**Supplementary Material**

**S.1. Algorithm for the calibration of the adjusted t-based method**

The calibration is to find $λ\_{adj}$ as an adjustment to $λ=0.5$. The bootstrap $λ\_{adj}$ for the lower confidence bound of $π$ can be obtained by the following algorithm.

1. For a given data set, calculate statistics ($̿\_{ij}$, $s\_{1}^{2}$, …, $s\_{4}^{2}$).
2. Simulate $B$ parametric bootstrap samples according to the fitted normal distribution and chi-squared distributions.
3. For a required confidence level $(1-α)$*,* initialize a pair of values ($λ\_{L}$*,* $λ\_{U}$) such that it contains 0.5. We start with $λ\_{L}=0.999$ and $λ\_{U}=0.001$.
4. Compute the corresponding empirical coverage probabilities $CP\_{λ\_{L}}$ and $CP\_{λ\_{U}}$ for the given $λ\_{L}$ and $λ\_{U}$, respectively, using the $B$ bootstrap samples generated in Step 2 by Equation (6).
5. Reset the values of $λ\_{L}$ and $λ\_{U}$ and recalculate the corresponding empirical coverage probability by Step 4 until $CP\_{λ\_{U}}-CP\_{λ\_{L}}\leq 0.05$ (a prespeciﬁed precision) or $\left|λ\_{L}-λ\_{U}\right|\leq 0.01$ (a prespeciﬁed value). If the above criterion is not satisfied, we let $λ\_{M}=(λ\_{L}+λ\_{U})/2$. Then, replace $λ\_{U}$ with $λ\_{M}$ and keep $λ\_{L}$ as the current value if $CP\_{λ\_{U}}\geq \left(1-α\right)$. Otherwise, keep $λ\_{U}$ as the current value and replace $λ\_{L}$ with $λ\_{M}$.
6. Determine

$λ\_{adj}=λ\_{L}+\frac{\left(λ\_{U}-λ\_{L}\right)(1-α-CP\_{λ\_{L}})}{CP\_{λ\_{U}}- CP\_{λ\_{L}}}$.

**S.2. R code for computing the point estimate and the lower confidence bound for**$ π$ **(Use the data set at 5:00 am in March as an illustration)**

B=5000

alpha=0.1

L=17.45

t=3

s=5

muhat=15.76

n=matrix(c(31,31,31,30,30,31,31,31,31,31,31,31,31,31,31),t,s)

KHbar=(t\*s)/sum(1/n)

S2=c(156.2097,177.6743,15.1103,17.5411)

df=c(t-1,s-1,(t-1)\*(s-1),sum(n)-t\*s)

h=c(1/(s\*KHbar),1/(t\*KHbar),(1-1/s-1/t)/KHbar,(KHbar-1)/KHbar)

c=c(1/(t\*s\*KHbar),1/(t\*s\*KHbar),-1/(t\*s\*KHbar),0)

tauhat2=sum(h\*S2)

### the estimate for pi ###

pihat=1-pnorm((L-muhat)/sqrt(tauhat2))

### the lower confidence limit for pi ###

sigmahat2=sum(c\*S2)

sigmahat2=ifelse(sigmahat2>0,sigmahat2,0)

V=(sum(c\*S2)^2)/sum((c\*S2)^2/df)

K=(sum(h\*S2)^2)/sum((h\*S2)^2/df)

tauhat0.5=K\*tauhat2/qchisq(0.5,K)

st=ifelse(sigmahat2>0,sqrt(sigmahat2)\*qt(1-alpha,V),0)

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

# t-based method #

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

if(pihat<0.2) LB=1-pnorm((L-muhat+st)/sqrt(tauhat0.5))

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

# adjusted-t based method #

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

if(pihat>=0.2){

### Step1 ###

sigmahat2\_T=(S2[1]-S2[3])/(s\*KHbar)

sigmahat2\_S=(S2[2]-S2[3])/(t\*KHbar)

sigmahat2\_TS=(S2[3]-S2[4])/KHbar

sigmahat2\_E=S2[4]

yij\_b=matrix(,t,s)

### Step2 ###

muhat\_b=array(,B)

S2\_b=matrix(,4,B)

for(Nl in 1:B){

 for(i in 1:t)

for(j in 1:s)

yij\_b[i,j]=rnorm(1,muhat,sqrt(max(0,sigmahat2\_T+sigmahat2\_S+sigmahat2\_TS+sigmahat2\_E/n[i,j])))

 yi\_b=array(,t)

 for(i in 1:t)yi\_b[i]=mean(yij\_b[i,])

 yj\_b=array(,s)

 for(j in 1:s)yj\_b[j]=mean(yij\_b[,j])

 muhat\_b[Nl]=mean(yi\_b)

 S2\_b[1,Nl]=s\*KHbar\*sum((yi\_b-muhat\_b[Nl])^2)/df[1]

 S2\_b[2,Nl]=t\*KHbar\*sum((yj\_b-muhat\_b[Nl])^2)/df[2]

 S2\_b[3,Nl]=0

 for(i in 1:t)for(j in 1:s)S2\_b[3,Nl]=S2\_b[3,Nl]+(yij\_b[i,j]-yi\_b[i]-yj\_b[j]+muhat\_b[Nl])^2

 S2\_b[3,Nl]=KHbar\*S2\_b[3,Nl]/df[3]

 }

 S2\_b[4,]=rchisq(B,df[4])\*S2[4]/df[4]

### Step3 ###

lamL=0.999

lamU=0.001

### Step4 ###

tauhat2\_b=array(,B)

sigmahat2\_b=array(,B)

K\_b=array(,B)

V\_b=array(,B)

taulamU=array(,B)

taulamL=array(,B)

bst=array(,B)

for(i in 1:B){

 tauhat2\_b[i]=sum(h\*S2\_b[,i])

 sigmahat2\_b[i]=sum(c\*S2\_b[,i])

 sigmahat2\_b[i]=ifelse(sigmahat2\_b[i]>0,sigmahat2\_b[i],0)

 K\_b[i]=(sum(h\*S2\_b[,i])^2)/sum((h\*S2\_b[,i])^2/df)

 V\_b[i]=(sum(c\*S2\_b[,i])^2)/sum((c\*S2\_b[,i])^2/df)

 taulamU[i]=K\_b[i]\*tauhat2\_b[i]/qchisq(lamU,K\_b[i])

 taulamL[i]=K\_b[i]\*tauhat2\_b[i]/qchisq(lamL,K\_b[i])

 bst[i]=ifelse(sigmahat2\_b[i]>0,sqrt(sigmahat2\_b[i])\*qt(1-alpha,V\_b[i]),0)

}

 pihat\_b=1-pnorm((L-muhat)/sqrt(tauhat2))

 LBlamU=1-pnorm((L-muhat\_b+bst)/sqrt(taulamU))

 LBlamL=1-pnorm((L-muhat\_b+bst)/sqrt(taulamL))

CP\_lamU=sum(LBlamU<=pihat\_b)/B

CP\_lamL=sum(LBlamL<=pihat\_b)/B

### Step5 ###

while(CP\_lamU - CP\_lamL >0.05 & abs(lamU-lamL)>0.01){

 lamM=(lamU+lamL)/2

 tauM=K\_b\*tauhat2\_b/qchisq(lamM,K\_b)

 pilamM=1-pnorm(((L-muhat\_b)+bst)/sqrt(tauM))

 CP\_lamM= sum(pilamM<=pihat\_b)/B

 lamU=ifelse(CP\_lamM >=1-alpha,lamM,lamU)

 lamL=ifelse(CP\_lamM >=1-alpha,lamL,lamM)

 CP\_lamU =ifelse(CP\_lamM >=1-alpha, CP\_lamM, CP\_lamU)

 CP\_lamL =ifelse(CP\_lamM >=1-alpha, CP\_lamL, CP\_lamM)

}

### Step6 ###

lamadj=lamL+(lamU-lamL)\*(1-alpha- CP\_lamL)/( CP\_lamU - CP\_lamL)

lamadj=ifelse(lamadj<0,0,lamadj)

lamadj=ifelse(lamadj>1,1,lamadj)

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

tauhatadj=K\*tauhat2/qchisq(lamadj,K)

LB=1-pnorm((L-muhat+st)/sqrt(tauhatadj))

if(is.na(LB)==T)LB=1-pnorm((L-muhat+st)/sqrt(tauhat0.5))

}

pihat

LB