script_analisis_1_IV.R

ACER

Thu Oct 04 11:49:12 2018

```
### PROYECTO PARCIAL 1 ###
### BIODISPONIBILIDAD Y BIOEQUIVALENCIA 2018 - 2###
### DEPARTAMENTO DE FARMACIA ###
### UNIVERSIDAD NACIONAL DE COLOMBIA ###

# Apertura de paquetes
library("ggplot2")
library("plyr")
library("reshape2")

# Apertura de datos
setwd(file.path("F:","Documentos","Estudio - Documentos","Farmacocinética 2012-1",
                "Bioequivalencia (2018)","Escritura"))
data = read.csv("./data/data_example_1_bioequivalence.csv",header=T,sep = ",",dec =".")

# Apertura de modelos PK
source("./scripts/modelos_compartimentos.R",echo=FALSE)


# Regresión No Lineal de Datos Administración IV --------------------------
m0 = length(unique(subset(data,Type == "IV")$ID)) # Número de Individuos
list_reg1 = list() # Creación de Lista

for (i in 1:m0) {
  list_reg1[[i]] = nls(Conc. ~ cmptm.1.IV(D0=Dosis, Vd, ke, t=Time),
                       data=subset(data,Type == "IV" & ID == i),
                       start = list(Vd=21,ke=0.25))
  }


# Obtener todos los resúmenes de modelos no lineales
list_reg1_summaries = lapply(list_reg1, function(x) summary(x))
# Obtener todos los R^2 de correlación para modelos
list_reg1_correlations = list()
for (i in 1:m0) {
  list_reg1_correlations[[i]] = cor(subset(data,Type == "IV" & ID == i)$Conc.,
                                    predict(list_reg1[[i]]),method="pearson")
}
# Obtener todos los perfiles para modelos no lineales
list_reg_1_profiles = lapply(list_reg1, function(x) profile(x, alpha = 0.05))

# Predicciones
tt <- c(seq(0,13,length=40))

list_reg1_predictions = list()
  for (i in 1:m0) {
    list_reg1_predictions[[i]]   = predict(list_reg1[[i]],
```

```
                                        newdata=list(t = tt),
                                        interval="confidence")
    }

# Dataframe para Graficar
df.reg1.1 = data.frame(ID = subset(data,Type=="IV")$ID,
                       ID2 = subset(data,Type=="IV")$ID2,
                       Time = subset(data,Type=="IV")$Time,
                       COBS = subset(data,Type=="IV")$Conc.,
                       CPRED = unlist(list_reg1_predictions))
# Gráfico en Páneles por Individuo
graph.reg1 = ggplot(subset(data,Type=="IV"),aes(Time,Conc.,group=ID2))+
  geom_point(col="black",fill=NA)+
  facet_wrap( ~ ID2,scales = "fixed",nrow=4,ncol=3)+
  guides(fill=guide_legend(title="TIPO"))+
  labs(x="Tiempo (horas)", y="Concentración (mcg/mL)") +
  scale_x_continuous(sec.axis=dup_axis(name=NULL,labels = NULL),
                     breaks = c(seq(0,14,by=2)),
                     limits=c(0,13)) +
  scale_y_continuous(sec.axis=dup_axis(name=NULL,labels = NULL),
                     breaks = c(seq(0,8,by=2)),
                     limits=c(0,7))+
  scale_color_manual(values=c("red4","blue4"),name="Sujeto")+
  geom_line(data=df.reg1.1, aes(x=df.reg1.1$Time,y=df.reg1.1$CPRED))+
  theme(panel.grid = element_line(colour="gray90",size =0.5),
        legend.text = element_blank(),
        panel.background=element_rect(colour = NA,fill = NA),
        panel.border = element_rect(colour = "black",fill = NA),
        strip.background = element_rect(colour="black",fill="cornsilk1"),
        strip.text.x = element_text(face="bold"))
# Gráfico de Spaguetti 1
graph.reg2 = ggplot(df.reg1.1,aes(Time,COBS,group=as.factor(ID),col=as.factor(ID)))+
  geom_point(aes(col=as.factor(ID)),shape=16,size=1)+
  geom_line(aes(x=Time,y=CPRED,col=as.factor(ID)),size=0.3)+
  guides(colour=guide_legend(title="Sujeto",ncol=2))+
  labs(x="Tiempo (horas)", y="Concentración (mcg/mL)") +
  scale_x_continuous(sec.axis=dup_axis(name=NULL,labels = NULL),
                     breaks = c(seq(0,14,by=2)),
                     limits=c(0,13)) +
  scale_y_continuous(sec.axis=dup_axis(name=NULL,labels = NULL),
                     breaks = c(seq(0,8,by=1)),
                     limits=c(0,7))+
  theme(panel.grid = element_line(colour="gray98",size =0.25),
        panel.background=element_rect(colour = NA,fill = NA),
        panel.border = element_rect(colour = "black",fill = NA))+
  scale_colour_brewer(palette = "Paired",name="Sujeto")
# Gráfico de Spaguetti 2
graph.reg3 = ggplot(df.reg1.1,aes(Time,COBS,group=as.factor(ID),col=as.factor(ID)))+
  geom_point(aes(col=as.factor(ID)))+
  geom_line(aes(x=Time,y=CPRED,col=as.factor(ID)))+
  guides(colour=guide_legend(title="Sujeto",ncol=2))+
  labs(x="Tiempo (horas)", y="Concentración (mcg/mL)") +
  scale_x_continuous(sec.axis=dup_axis(name=NULL,labels = NULL),
```

```
                    breaks = c(seq(0,14,by=2)),
                    limits=c(0,13)) +
  scale_y_continuous(sec.axis=dup_axis(name=NULL,labels = NULL),
                    breaks = c(seq(0,8,by=0.5)),trans = "log10",
                    limits=c(4E-01,7))+
  theme(panel.grid = element_line(colour="gray98",size =0.25),
        panel.background=element_rect(colour = NA,fill = NA),
        panel.border = element_rect(colour = "black",fill = NA))+
  scale_colour_brewer(palette = "Paired",name="Sujeto")

# Resumen de Parámetros
l1 = length(list_reg1)
v1 = vector(length = l1)
v2 = vector(length = l1)
v3 = vector(length = l1)
v4 = vector(length = l1)
v5 = vector(length = l1)
v6 = vector(length = l1)
v9 = vector(length = l1)
v10 = vector(length = l1)
for (i in 1:l1) {
  v1[i] = list_reg1_summaries[[i]]$parameters[1,1]
  v2[i] = list_reg1_summaries[[i]]$parameters[1,2]
  v3[i] = list_reg1_summaries[[i]]$parameters[2,1]
  v4[i] = list_reg1_summaries[[i]]$parameters[2,2]
  v5[i] = list_reg1_summaries[[i]]$sigma
  v6[i] = list_reg1_correlations[[i]]
  v9[i] = sqrt(deviance(list_reg1[[i]])/df.residual(list_reg1[[i]])) # RSE
  v10[i] = df.residual(list_reg1[[i]])
  }


A = data.frame(ID = unique(data$ID),
               ID2 = unique(data$ID2),
               D0 = rep(140,l1),
               Vd.value = v1,
               Vd.sd = v2,
               ke.value = v3,
               ke.sd = v4,
               CL.value = v1*v3,
               Sigma = v5,
               R2 = v6,
               RSE = v9,
               DF = v10)

A$C0.value = A$D0/A$Vd.value
A$t.vida.media = 0.693/A$ke.value
#### Matriz para cálculo AUC
B = dcast(subset(data,Type == "IV")[,c(1:5)], Time ~ ID, value.var="Conc.")
B[11,] = c(0.0, A$C0.value) # Adición del primer valor en curva de concentración
#
{attach(B)
B <- B[order(Time),]
```

```
detach(B)  }
B
```

```
##    Time        1        2        3       4        5        6        7
## 11  0.0 6.665149 6.867635 6.462662 7.00177 6.328529 6.398929 6.931367
## 1   0.5 5.940000 6.120000 5.760000 6.24000 5.640000 5.700000 6.180000
## 2   1.0 5.300000 5.460000 5.140000 5.56000 5.040000 5.090000 5.510000
## 3   1.5 4.720000 4.860000 4.580000 4.96000 4.480000 4.530000 4.910000
## 4   2.0 4.210000 4.340000 4.080000 4.42000 4.000000 4.040000 4.380000
## 5   3.0 3.340000 3.440000 3.240000 3.51000 3.170000 3.210000 3.470000
## 6   4.0 2.660000 2.740000 2.580000 2.79000 2.530000 2.550000 2.770000
## 7   6.0 1.680000 1.730000 1.630000 1.76000 1.600000 1.610000 1.750000
## 8   8.0 1.060000 1.090000 1.030000 1.11000 1.010000 1.020000 1.100000
## 9  10.0 0.670000 0.690000 0.650000 0.70000 0.640000 0.640000 0.700000
## 10 12.0 0.420000 0.430000 0.410000 0.44000 0.400000 0.400000 0.440000
##           8        9       10       11       12
## 11 7.198305 6.128234 6.866816 6.326535 6.387782
## 1  6.410000 5.460000 6.120000 5.630000 5.680000
## 2  5.720000 4.880000 5.300000 5.080000 5.070000
## 3  5.100000 4.340000 4.960000 4.430000 4.510000
## 4  4.550000 3.870000 4.340000 3.990000 4.020000
## 5  3.610000 3.070000 3.340000 3.150000 3.190000
## 6  2.870000 2.450000 2.740000 2.510000 2.530000
## 7  1.810000 1.550000 1.630000 1.580000 1.590000
## 8  1.140000 0.970000 1.040000 1.030000 1.000000
## 9  0.720000 0.620000 0.660000 0.630000 0.620000
## 10 0.450000 0.390000 0.430000 0.410000 0.380000
```

```
# Cálculo de AUC
AUC = matrix(nrow=dim(B)[1],ncol=(dim(B)[2])-1)
for (i in (1:dim(B)[1])) {
  for (j in (2:dim(B)[2])) {
    if (i < (dim(B)[1])) {
      AUC[i,(j-1)] = (B[i,j]+B[(i+1),j])*(B[i+1,1]-B[i,1])/2  # Acá se toman todos los datos experiment
    } else {
      AUC[i,(j-1)] = (B[i,j]/A[(j-1),6]) # En el último punto, se hace una extrapolación
    }
  }
}
# Adición de AUC a matriz A
A$AUC.trunc = apply(AUC[-11,],2,sum)
A$AUC.total = apply(AUC,2,sum)
```