Supplementary Material: An R code for estimating parameters and plotting

```
##Parameter estimation for the SBS and SW distributions
library(rtdists)
library(ghyp)
library(fitdistrplus)
library(fBasics)
library(gamlss.dist)
library(retimes)
library(statmod)
library(MASS)
library(bssn)
library(xtable)

##load the data
data <- read.csv("RT_new_data.csv", sep = ";", dec = ",")

##Filter the NA entries
data_no.na <- na.omit(data)

##Consider only correct answers
data_acc <- data_no.na[data_no.na$acc == 1,]

##Some filters
##First filter Condition == "RI" & data_acc$Bloque == "B1"
& data_acc$stim == 4 & data_acc$resp == 2
data_acc_RI_B1_Stim4_resp2 <- data_acc[data_acc$Condition == "RI"
& data_acc$Bloque == "B2" & data_acc$stim == 4 & data_acc$resp == 2, ]

##filtered data set
sample_t <- data_acc_RI_B1_Stim4_resp2$RT

##Plotting the histograms
hist(sample_t, probability = TRUE)

##Parameter estimation process for the SBS
#Density of the SBS
dshiftedBSSN <- function(t, alpha, beta, theta){
#Both specifications can be used
return(dbssn(t-theta, alpha, beta, lambda=0))
#(1/(2*alpha))*(1/sqrt(beta*(t-theta))+sqrt(beta/(t-theta)^3))
#*dnorm((1/alpha)*(sqrt((t-theta)/beta)-1/sqrt((t-theta)/beta)))
}
#log-likelyhood function of the SBS to be optimized
```

```r
lshiftedBSSN <- function(par, t){
alpha <- par[1]
beta <- par[2]
theta <- par[3]
n <- length(t)
-n*log(alpha)-(n/2)*log(beta)+sum(log(t-theta+beta))-
(1/(2*alpha^2)*sum((t-theta)/beta+(beta/(t-theta))-2))
}

#Gradient function of the BS distribution
#to use in the Newton Raphson method below
miFun = function(t) {
alpha <- t[1]
beta <- t[2]
n = length(x)
h =(sum(1/(beta + x)))
c(-n/alpha + sum(x/beta)/alpha^3 + (sum(beta/x))/alpha^3 - (2*n)/alpha^3,
h - n/(2*beta) + sum(x)/(2*alpha^2*beta^2) - sum(1/x)/(2*alpha^2))
}

#Actualization vector
sigDx <- function(ff, x) {
solve(jacobian(ff, x), -ff(x))
}

#Norm calculation, decition rule
modulus <- function(x) sqrt(sum(x^2))

#Newton Raphson method
#Iterative optimization method
NwtRph <- function(ff, x0, eps=0.0001, lim=10000, absComp=F) {
n <- 0
repeat {
difx <- sigDx(ff, x0)
x <- x0 + difx
r <- modulus(difx)
if (absComp) {
if (r <= eps) return(x)
} else {
if (r <= eps*modulus(x0)) return(x)
}

if (n > lim) return (NULL)
n <- n+1
x0 <- x
}
}

#Initial guess for the parameters in the Newton Raphson algorithm
p_tst <- c(1,1) #(alpha, beta)
```

```
theta <- min(sample_t)-0.01
p_guess <- c(p_tst, theta)

#Optimization process: Here the optimization is performed in two steps
#1.- A grid search for theta;
#2.-  Newton's algorithm for alpha and beta.
#The function to be optimized is the log-likelyhood
ind <- 0
log_lik_sbssn <- lshiftedBSSN(p_guess, sample_t)
while(theta>(min(sample_t))-50){
s_sample_t <- sample_t-theta
x <- s_sample_t
p_aux <- NwtRph(miFun, p_tst)
if((2*3-2*lshiftedBSSN(c(p_aux, theta), sample_t))<
(2*3-2*lshiftedBSSN(p_guess, sample_t))){
p_guess <- c(p_aux, theta)
}
theta <- theta - 0.01
ind <- ind + 1
print(ind)
}

#Goodness of fit
#Adjusted parameters
alpha <- p_guess[1]
beta <- p_guess[2]
theta <- p_guess[3]

#AIC for the model
2*3-2*lshiftedBSSN(p_guess, sample_t)

##Plotting the histograms for the SBS model
hist(sample_t, probability = TRUE, breaks = 20,
xlim=c(min(sample_t)-10, max(sample_t+10)))
xx <- seq(min(sample_t)-0.5, max(sample_t+0.5), 0.1)
lines(xx, dshiftedBSSN(xx, alpha, beta, theta))

# ##QQ-plots for the SBS model
sampleSbssn <- rbssn(length(sample_t),alpha, beta, 0)+theta
#
qqplot(sample_t, sampleSbssn[abs(sampleSbssn)]
,main=paste("QQ-plot␣SBS"),xlab = "Theoretical␣Quantiles",
ylab = "Sample␣Quantiles",
xlim = c(500, 1400), ylim=c(500,1400))

#Determination coefficient for the SBS model
cor(sort(sample_t),sort(sampleSbssn[abs(sampleSbssn)]))

#Incresing rate of the latent cognitive process - Gaussian noise
print(paste('Estimate␣on␣the␣Incresing␣rate␣of␣the␣latent␣cognitive␣process␣for
Gaussian␣noise',alpha^2))
print(paste('The␣estimator␣for␣the␣coefficient␣of␣variation␣after', theta,'is',
sqrt(beta)*alpha))

#Fit of the SW model
##data set
sample_t <- data_acc_RI_B1_Stim4_resp2$RT

##Parameter estimation for the Shifted Wald
sample_size<-length(sample_t)
N<-length(sample_size)

##Estimates for the Shifted Wald distribution
Shifted_wald_est<-nigFit(sample_t, method = "mle", doplot = F)

##Recovering the parameters of every ajusted model
parShifted_wald_est<-as.numeric(Shifted_wald_est@fit$par)


# ##QQ-plots SW
samplenig<-rnig(length(sample_t),parShifted_wald_est[1],
parShifted_wald_est[2],  parShifted_wald_est[3])+theta
#
qqplot(sample_t, samplenig,main=paste("QQ-plot␣Shifted␣Wald"),
xlab = "Theoretical␣Quantiles",
ylab = "Sample␣Quantiles", xlim = c(500, 1400), ylim=c(500,1400))

#Determination coefficient
cor(sort(sample_t),sort(samplenig))

##Plot os the BS distribution for different parameters
plot(seq(0,4,00.1), dbssn(seq(0,4,00.1), alpha = 0.5, beta = 1, lambda = 0),
type = 'l', col = 2, ylab = 'BS␣density', xlab = 'variable␣values',
main = 'Plot␣of␣the␣BS␣density', xlim = c(0,3), ylim = c(0,1.5))

lines(seq(0,4,0.01), dbssn(seq(0,4,0.01), alpha = 0.8, beta = 1,
lambda = 0), col = 3)

lines(seq(0,4,0.01), dbssn(seq(0,4,0.01), alpha = 1.5, beta = 1,
lambda = 0), col = 4)

lines(seq(0,4,0.01), dbssn(seq(0,4,0.01), alpha = 0.2, beta = 1.5,
lambda = 0), col = 5)
```