Supplemental material

Section A: Measurement performance for i.i.d. process with sample size $N = 15, 25, 30$

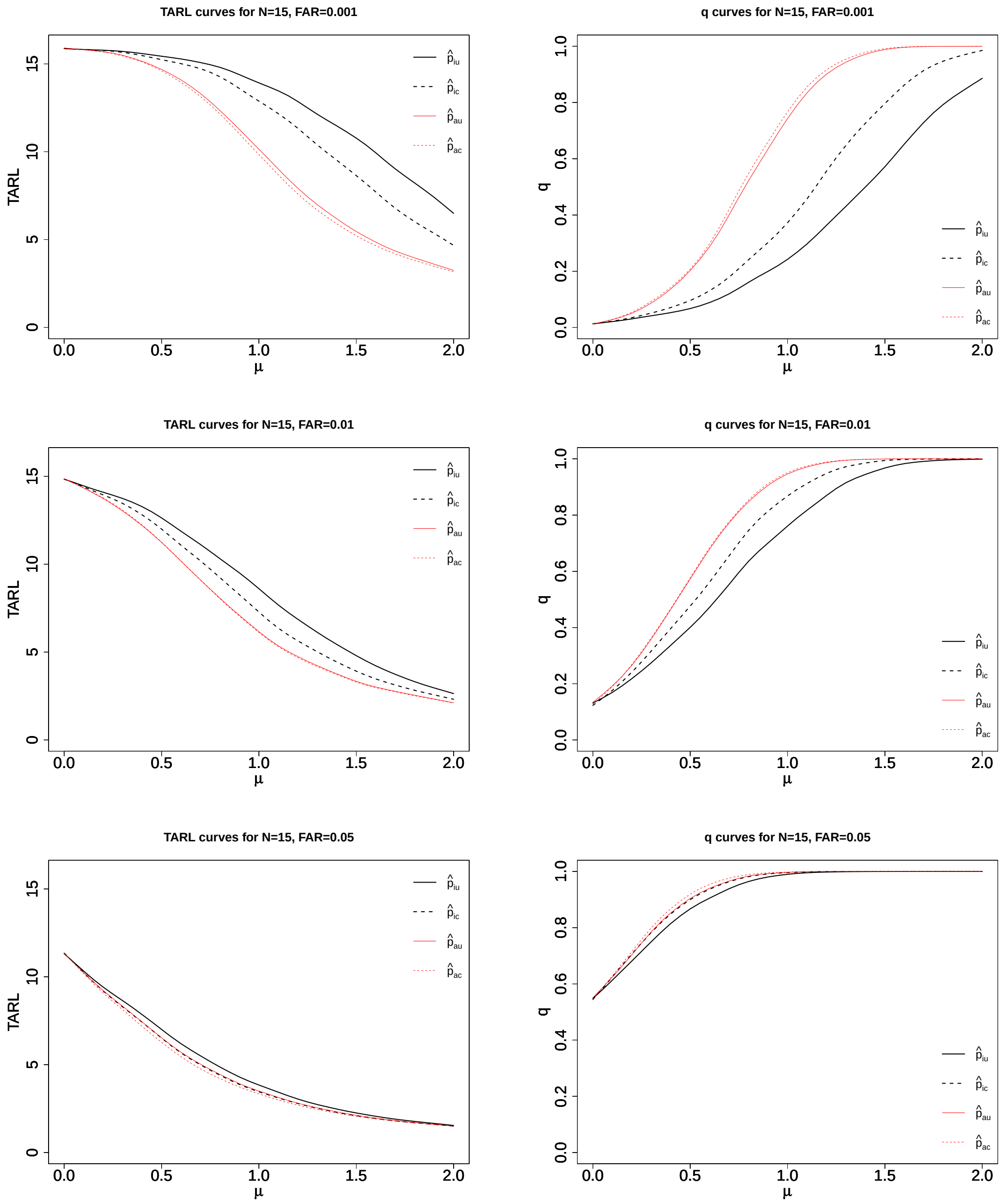


Fig. 11 TARL and q performance under normal distribution ($N = 15$)

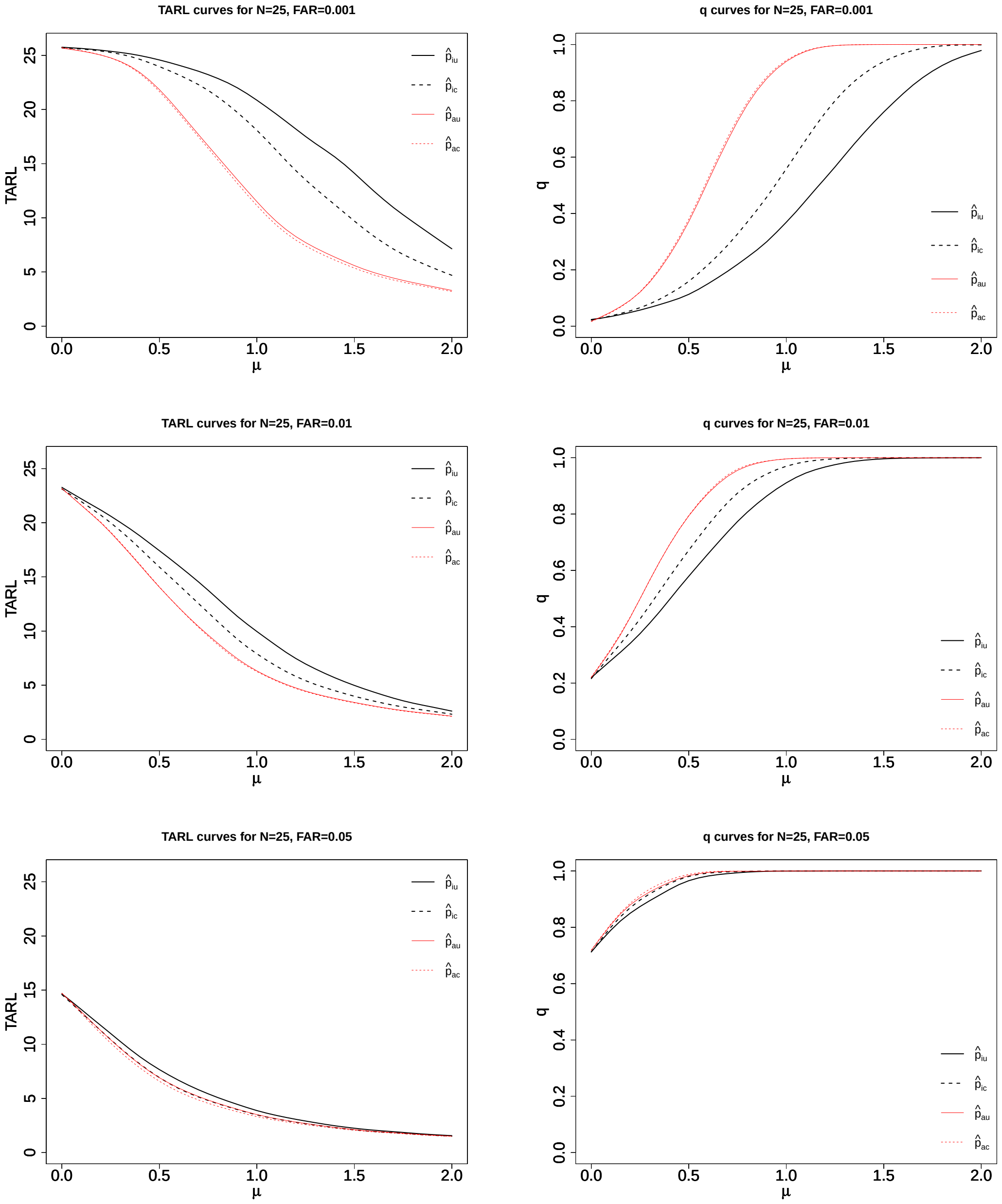


Fig. 12 TARL and q performance under normal distribution ($N = 25$)

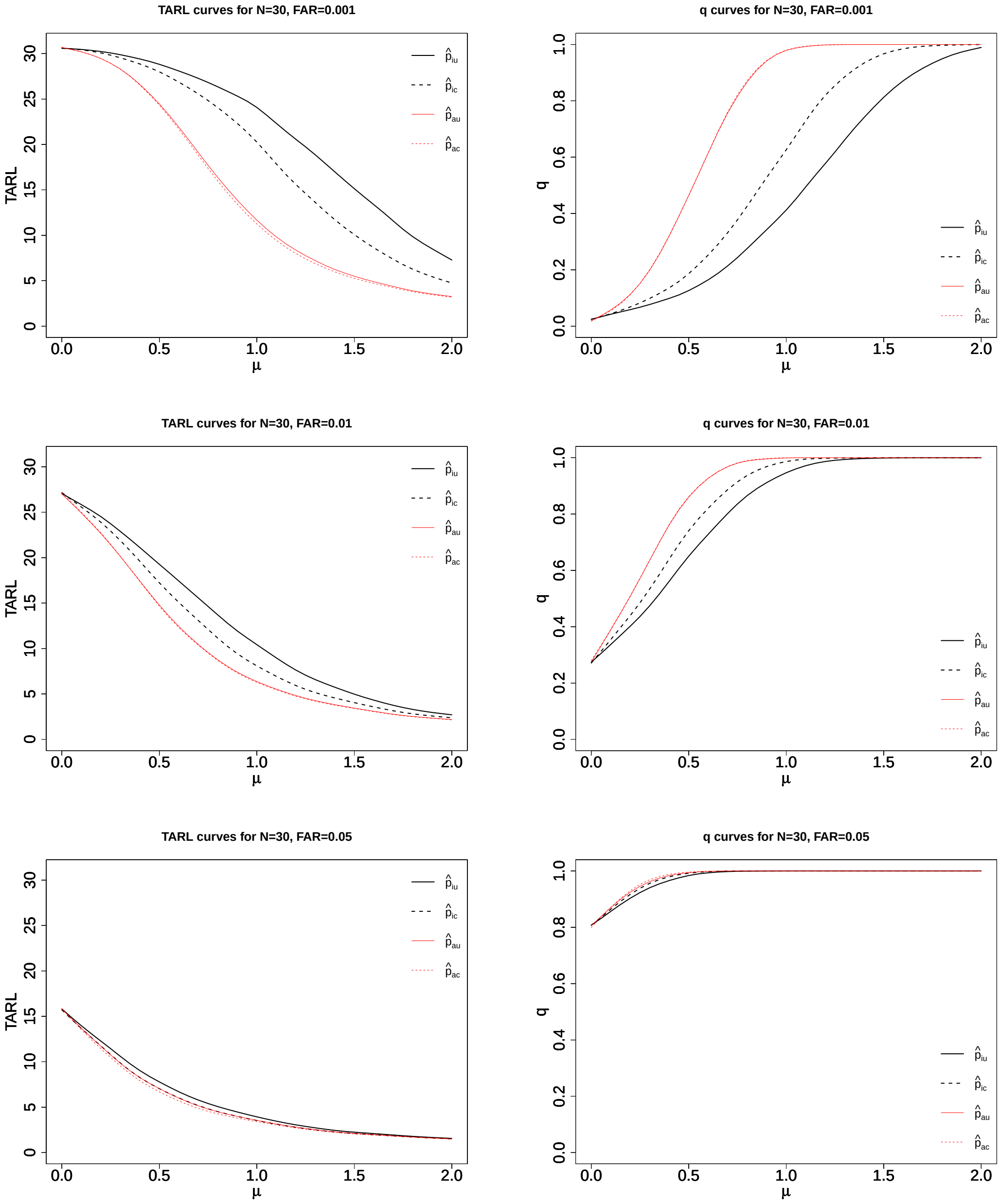


Fig. 13 TARL and q performance under normal distribution ($N = 30$)

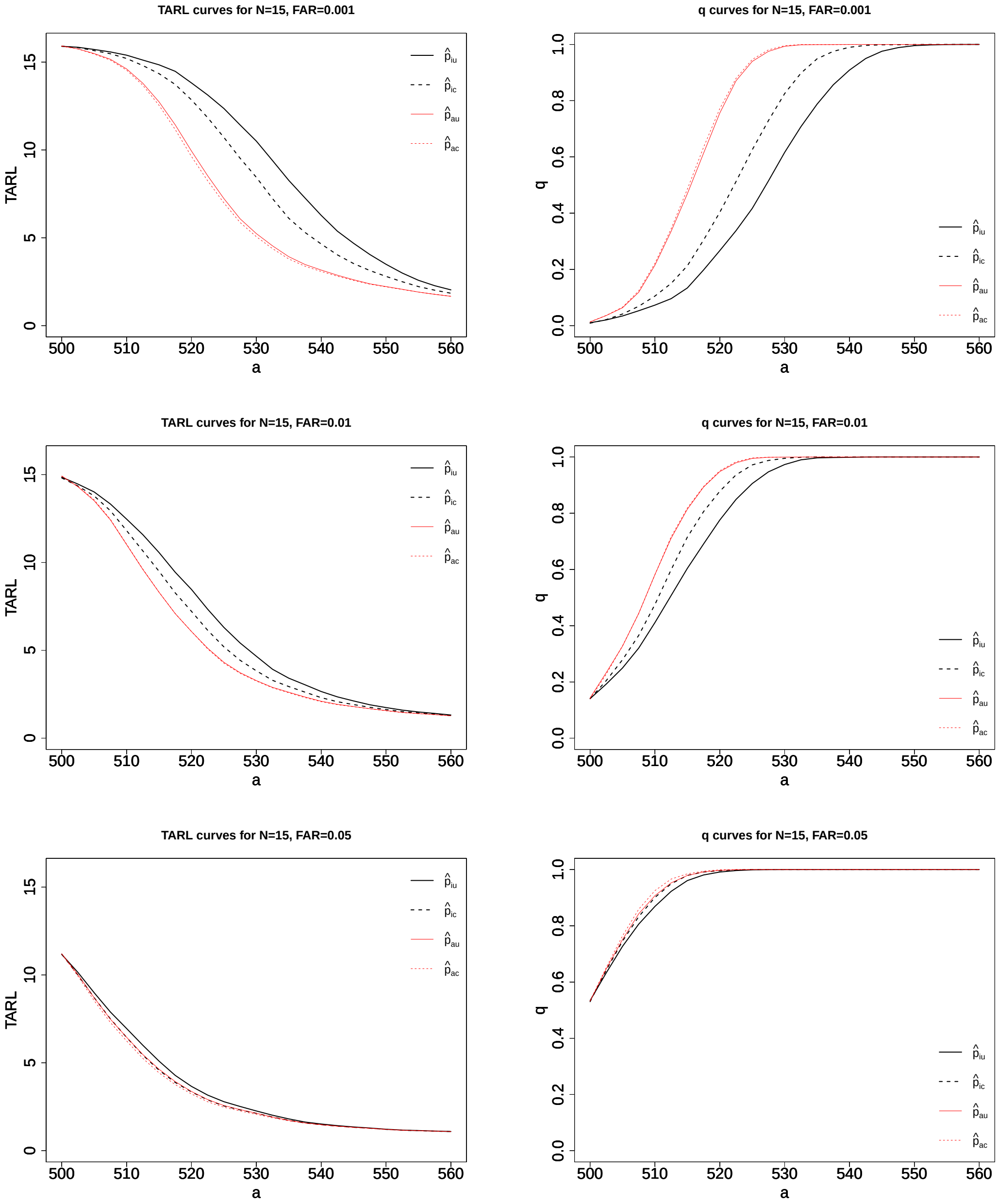


Fig. 14 TARL and q performance under beta distribution ($N = 15$)

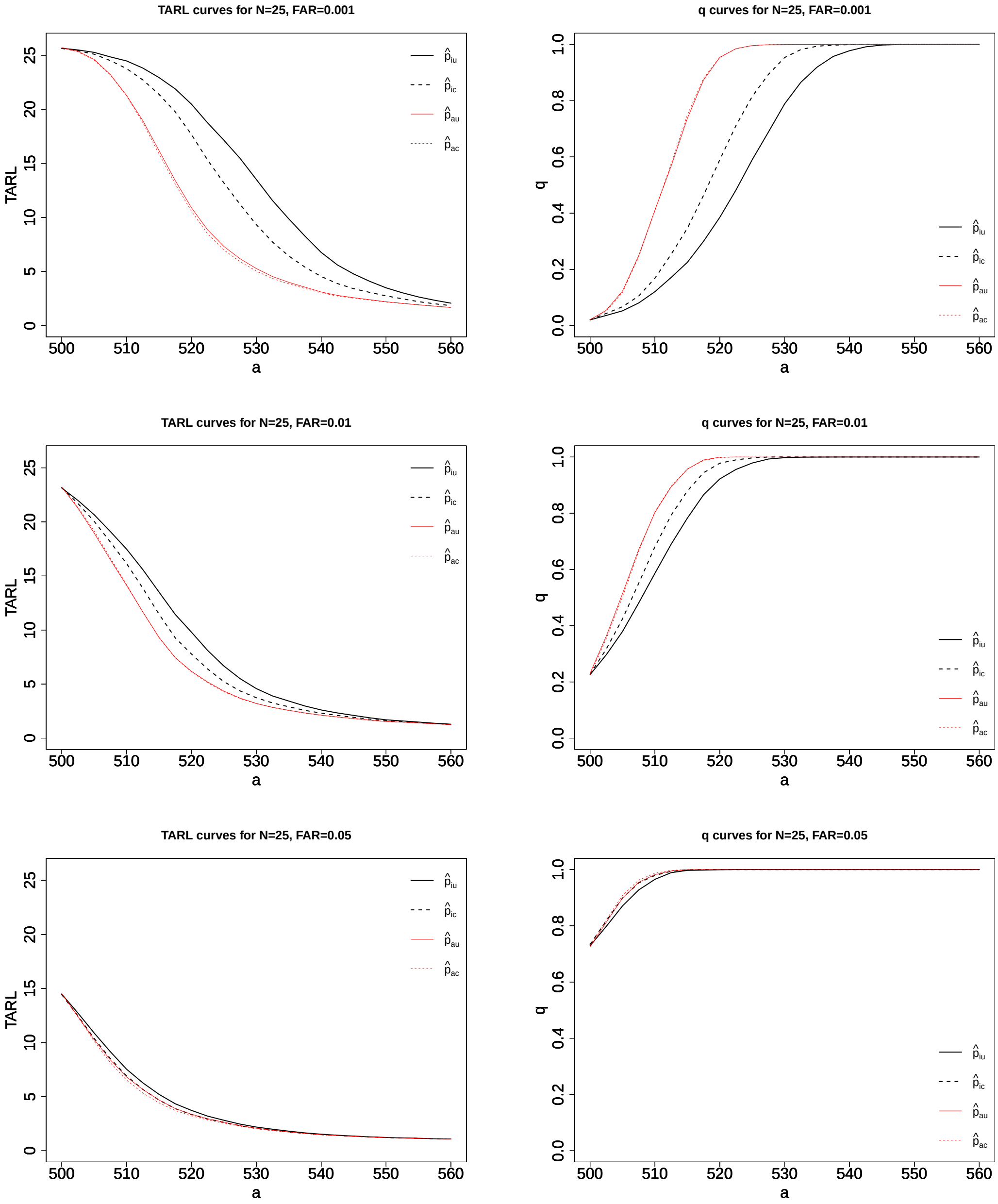
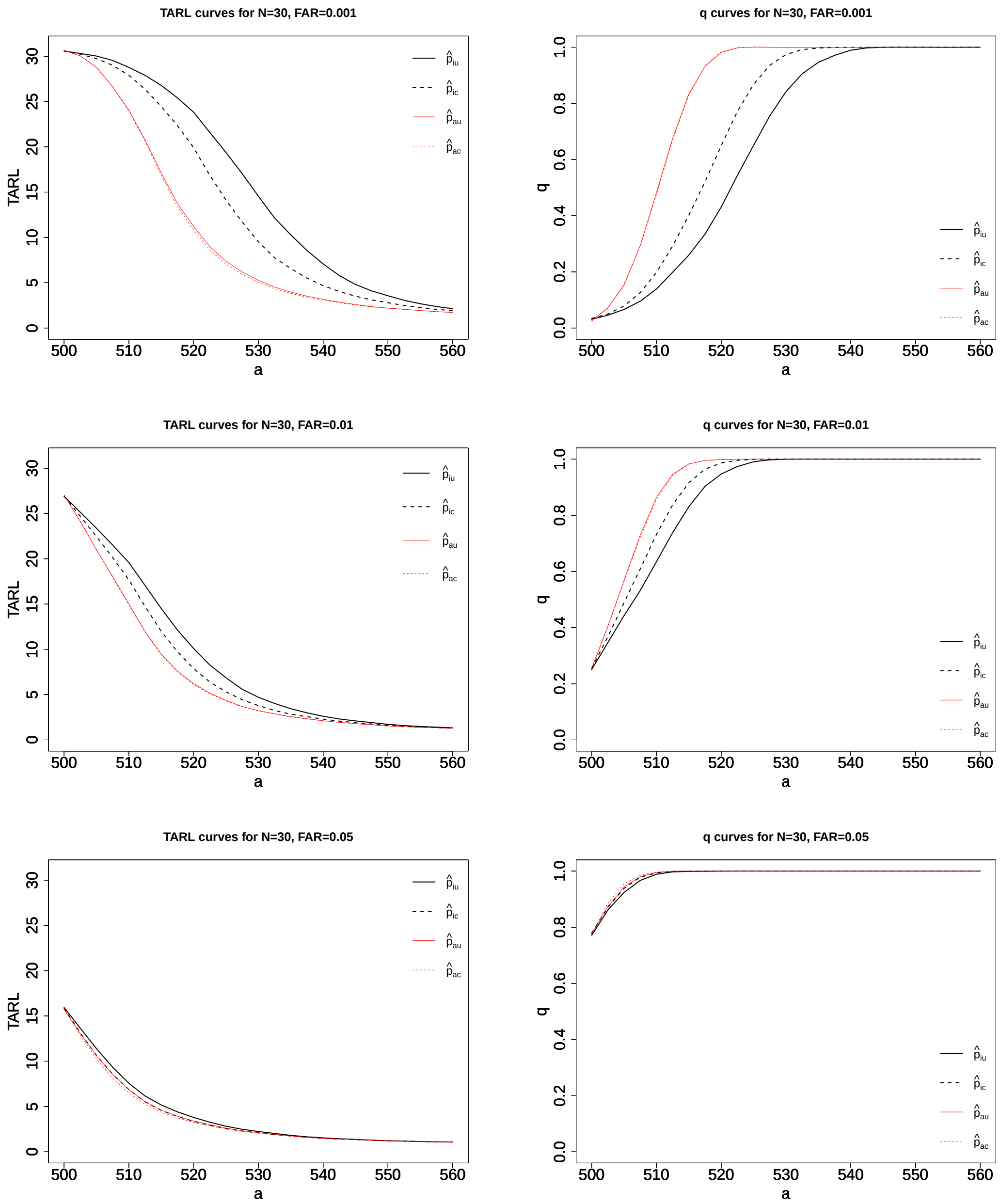


Fig. 15 TARL and q performance under beta distribution ($N = 25$)



TARL curves for N=30, FAR=0.001

q curves for N=30, FAR=0.001

TARL curves for N=30, FAR=0.01

q curves for N=30, FAR=0.01

TARL curves for N=30, FAR=0.05

q curves for N=30, FAR=0.05

Fig. 16 TARL and q performance under beta distribution ($N = 30$)

Section B: R code for fractional nonconformance estimation

```
#Input parameters
p=0.03 #AQL
n=20 #sample size
q=0.999 #1-q is false alarm rate
m=100000 #number of simulation to get empirical distribution
reps=2000 #number of runs to obtain emperical TARL and q
mu0=0 #in-control process mean
sd_y=1 #in-control process standard deviation
k=0.25 #ratio of Var(Z)/Var(Y)
sd_z=sqrt(k*sd_y^2) #measurement error standard deviation


#estimate is the function to estimate fractional nonconformance
estimate=function(n,p)
{
sample=rnorm(n,mu0,sd_y)
U.spec=-qnorm(p, mu0, sd_y)
#Upper limit based on AQL
frac.est=1-pnorm(U.spec, sample, sd_z)
#individual unconditional fractional nonconformance
cum.n=1:length(sample)
cum.p=cumsum(frac.est)/cum.n
#average unconditional fractional nonconformance
pbar=sum(frac.est)/length(sample)
ybar=cumsum(sample)/cum.n
a=(sample-U.spec)-k*(sample-ybar)
b=sd_z*sqrt(1-k)
wu=a/b
frac.est2=pnorm(wu)
#individual conditional fractional nonconformance
pbar2=sum(frac.est2)/length(sample)
cum.p2=cumsum(frac.est2)/cum.n
#average conditional fractional nonconformance
cbind(frac.est,frac.est2,cum.p,cum.p2)
}

#simulation m times to get empirical distribution
results=replicate(m,estimate(n,p))

#UCL is the function to estimate Upper Control Limit
UCL=function(q)
{
results2=t(results[,2,])
UCL.p2=NULL
UCL.p2[1]=quantile(results2[,1],q)
a=results2
for(i in 2:n)
{
b=assign(paste0("temp",i-1),a)
#include in-control previous points only
```

```
a=subset(b,b[,i-1]<UCL.p2[i-1])
UCL.p2[i]=quantile(a[,i],q)
}

results3=t(results[,3,])
UCL.p3=NULL
UCL.p3[1]=quantile(results3[,1],q)
a=results3
for(i in 2:n)
{
b=assign(paste0("temp",i-1),a)
a=subset(b,b[,i-1]<UCL.p3[i-1])
UCL.p3[i]=quantile(a[,i],q)
}

results4=t(results[,4,])
UCL.p4=NULL
UCL.p4[1]=quantile(results4[,1],q)
a=results4
for(i in 2:n)
{
b=assign(paste0("temp",i-1),a)
a=subset(b,b[,i-1]<UCL.p4[i-1])
UCL.p4[i]=quantile(a[,i],q)
}

rbind(UCL.p2,UCL.p3,UCL.p4)
}

UCL.p1=quantile(results[,1,],q) #UCL.p1 for individual unconditional case
UCL.p2=UCL(q)[1,] #UCL.p2 for individual conditional case
UCL.p3=UCL(q)[2,] #UCL.p3 for average unconditional case
UCL.p4=UCL(q)[3,] #UCL.p4 for average conditional case


#TARL is the function to estimate TARL
TARL=function(p,mu){
RL=function(n,p,mu)
{
sample=rnorm(n,mu,sd_y)
U.spec=-qnorm(p, mu0, sd_y)
frac.est=1-pnorm(U.spec, sample, sd_z)
cum.n=1:length(sample)
cum.p=cumsum(frac.est)/cum.n
ybar=cumsum(sample)/cum.n
a=(sample-U.spec)-k*(sample-ybar)
b=0.5*sqrt(1-k)
wu=a/b
frac.est2=pnorm(wu)
cum.p2=cumsum(frac.est2)/cum.n

test1=as.integer(frac.est>UCL.p1)
for(i in 1:n)
```

```
{if (test1[i]==1) {RL1=i;break}
else{RL1=n+1}
}
RL1 #RL for individual unconditional case

test2=as.integer(frac.est2>UCL.p2)
for(j in 1:n)
{if (test2[j]==1) {RL2=j;break}
else{RL2=n+1}
}
RL2 #RL for individual conditional case

test3=as.integer(cum.p>UCL.p3)
for(k in 1:n)
{if (test3[k]==1) {RL3=k;break}
else{RL3=n+1}
}
RL3 #RL for average unconditional case

test4=as.integer(cum.p2>UCL.p4)
for(m in 1:n)
{if (test4[m]==1) {RL4=m;break}
else{RL4=n+1}
}
RL4 #RL for average conditional case

cbind(RL1,RL2,RL3,RL4)
}

t=t(replicate(reps,RL(n,p,mu))[1,,])
tt=c(sum(t[,1]),sum(t[,2]),sum(t[,3]),sum(t[,4]))
arl=tt/reps
arl
}

mu_list=seq(0,2,0.05) #mean of Y shift from 0 to 2
resultstarl=mapply(FUN=TARL,p=0.03,mu=mu_list)


#plot TARL vs mean shift
par(mar=c(5,5,5,5))
plot(mu_list, resultstarl[1,],xlim = c(0, 2),ylim=c(0,n+1),
xlab=expression(mu), ylab="TARL",pch=NA,cex.axis = 1.5,cex.lab=1.5)
lines(smooth.spline(mu_list, resultstarl[1,], spar=0.5),type="l",lty=1,lwd=2,col="black")
par(new=T)
plot(mu_list, resultstarl[2,],xlim = c(0, 2),ylim=c(0,n+1),
xlab=expression(mu), ylab="TARL",pch=NA,cex.axis = 1.5,cex.lab=1.5)
lines(smooth.spline(mu_list, resultstarl[2,], spar=0.5),type="l",lty=2,lwd=2,col="black")
par(new=T)
plot(mu_list, resultstarl[3,],xlim = c(0, 2),ylim=c(0,n+1),
xlab=expression(mu), ylab="TARL",pch=NA,cex.axis = 1.5,cex.lab=1.5)
lines(smooth.spline(mu_list, resultstarl[3,], spar=0.5),type="l",lty=1,lwd=1,col="red")
par(new=T)
```

```
plot(mu_list, resultstarl[4,],xlim = c(0, 2),ylim=c(0,n+1),
xlab=expression(mu), ylab="TARL",pch=NA,cex.axis = 1.5,cex.lab=1.5)
lines(smooth.spline(mu_list, resultstarl[4,], spar=0.5),type="l",lty=2,lwd=1,col="red")
legend("topright",c(expression(hat(p)[iu]),expression(hat(p)[ic]),
expression(hat(p)[au]),expression(hat(p)[ac])),col=c("black","black","red","red"),
lty=c(1,2,1,2),lwd=c(2,2,1,1),pch=c(NA,NA), merge=FALSE ,cex = 1.1, y.intersp=2,
bty = "n")
title(paste0("TARL curves for"," N=",n,", FAR=",1-q))


#Prob_sig is the function to estimate q
Prob_sig=function(p,mu){
signal=function(n,p,mu)
{
sample=rnorm(n,mu,sd_y)
U.spec = -qnorm(p, mu0, sd_y)
frac.est=1-pnorm(U.spec, sample, sd_z)
cum.n=1:length(sample)
cum.p=cumsum(frac.est)/cum.n
ybar=cumsum(sample)/cum.n
a=(sample-U.spec)-k*(sample-ybar)
b=0.5*sqrt(1-k)
wu=a/b
frac.est2=pnorm(wu)
cum.p2=cumsum(frac.est2)/cum.n

test1=as.integer(frac.est>UCL.p1)
for(i in 1:n)
{if (test1[i]==1) {RL1=1;break}
else{RL1=0}
}
RL1

test2=as.integer(frac.est2>UCL.p2)
for(j in 1:n)
{if (test2[j]==1) {RL2=1;break}
else{RL2=0}
}
RL2

test3=as.integer(cum.p>UCL.p3)
for(k in 1:n)
{if (test3[k]==1) {RL3=1;break}
else{RL3=0}
}
RL3

test4=as.integer(cum.p2>UCL.p4)
for(m in 1:n)
{if (test4[m]==1) {RL4=1;break}
else{RL4=0}
}
RL4
cbind(RL1,RL2,RL3,RL4)
}

t=t(replicate(reps,signal(n,p,mu))[1,,])
tt=c(sum(t[,1]),sum(t[,2]),sum(t[,3]),sum(t[,4]))
prob_g=tt/reps
prob_g
}

mu_list=seq(0,2,0.05)
resultssig=mapply(FUN=Prob_sig,p=0.03,mu=mu_list)


#plot q vs mean shift
plot(mu_list, resultssig[1,],xlim = c(0, 2),ylim=c(0,1),
xlab=expression(mu), ylab="q",pch=NA,cex.axis = 1.5,cex.lab=1.5)
lines(smooth.spline(mu_list, resultssig[1,], spar=0.5),type="l",lty=1,lwd=2,col="black")
par(new=T)
plot(mu_list, resultssig[2,],xlim = c(0, 2),ylim=c(0,1),
xlab=expression(mu), ylab="q",pch=NA,cex.axis = 1.5,cex.lab=1.5)
lines(smooth.spline(mu_list, resultssig[2,], spar=0.5),type="l",lty=2,lwd=2,col="black")
par(new=T)
plot(mu_list, resultssig[3,],xlim = c(0, 2),ylim=c(0,1),
xlab=expression(mu), ylab="q",pch=NA,cex.axis = 1.5,cex.lab=1.5)
lines(smooth.spline(mu_list, resultssig[3,], spar=0.5),type="l",lty=1,lwd=1,col="red")
par(new=T)
plot(mu_list, resultssig[4,],xlim = c(0, 2),ylim=c(0,1),
xlab=expression(mu), ylab="q",pch=NA,cex.axis = 1.5,cex.lab=1.5)
lines(smooth.spline(mu_list, resultssig[4,], spar=0.5),type="l",lty=2,lwd=1,col="red")
legend("bottomright",c(expression(hat(p)[iu]),expression(hat(p)[ic]),
expression(hat(p)[au]),expression(hat(p)[ac])),col=c("black","black","red","red"),
lty=c(1,2,1,2),lwd=c(2,2,1,1),pch=c(NA,NA), merge=FALSE ,cex = 1.1, y.intersp=2,
bty = "n")
title(paste0("q curves for"," N=",n,", FAR=",1-q))
```