J. Braz. Chem. Soc., Vol. 24, No. 9, S1-S3, 2013.
Printed in Brazil - ©2013 Sociedade Brasileira de Química
0103 - 5053 $6.00+0.00

Supplementary Information

Construction of Analytical Curve Fit Models for Simvastatin using Ordinary and Weighted Least Squares Methods

Flávia D. Marques-Marinho,*[,a] Ilka A. Reis[b] and Cristina D. Vianna-Soares[a]

[a]Department of Pharmaceutical Products, Faculty of Pharmacy and [b]Department of Statistics, Institute of Exact Sciences, Federal University of Minas Gerais, Av. Pres. Antônio Carlos, 6627, 31270-901 Belo Horizonte-MG, Brazil

```
# response and conc are vectors containing the nT observations for the response
# and the concentration, respectively.
conc.qd<-conc^2             # creating the quadratic term
# Adjusting the model using WSLM(weights are calculated as in Fig. A.3)
# To adjust a OSLM, just drop the option "weights" in lm()command.
model.conc.pond.qd<-lm(response ~ conc + conc.qd, weights=diag(W3))
residues.stu.pond.qd<-rstudent(model.conc.pond.qd) ## studentized residuals
ylim<-c(min(residues.stu.pond.qd),max(residues.stu.pond.qd))
xlim<-c(min(conc),max(conc))
plot(conc,residues.stu.pond.qd,col="purple",ylim=ylim,xlim=xlim) ; abline(h=0)
# Checking for outliers.
outliers<-(1:length(response))[abs(residues.stu.pond.qd)>3.0]
# Remove the outliers, if necessary.
data.noout.pond.qd<-cbind(response, conc, conc.qd, residues.stu.pond.qd)
data.noout.pond.qd<-data.noout.pond.qd[-outliers,]
# Adjust the model to the new dataset, if necessary and test outliers again.
model.conc.pond.qd<-lm(data.noout.pond.qd[,1]~ data.noout.pond.qd [,2]+
data.noout.pond.qd[,3])
residues.stu.pond.qd<-rstudent(model.conc.pond.qd) ## studentized residuals
plot(data.noout.pond.qd[,1],residues.stu.pond.qd);abline(h=0) # checking the
# residuals
# checking for outliers
outliers<-(1:length(data.noout.pond.qd[,1]))[abs(residues.stu.pond.qd)>3.0]
# Remove the outliers and adjust the model to the new dataset (if necessary)
data.noout.pond.qd<-cbind(data.noout.pond.qd[,1], data.noout.pond.qd[,2],
data.noout.pond.qd[,3], data.noout.pond.qd[,4])
data.noout.pond.qd<-data.noout.pond.qd[-outliers,]
# Checking the assumptions about the model errors
data.levene<-data.frame(cbind(rstudent(model.conc.pond.qd), conc))
# Testing for Variance homogeneity
leveneTest(data.levene[,1]~as.factor(data.levene[,2]),data=data.levene)
bartlett.test(rstudent(model.conc.pond.qd) ~ as.factor(conc))
# Testing for residuals normality
shapiro.test(rstudent(model.conc.pond.qd))   #Normality test
qqPlot(rstudent(model.conc.pond.qd))
anova(model.conc.pond.qd)    # Examining the coefficients estimates statistical
                             # significance
# Calculating MEP and MRE
data<-cbind(response,conc,conc.qd)  #Use data.noout if there are outliers
X<-cbind(rep(1,length(data[,2])),data[,2], data[,3] )
MEP.i<-numeric(length(data[,1]))
MRE.i<-numeric(length(data[,1]))
for (i in 1:length(data[,1])) {
 model<-lm(data[-i,1]~ data[-i,2] + data[-i,3],weights=diag(W3)[-i])
 betas<-solve(t(X[-i,])%*%W3[-i,-i]%*%X[-i,])%*%t(X[-i,])%*%W3[-i,-i]%*%
              data[-i,1]
predicted<-X%*%betas
MEP.i[i]<-(data[i,1]-predicted [i])^2
delta<-(betas[2]^2) - 4*betas[3]*(betas[1]-data[,1])
# Calculating the roots of the second degree equation
predicted <-cbind(-betas[2]-sqrt(delta),-betas[2]+sqrt(delta))/(2*betas[3])
predicted <-apply(abs(predicted),1,min)  # Choosing the smaller root
MRE.i[i]<-(abs(data[i,2]-predicted [i]))/data[i,2]
                                }
(MEP<-sum(MEP.i)/length(data[,1]))
(MRE<-sum(MRE.i)/length(data[,1]))
```

Figure S1. R script to adjust and check the quadratic model in equation 1 for a single day using the WLSM when independent variable (X) is expressed as concentration.

*e-mail: flaviadmar@hotmail.com

```
# response.day1, response.day2 and mass.day1, mass.day2 are vectors containing
# the nT observations for the response and the mass in different days,
# respectively.
response.days<-c(response.day1,response.day2)
mass.days<-c(mass.day1,mass.day2)
ind.day<-c(rep(1,length(response.day1)),rep(0,length(response.day2)))
# Adjusting the model using WSLM
# To adjust a OSLM, just drop the option "weights" in lm()
command.model.days.pond.ln<-lm(response.days ~ mass.days + ind.day +
ind.day*mass.days, weights=diag(W.day))
residues.stu.pond.ln<-rstudent(model.days.pond.ln) # studentized residuals
ylim<-c(min(residues.stu.pond.ln),max(residues.stu.pond.ln))
xlim<-c(min(mass.days),max(mass.days))
# Checking the residuals
plot(mass[ind.day==0],residues.stu.pond.ln[ind.day==0],col="red",
     ylim=ylim,xlim=xlim) ; abline(h=0) ;par(new=T)
plot(mass[ind.day==1],residues.stu.pond.ln[ind.day==1],col="blue",
     ylim=ylim,xlim=xlim)
# Check for outliers. Remove them, if necessary.
# Adjust the model to the new dataset, if necessary.
# Checking the assumptions about the model errors
data.levene<-data.frame(cbind(rstudent(model.days.pond.ln),mass.days))
# Testing for Variance homogeneity
leveneTest(data.levene[,1]~as.factor(data.levene[,2]),data=data.levene)
bartlett.test(rstudent(model.days.pond.ln) ~ as.factor(mass.days))
# Testing for residuals normality
shapiro.test(rstudent(model.days.pond.ln))
qqPlot(rstudent(model.days.pond.ln))
anova(model.days.pond.ln)     # Examining the coefficients estimates statistical
                              # significance
# Examine the significance of the coefficient of the interaction term
#(ind.day*mass.days). If it is not significant, adjust the commom model
model.common.pond.ln<-lm(response.days ~ mass.days,weights=diag(W.day))
residues.stu.common.pond.ln<-rstudent(model.common.pond.ln)
                                              # studentized residuals
# Checking for outliers
outliers<-(1:length(response.days))[abs(residues.stu.common.pond.ln)>3.0]
# Removing the outliers (if necessary)
data.noout<-cbind(response.days, mass.days, residues.stu.common.pond.ln)
data.noout<-data.noout[-outliers,]
# Adjust the model using WSLM to the new dataset, if necessary
# Calculating MEP and MRE
data<- cbind(response.days,mass.days)   # Use data.noout if there are outliers
MEP.i<-numeric(length(data[,1]))
MRE.i<-numeric(length(data[,1]))
for (i in 1:length(data[,1])) {
 model<-lm(data[-i,1]~ data[-i,2],weights=diag(W.day)[-i] )
 betas<-solve(t(X[-i,])%*%W.day [-i,-i]%*%X[-i,])%*%t(X[-i,])%*%
                W.day [-i,-i]%*%data[-i,1]
 predicted<-(data[,1]-betas[1])/betas[2]
 MRE.i[i]<-(abs(data[i,2]-predicted [i]))/data[i,2]
 predicted<-X%*%betas
 MEP.i[i]<-(data[i,1]-predicted [i])^2
                              }
(MEP<-sum(MEP.i)/length(data[,1]))    # MEP
(MRE<-sum(MRE.i)/length(data[,1]))    # MRE
```

Figure S2. R script to adjust and check the linear version of the model in equation 3 using the WLSM when independent variable (X) is expressed as mass.

```
# mean.day and var.day are vectors containing the mean and the variance of the
# replicates in each value of concentration (or mass), respectively.

# Examining the relationship between the variance and the mean of response
plot(mean.day ,var.day)

# Example: possible relationship: Var = exponential(Mean) --> ln(Var) = Media

# The ln function is applied to the Var to make the relationship between Var and
# Mean to be linear.
log.var<-log(var.day)

# Estimating the linear model
model<-lm(log.var ~ mean.day)

# Predicting the ln(Var) given the values of the Mean
ln.Var.predicted<- predict(model)

# The weights are the inverse of the predicted variances
W.day<-1/rep(exp(ln.Var.predicted),3)     #the weights are replicated since are
                      #three replicates in each concentration (or mass)

# Building the weights matrix W
n<-length(W.day)
W<-matrix(numeric(n*n),ncol=n)
diag(W)<-W.day
```

Figure S3. R script to examine the relationship between the variance and the mean of response and to calculate the weights to be used in WLSM.