Instructions

1. Copy Section 1 into R

2. Edit the code in Section 2 to enter
- the number of pre-measured parts in the baseline denoted by N
- the proportion of previously passed parts in the sample (f=0 is recommended)
- the guessed parameter values alpha, beta and the pass rate
- the required standard deviations for the estimates of alpha and beta

3. Copy and paste Sections 2 as edited and Section 3 into R.

4. To re-run the program with different inputs, re-edit section 2 and repeat step 3.

```
#############################################################################
# Section 1
#############################################################################

# Copy and paste into R all the following commands up to Section 2

rm(list=ls(all=TRUE))

sample_size<-function(data) {

N<-data[1]
f<-data[2]
theta0<-data [3]
theta1<-1-data[4]
tau<-data[5]
std_theta0_0<-data[6]
std_theta1_0<-data[7]

phi<-(tau-theta0)/(theta1-theta0)

p_conf<-(theta1*phi*f)/tau+((1-theta1)*phi*(1-f))/(1-tau)

std_theta1<-numeric(0)
std_theta0<-numeric(0)
std_phi<-numeric(0)
p_no_nonconform<-numeric(0)

n_r<-c(0,0)

for (j in 3:15) {
```

```
r<-j

pass<-seq(from=0, to=r, by=1)

E_I11_1<-numeric(0)
E_I22_1<-numeric(0)
E_I33_1<-numeric(0)
E_I12_1<-numeric(0)
E_I13_1<-numeric(0)
E_I23_1<-numeric(0)
Pr_1<-numeric(0)

E_I11_2<-numeric(0)
E_I22_2<-numeric(0)
E_I33_2<-numeric(0)
E_I12_2<-numeric(0)
E_I13_2<-numeric(0)
E_I23_2<-numeric(0)
Pr_0<-numeric(0)

for (i in 1:(r+1)) {

s<-pass[i]

prob1<-choose(r,s)*((theta1^(s+1))*((1-theta1)^(r-s))*phi+(theta0^(s+1))*((1-theta0)^(r-
s))*(1-phi))/(theta1*phi+theta0*(1-phi))

a1<-(theta1^(s+1))*((1-theta1)^(r-s))*phi+(theta0^(s+1))*((1-theta0)^(r-s))*(1-phi)

prob0<-choose(r,s)*((theta1^s)*((1-theta1)^(r-s+1))*phi+(theta0^s)*((1-theta0)^(r-
s+1))*(1-phi))/((1-theta1)*phi+(1-theta0)*(1-phi))

I_11_1<--(((theta1^(s+1))*((1-theta1)^(r-s))-(theta0^(s+1))*((1-theta0)^(r-
s)))^2)/((((theta1^(s+1))*((1-theta1)^(r-s))*phi+(theta0^(s+1))*((1-theta0)^(r-s))*(1-
phi))^2)+((theta1-theta0)/(theta1*phi+theta0*(1-phi)))^2

if (is.finite(prob1*I_11_1)==”TRUE”) {
E_I11_1<-c(E_I11_1,prob1*I_11_1)} else {
E_I11_1<-c(E_I11_1,0)}

I_22_1<-(phi/a1)*((s+1)*(s*(theta1^(s-1))*((1-theta1)^(r-s))-(theta1^s)*(r-s)*((1-
theta1)^(r-s-1)))-(r-s)*((s+1)*(theta1^s)*((1-theta1)^(r-s-1))-(theta1^(s+1))*(r-s-1)*((1-
theta1)^(r-s-2))))-((phi/a1)*((s+1)*(theta1^s)*((1-theta1)^(r-s))-(theta1^(s+1))*((1-
theta1)^(r-s-1))*(r-s)))^2+(phi/(theta1*phi+theta0*(1-phi)))^2
```

```
if (is.finite(prob1*I_22_1)==”TRUE”) {
E_I22_1<-c(E_I22_1,prob1*I_22_1)} else {
E_I22_1<-c(E_I22_1,0)}

I_33_1<-((1-phi)/a1)*((s+1)*(s*(theta0^(s-1))*((1-theta0)^(r-s))-(r-s)*(theta0^s)*((1-theta0)^(r-s-1)))-(r-s)*((s+1)*(theta0^s)*((1-theta0)^(r-s-1))-(r-s-1)*(theta0^(s+1))*((1-theta0)^(r-s-2))))-(((1-phi)/a1)*((s+1)*(theta0^s)*((1-theta0)^(r-s))-(r-s)*(theta0^(s+1))*((1-theta0)^(r-s-1))))^2+((1-phi)/(theta1*phi+theta0*(1-phi)))^2

if (is.finite(prob1*I_33_1)==”TRUE”) {
E_I33_1<-c(E_I33_1,prob1*I_33_1)} else {
E_I33_1<-c(E_I33_1,0)}

I_12_1<-(((s+1)*(theta1^s)*((1-theta1)^(r-s))-(r-s)*(theta1^(s+1))*((1-theta1)^(r-s-1)))/a1)*(1-(phi*((theta1^(s+1))*((1-theta1)^(r-s))-(theta0^(s+1))*((1-theta0)^(r-s))))/a1)-1/(theta1*phi+theta0*(1-phi))+phi*(theta1-theta0)/(theta1*phi+theta0*(1-phi))^2

if (is.finite(prob1*I_12_1)==”TRUE”) {
E_I12_1<-c(E_I12_1,prob1*I_12_1)} else {
E_I12_1<-c(E_I12_1,0)}

I_13_1<--(((s+1)*(theta0^s)*((1-theta0)^(r-s))-(r-s)*(theta0^(s+1))*((1-theta0)^(r-s-1)))/a1)*(1+((1-phi)*((theta1^(s+1))*((1-theta1)^(r-s))-(theta0^(s+1))*((1-theta0)^(r-s))))/a1)+1/(theta1*phi+theta0*(1-phi))+((1-phi)*(theta1-theta0))/(theta1*phi+theta0*(1-phi))^2

if (is.finite(prob1*I_13_1)==”TRUE”) {
E_I13_1<-c(E_I13_1,prob1*I_13_1)} else {
E_I13_1<-c(E_I13_1,0)}

I_23_1<--((1-phi)*phi*((s+1)*(theta0^s)*((1-theta0)^(r-s))-(r-s)*(theta0^(s+1))*((1-theta0)^(r-s-1)))*((s+1)*(theta1^s)*((1-theta1)^(r-s))-(r-s)*(theta1^(s+1))*((1-theta1)^(r-s-1))))/(a1^2)+((1-phi)*phi)/(theta1*phi+theta0*(1-phi))^2

if (is.finite(prob1*I_23_1)==”TRUE”) {
E_I23_1<-c(E_I23_1,prob1*I_23_1)} else {
E_I23_1<-c(E_I23_1,0)}

Pr_1<-c(Pr_1,prob1)

a2<-(theta1^s)*((1-theta1)^(r-s+1))*phi+(theta0^s)*((1-theta0)^(r-s+1))*(1-phi)

I_11_2<--(((theta1^s)*((1-theta1)^(r-s+1))-(theta0^s)*((1-theta0)^(r-s+1)))/a2)^2+((theta0-theta1)/((1-theta1)*phi+(1-theta0)*(1-phi)))^2
```

```
if (is.finite(prob0*I_11_2)==”TRUE”) {
E_I11_2<-c(E_I11_2,prob0*I_11_2)} else {
E_I11_2<-c(E_I11_2,0)}

I_22_2<-(phi*(s*((s-1)*(theta1^(s-2))*((1-theta1)^(r-s+1))-(r-s+1)*(theta1^(s-1))*((1-theta1)^(r-s)))-(r-s+1)*(s*(theta1^(s-1))*((1-theta1)^(r-s))-(r-s)*(theta1^s)*((1-theta1)^(r-s-1)))))/a2-((phi*(s*(theta1^(s-1))*((1-theta1)^(r-s+1))-(r-s+1)*(theta1^s)*((1-theta1)^(r-s))))/a2)^2+(phi/((1-theta1)*phi+(1-theta0)*(1-phi)))^2

if (is.finite(prob0*I_22_2)==”TRUE”) {
E_I22_2<-c(E_I22_2,prob0*I_22_2)} else {
E_I22_2<-c(E_I22_2,0)}

I_33_2<-((1-phi)*(s*((s-1)*(theta0^(s-2))*((1-theta0)^(r-s+1))-(r-s+1)*(theta0^(s-1))*((1-theta0)^(r-s)))-(r-s+1)*(s*(theta0^(s-1))*((1-theta0)^(r-s))-(r-s)*(theta0^s)*((1-theta0)^(r-s-1)))))/a2-(((1-phi)*(s*(theta0^(s-1))*((1-theta0)^(r-s+1))-(r-s+1)*(theta0^s)*((1-theta0)^(r-s))))/a2)^2+((1-phi)/((1-theta1)*phi+(1-theta0)*(1-phi)))^2

if (is.finite(prob0*I_33_2)==”TRUE”) {
E_I33_2<-c(E_I33_2,prob0*I_33_2)} else {
E_I33_2<-c(E_I33_2,0)}

I_12_2<-((s*(theta1^(s-1))*((1-theta1)^(r-s+1))-(r-s+1)*(theta1^s)*((1-theta1)^(r-s)))/a2)*(1-(phi*((theta1^s)*((1-theta1)^(r-s+1))-(theta0^s)*((1-theta0)^(r-s+1))))/a2)+1/((1-theta1)*phi+(1-theta0)*(1-phi))-(phi*(theta0-theta1))/((1-theta1)*phi+(1-theta0)*(1-phi))^2

if (is.finite(prob0*I_12_2)==”TRUE”) {
E_I12_2<-c(E_I12_2,prob0*I_12_2)} else {
E_I12_2<-c(E_I12_2,0)}

I_13_2<--((s*(theta0^(s-1))*((1-theta0)^(r-s+1))-(r-s+1)*(theta0^s)*((1-theta0)^(r-s)))/a2)*(1+((1-phi)*((theta1^s)*((1-theta1)^(r-s+1))-(theta0^s)*((1-theta0)^(r-s+1))))/a2)-1/((1-theta1)*phi+(1-theta0)*(1-phi))-((1-phi)*(theta0-theta1))/((1-theta1)*phi+(1-theta0)*(1-phi))^2

if (is.finite(prob0*I_13_2)==”TRUE”) {
E_I13_2<-c(E_I13_2,prob0*I_13_2)} else {
E_I13_2<-c(E_I13_2,0)}

I_23_2<--(phi*(1-phi)*(s*(theta1^(s-1))*((1-theta1)^(r-s+1))-(r-s+1)*(theta1^s)*((1-theta1)^(r-s)))*(s*(theta0^(s-1))*((1-theta0)^(r-s+1))-(r-s+1)*(theta0^s)*((1-theta0)^(r-s))))/a2^2+phi*(1-phi)/((1-theta1)*phi+(1-theta0)*(1-phi))^2

if (is.finite(prob0*I_23_2)==”TRUE”) {
```

```
E_I23_2<-c(E_I23_2,prob0*I_23_2)} else {
E_I23_2<-c(E_I23_2,0)}

Pr_0<-c(Pr_0,prob0)

}

sum(Pr_1)
sum(Pr_0)

n<-10
n1<-round(f*n)
n0<-round((1-f)*n)
EN1<-(N-n)*tau
EN0<-(N-n)*(1-tau)

Iphiphi<-EN1*(theta1-theta0)^2/(theta1*phi+theta0*(1-phi))^2+EN0*(-
theta1+theta0)^2/((1-theta1)*phi+(1-theta0)*(1-phi))^2

Itheta1theta1<-EN1*phi^2/(theta1*phi+theta0*(1-phi))^2+EN0*phi^2/((1-
theta1)*phi+(1-theta0)*(1-phi))^2

Itheta0theta0<-EN1*(1-phi)^2/(theta1*phi+theta0*(1-phi))^2+EN0*(-1+phi)^2/((1-
theta1)*phi+(1-theta0)*(1-phi))^2

Iphitheta1<--EN1/(theta1*phi+theta0*(1-phi))+EN1*(theta1-
theta0)*phi/(theta1*phi+theta0*(1-phi))^2+EN0/((1-theta1)*phi+(1-theta0)*(1-phi))-
EN0*(-theta1+theta0)*phi/((1-theta1)*phi+(1-theta0)*(1-phi))^2

Iphitheta0<-EN1/(theta1*phi+theta0*(1-phi))+EN1*(theta1-theta0)*(1-
phi)/(theta1*phi+theta0*(1-phi))^2-EN0/((1-theta1)*phi+(1-theta0)*(1-phi))+EN0*(-
theta1+theta0)*(-1+phi)/((1-theta1)*phi+(1-theta0)*(1-phi))^2

Itheta1theta0<-EN1*phi*(1-phi)/(theta1*phi+theta0*(1-phi))^2-EN0*phi*(-1+phi)/((1-
theta1)*phi+(1-theta0)*(1-phi))^2

J11<--(n1*sum(E_I11_1)+n0*sum(E_I11_2))+Iphiphi
J12<--(n1*sum(E_I12_1)+n0*sum(E_I12_2))+Iphitheta1
J13<--(n1*sum(E_I13_1)+n0*sum(E_I13_2))+Iphitheta0
J22<--(n1*sum(E_I22_1)+n0*sum(E_I22_2))+Itheta1theta1
J23<--(n1*sum(E_I23_1)+n0*sum(E_I23_2))+Itheta1theta0
J33<--(n1*sum(E_I33_1)+n0*sum(E_I33_2))+Itheta0theta0

J<-matrix(c(J11,J12,J13,J12,J22,J23,J13,J23,J33),nrow=3,ncol=3)

V<-solve(J)
```

```
std_phi_partial<-sqrt(V[1,1])
std_theta1_partial<-sqrt(V[2,2])
std_theta0_partial<-sqrt(V[3,3])

while((std_theta1_partial>std_theta1_0)|(std_theta0_partial>std_theta0_0))

{

n<-n+1

n1<-round(f*n)
n0<-round((1-f)*n)

EN1<-(N-n)*tau
EN0<-(N-n)*(1-tau)

Iphiphi<-EN1*(theta1-theta0)^2/(theta1*phi+theta0*(1-phi))^2+EN0*(-
theta1+theta0)^2/((1-theta1)*phi+(1-theta0)*(1-phi))^2

Itheta1theta1<-EN1*phi^2/(theta1*phi+theta0*(1-phi))^2+EN0*phi^2/((1-
theta1)*phi+(1-theta0)*(1-phi))^2

Itheta0theta0<-EN1*(1-phi)^2/(theta1*phi+theta0*(1-phi))^2+EN0*(-1+phi)^2/((1-
theta1)*phi+(1-theta0)*(1-phi))^2

Iphitheta1<--EN1/(theta1*phi+theta0*(1-phi))+EN1*(theta1-
theta0)*phi/(theta1*phi+theta0*(1-phi))^2+EN0/((1-theta1)*phi+(1-theta0)*(1-phi))-
EN0*(-theta1+theta0)*phi/((1-theta1)*phi+(1-theta0)*(1-phi))^2

Iphitheta0<-EN1/(theta1*phi+theta0*(1-phi))+EN1*(theta1-theta0)*(1-
phi)/(theta1*phi+theta0*(1-phi))^2-EN0/((1-theta1)*phi+(1-theta0)*(1-phi))+EN0*(-
theta1+theta0)*(-1+phi)/((1-theta1)*phi+(1-theta0)*(1-phi))^2

Itheta1theta0<-EN1*phi*(1-phi)/(theta1*phi+theta0*(1-phi))^2-EN0*phi*(-1+phi)/((1-
theta1)*phi+(1-theta0)*(1-phi))^2

J11<--(n1*sum(E_I11_1)+n0*sum(E_I11_2))+Iphiphi
J12<--(n1*sum(E_I12_1)+n0*sum(E_I12_2))+Iphitheta1
J13<--(n1*sum(E_I13_1)+n0*sum(E_I13_2))+Iphitheta0
J22<--(n1*sum(E_I22_1)+n0*sum(E_I22_2))+Itheta1theta1
J23<--(n1*sum(E_I23_1)+n0*sum(E_I23_2))+Itheta1theta0
J33<--(n1*sum(E_I33_1)+n0*sum(E_I33_2))+Itheta0theta0

J<-matrix(c(J11,J12,J13,J12,J22,J23,J13,J23,J33),nrow=3,ncol=3)
V<-solve(J)

std_phi_partial<-sqrt(V[1,1])
std_theta1_partial<-sqrt(V[2,2])
std_theta0_partial<-sqrt(V[3,3])
}

n_r<-rbind(n_r,c(n,r))

if (p_conf^n<0.00001) {
p_no_nonconform<-c(p_no_nonconform, 0)} else
{ p_no_nonconform<-c(p_no_nonconform, p_conf^n)}

std_theta1<-c(std_theta1,std_theta1_partial)
std_theta0<-c(std_theta0,std_theta0_partial)
std_phi<-c(std_phi,std_phi_partial)
}

n_r<-n_r[-1,]

sample_size_choices<-cbind(n=n_r[,1],r=n_r[,2],nxr=n_r[,1]*
n_r[,2],std_alpha=round(std_theta0,4),std_beta=round(std_theta1,4),std_pic=round(std_p
hi,4), p_no_nonconform);

sample_size_choices
}

########################################################################
## Section 2
########################################################################

## Enter inputs corresponding to your study design, the guessed parameter values and
required precisions in the command editor you use, by changing the values of N, f, alpha,
beta, pi_p, std_alpha_0,std_beta_0. Note that the current values correspond to the
example given in the paper “Assessment of a Binary Measurement System in Current
Use”, by Danila et al. (2009) (see Table 1).

## After editing, paste Section 2 into R

## Give the total number of parts in the baseline, i.e. the total number of parts previously
measured by the BMS (N):

N<-5000

## Give the proportion of previously passed parts in the sample (f)
```

```
f<-0

## Give the guessed parameter values:

alpha<-0.01
beta<-0.02
pi_p<-0.92
pi_c<-(pi_p-alpha)/(1-alpha-beta)
pi_c

if (pi_c>1) {
‘impossible parameter values; re-enter alpha or beta or pi_p’}

pi_p<-0.9
pi_c<-(pi_p-alpha)/(1-alpha-beta)
pi_c

## Give the desired precision for your parameters (std_alpha_0 and std_beta_0):

std_alpha_0<-0.005
std_beta_0<-0.005


##############################################################################
## Section 3
##############################################################################

## Copy and paste the next commands into R

input<-c(N,f, alpha,beta,pi_p,std_alpha_0,std_beta_0)

current_input_values<-cbind(N=input[1],f=input[2],
alpha=input[3],beta=input[4],pi_p=input[5],std_alpha_0=input[6],std_beta_0=input[7])

current_input_values

sample_size(input)
```