**Stat\_consulting\_RcodeM**

## Remove any existing dataset

## or analysis from your

## workspace

rm(list=ls())

options(warn=-1)

### Load libraries

require(data.table)

require(lubridate)

require(ggplot2)

### Change your directory

setwd("C:/Users/stuwe/Desktop/Stat analysis from stat lab")

### Read different squirrel file"

squirrel <- fread("mphys\_ra.csv",

data.table = F)

### manipulate date

squirrel$date <- as.Date(squirrel$date,format = '%m/%d/%Y')

squirrel$year <- year(squirrel$date)

squirrel$months <- month(squirrel$date, label = T)

squirrel$day <- day(squirrel$date)

## Log Transformation

names(squirrel)[c(4,6)] <- c('Estradiol','Progesterone')

squirrel$Estradiol <- log(squirrel$Estradiol)

squirrel$Progesterone <- log(squirrel$Progesterone)

### Split year to year

year2015 = squirrel[squirrel$year == 2015, -5]

year2016 = squirrel[squirrel$year == 2016, -5]

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

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

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

##

##

## Functions to analyze Data

##

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

## localMaxima : identifies peaks

## in the data

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

localMaxima <- function(x) {

# Use -Inf instead if x is numeric (non-integer)

y <- diff(c(-.Machine$integer.max, x)) > 0L

rle(y)$lengths

y <- cumsum(rle(y)$lengths)

y <- y[seq.int(1L, length(y), 2L)]

if (x[[1]] == x[[2]]) {

y <- y[-1]

}

y

}

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

## analyze\_yearly\_trends :

## - compares yearly estradiol

## and progesterone values

## - plots peaks and events

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

analyze\_yearly\_trends <- function(m1, lag=0, cutoff\_estradiol=2000,

cutoff\_progesterone=400, year='2015',

up\_pro=300,

bot\_pro=100,

up\_est=3000,

bot\_est=1500){

### Detect events

estra\_max <- numeric(nrow(m1))

estra\_max[localMaxima(m1$Estradiol)] <- 1

estra\_max[which(m1$Estradiol < log(cutoff\_estradiol))] <- 0

proge\_max <- numeric(nrow(m1))

proge\_max[localMaxima(m1$Progesterone)] <- 1

proge\_max[which(m1$Progesterone < log(cutoff\_progesterone))] <- 0

event <- numeric(nrow(m1))

for(i in 1:length(event)){

if( estra\_max[i]==1 & sum(proge\_max[i:(i+lag)])>0 &

day(m1$date[i+lag]) - day(m1$date[i]) <= lag){

#event[i:(i+lag)] <- 1

event[i] <- 1

}

}

max\_mat <- data.frame(estra\_max, proge\_max, event)

### Plot all events

plot(m1$date, m1$Progesterone, type='l', ylim=c(4,9), lwd=2, col='green',

main=paste('Estradiol ~ Progesterone',year,sep=':'), xlab='Day',ylab='Log Concentration')

lines(m1$date, m1$Estradiol, type='l', lwd=2, col='blue')

event <- m1$Estradiol \* max\_mat$event

event2 <- m1$Progesterone \* max\_mat$event

points(m1$date, event, col='red', lwd=5)

points(m1$date, event2, col='red',lwd=5)

segments(x0 =m1$date , y0 = event2, x1 = m1$date, y1=event, lwd=2, lty=2, col='red' )

legend('topright', c('Estradiol','Progesterone','Ovulation'),

col=c('blue','green','red'), lwd=2, pch=c(1,1,1), cex=0.8)

abline(h=log(up\_est), lty=2, lwd=2)

abline(h=log(bot\_est), lty=2, lwd=2)

abline(h=log(up\_pro), lty=2, col='brown', lwd=2)

abline(h=log(bot\_pro), lty=2, col='brown', lwd=2)

legend('topleft', c(paste('Estr', c(up\_est,bot\_est),sep='=')), lty=2, col='black', lwd=2)

legend('bottomleft',c(paste('Proge', c(up\_pro,bot\_pro),sep='=')), lty=2, col='brown', lwd=2)

}

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

## analyze\_monthly\_trends :

## - compares monthly estradiol

## and progesterone values

## - plots peaks and events

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

analyze\_monthly\_trends <- function(m1, lag=0, cutoff\_estradiol=2000,

cutoff\_progesterone=400){

### Detect events

estra\_max <- numeric(nrow(m1))

estra\_max[localMaxima(m1$Estradiol)] <- 1

estra\_max[which(m1$Estradiol < log(cutoff\_estradiol))] <- 0

proge\_max <- numeric(nrow(m1))

proge\_max[localMaxima(m1$Progesterone)] <- 1

proge\_max[which(m1$Progesterone < log(cutoff\_progesterone))] <- 0

event <- numeric(nrow(m1))

df = data.frame(estra\_max,

proge\_max,

m1$date)

for(i in 1:length(event)){

date\_search = min(i+lag, length(event))

if( estra\_max[i]==1 & sum(proge\_max[i:(i+lag)])>0 &

day(m1$date[date\_search]) - day(m1$date[i]) <= lag){

event[i] <- 1

#event[i:(i+lag)] <- 1

}

}

df$event <- event

event <- m1$Estradiol \* event

max\_mat <- data.frame(estra\_max, proge\_max, event)

### Do monthly panel plots

m1$event <- event

newM1 = melt(m1,id.vars = c('id','day','date','months','year'))

p1 = ggplot(newM1, aes(x = day, value, col=variable)) + geom\_line() + facet\_grid(months~.) +

theme\_bw() + theme(

plot.title = element\_text(color="black", size=24, face="bold", hjust = 0.5),

axis.title.x = element\_text(color="black", size=18, face="bold"),

axis.title.y = element\_text(color="black", size=18, face="bold"),

axis.text.x = element\_text(color="black", size=12, face="bold")) +

ggtitle('Ovulation') + xlab('Date') + ylab('Log Concentration')

### Look at time between peaks

time\_between\_peaks <- diff(which(event > 0 ))

if(sum(event>0) < 5){

print('insufficient data to run KS Test')

} else{

print(summary\_peaks(time\_between\_peaks))

print(test\_between\_peaks\_distribution(time\_between\_peaks))

fit1 <- fitdistr(time\_between\_peaks, "exponential")

#

# true\_Exp <- rexp(10000, fit1$estimate)

#

# plot(ecdf(time\_between\_peaks), xlim=range(c(time\_between\_peaks, true\_Exp)), col="dodgerblue", main='KS Test Visualization',

# ylab=paste("Wait-time b/w Events CDF"), xlab= "")

# plot(ecdf(true\_Exp), add=TRUE, lty="dashed", col="purple", ylab="", xlab="")

# legend("right", legend=c("2016 observed", "Exponential Dist'n"), col=c("dodgerblue", "purple"),

# lty="dashed", lwd=2 )

#

p2 <- recordPlot()

}

print(p1)

return(list(summary\_peaks(time\_between\_peaks),

time\_between\_peaks,

df))

}

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

## summary\_peaks :

## - provides tabular summary of

## peaks

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

summary\_peaks <- function(time\_between\_peaks){

numPeaks <- length(time\_between\_peaks)+1

AvgLength<- mean(time\_between\_peaks)

AvgStd <- sqrt(var(time\_between\_peaks))

if(numPeaks < 5){

Pvalue <- NA

} else{

require(MASS)

fit1 <- fitdistr(time\_between\_peaks, "exponential")

KS.res <- ks.test(time\_between\_peaks, "pexp", fit1$estimate)

Pvalue = KS.res$p.value

}

summary\_stats <- data.frame(NumberOfOvulation = numPeaks,

Avg.Time\_BW\_Ovu = AvgLength,

Median.Time\_BW\_Ovu = median(time\_between\_peaks),

# Max\_Time\_Between\_Peaks = max(time\_between\_peaks),

# Min\_Time\_Between\_Peaks = min(time\_between\_peaks),

Std\_Deviation = AvgStd,

Pvalue=Pvalue)

return(round(summary\_stats,2))

}

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

## test\_between\_peaks\_distribution :

## - conducts KS test

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

test\_between\_peaks\_distribution <- function(time\_between\_peaks){

cat('\nTo determine the strength of the distribution of the data we run a KS test

from fitting an exponential distribution to the observed data\n')

require(MASS)

fit1 <- fitdistr(time\_between\_peaks, "exponential")

KS.res <- ks.test(time\_between\_peaks, "pexp", fit1$estimate)

print(KS.res)

if(KS.res$p.value > 0.05){

cat(paste('The p-value > 0.05 suggests that the spontaneous ovulation events

are occuring at regular intervals following an exponential distribution

with rate parameter =', round(fit1$estimate,2)))

} else {

cat('The p-value < 0.05 suggests that the spontaneous ovulation events are not

occuring at regular intervals, and thus do not follow an exponential distribution

of equal-peak intervals')

}

return(KS.res)

}

### Yearly summary

years\_plot <- function(days\_lag, cutoff\_estradiol, cutoff\_progesterone,

up\_pro = 300,

bot\_pro = 100,

up\_est = 3000,

bot\_est = 1500){

par(mfrow=c(2,1), mar=c(5,5,5,5))

analyze\_yearly\_trends(m1=year2015, lag = days\_lag,

cutoff\_estradiol = cutoff\_estradiol,

cutoff\_progesterone = cutoff\_progesterone,

year='2015',

up\_pro = up\_pro,

bot\_pro = bot\_pro,

up\_est = up\_est,

bot\_est = bot\_est)

analyze\_yearly\_trends(m1=year2016, lag = days\_lag,

cutoff\_estradiol = cutoff\_estradiol,

cutoff\_progesterone = cutoff\_progesterone,

year='2016',

up\_pro = up\_pro,

bot\_pro = bot\_pro,

up\_est = up\_est,

bot\_est = bot\_est)

}

##

##

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

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

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

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

##

## Analyze Results

##

### Set custom parameters

cutoff\_estradiol = 1000

cutoff\_progesterone = 200

days\_lag = 0

## Analyze squirrel data using

## Monthly Panels + KS Test

#jpeg('C:/Users/stuwe/Desktop/Stat analysis from stat lab/monthy\_ovulation2015.jpeg')

time\_between\_peaks2015 <- analyze\_monthly\_trends(m1= year2015, lag=days\_lag,

cutoff\_estradiol = cutoff\_estradiol,

cutoff\_progesterone = cutoff\_progesterone)

#dev.off()

#jpeg('C:/Users/stuwe/Desktop/Stat analysis from stat lab/monthy\_ovulation2016.jpeg')

time\_between\_peaks2016 <- analyze\_monthly\_trends(m1=year2016, lag=days\_lag,

cutoff\_estradiol = cutoff\_estradiol,

cutoff\_progesterone = cutoff\_progesterone)

#dev.off()

peakSummary = data.frame(rbind(time\_between\_peaks2015[[1]],

time\_between\_peaks2016[[1]]), row.names = c('2015','2016'))

print(peakSummary)

write.csv(peakSummary, file='C:/Users/stuwe/Desktop/Stat analysis from stat lab/ovulation\_summary.csv')

## Analyze squirrel data using

## Yearly analysis

#jpeg('C:/Users/stuwe/Desktop/Stat analysis from stat lab/yearly\_comparison.jpeg', width = 900, height= 1200)

years\_plot(days\_lag = days\_lag ,

cutoff\_estradiol = cutoff\_estradiol,

cutoff\_progesterone = cutoff\_progesterone,

up\_pro = 300,

bot\_pro = 100,

up\_est = 2500,

bot\_est = 1200)

#dev.off()