstesTranscript, colnames(dat2)):dim(dat2)[2]] / norm\_fac2

colnames(qPCR) <- paste(colnames(qPCR),"qPCR",sep = "\_")

rownames(qPCR) <- dat2[,1]

q.stat<-qPCR

qPCR <- as.data.frame(t(qPCR))

qPCR <- cbind(sub("\_qPCR", "", rownames(qPCR)), qPCR)

colnames(qPCR)[1] <- "gene"

write.csv(qPCR, paste0(paste(datum,"qPCR\_VitD", norm\_method, normgroup, paste(housekeeping\_genes, collapse="\_"),sep="\_"),".csv"))

qPCR

write.xlsx (qPCR, "Expression.xlsx" )

#### END ####

pdata<-read.csv("pheno\_piglet\_PC.csv")

#### preparation for linear model #####

q.stat<-qPCR

q.stat<-log2(q.stat) ###log transform ???

qRT<-merge(dat[,c(1:3)],q.stat,by.x="Sample",by.y=0,order=FALSE)

qRT<-merge(qRT,pdata,by="ID",order=FALSE)

## loop starts from here

#effect<-c("Diet.Piglet","Sex","Slaughter.order") #colnames(qRT2)

effect<-c("Diet.Piglet","Slaughter.order") #colnames(qRT2)

out.all <- data.frame()

for (g in 1:length(unique(dat$tissue))) {

tissue<-unique(dat$tissue)[g] #different tissues 1,2...

Index <- grep(tissue, qRT$tissue)

q.stat2<-qRT[Index,4:12]

qRT2<-qRT[Index,c(1,3,14:21)]

qRT2$Diet.Piglet<-sub("[0-9]+","",qRT2$Diet.Piglet)

qRT2$Diet.Piglet <- as.factor(qRT2$Diet.Piglet)

qRT2$Sow\_ID <- as.factor(qRT2$Sow\_ID)

q.stat2<-q.stat2[colSums(!is.na(q.stat2)) > 0]

qRT2$batch <- as.factor(qRT2$batch)

for (i in 1:dim(q.stat2)[2]) {

#lm.i<-lmer(paste0("q.stat2[,i]~", paste(effect,collapse="+")),na.action=na.omit,data=qRT2)

lm.i<-lm(paste0("q.stat2[,i]~", paste(effect,collapse="+")),na.action=na.omit,data=qRT2)

#lm.i<-lm(q.stat2[,i] ~ Diet.Piglet,na.action=na.omit,data=qRT2)

z<-aggregate(q.stat2[,i] ~ qRT2$Diet.Piglet, q.stat2, function(x) cbind(M = mean(x), SE = sd(x)/sqrt(length(x))))

z1<-cbind.data.frame(colnames(q.stat2)[i],z[1,2],z[2,2],Anova(lm.i,type=3)[2,4])

colnames(z1)<-c("gene","high\_P(m)","high\_P(SE)","low\_P(m)","low\_P(SE)","p-value")

z1<-as.data.frame(z1)

if (i==1) {

q.sig<-z1

}else {

q.sig<-rbind.data.frame(q.sig,z1,stringsAsFactors = FALSE)

}}

#cat("\n","tissue:",tissue," outlier? ",outlier,"\n")

#print(q.sig)

q.sig <- cbind.data.frame(sub("\_qPCR", "", q.sig$gene), q.sig)

rownames(q.sig)<-q.sig$gene

q.sig<-q.sig[-2]

colnames(q.sig)[1] <- "gene"

q.sig$new<-((q.sig[2]-q.sig[4])/abs(q.sig[2]-q.sig[4]))\*2^(abs(q.sig[2]-q.sig[4]))

FC.qPCR<-q.sig[,c(1,7)]

colnames(FC.qPCR) <- c("gene","FCqPCR")

pqVals <- p.adjust(q.sig[,6],"fdr")

qPCRout<-cbind("gene"=FC.qPCR[,1],tissue,"FC"=q.sig$new$`high\_P(m)`,q.sig[,2:6],"q-value"=pqVals)

out.all<-rbind.data.frame(out.all,qPCRout)

}

subset(out.all,out.all$`p-value`<=0.1)

write.table(out.all, paste(datum,paste0(paste(housekeeping\_genes, collapse="\_"), ".csv"),sep="\_"), col.names=TRUE, row.names=FALSE, quote=FALSE,sep=',')

# Boxplot for all 9 genes

library(ggplot2)

library(magrittr)

library(dplyr)

library(forcats)

setwd("Z:/30Deser/user/wubuli/Austausch/PCR\_Sus Scrofa/LightCycler480")

#dat<-read.csv("Z:/30Deser/user/wubuli/Austausch/PCR\_Sus Scrofa/LightCycler480/qpcr-norm2.csv")

dat2<-read.csv("Z:/30Deser/user/wubuli/Austausch/PCR\_Sus Scrofa/LightCycler480/050918\_norm-qPCR-means\_all\_\_RPL32.csv",stringsAsFactors = FALSE)

#dat2<-read.csv("Z:/30Deser/user/wubuli/Austausch/PCR\_Sus Scrofa/LightCycler480/10052019\_norm-H-L pig qPCR-means\_per\_group\_tissue\_RPL32.csv",stringsAsFactors = FALSE)

rownames(dat2)<-dat2[,2]

dat<-t(dat2[,c(-1,-2)])

rownames(dat)<-gsub("[.]"," ",rownames(dat))

ID<-gsub("^[A-z]+ ","",rownames(dat))

ID<-gsub("[A-z]+ ","",rownames(dat))

pheno<-cbind.data.frame(Sample=rownames(dat),Tissue=gsub(" [0-9]+","",rownames(dat)),ID=ID)

dat<-merge(pheno,dat,by.x="Sample",by.y=0)

## Order by default

##dat[is.na(dat)]<-0

#ggplot(data=subset(dat,!is.na(dat)),aes(x=tissue, y=SLC34A3\_qPCR)) + stat\_boxplot(geom ='errorbar') + geom\_dotplot(binaxis ="y",binwidth = 1.0, stackdir = "center", dotsize = 0.4) +

# geom\_boxplot(color="blue",fill="blue", alpha=0.2, outlier.colour = "red", outlier.fill = "red", outlier.size=2) +

# stat\_summary(fun.y = mean, geom="point", shape=20, size=3, color="green", fill="green") +

## stat\_summary(fun.data = "mean\_cl\_normal", aes(shape="mean"), colour = "red", geom="point") +

## scale\_shape\_manual("", values=c("mean"="x")) +

# theme(axis.text.x=element\_text(angle = 60, hjust = 1)) +

# xlab("Tissues") +ylab("Transcript Copy Numbers") +

# ggtitle("SLC34A3 gene Expression plot") + annotation\_custom(my\_grob) +

# ylim(0,30)

library(grid)

my\_text <- "Red dot: Outlier \n Green dot: Mean"

my\_grob = grid.text(my\_text, x=0.9, y=0.9, gp=gpar(col="black", fontsize=8, fontface="bold"))

## Order Using median

#dat[is.na(dat)]<-0

#dat %>%

# mutate(Tissue = fct\_reorder(Tissue, SLC17A1, fun=median)) %>%

#ggplot(aes(x=reorder(Tissue, SLC17A1), y=SLC17A1)) +

# geom\_boxplot(color="blue",fill="blue", alpha=0.2,outlier.colour = "red",

# outlier.fill="red", outlier.size=2) + stat\_boxplot(geom ='errorbar') +

# stat\_summary(fun.y=mean, geom="point", shape=20, size=3, color="green", fill="green") +

# theme(axis.text.x=element\_text(angle = 60, hjust = 1)) +

# xlab("Tissues") +ylab("Transcript Copy Numbers") + ggtitle("SLC17A1 gene Expression plot")+

# ylim(0,20) + annotation\_custom(my\_grob)

# Reorder following a precise order(by name)

#names<-c(rep("Aorta",5),rep("bone",5),rep("caecum",5),rep("colon.d",5),rep("colon.p",5),rep("duod",5),rep("ileum",5),rep("jeju.d",5),rep("jeju.p",5),rep("ki.cort",5),rep("ki.med",5),rep("liver",5),rep("lung",5),rep("stom",4))

#dat[is.na(dat)]<-0

dat$Tissue<-factor(dat$Tissue,levels=c("ki cort","ki med","liver","stom","duod","jeju p","jeju d","ileum","caecum","colon p","colon d","bone","lung","Aorta","Muscle"))

dat.org<-dat

dat.org <- dat.org[order(as.numeric(dat.org$Tissue)),]

dat<-dat.org

for (i in 4:12){

cat(paste0(colnames(dat)[i]))

tmp <- na.omit(sapply(split(dat[,i], dat$Tissue), mean, na.rm=T)) # calculates mean (omitting NAs)

dat[dat$Tissue %in% names(tmp)[tmp<=4],i] <-NA # substitutes mean<10 vals by NAs

}

write.csv(dat,file = "qPCR Normalized data\_larger4\_trial2.csv") # saves output!

#write table for every gene (or combine)

#png(paste0(colnames(dat)[i],".png"),units="in", width=5, height=5, res=300)

print(ggplot(data=dat,aes(dat$tissue,y=dat[,i])) +

geom\_boxplot(color="blue",fill="blue", alpha=0.2,outlier.colour = "red",

outlier.fill="red", outlier.size=2) + stat\_boxplot(geom ='errorbar',width=0.2) +

stat\_summary(fun.y=mean, geom="point", shape=20, size=3, color="green", fill="green") +

theme(axis.text.x=element\_text(angle = 60, hjust = 1)) +

xlab("Tissues") +ylab("Transcript Copy Numbers") + ggtitle(paste0(colnames(dat)[i]," gene Expression plot"))+

scale\_y\_log10(breaks=c(10,100,1000,10000,33000)) + annotation\_custom(my\_grob))

#dev.off()

#}

dat[is.na(dat)]<-0

dat %>%

mutate(Tissue = fct\_reorder(Tissue, SLC17A1, fun=mean)) %>%

ggplot(aes(x=reorder(Tissue, SLC17A1), y=SLC17A1)) +

geom\_boxplot(color="blue",fill="blue", alpha=0.2,outlier.colour = "red",

outlier.fill="red", outlier.size=2) + stat\_boxplot(geom ='errorbar') +

stat\_summary(fun.y=mean, geom="point", shape=20, size=3, color="green", fill="green") +

theme(axis.text.x=element\_text(angle = 60, hjust = 1)) +

xlab("Tissues") +ylab("Transcript Copy Numbers") + ggtitle("SLC17A1 gene Expression plot")+

scale\_y\_log10(breaks=c(10,25, 50, 75, 100, 150, 50000)) + annotation\_custom(my\_grob)

#Normalization of data using housekeeping gene (Modified\_Data\_Gene\_Expression\_qPCR only)

library(openxlsx)

library(qvalue)

library(lme4)

library(lmerTest)

library(car)

setwd("~/PCR\_Sus\_Scrofa\_V7/R")

outlier<-"n" # y or n

dat<-read.csv("~/PCR\_Sus\_Scrofa\_V7/LightCycler480/qPCR\_Result.csv") #edited file

dat<-dat[,c(1,5,6,7)]

dat$gene<-sub("\_.\*","",dat$Experiment.Name)

dat<-dat[!dat$SampleName=="NC",]

dat<-dat[-grep("std.\*",dat$SampleName),]

dat<-dat[,c(5,2,3,4)]

colnames(dat)<-c("gene","Sample","CP","Conc")

data<-dat[,-3]

data<-data[order(data[,1],data[,2]),]

data1 <- data[seq\_along(data[,1]) %% 2==0,]

data1$ID<-paste0(data1$gene,"x",data1$Sample)

data2 <- data[!seq\_along(data[,1]) %% 2==0,]

data2$ID<-paste0(data2$gene,"x",data2$Sample)

dat.comb<-merge(data1,data2,by="ID")

dat.comb<-dat.comb[,c(3,2,4,7)]

colnames(dat.comb)<-c("Sample","gene","rep1","rep2")

a<-split(dat.comb,dat.comb$gene)

a<-lapply(a, function (x) {colnames(x)[c(3,4)]<-c(paste0(x[1,2],1),paste0(x[1,2],2));x})

a<-lapply(a, function(x) { x["gene"] <- NULL; x })

res<-Reduce(function(x, y) merge(x, y, all=TRUE,by="Sample"), a)

pheno<-cbind.data.frame(Sample=res$Sample,tissue=gsub("[0-9]+","",res$Sample),ID=sub("[A-z]+","",res$Sample))

dat<-merge(pheno,res,by.x="Sample",by.y="Sample")

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

### Edit!!! ###

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

datum <- c("12012021"); #date

housekeeping\_genes <- c("RPL32")

ErstesTranscript <- "CYP24A1"; ### give name of first gene (excluding HK)

norm\_method <- "per\_group" ### "per\_group", or "all"

normgroup <- "tissue" ### if "per\_group" provide the group

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

### reorder columns that HK come first ###

Tieranzahl = dim(dat)[1]; #number of samples

pos\_HK<-grep(housekeeping\_genes[1],colnames(dat))

pos\_gene<-grep(ErstesTranscript[1],colnames(dat))

if (pos\_HK[1]<pos\_gene[1]){

dat<-dat[,c(1:(min(pos\_HK)-1),pos\_HK,(max(pos\_HK)+1):dim(dat)[2])]

} else

dat<-dat[,c(1:(min(pos\_gene)-1),pos\_HK,pos\_gene,(max(pos\_gene)+1):(min(pos\_HK)-1),(max(pos\_HK)+1):dim(dat)[2])]

colnames(dat)<-gsub("[0-9]$","",colnames(dat))

housekeeping\_genes\_out <- c()

if(length(housekeeping\_genes\_out)>0){

dat <- dat[,-grep(housekeeping\_genes\_out, colnames(dat))]

} else

rm(housekeeping\_genes\_out)

pos <- match(housekeeping\_genes[1], colnames(dat))

#outlier test

library(outliers)

library(ggplot2)

grubbs.flag <- function(x) {

outliers <- NULL

test <- x

grubbs.result <- grubbs.test(test)

pv <- grubbs.result$p.value

while(pv < 0.05) {

outliers <- c(outliers,as.numeric(strsplit(grubbs.result$alternative," ")[[1]][3]))

test <- x[!x %in% outliers]

grubbs.result <- grubbs.test(test)

pv <- grubbs.result$p.value

}

return(data.frame(X=x,Outlier=(x %in% outliers)))

}

dat.outtest <- dat[,c(pos:dim(dat)[2])]

dat.outtest2a <- dat.outtest[,seq(1,by=2,ncol(dat.outtest))]

dat.outtest2b <- dat.outtest[,seq(2,by=2,ncol(dat.outtest))]

colnames(dat.outtest2b) <- colnames(dat.outtest2a)

dat.outtest <- rbind(dat.outtest2a, dat.outtest2b)

dat.outtest.list <- apply(dat.outtest, 2, grubbs.flag)

dat.out2 <- data.frame(ID=c(1:Tieranzahl))

for(i in c(1:dim(dat.outtest)[2])) {

truefalse <- as.data.frame(dat.outtest.list[i])[,2]

dat.out2 <- cbind(dat.out2, truefalse)

colnames(dat.out2)[i+1] <- colnames(dat.outtest)[i]

}

dat.out2 <- cbind(dat.out2[1:(nrow(dat.out2)/2),],dat.out2[-(1:(nrow(dat.out2)/2)),])[,c(matrix(data = 1:(ncol(dat.out2)\*2), nrow=2, byrow=TRUE))]

dat.out2 <- dat.out2[,-2]

#dat.out2

#write.table(dat.out2, paste(datum,"qPCR\_outlier.txt",sep="\_"), col.names=TRUE, row.names=FALSE, quote=FALSE,sep='\t')

#### remove outlier #####

# no outlier found for u, we skip this part ...

if (outlier=="y"){ # something is missing here ... outlier is not defined correctly

df2<-cbind("ID"=dat.out2[,1],dat[,c(pos:dim(dat)[2])])

remov<-which(dat.out2=="TRUE",arr.ind=T)

if (dim(remov)[1]>0) {

for (i in 1:dim(remov)[1])

df2[remov[i,1],remov[i,2]]<-NA

dat<-cbind(dat[,c(1:pos-1)],df2[,c(2:dim(df2)[2])])

}else { dat<-dat}

}

######

#Generate means per transcript from replicates

dat2 <- data.frame(ID=c(1:Tieranzahl))

for(i in c(1: c(length(pos:dim(dat)[2])/2) )){

Means <- rowMeans(dat[,(2\*i+pos-2):(2\*i+pos-1)],na.rm=TRUE)

dat2 <- cbind(dat2, Means)

colnames(dat2)[i+1] <- colnames(dat[(2\*i+pos-2)])

}

dat2 <- cbind(dat[,(1:c(pos-1))], dat2[,2:dim(dat2)[2]])

ui<-sapply(dat2,is.nan)

dat2[ui]<-NA

#dat2

#mean per group or over all samples using housekeeping genes

dat3 <- dat2

if(norm\_method == "per\_group") {

for(i in match(housekeeping\_genes, colnames(dat2))){ # Mittelwertberechnungen per Gruppe für housekeeping\_genes

means=sapply(split(dat3[1:Tieranzahl,i], dat3[1:Tieranzahl, match(normgroup, colnames(dat2))]), mean, na.rm=TRUE)

dat3[,i]=means[dat3[,match(normgroup, colnames(dat2))]]

}

} else

for(i in match(housekeeping\_genes, colnames(dat2))){ # Mittelwertberechnungen über alle für housekeeping\_genes

means=mean(dat3[1:Tieranzahl,i],na.rm=TRUE)

dat3[,i]=means

}

#dat3

#geometric mean of averaged values based on housekeeping genes

norm\_fac1 <- dat2[,match(housekeeping\_genes, colnames(dat2))]/dat3[,match(housekeeping\_genes, colnames(dat3))]

norm\_fac2 <- c(1:Tieranzahl)

if(length(housekeeping\_genes)>1){

norm\_fac2=apply(norm\_fac1, 1, function(x) exp(mean(log(x),na.rm=TRUE))) #geometric.mean(norm\_fac[i,]) per row

}else

norm\_fac2=norm\_fac1

#divid target genes by the geometric mean of housekeeping\_genes, rename +combine +transpose colnames/rownames

qPCR <- dat2[,match(ErstesTranscript, colnames(dat2)):dim(dat2)[2]] / norm\_fac2

colnames(qPCR) <- paste(colnames(qPCR),"qPCR",sep = "\_")

rownames(qPCR) <- dat2[,1]

q.stat<-qPCR

qPCR <- as.data.frame(t(qPCR))

qPCR <- cbind(sub("\_qPCR", "", rownames(qPCR)), qPCR)

colnames(qPCR)[1] <- "gene"

write.csv(qPCR, paste0(paste(datum,"qPCR\_VitD", norm\_method, normgroup, paste(housekeeping\_genes, collapse="\_"),sep="\_"),".csv"))

qPCR

write.xlsx (qPCR, "Expression.xlsx")