Customizable R code to accompany:

Timmons, A. C., & Preacher, K. J. (in press). The importance of temporal design: How do measurement intervals affect the accuracy and efficiency of parameter estimates in longitudinal research? *Multivariate Behavioral Research*.

```
#########################################################################
#########################################################################
# INSTRUCTIONS                                                          #
#########################################################################
#########################################################################

# Simulated Code for Two-Parameter Functions

# This simulation will only work for two-parameter functions (e.g.,
# power). Please see other code for three parameters.

# Do not choose 0 as a parameter estimate. If you choose 0, the
# simulation will not run.

# This code will take several minutes to run.

# Convergence errors may print but the simulation will
# continue to run until 1,000 estimates converge.

# You will need to install the package nlme to run this code (see code
below).

# install.packages("nlme", dependencies = T)

# FOLLOW STEPS 1-4 IN THE CAPITALIZED COMMENTS AND THEN RUN THE
# ENTIRE CODE. LOOK IN COMMENTS FOR EXAMPLES.

library(nlme)

#########################################################################
#########################################################################
# STEP 1: REPLACE AAA WITH THE NUMBER AND SPACING OF MEASUREMENT     #
# OCCASIONS YOU WOULD LIKE TO SIMULATE.                              #
#########################################################################
#########################################################################

temporal.design <- c(AAA)

# Example
# temporal.design <- c(0, 12, 24, 36, 48, 60)

#########################################################################
#########################################################################
# STEP 2: REPLACE BBB, CCC, DDD, AND EEE WITH THE PARAMETER          #
```

```
# VALUES YOU WOULD LIKE TO SIMULATE. ALSO REPLACE FFF WITH THE       #
# NUMBER CLUSTERS YOU WOULD LIKE.                                    #
#                                                                    #
# b0 = VALUE OF FIRST COEFFICIENT                                    #
# b1 = VALUE OF SECOND COEFFICIENT                                   #
# ve = LEVEL ONE VARIANCE                                            #
# tau = LEVEL TWO VARIANCE                                           #
# nn = NUMBER OF CLUSTERS                                            #
#                                                                    #
######################################################################
######################################################################

b0 <- BBB
b1 <- CCC
ve <- DDD
tau <- EEE
nn <- FFF

# Example
# b0 <- 9
# b1 <- .7
# ve <- 400
# tau <- 4
# nn <- 50

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

set.seed(42596)

ve.sqrt <- sqrt(ve)
tau.sqrt <- sqrt(tau)

u0j <- rnorm(nn, 0, tau.sqrt)

out <- matrix(nrow = 1000,ncol = 4)
i <- 0
for (loop in 1:1000){
i <- i+1

temporal.design.1 <- vector(mode="numeric",length =
nn*length(temporal.design))

######################################################################
######################################################################
# STEP 3: REPLACE GGG WITH THE EQUATION FOR THE FUNCTIONAL FORM YOU  #
# WOULD LIKE TO SIMULATE USING b0 AND b1 FOR THE PARAMETER VALUES.   #
# MAKE X "temporal.design." TO ADD A RANDOM EFFECT, PUT THE          #
# PARAMETER VALUE + u0j[j] IN PARENTHESES, e.g., (b0 + u0j[j]).      #
######################################################################
######################################################################
```

```
for (j in 1:nn) {
  for (x in temporal.design) {
    temp <- rnorm(length(temporal.design), GGG, ve.sqrt)
  }

# Example
# for (j in 1:nn) {
#  for (x in temporal.design) {
#    temp <- rnorm(length(temporal.design), (b0 +
u0j[j])*temporal.design^b1, ve.sqrt)

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

  for (x in 1:length(temporal.design)) {
    temporal.design.1[(j - 1)*length(temporal.design) + x] <- temp[x]
  }
}

q2 <- rep(1:nn,each = length(temporal.design))
q1 <- rep(1:length(temporal.design), times = nn)
time<-rep(temporal.design, time = nn)
dataframe.temporal.design <- data.frame(q2, q1, time, temporal.design.1)

#########################################################################
#########################################################################
# STEP 4: REPLACE HHH WITH THE EQUATION FOR THE FUNCTIONAL FORM YOU  #
# WOULD LIKE TO SIMULATE AS DONE ABOVE EXECPT REPLACE b0 WITH par1,  #
# b1 WITH par2, AND temporal.design WITH time. ALSO, DO NOT ADD      #
# u0j[j]). INSTEAD, REPLACE III WITH EITHER par1 OR par2,            #
# DEPENDING ON WHICH IS THE RANDOM EFFECT.                           #
#########################################################################
#########################################################################

try(fit.temporal.design <- nlme(temporal.design.1 ~ HHH, fixed = list(par1 +
par2 ~ 1), random = III ~ 1 | q2, data = dataframe.temporal.design, start =
c(par1 = b0, par2 = b1)))

# Example
# try(fit.temporal.design <- nlme(temporal.design.1 ~ par1*time^par2, fixed
= list(par1 + par2 ~
# 1), random = par1 ~ 1 | q2, data = dataframe.temporal.design, start =
c(par1 = b0, par2 = b1)))

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

out[i,1]=summary(fit.temporal.design)$tTable[1]
out[i,2]=summary(fit.temporal.design)$tTable[2]
out[i,3]=summary(fit.temporal.design)$tTable[3]
out[i,4]=summary(fit.temporal.design)$tTable[4]
```

```
}

outdataframe <- data.frame(out)
attach(outdataframe)

par1.temporal.design <- X1
par2.temporal.design <- X2
SE1.temporal.design <- X3
SE2.temporal.design <- X4

PRB.par1.temporal.design <- ((mean(par1.temporal.design)-b0)/b0)*100
ESD.par1.temporal.design <- sd(par1.temporal.design)

PRB.par2.temporal.design <- ((mean(par2.temporal.design)-b1)/b1)*100
ESD.par2.temporal.design <- sd(par2.temporal.design)

SE.par1.temporal.design <- mean(SE1.temporal.design)
SE.par2.temporal.design <- mean(SE2.temporal.design)

#########################################################################
#########################################################################
# OUTPUT:                                                               #
#########################################################################
#########################################################################

(mean(par1.temporal.design))
# Mean estimate of parameter 1 across 1,000 trials

(mean(par2.temporal.design))
# Mean estimate of parameter 2 across 1,000 trials

(PRB.par1.temporal.design)
# Percent relative bias of parameter 1

(PRB.par2.temporal.design)
# Percent relative bias of parameter 2

(SE.par1.temporal.design)
# Standard error of parameter 1

(SE.par2.temporal.design)
# Standard error of parameter 2

(ESD.par1.temporal.design)
# Empirical standard deviation of parameter 1

(ESD.par2.temporal.design)
# Empirical standard deviation of parameter 2

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

```
#########################################################################
#########################################################################
# INSTRUCTIONS                                                          #
#########################################################################
#########################################################################

# Simulated Code for Three-Parameter Functions

# This simulation will only work for three-parameter functions (e.g.,
# Gompertz). Please see other code for two parameters.

# Do not choose 0 as a parameter estimate. If you choose 0, the
# simulation will not run.

# This code will take several minutes to run.

# Convergence errors may print but the simulation will
# continue to run until 1,000 estimates converge.

# You will need to install the package nlme to run this code (see code
below).

# install.packages("nlme", dependencies = T)

# FOLLOW STEPS 1-4 IN THE CAPITALIZED COMMENTS AND THEN RUN THE
# ENTIRE CODE. LOOK IN COMMENTS FOR EXAMPLES.

library(nlme)

#########################################################################
#########################################################################
# STEP 1: REPLACE AAA WITH THE NUMBER AND SPACING OF MEASUREMENT        #
# OCCASIONS YOU WOULD LIKE TO SIMULATE.                                 #
#########################################################################
#########################################################################

temporal.design <- c(AAA)

# Example
# temporal.design <- c(0, 12, 24, 36, 48, 60)


#########################################################################
#########################################################################
# STEP 2: REPLACE BBB, CCC, DDD, EEE, AND FFF WITH THE PARAMETER        #
# VALUES YOU WOULD LIKE TO SIMULATE. ALSO REPLACE GGG WITH THE          #
# NUMBER CLUSTERS YOU WOULD LIKE.                                       #
#                                                                       #
# b0 = VALUE OF FIRST COEFFICIENT                                       #
# b1 = VALUE OF SECOND COEFFICIENT                                      #
```

```
# b2 = VALUE OF THE THIRD COEFFICIENT                                  #
# ve = LEVEL ONE VARIANCE                                              #
# tau = LEVEL TWO VARIANCE                                             #
# nn = NUMBER OF CLUSTERS                                              #
#                                                                      #
########################################################################
########################################################################

b0 <- BBB
b1 <- CCC
b2 <- DDD
ve <- EEE
tau <- FFF
nn <- GGG

# Example
# b0 <- 500
# b1 <- 4
# b2 <- .2
# ve <- 400
# tau <- 900
# nn <- 50

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

set.seed(42596)

ve.sqrt <- sqrt(ve)
tau.sqrt <- sqrt(tau)

u0j <- rnorm(nn, 0, tau.sqrt)

out <- matrix(nrow = 1000,ncol = 6)
i <- 0
for (loop in 1:1000){
i <- i+1

temporal.design.1 <- vector(mode="numeric",length =
nn*length(temporal.design))

########################################################################
########################################################################
# STEP 3: REPLACE HHH WITH THE EQUATION FOR THE FUNCTIONAL FORM YOU  #
# WOULD LIKE TO SIMULATE USING b0, b1, and b2 FOR THE PARAMETER      #
# VALUES. MAKE X "temporal.design." TO ADD A RANDOM EFFECT, PUT THE  #
# PARAMETER VALUE + u0j[j] IN PARENTHESES, e.g., (b0 + u0j[j]).      #
########################################################################
########################################################################

for (j in 1:nn) {
  for (x in temporal.design) {
```

```
    temp <- rnorm(length(temporal.design), HHH, ve.sqrt)
  }

# Example
# for (j in 1:nn) {
#  for (x in temporal.design) {
#    temp <- rnorm(length(temporal.design), (b0 + u0j[j])*exp(-exp(b1-
b2*temporal.design)), ve.sqrt)

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

  for (x in 1:length(temporal.design)) {
    temporal.design.1[(j - 1)*length(temporal.design) + x] <- temp[x]
  }
}

q2 <- rep(1:nn,each = length(temporal.design))
q1 <- rep(1:length(temporal.design), times = nn)
time<-rep(temporal.design, time = nn)
dataframe.temporal.design <- data.frame(q2, q1, time, temporal.design.1)



#########################################################################
#########################################################################
# STEP 4: REPLACE III WITH THE EQUATION FOR THE FUNCTIONAL FORM YOU  #
# WOULD LIKE TO SIMULATE AS DONE ABOVE EXECPT REPLACE b0 WITH par1,  #
# b1 WITH par2, b2 WITH par3, AND temporal.design WITH time.        #
# ALSO, DO NOT ADD u0j[j]). INSTEAD, REPLACE JJJ WITH EITHER par1,   #
# par2, OR par3, DEPENDING ON WHICH IS THE RANDOM EFFECT.           #
#########################################################################
#########################################################################

try(fit.temporal.design <- nlme(temporal.design.1 ~ III, fixed = list(par1 +
par2 + par3 ~ 1), random = JJJ ~ 1 | q2, data = dataframe.temporal.design,
start = c(par1 = b0, par2 = b1, par3 = b2)))

# Example
# try(fit.temporal.design <- nlme(temporal.design.1 ~ par1*exp(-exp(par2-
par3*time)),
# fixed = list(par1 + par2 + par3 ~ 1), random = par1 ~ 1 | q2, data =
dataframe.temporal.design,
# start = c(par1 = b0, par2 = b1, par3 = b2)))

out[i,1]=summary(fit.temporal.design)$tTable[1]
out[i,2]=summary(fit.temporal.design)$tTable[2]
out[i,3]=summary(fit.temporal.design)$tTable[3]
out[i,4]=summary(fit.temporal.design)$tTable[4]
out[i,5]=summary(fit.temporal.design)$tTable[5]
out[i,6]=summary(fit.temporal.design)$tTable[6]
}
```

```
outdataframe <- data.frame(out)
attach(outdataframe)

par1.temporal.design <- X1
par2.temporal.design <- X2
par3.temporal.design <- X3
SE1.temporal.design <- X4
SE2.temporal.design <- X5
SE3.temporal.design <- X6

PRB.par1.temporal.design <- ((mean(par1.temporal.design)-b0)/b0)*100
ESD.par1.temporal.design <- sd(par1.temporal.design)

PRB.par2.temporal.design <- ((mean(par2.temporal.design)-b1)/b1)*100
ESD.par2.temporal.design <- sd(par2.temporal.design)

PRB.par3.temporal.design <- ((mean(par3.temporal.design)-b2)/b2)*100
ESD.par3.temporal.design <- sd(par3.temporal.design)

SE.par1.temporal.design <- mean(SE1.temporal.design)
SE.par2.temporal.design <- mean(SE2.temporal.design)
SE.par3.temporal.design <- mean(SE3.temporal.design)

#########################################################################
#########################################################################
# OUTPUT:                                                               #
#########################################################################
#########################################################################

(mean(par1.temporal.design))
# Mean estimate of parameter 1 across 1,000 trials

(mean(par2.temporal.design))
# Mean estimate of parameter 2 across 1,000 trials

(mean(par3.temporal.design))
# Mean estimate of parameter 3 across 1,000 trials

(PRB.par1.temporal.design)
# Percent relative bias of parameter 1

(PRB.par2.temporal.design)
# Percent relative bias of parameter 2

(PRB.par3.temporal.design)
# Percent relative bias of parameter 3

(SE.par1.temporal.design)
# Standard error of parameter 1

(SE.par3.temporal.design)
```

```
# Standard error of parameter 3

(SE.par2.temporal.design)
# Standard error of parameter 2

(ESD.par1.temporal.design)
# Empirical standard deviation of parameter 1

(ESD.par2.temporal.design)
# Empirical standard deviation of parameter 2

(ESD.par3.temporal.design)
# Empirical standard deviation of parameter 3

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