OPIM 5603 RStudio Session 10
Wednesday, November 6, 2019

Hypothesis Testing and Confidence Intervals
via Sampling Distributions generated in R

```
source("myFunctions.R")     # load custom functions
```

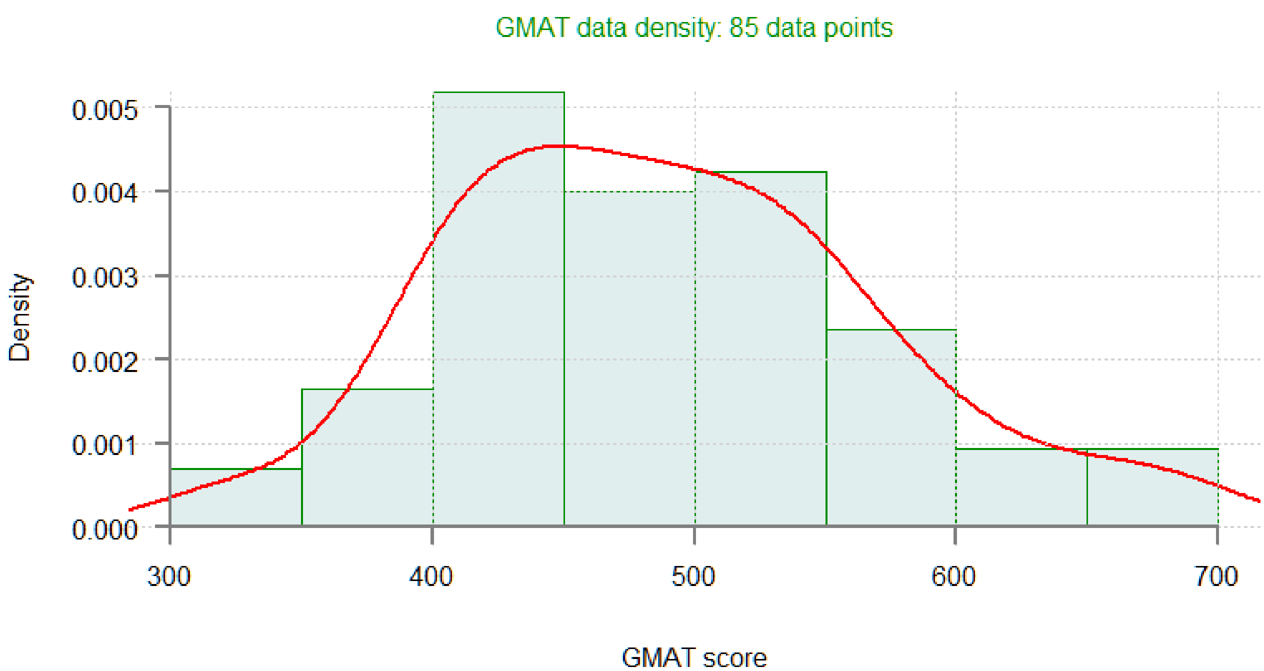
First Approach: Make (a justified) assumption about population distr.

Example 1: (example from Session 8)

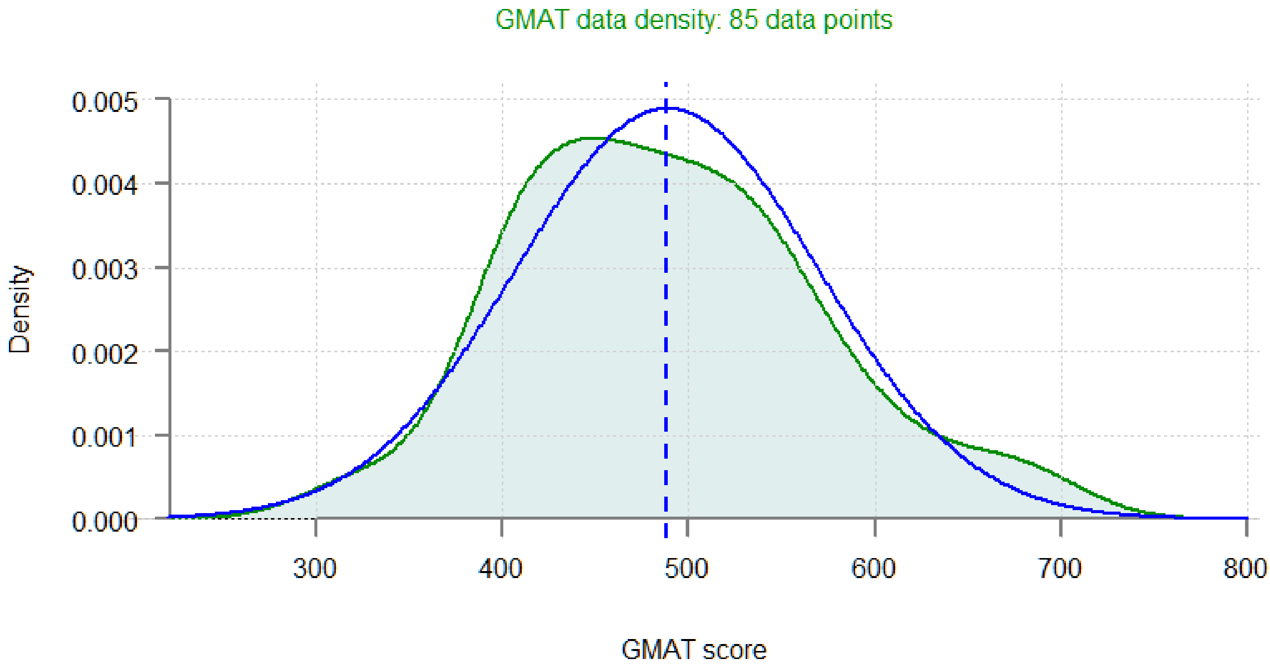
```
# Test the null hypothesis that the population mean of GMAT data is 510.

  adm <- read.csv("Data08/Admission.csv"); head(adm)
  GMAT = adm$GMAT
  summary(GMAT)
```

```
   Min.    1st Qu. Median  Mean    3rd Qu. Max.
   313.0   425.0   482.0   488.4   538.0   693.0
```

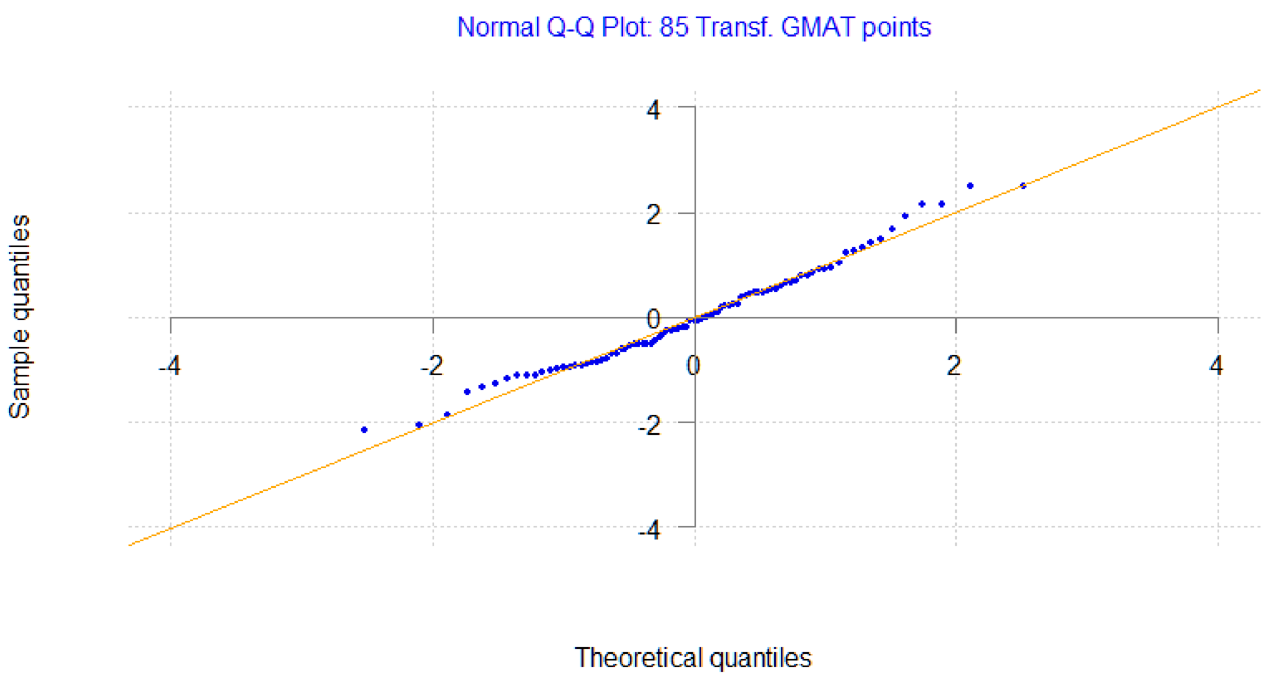
```
# From Session 9: GMAT data density, sample mean, etc.
  myhd(GMAT,ylim=c(0,0.005),col="azure2",border="green4",
       xlab="GMAT score",main="GMAT data density")
```



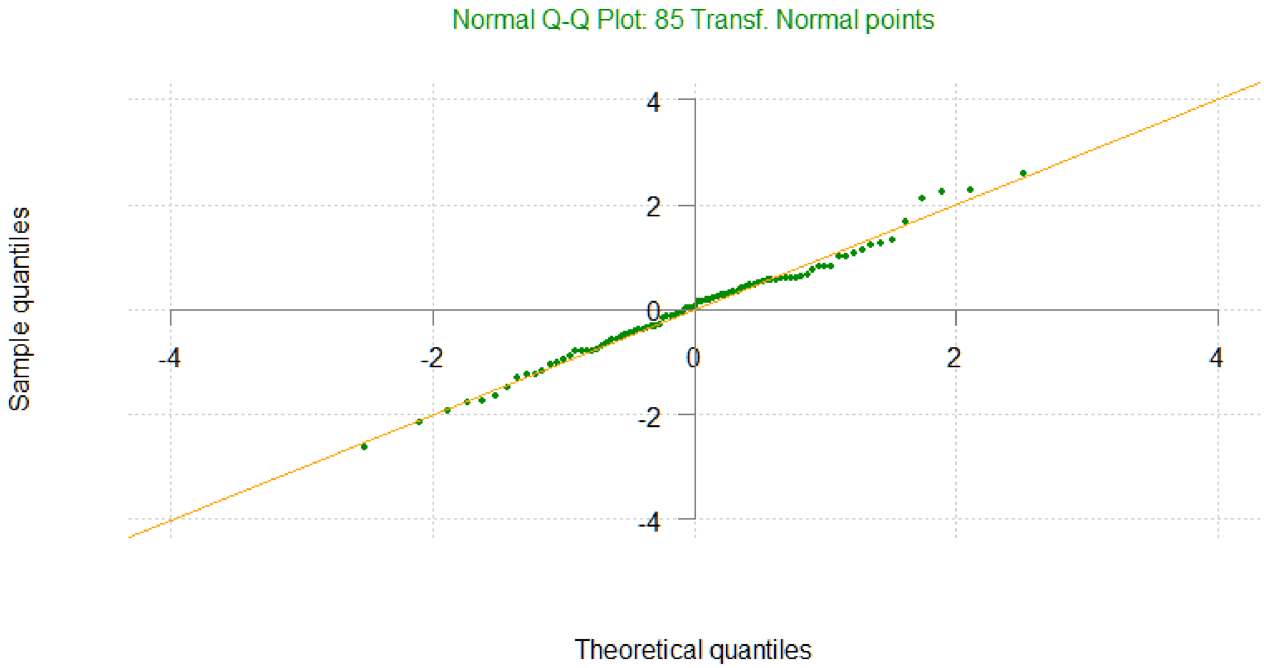

```
  myed(GMAT,ylim=c(0,0.005),pcol="azure2",dcol="green4",
       xlab="GMAT score",main="GMAT data density")
  sm = mean(GMAT); sm
  ssd = sd(GMAT); ssd
  x = seq(220,800)
  lines(x,dnorm(x,mean=sm,sd=ssd),lwd=2,col="blue")
  abline(v=sm,lwd=2,lty=2,col="blue")
```




```
# Is GMAT data normally distributed? Try with a snQQplot:
  snQQplot(trv(GMAT),main="Transf. GMAT")
```

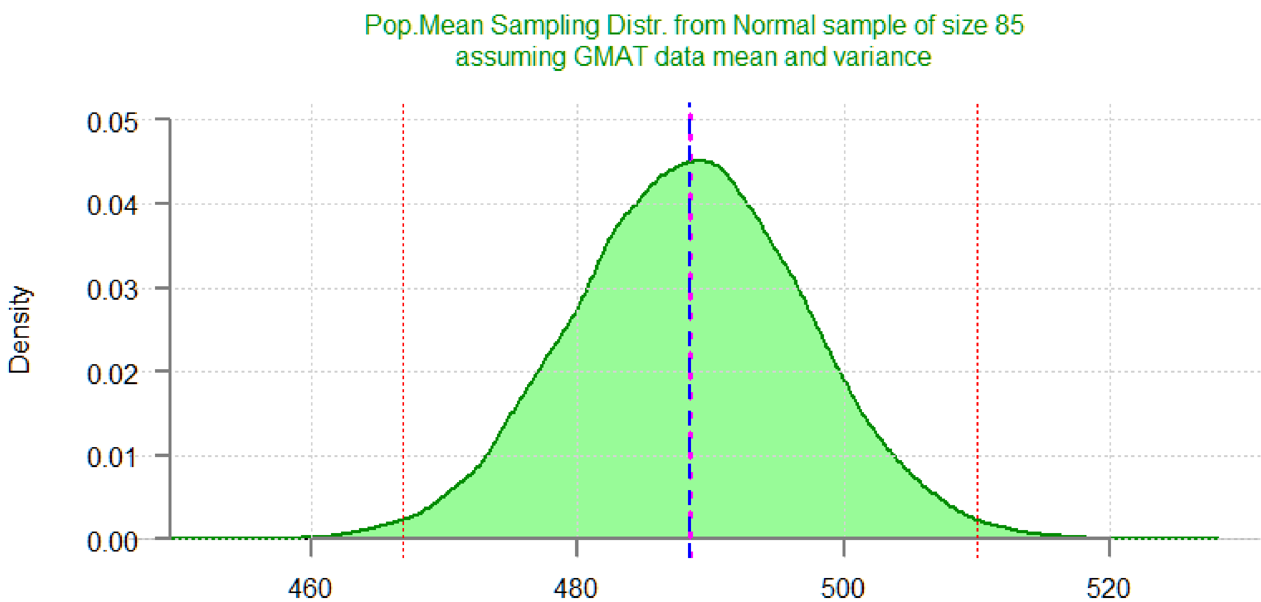



```
# How different is it from the snQQplot of the true Normal random sample?
  snQQplot(trv(rnorm(85,sm,ssd)),col="green4",main="Transf. Normal")
```




```
# KEY: Based on observed GMAT density and the quantile plot WE ASSUME
#      the population distribution is Normal.
#      (discussed in great detail, and justified, in Session 9+)
# The best (unbiased) estimators for population mean and variance are
# Sample Mean and Sample Variance, respectively.
# Thus we assume population distribution is
#              N(sm,ssd^2) = N(488.4,81.5(square))

# The sample size is n = 85 (there are 85 rows in the dataframe).
  n = length(GMAT)

# To create one sample
  rnorm(n,sm,ssd)

# To compute the metric (mean) for one sample
  mean(rnorm(n,sm,ssd))

# Finally, replicate this 50 thousand times to get a
# Sampling Distribution Vector for the Population Mean
  m = 50000
  sdv1a = replicate(m,mean(rnorm(n,sm,ssd)))

# Plot the empirical density of the sampling distribution vector

# Note: 'Sampling Distribution' and 'Sampling Distribution Vector'
#   are synonyms. The generated vector is a 'representative'
#   of the distribution we (usually) don't know exactly.
#   The way we visialize vector 'representing' the distribution
#   is via the empirical density.  Hence it is reasonable to refer to
#   the Empirical Density of Sampling Distribution Vector as
#   'Sampling Distribution' as well.

  myed(sdv1a,cntFlg=F,
      main=paste("Pop.Mean Sampling Distribution from Normal sample of size",n),
      sub="assuming GMAT data mean and variance")

  myed(sdv1a,ylim=c(0,0.05),cntFlg=F,
      main=paste("Pop.Mean Sampling Distr. from Normal sample of size",n),
      sub="assuming GMAT data mean and variance")
```

```
# without custom functions:
  # d_sdv1a = density(sdv1a)
  # plot(d_sdv1a,lwd=3,col="green2",
  #      main=paste("Pop.Mean Sampling Distribution from Normal sample of size",n),
  #      "assuming GMAT data mean and variance")); grid()
  # polygon(d_sdv1a,col="lightgreen")

# Note: in this case we know the true distribution of the sampling
#       distribution: N(488.4, 81.5/sqrt(85))
# Question: what is the "half-mass" point (median)?
  abline(v=median(sdv1a),lwd=3,col="magenta",lty=3)
  # Where does the sm fit in?
  abline(v=sm,lwd=2,lty=2,col="blue")

# Null-hypothesis H0: GMAT population mean is 510
  mu0 = 510

# How far from sm is the mu0?
  abline(v=mu0,lwd=1.5,col="red",lty=3)
  abline(v=sm+sm-mu0,lwd=1.5,col="red",lty=3)
```




```
# Empirical p-value (see the plot)
  left_p = length(sdv1a[sdv1a<sm+sm-mu0])
  right_p = length(sdv1a[sdv1a>mu0])
  pval = (left_p+right_p)/length(sdv1a); pval

# In order to avoid keeping track of which of the two values,
# mu0 and sm+sm-mu0 is larger/smaller,
  lowbd = min(mu0,sm+sm-mu0)     # lower bound
  uppbd = max(mu0,sm+sm-mu0)     # upper bound

# Compute the probabilities 'beyond' these lines to obtain
# the p-values for the hypothesis
  left_p = length(sdv1a[sdv1a<lowbd])
  right_p = length(sdv1a[sdv1a>uppbd])
  pval = (left_p+right_p)/length(sdv1a); pval
```

```
[1] 0.01534
```

```
# We reject the hypothesis at significance level 0.05.
```

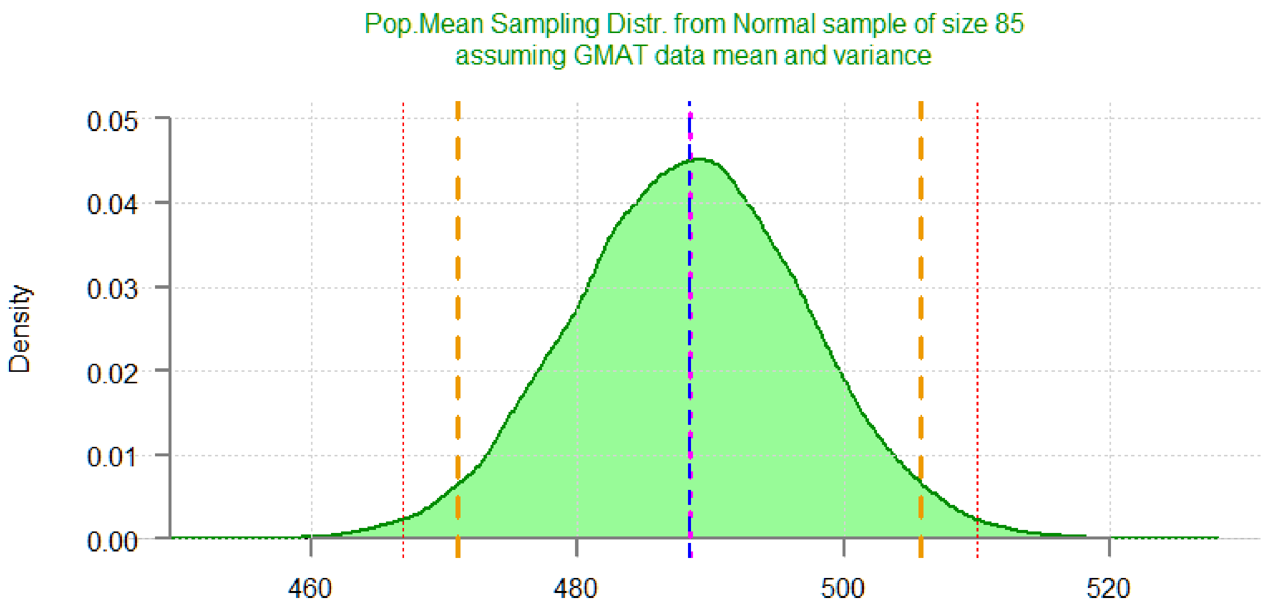
```
# Since this is a test for the population mean, we can compare to built-in t-test:
  t.test(GMAT,mu=mu0,conf.level = 0.95)
```

```
        One Sample t-test
data:  GMAT
t = -2.4375, df = 84, p-value = 0.0169
alternative hypothesis: true mean is not equal to 510
95 percent confidence interval:   470.8631 506.0310
sample estimates:
  mean of x      488.4471
```

```
# Related question: What is the empirical conf.interval for 95% confidence?
  myeCI(sdv1a,0.95)
```

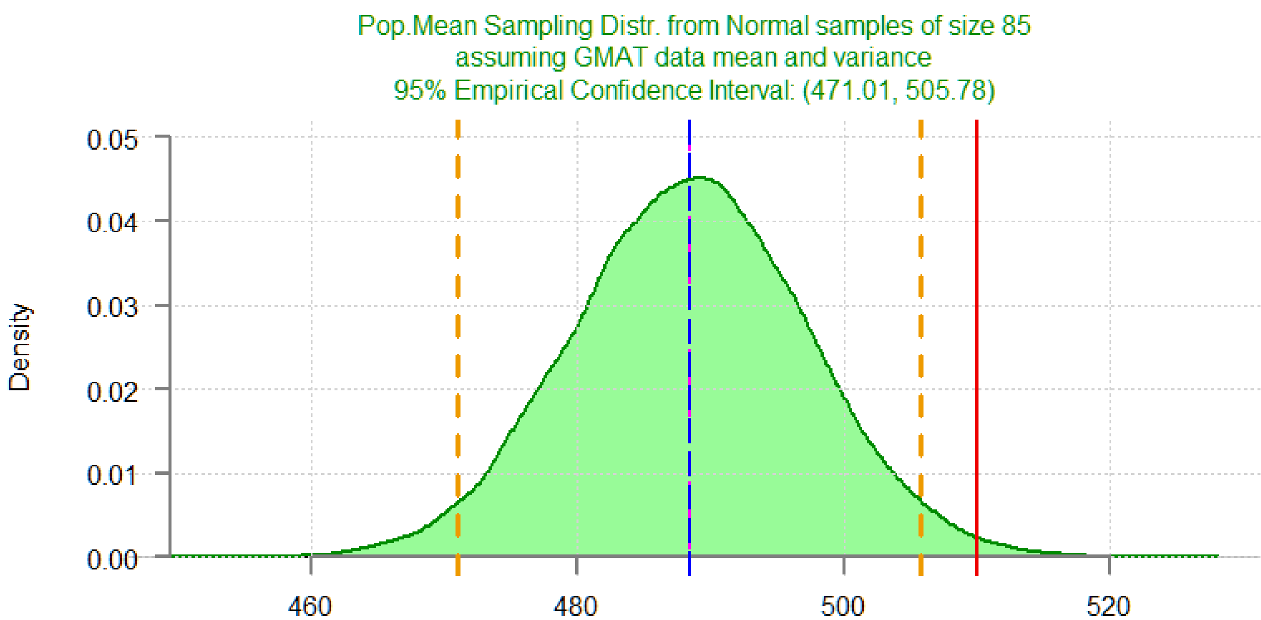
```
    2.5%    97.5%
471.0071 505.7843
```

```
  abline(v=myeCI(sdv1a,0.95),lwd=3,lty=2,col="orange2")
```



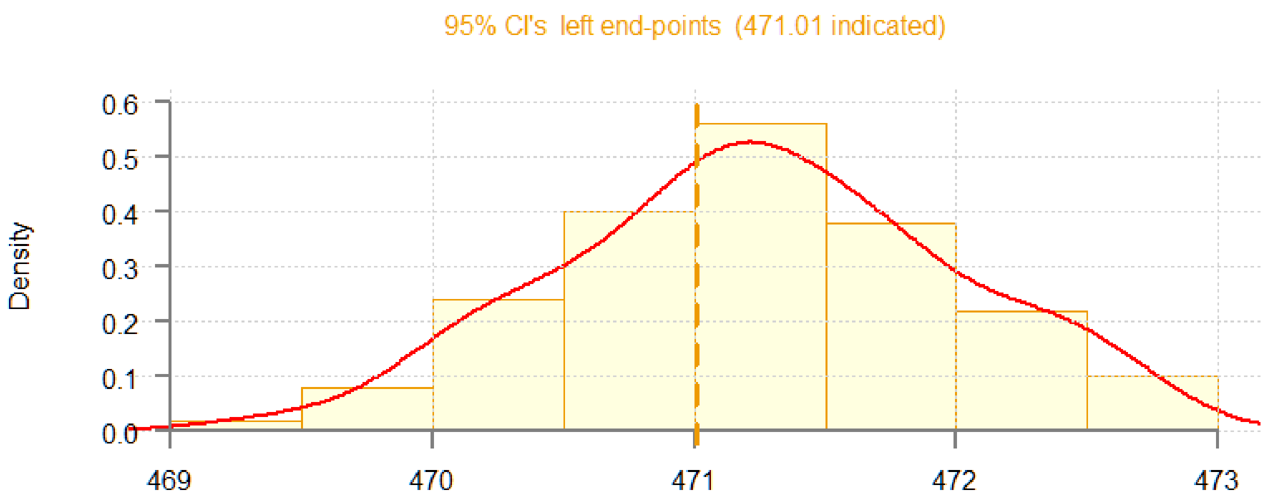

```
# without custom functions:
  # left_ep = quantile(sdv,0.025)
  # right_ep = quantile(sdv,0.975)
  # CI = c(left_ep,right_ep); CI
  # abline(v=CI,lwd=1.5,col="purple")

# As in the classical t-test we shall often AVOID the computation of p-values
# and use empirical confidence interval instead.

# Conclusion: Since the hypothesized value mu0 = 510 is outside the two-sided
#             empirical 95% confidence interval, we reject the null hypothesis
#             in favor of the alternative: Population mean is not equal 510.


# Useful wrap (to avoid cutting and pasting same lines over and over again)
# Function plots the sampling distribution density and marks confidence interval
#   on the plot. It also returns the confidence interval endpoints.
```

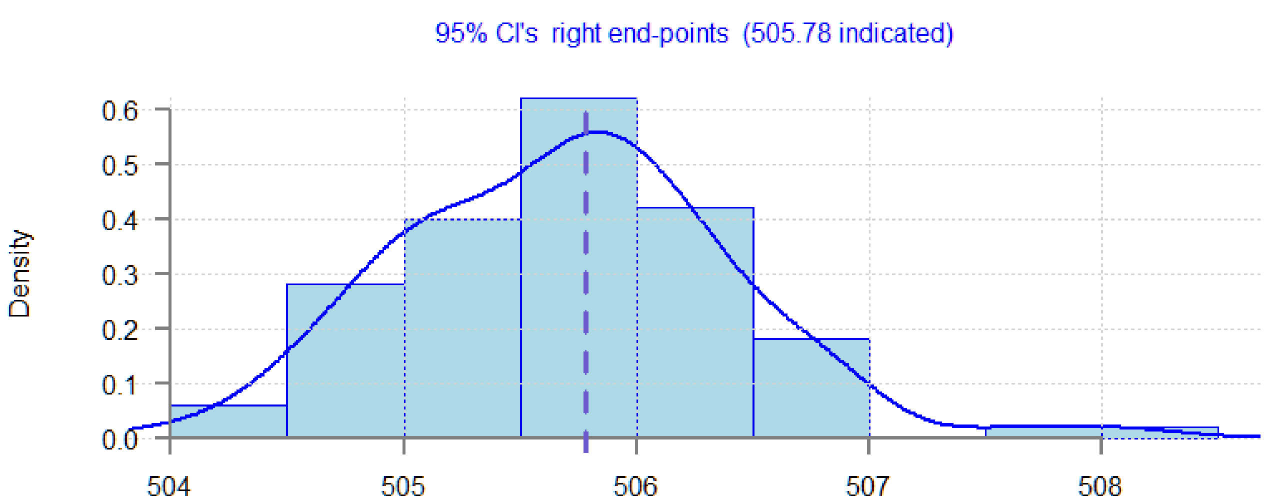
```
  mysdci = function(sdv,xlim=NULL,ylim=NULL,xlab="",main=NULL,sub=NULL,cntFlg=F,
                    pcol="palegreen",dlwd=2,dcol="green4",axes=T,conf=0.95,rnd=2)
  {
    CI = myeCI(sdv,conf)
    ssub = paste(100*conf,"% Empirical Confidence Interval: (",
                 round(CI[1],rnd),", ",round(CI[2],rnd),")",sep="")
    if (!is.null(sub)) sub = c(sub,ssub) else sub = ssub
    myed(sdv,xlim=xlim,ylim=ylim,xlab=xlab,main=main,sub=sub,cntFlg=cntFlg,
         pcol=pcol,dlwd=dlwd,dcol=dcol,axes=axes)
    abline(v=median(sdv),lwd=2,col="magenta",lty=2)
    abline(v=CI,lwd=3,lty=2,col="orange2")
    return(CI)
  }

# Apply it to 'sdv1a' sampling distribution vector:
  mysdci(sdv1a,ylim=c(0,0.05),
         main=paste("Pop.Mean Sampling Distr. from Normal samples of size",n),
         sub="assuming GMAT data mean and variance")
  abline(v=sm,lwd=2,col="blue",lty=5)        # GMAT data sample mean (average)
  abline(v=mu0,lwd=2,col="red2")             # hypothesized value: mu0 = 510
```




```
# Note: In classical theory CI for population mean depends only on
#       the sample and the confidence level.
# In this case the empirical CI depends on the generated sdv.
# How much of the variation should we see?
# We can certainly repeat CI calculations say M = 100 times.
# But we must reduce m to avoid long execution times

  m = 1000; M = 100
  CIsdv1a = rpl(M,myeCI(rpl(m,mean(rnorm(n,sm,ssd)))))

  par(mfrow=c(2,1))
  myhd(CIsdv1a[1,],breaks=10,cntFlg=F,main="95% CI's left end-points")
  myhd(CIsdv1a[2,],breaks=10,cntFlg=F,main="95% CI's right end-points",
       col="lightblue",border="blue2",dcol="blue")
  par(mfrow=c(1,1))

  # wrap it as an auxiliary function:
```

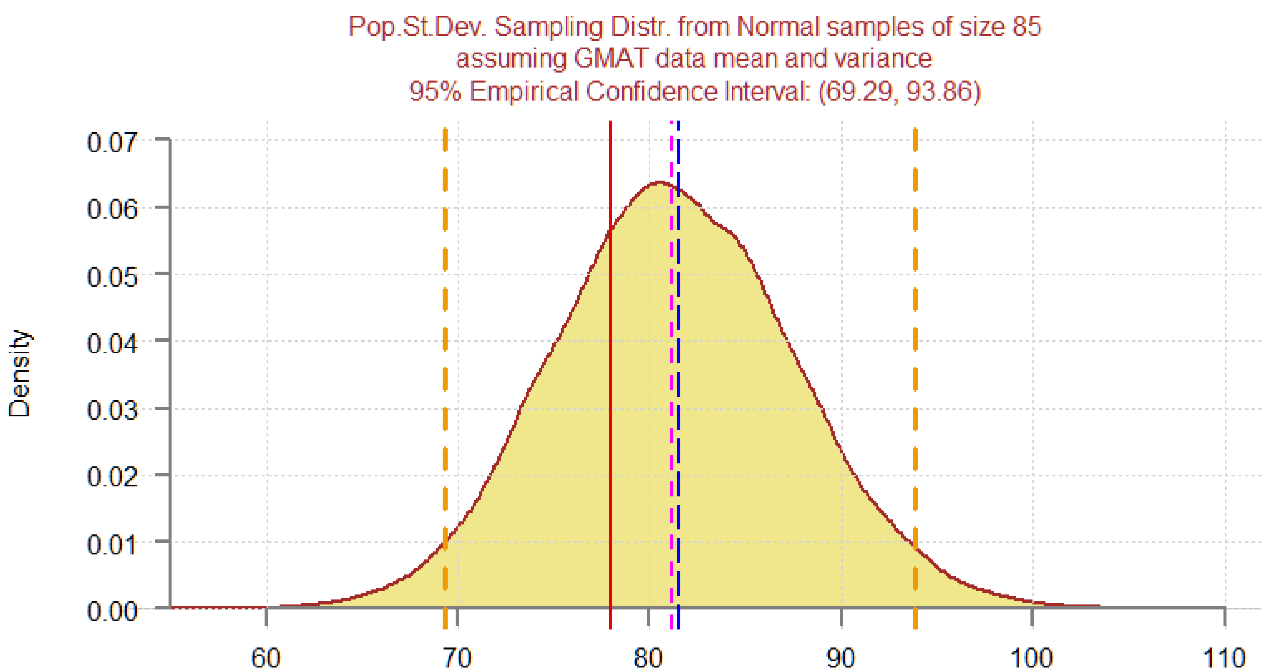
```
  myCIsdv = function(CIsdv,marker=NULL,ylim=NULL,conf=0.95,rnd=2)
{
  par(mfrow=c(2,1))
  b = ceiling(sqrt(length(CIsdv)/2))
  main = paste(100*conf,"% CI's ",sep=""); sub=NULL
  if (!is.null(marker))
    sub = paste(" (",round(marker[1],rnd)," indicated)",sep="")
  myhd(CIsdv[1,],breaks=b,ylim=ylim,cntFlg=F,main=paste(main,"left end-points",sub))
  abline(v=marker[1],lwd=3,lty=2,col="orange2")
  if (!is.null(marker))
    sub = paste(" (",round(marker[2],rnd)," indicated)",sep="")
  myhd(CIsdv[2,],breaks=b,ylim=ylim,cntFlg=F,main=paste(main,"right end-points",sub),
       col="lightblue",border="blue2",dcol="blue")
  abline(v=marker[2],lwd=3,lty=2,col="slateblue")
  par(mfrow=c(1,1))
}

myCIsdv(CIsdv1a,myeCI(sdv1a))
myCIsdv(CIsdv1a,myeCI(sdv1a),ylim=c(0,0.6))
```



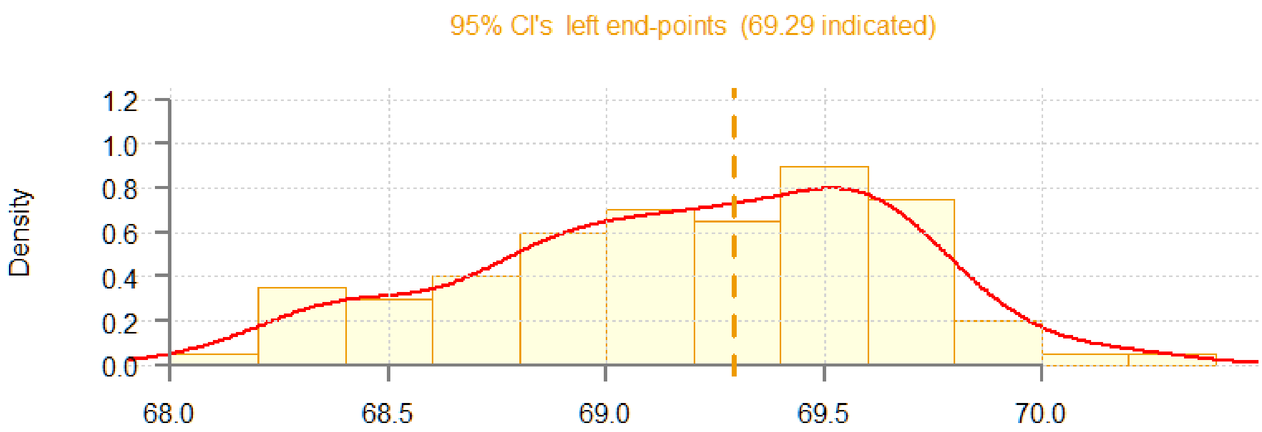


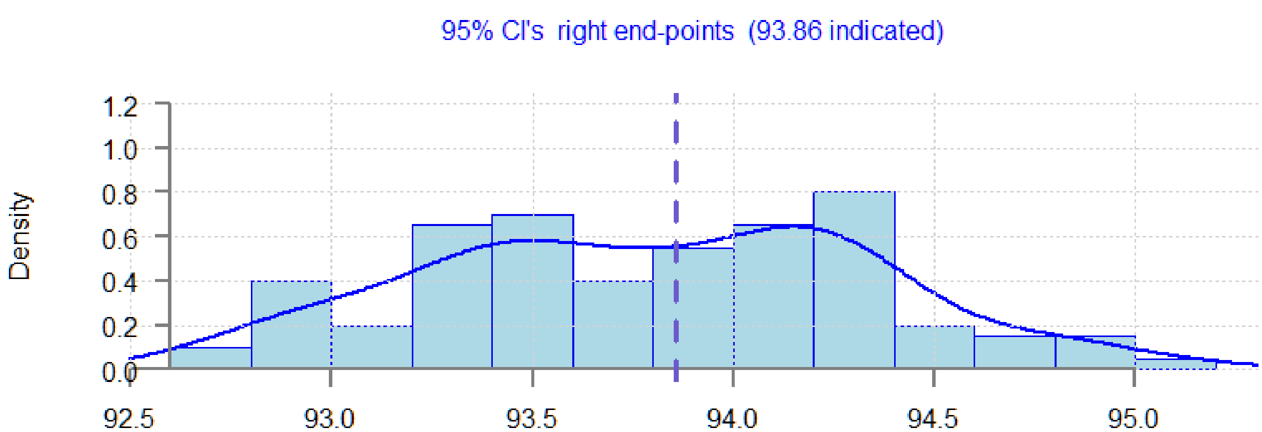
Example 2:

```
# Empirical Confidence Interval for population standard deviation
# "Null-hypothesis: population standard deviation is 78"

# Lineup as before: Sample size, mean, and standard deviation of GMAT data
  # n = length(GMAT); n
  # sm = mean(GMAT); sm
  # ssd = sd(GMAT); ssd

# Hypothesized Population Standard Deviation value
  sd0 = 78

# "Metric" is the  sample standard deviation: we need to replicate it many times
  sd(rnorm(n,sm,ssd))

# Get the Sample Standard Deviation of 50,000 replicates
#   (stored as sampling distribution vector) and plot its density
  m = 50000
  sdv2a = replicate(m,sd(rnorm(n,sm,ssd)))
  mysdci(sdv2a,ylim=c(0,0.07),pcol="khaki",dcol="brown",
         main=paste("Pop.St.Dev. Sampling Distr. from Normal samples of size",n),
         sub="assuming GMAT data mean and variance")
  abline(v=sd0,lwd=2,col="red2")          # hypothesized value: sd0 = 78
  abline(v=ssd,lwd=2,col="blue",lty=5)     # sample sd of GMAT data
```




```
# Test conclusion: We accept the null-hypothesis at significance level 0.05.

# CI variation:
  m = 1000; M = 100
  CIsdv2a = rpl(M,myeCI(rpl(m,sd(rnorm(n,sm,ssd)))))
  myCIsdv(CIsdv2a,myeCI(sdv2a))
  myCIsdv(CIsdv2a,myeCI(sdv2a),ylim=c(0,1.2))
```

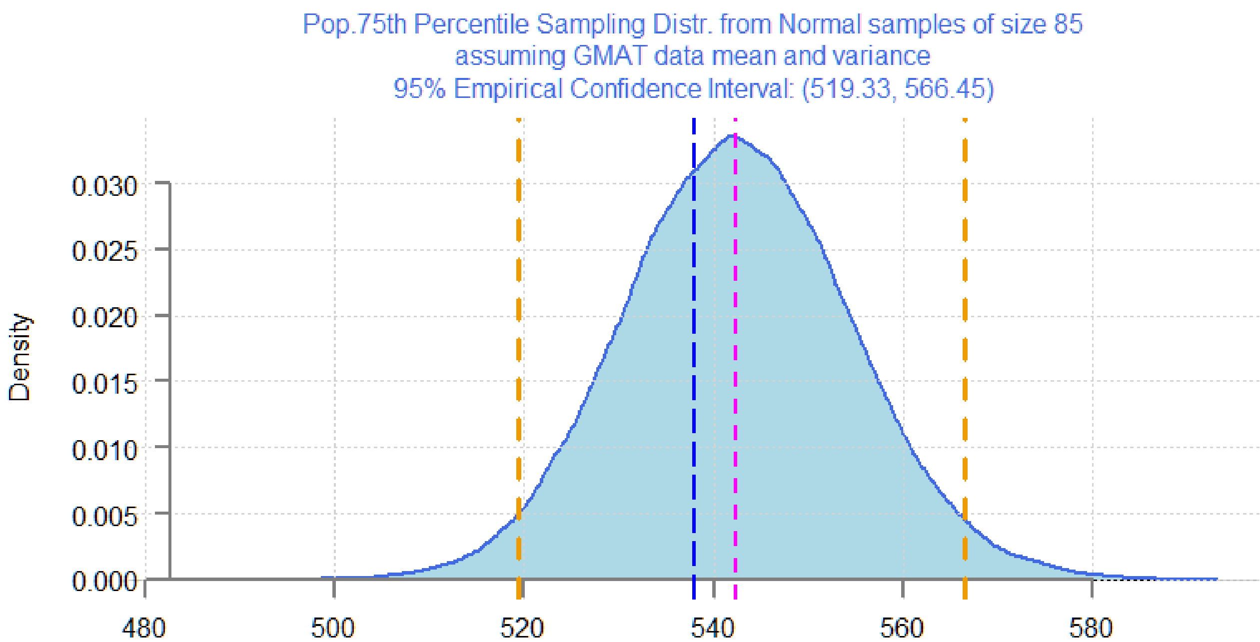




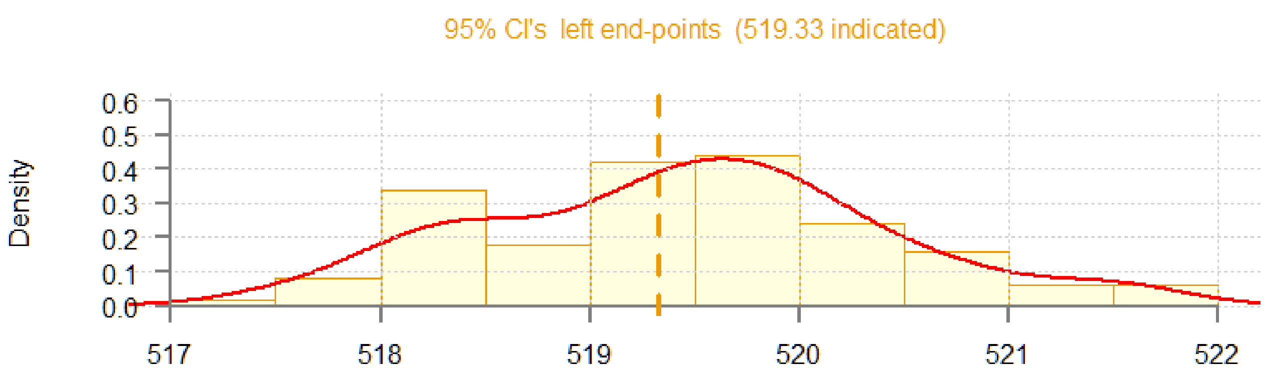
Example 3:

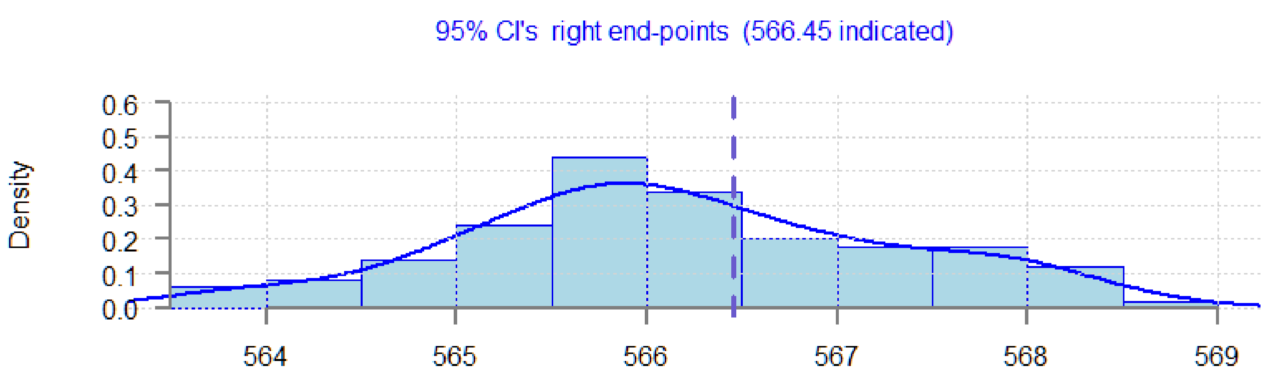
```
# Empirical Confidence Interval for the population 75th percentile
# "Null-hypothesis: Population 75th percentile is 600"

  qnt = 0.75          # 75th percentile = 0.75 quantile
# qnt = 0.99          # 99th percentile
# qnt = 1             # maximum

# Sample size, mean, standard deviation, and the desired percentile of GMAT data
  # n = length(GMAT); n
  # sm = mean(GMAT); sm
  # ssd = sd(GMAT); ssd
  sqnt = quantile(GMAT,qnt); sqnt

# Hypothesized Population 75th Percentile (Third Quartile) value
  qnt0 = 600

# "Metric" is qnt quantile
  quantile(rnorm(n,sm,ssd),qnt)

# Get the desired quantile of 50,000 replicates and plot density
  m = 50000
  sdv3a = replicate(m,quantile(rnorm(n,sm,ssd),qnt))
  mysdci(sdv3a,pcol="lightblue",dcol="royalblue",
         main=p("Pop.",100*qnt,"th Percentile Sampling Distr.
                              from Normal samples of size ",n,sep=""),
         sub="assuming GMAT data mean and variance")
```

```
abline(v=qnt0,lwd=2,col="red2")          # hypothesized value: qnt0 = 600
abline(v=sqnt,lwd=2,col="blue",lty=5)    # GMAT data 75th Empirical Percentile
```




```
# Test conclusion: Since 600 is outside the conf interval we reject the null-hypothesis

# CI variation:
  m = 1000; M = 100
  CIsdv3a = rpl(M,myeCI(rpl(m,quantile(rnorm(n,sm,ssd),qnt))))
  myCIsdv(CIsdv3a,myeCI(sdv3a))
  myCIsdv(CIsdv3a,myeCI(sdv3a),ylim=c(0,0.6))
```





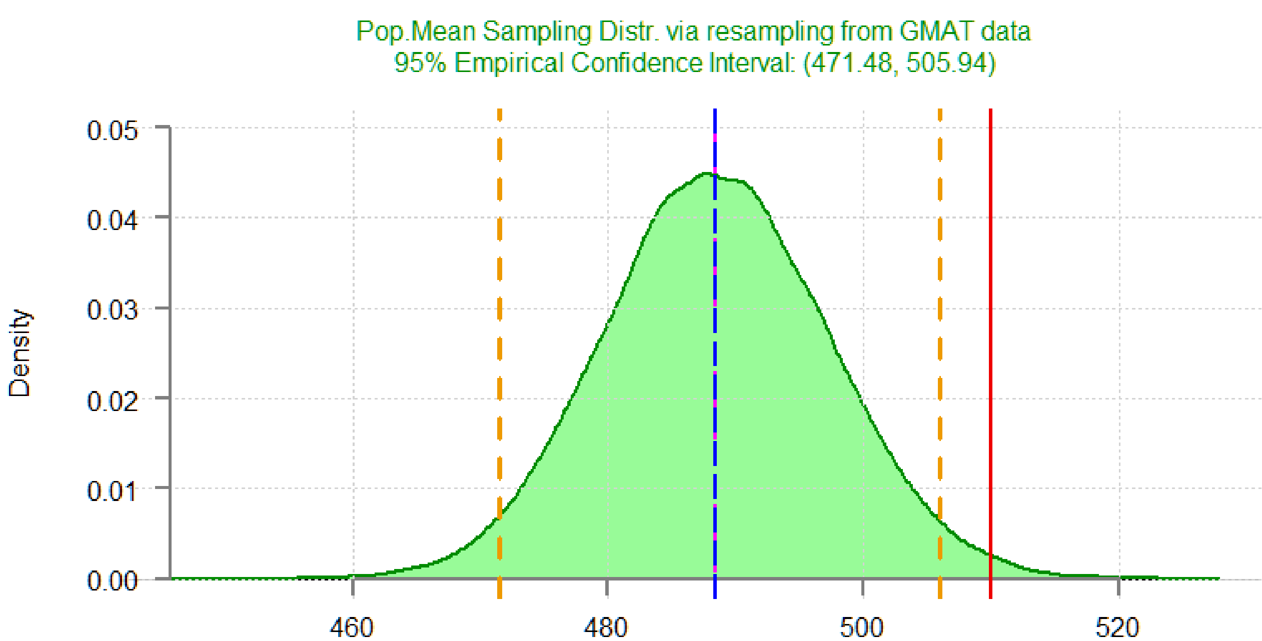

```
# TRY repeating for other percentiles
```

```
-------------------------------------------------------------------------------------

     Second approach: Make NO assumptions about population distribution,
                      instead use only the data points and create
                      simulated samples from these points (re-sampling)

-------------------------------------------------------------------------------------
```

```r
# The sampling distr. will be obtained by "resampling": use the built-in function "sample"
  x = seq(1,5)

  sample(x,5,replace=T)    # we use re-sampling with replacement
  sample(x,5,replace=F)

  sample(x,12,replace=T)
  # sample(x,12,replace=F)
# ------------------------------------------------------------------------------------
```

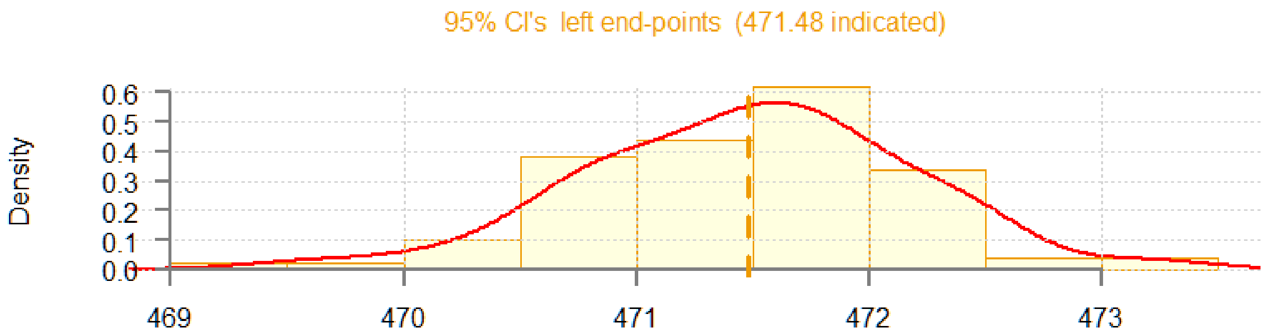
Example 1 via resampling

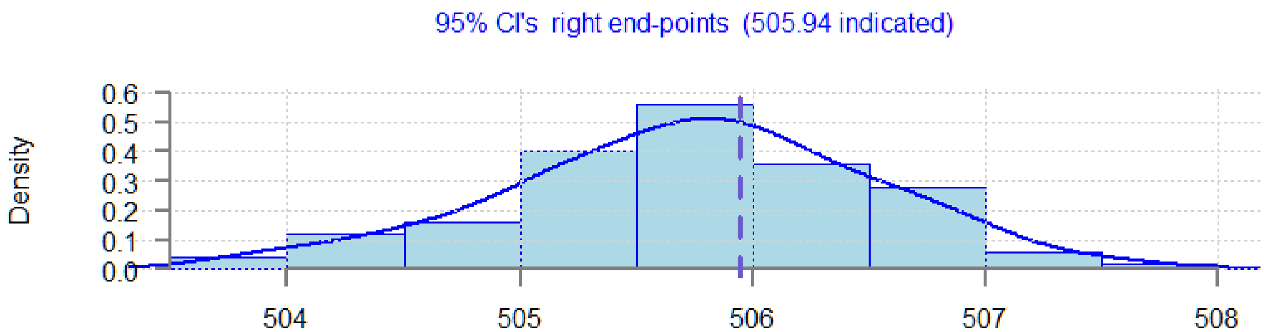
```r
# Empirical Confidence Interval for the population mean
# "Null-hypothesis: the population mean of GMAT data is 510"

# Sample size (note: we do not need sm and ssd anymore)
  n = length(GMAT); n

# In our case, we re-sample from the GMAT vector
  s = sample(GMAT,n,replace=T)

# check the mean of the sample obtained by re-sampling
  mean(s)
# Replicate it
  m = 50000
  sdv1b = replicate(m,mean(sample(GMAT,n,T)))
  # and plot sampling distribution and CI
  mysdci(sdv1b,ylim=c(0,0.05),
         main="Pop.Mean Sampling Distr. via resampling from GMAT data")
  abline(v=mu0,lwd=2,col="red2")          # hypothesized value: mu0 = 510
  abline(v=sm,lwd=2,col="blue",lty=5)     # GMAT data sample mean (average)
```



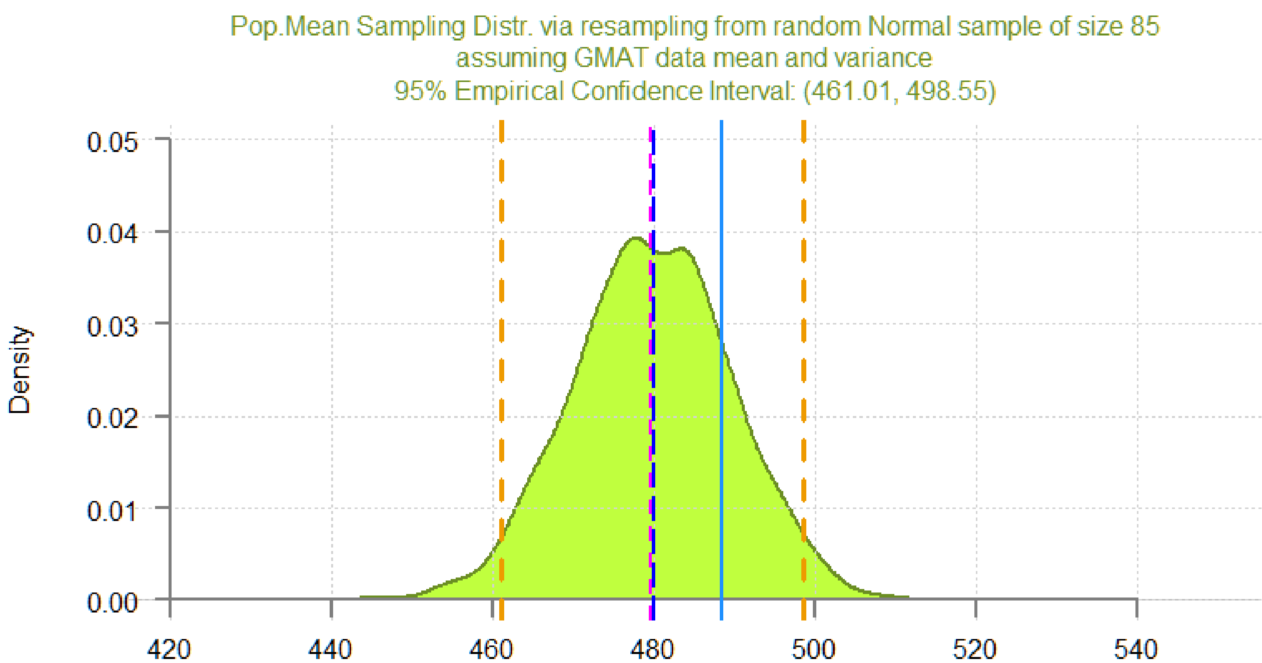

```r
# CI variation:
  m = 1000; M = 100
  CIsdv1b = rpl(M,myeCI(rpl(m,mean(sample(GMAT,n,T)))))
```

```
myCIsdv(CIsdv1b,myeCI(sdv1b))
myCIsdv(CIsdv1b,myeCI(sdv1b),ylim=c(0,0.6))
```



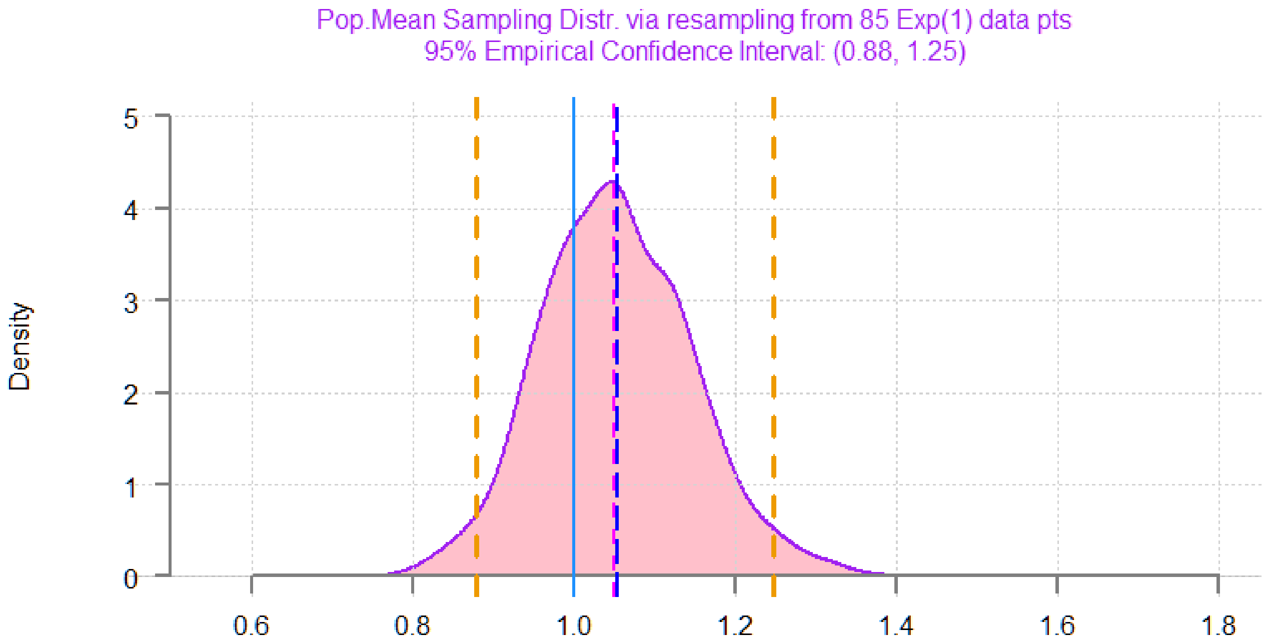



```
# Try it for another N(sm,ssd^2) vector
  w = rnorm(n,sm,ssd)
  sdvn = replicate(m,mean(sample(w,n,T)))
  mysdci(sdvn,pcol="olivedrab1",dcol="olivedrab",xlim=c(420,550),ylim=c(0,0.05),
         main=p("Pop.Mean Sampling Distr. via resampling from random Normal sample of size",n),
         sub="assuming GMAT data mean and variance")
  abline(v=mean(w),lwd=2,col="blue",lty=5)       # w sample mean (average)
  abline(v=sm,lwd=2,col="dodgerblue",lty=1)      # GMAT average
```



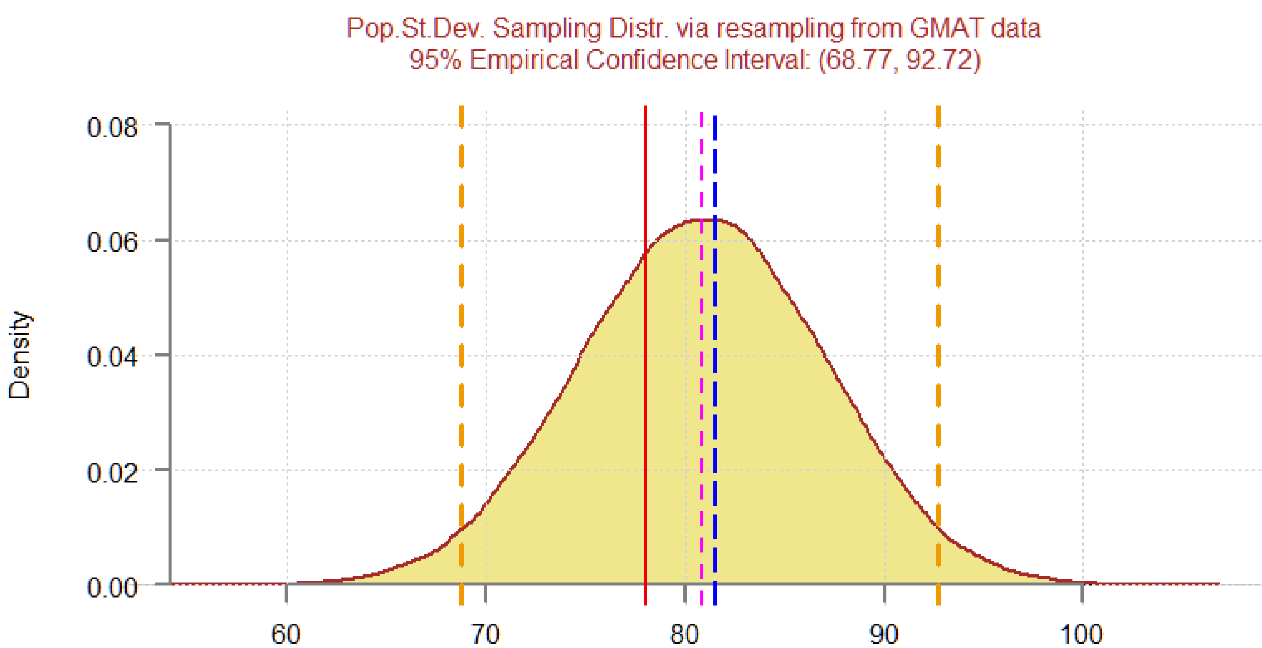

```
# Try it for another population distribution
```

```
w = rexp(n)
sdve = replicate(m,mean(sample(w,n,T)))
mysdci(sdve,pcol="pink",dcol="purple",xlim=c(0.5,1.8),ylim=c(0,5),
       main=p("Pop.Mean Sampling Distr. via resampling from",n,"Exp(1) data pts"))
abline(v=mean(w),lwd=2,col="blue",lty=5)       # w sample mean (average)
abline(v=1,lwd=2,col="dodgerblue",lty=1)       # pop mean of Exp(1)
```

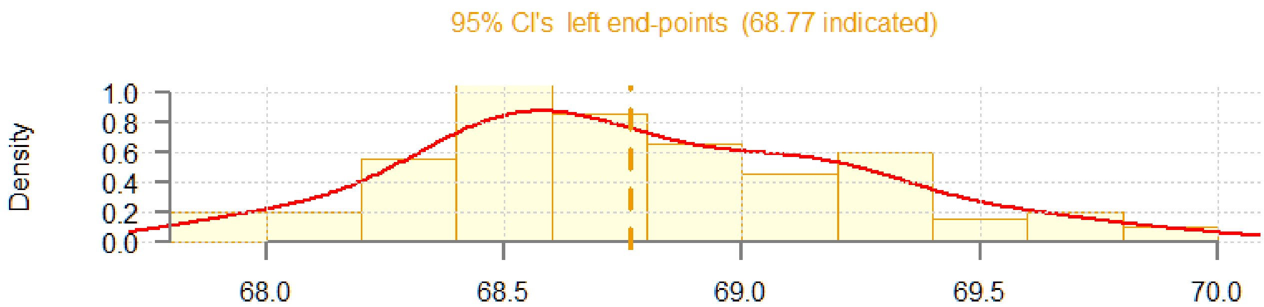


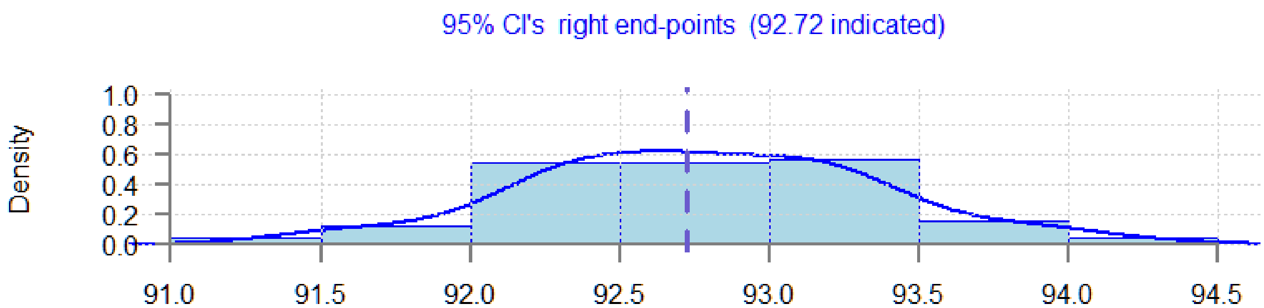
Example 2 via resampling

```
# Empirical Confidence Interval for the population standard deviation
# "Null-hypothesis: population standard deviation is 78"

  m = 50000
  sdv2b = replicate(m,sd(sample(GMAT,n,T)))
  mysdci(sdv2b,ylim=c(0,0.08),pcol="khaki",dcol="brown",
         main="Pop.St.Dev. Sampling Distr. via resampling from GMAT data")
  abline(v=sd0,lwd=2,col="red2")            # hypothesized value: sd0 = 78
  abline(v=ssd,lwd=2,col="blue",lty=5)      # GMAT data Sample Standard Deviation
```



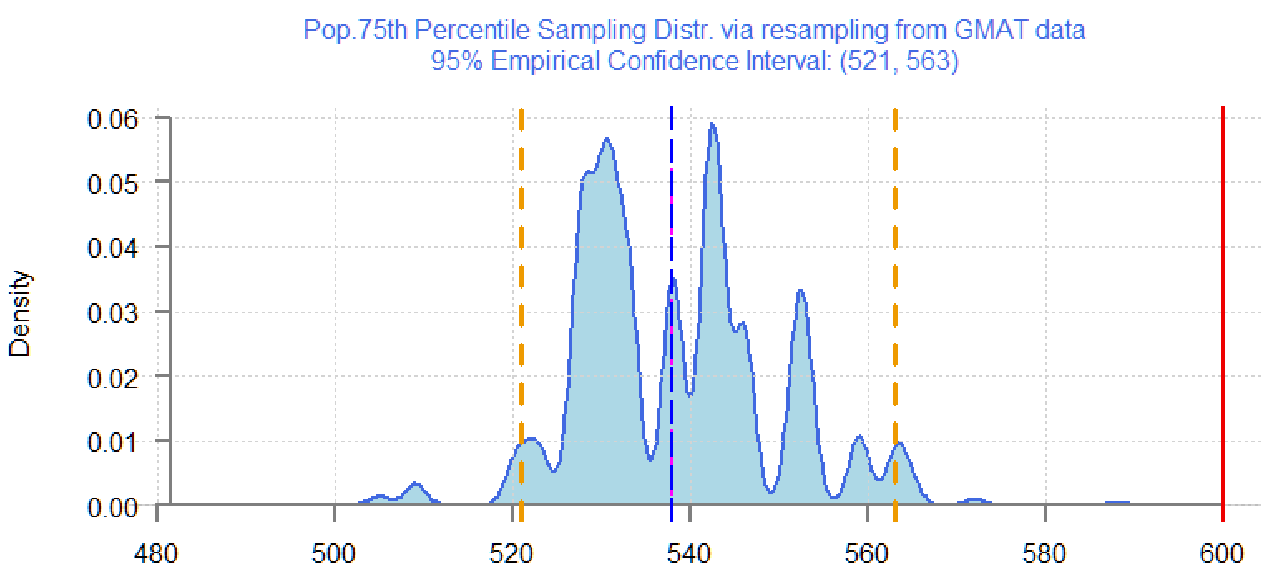
```
# CI variation:
  m = 1000; M = 100
  CIsdv2b = rpl(M,myeCI(rpl(m,sd(sample(GMAT,n,T)))))
  myCIsdv(CIsdv2b,myeCI(sdv2b))
  myCIsdv(CIsdv2b,myeCI(sdv2b),ylim=c(0,1))
```



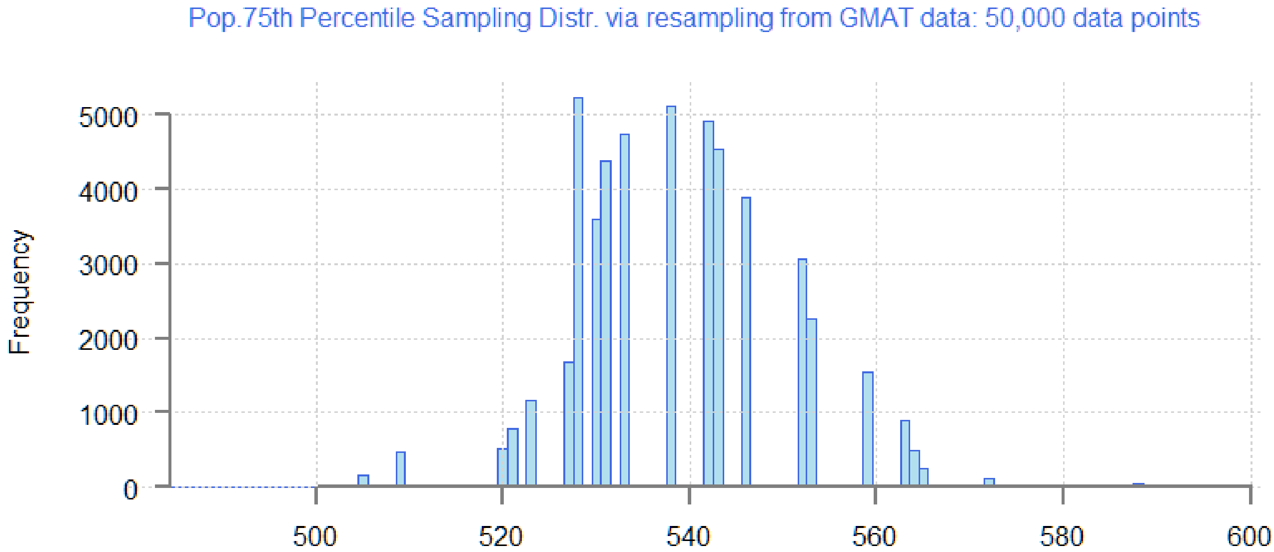


Example 3 via resampling

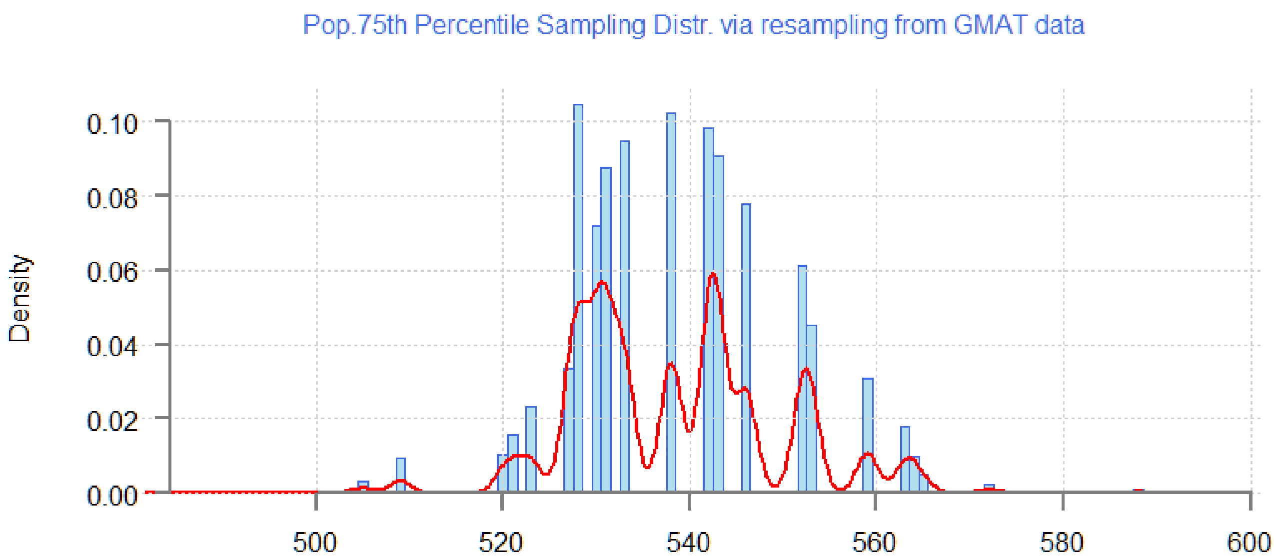
```
# Empirical Confidence Interval for the population 75th percentile
# "Null-hypothesis: 75th percentile of the population is 600"

  qnt = 0.75               # qnt = 0.5
  m = 50000

  sdv3b = replicate(m,quantile(sample(GMAT,n,T),qnt))
  # auxiliary string for the super-long chart title
  main = p("Pop.",100*qnt,
           "th Percentile Sampling Distr. via resampling from GMAT data",sep="")
  mysdci(sdv3b,pcol="lightblue",dcol="royalblue",main=main)
  abline(v=qnt0,lwd=2,col="red2")          # hypothesized value: qnt0 = 600
  abline(v=sqnt,lwd=2,col="blue",lty=5)    # GMAT data 75th Empirical Percentile
```



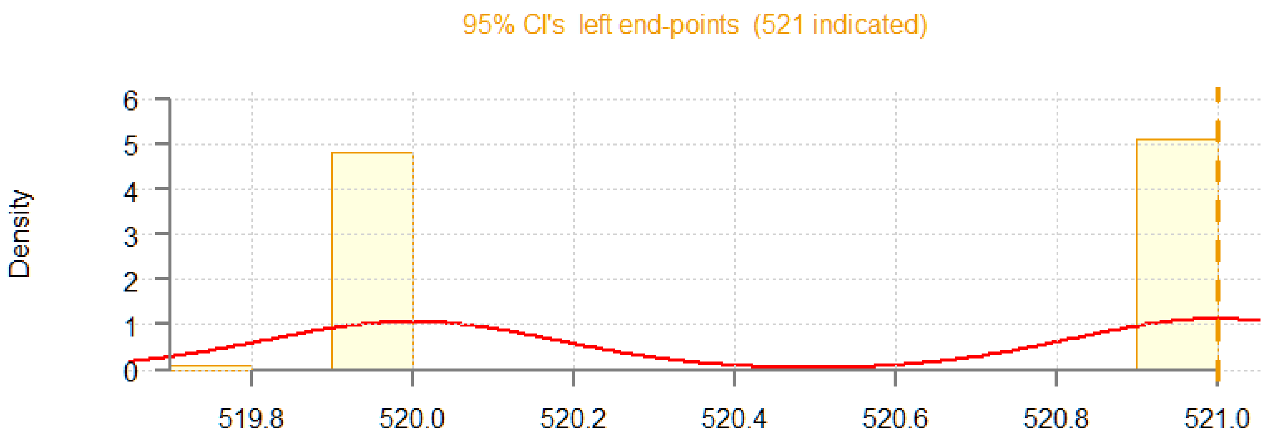
```
# Why such bumpiness?
  summary(sdv3b)
# Key: How many distinct values does vector 'sampldist' have?
  length(sdv3b); length(unique(sdv3b))

# Empirical density does a poor job in representing sparse data; try histogram
  myhist(sdv3b,freq=T,main=main,col="lightblue2",border="royalblue")
  myhist(sdv3b,freq=T,breaks=seq(min(sdv3b)-0.5,max(sdv3b)+0.5),
         main=main,col="lightblue2",border="royalblue")
```

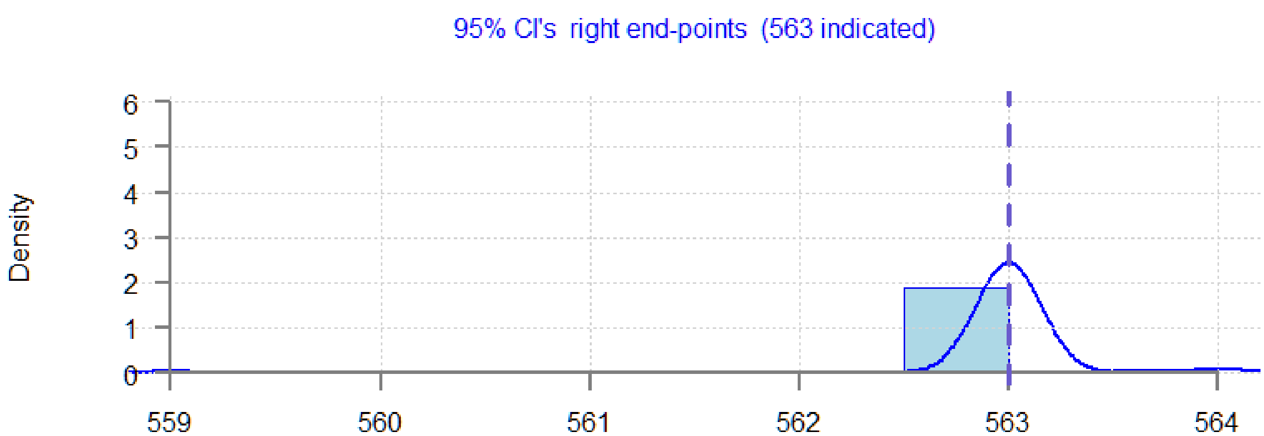



```
  myhd(sdv3b,breaks=seq(min(sdv3b)-0.5,max(sdv3b)+0.5),cntFlg=F,main=main,
      col="lightblue2",border="royalblue")
```




```
  myhd2(sdv3b,cntFlg=F,main=main,col="lightblue2",border="royalblue")

# sampling distribution for percentiles is very bumpy; What about CIs?

# CI variation:
  m = 1000; M = 100
  CIsdv3b = rpl(M,myeCI(rpl(m,quantile(sample(GMAT,n,T),qnt))))
  myCIsdv(CIsdv3b,myeCI(sdv3b))
  myCIsdv(CIsdv3b,myeCI(sdv3b),ylim=c(0,6))
```





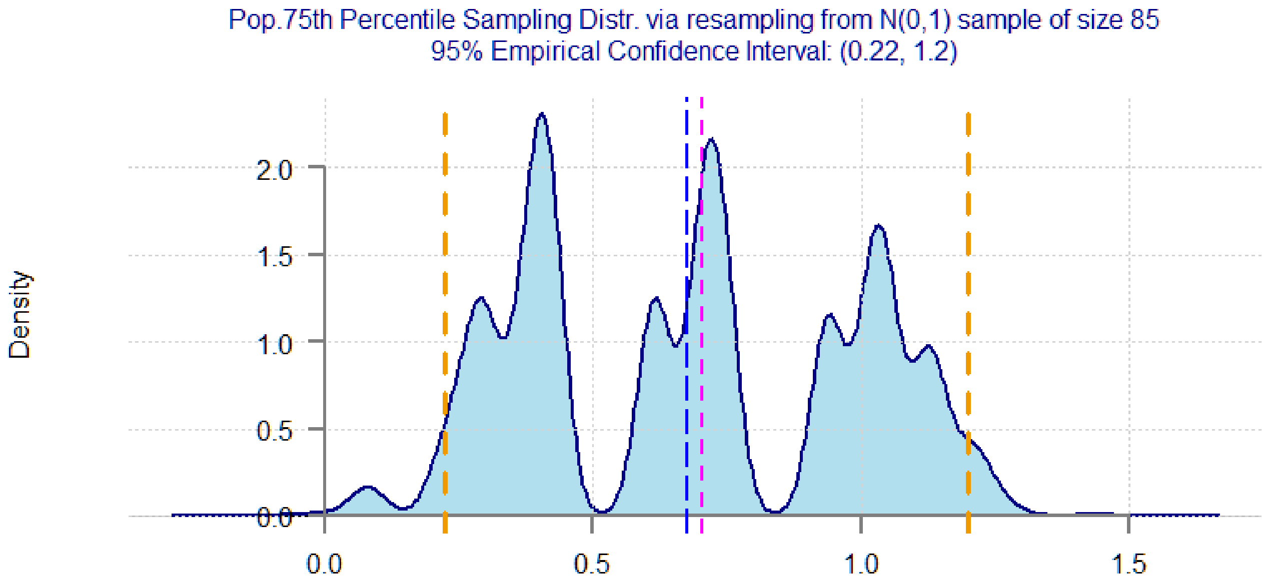

```
# Unusual distribution? Check the actual frequencies:
  table(CIsdv3b[1,])   # Frequency table of left end-points
```

```
    519.725      520  520.975      521
          1       48        4       47
```

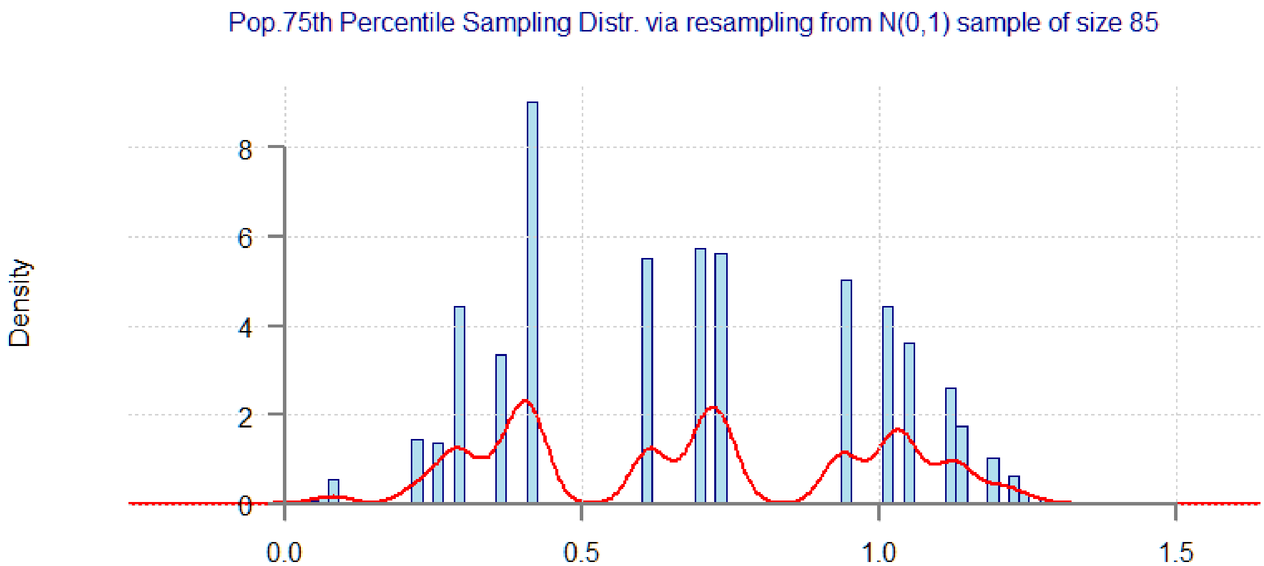
```
  table(CIsdv3b[2,])   # Frequency table of rightt end-points
```

```
    559      563  563.025      564
      1       95        1        3
```

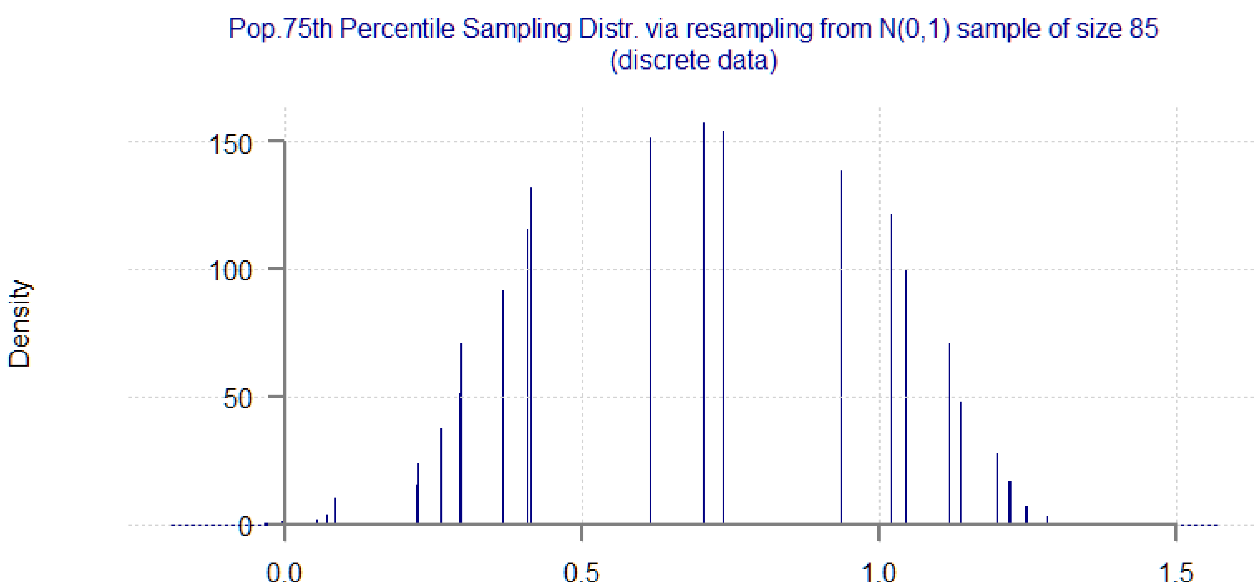
```
# Recall: sampling distribution for percentiles is very bumpy: this will be the
#         case for all percentiles: try median (uncomment "qnt = 0.5" line)

# Try it for a Normal N(0,1) sample of size 85 = length(GMAT)
  n = length(GMAT)
  # n = 5000
  w = rnorm(n)
  qnt = 0.75
  m = 50000
  sdvnr = replicate(m,quantile(sample(w,n,T),qnt))

  summary(sdvnr)
  length(sdvnr); length(unique(sdvnr))

  main = p("Pop.",100*qnt,
         "th Percentile Sampling Distr. via resampling from N(0,1) sample of size ",n,sep="")

  mysdci(sdvnr,pcol="lightblue2",dcol="navy",main=main)
  abline(v=qnorm(qnt),lwd=2,col="blue",lty=5)     # Pop. 75th Percentile of N(0,1)
```



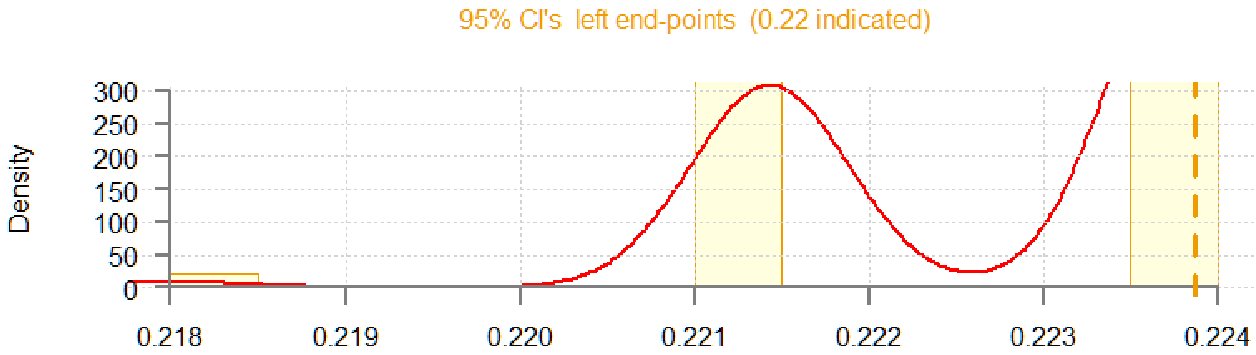

```
myhd(sdvnr,breaks=seq(min(sdvnr),max(sdvnr),l=101),cntFlg=F,
     main=main,col="lightblue2",border="navy")
```

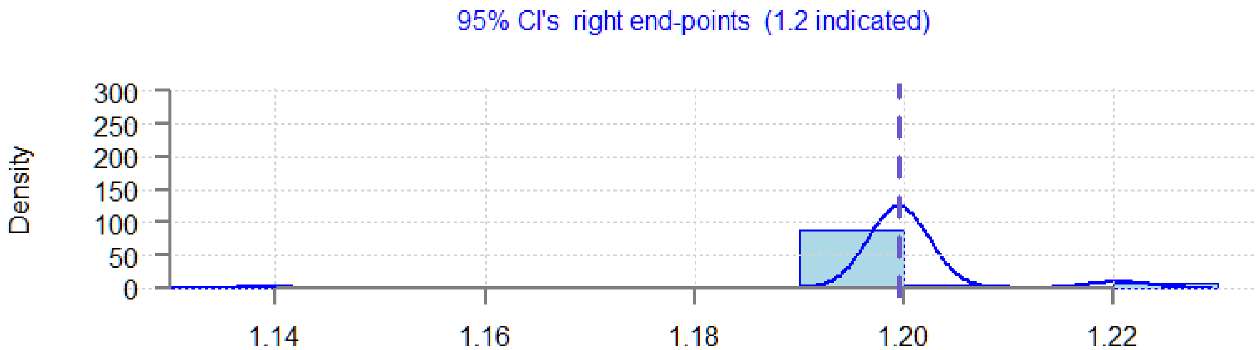



```
myhd2(sdvnr,cntFlg=F,main=main,col="lightblue2",border="navy")
```



```
# CI variation:
  m = 1000; M = 100
  CIsdvnr = rpl(M,myeCI(rpl(m,quantile(sample(w,n,T),qnt))))
  myCIsdv(CIsdvnr,myeCI(sdvnr))
```






```
  table(CIsdvnr[1,])   # Frequency table of left end-points
```

```
0.218016745973082 0.221430441571721 0.223802533914368 0.223863356794949
                1                35                 6                58
```

```
  table(CIsdvnr[2,])   # Frequency table of right end-points
```

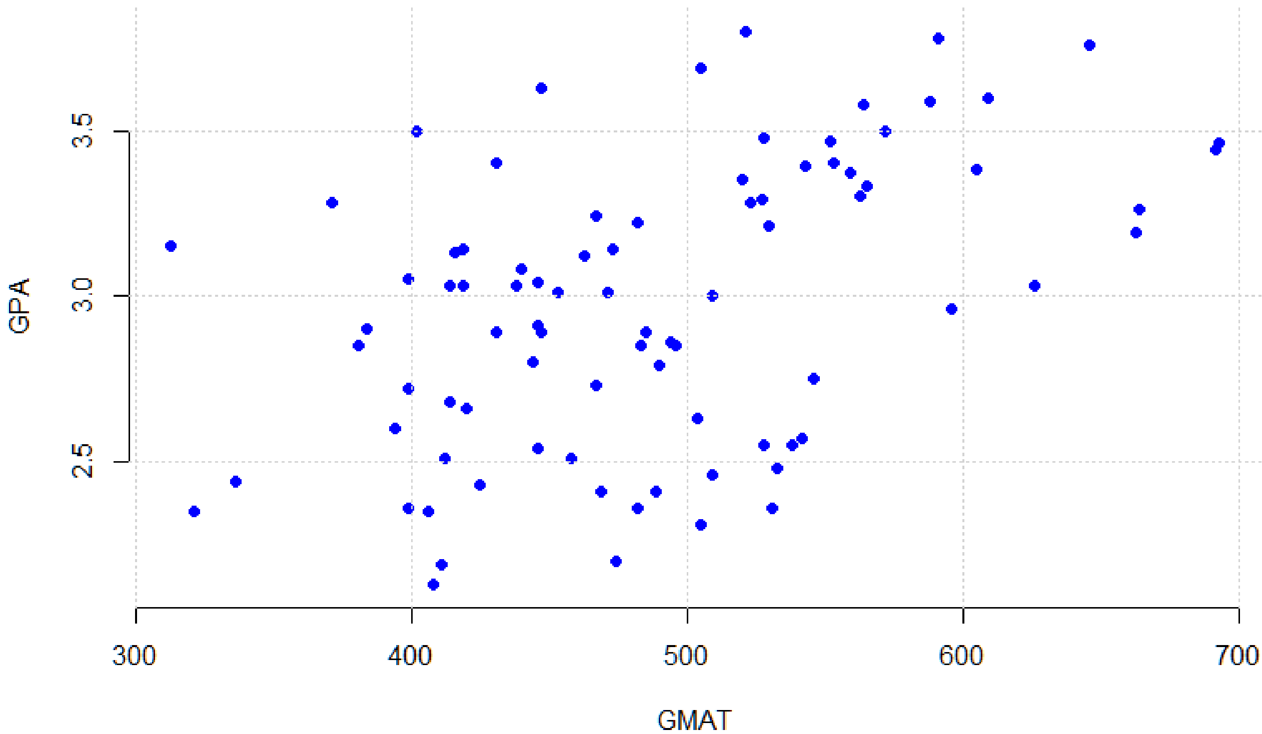
```
1.1388181781256 1.19948646428119  1.2000114595014  1.2204862730898
              1               88                4                7
```

```
# Same problems for larger samples (uncomment "n = 5000" line)

# Conclusion: Bumpier sampling distribution plots, yet confidence intervals are 'stable'

# ------------------------------------------------------------------------------------------
```

Example 4:

```
# Can we check if the GPA and GMAT scores from the admission data set are independent?
#
# The question really is: are the underlying random variables, GPA and GMAT scores of
# the student population, independent?
#
# Independence of random variables cannot be checked by calculation.
# Furthermore, we only have a sample of pairs (GPA, GMAT) of size n = 85.
#
# But we can check un-correlated-ness!
# Correlation: "a quantity measuring the extent of interdependence"
#
# Independent random variables are uncorrelated (correlation = 0).
# The converse is not true: There exist dependent random variables whose correlation iz zero.

# We test the following:
```

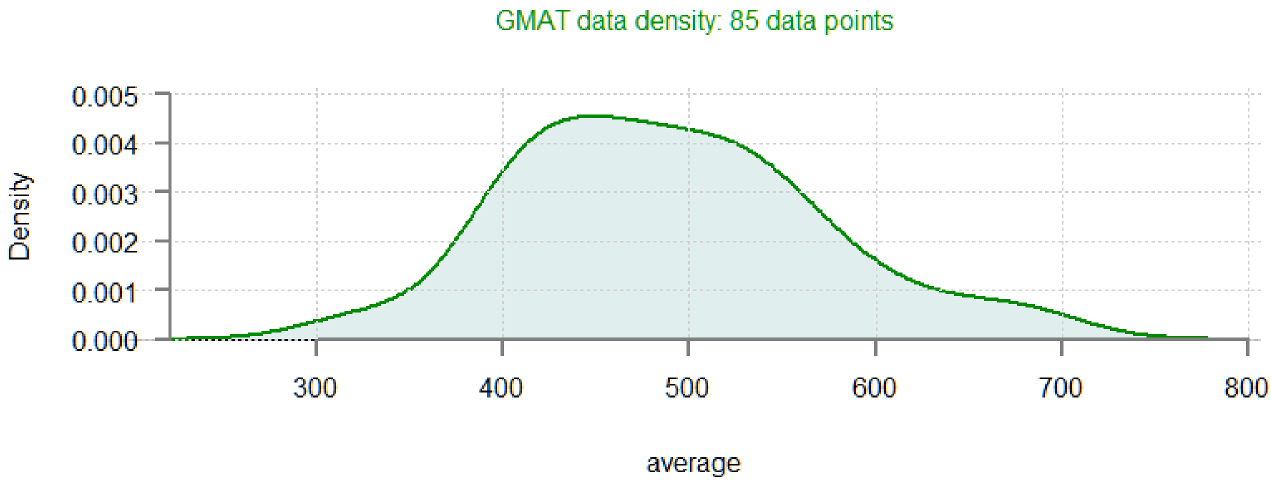
```
# Null-hypothesis: GPA and GMAT scores are uncorrelated!

# If the null-hypothesis is accepted, i.e., the GPA and GMAT scores are uncorrelated,
# they may or may not be independent.
# However, if the null hypothesis is rejected in favor of the alternative:
# GPA and GMAT scores are not uncorrelated, hence not independent!

  GPA = adm$GPA

# Scatter plot the GPA scores against the GMAT scores
  plot(GMAT,GPA,pch=19,col="blue",frame.plot=F); grid()
```

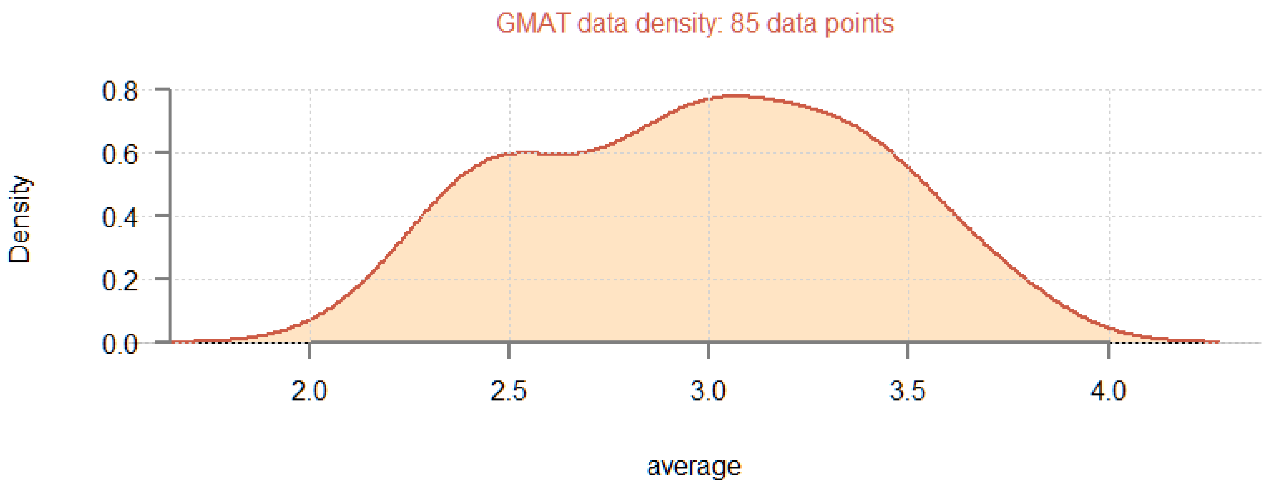



```
# What is the sample correlation?
  Scor = cor(GPA,GMAT); Scor
```

```
[1] 0.4606332
```

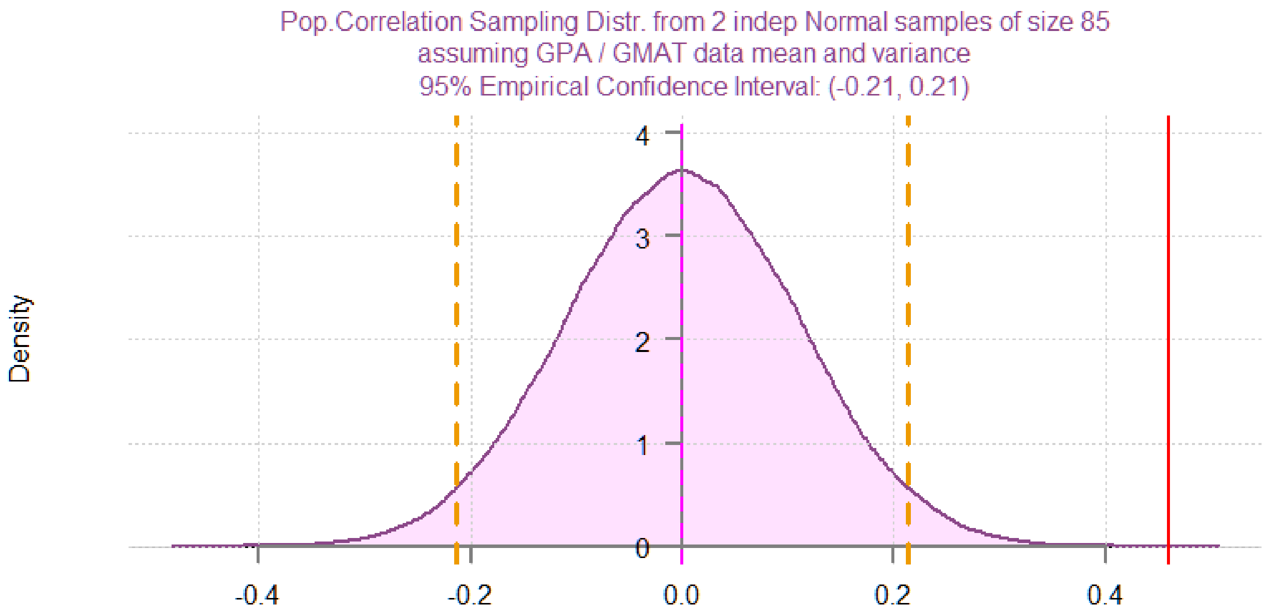
```
  sm1 = mean(GPA);  ssd1 = sd(GPA)
  sm2 = mean(GMAT); ssd2 = sd(GMAT)

# the empirical densities
  par(mfrow=c(2,1))                       # set 2x1 plots
  myed(GMAT,ylim=c(0,0.005),pcol="azure2",dcol="green4",
       xlab="average",main="GMAT data density")
  myed(GPA,pcol="bisque",dcol="coral3",
       xlab="average",main="GMAT data density")
  par(mfrow=c(1,1))                       # set 1x1 plot
```



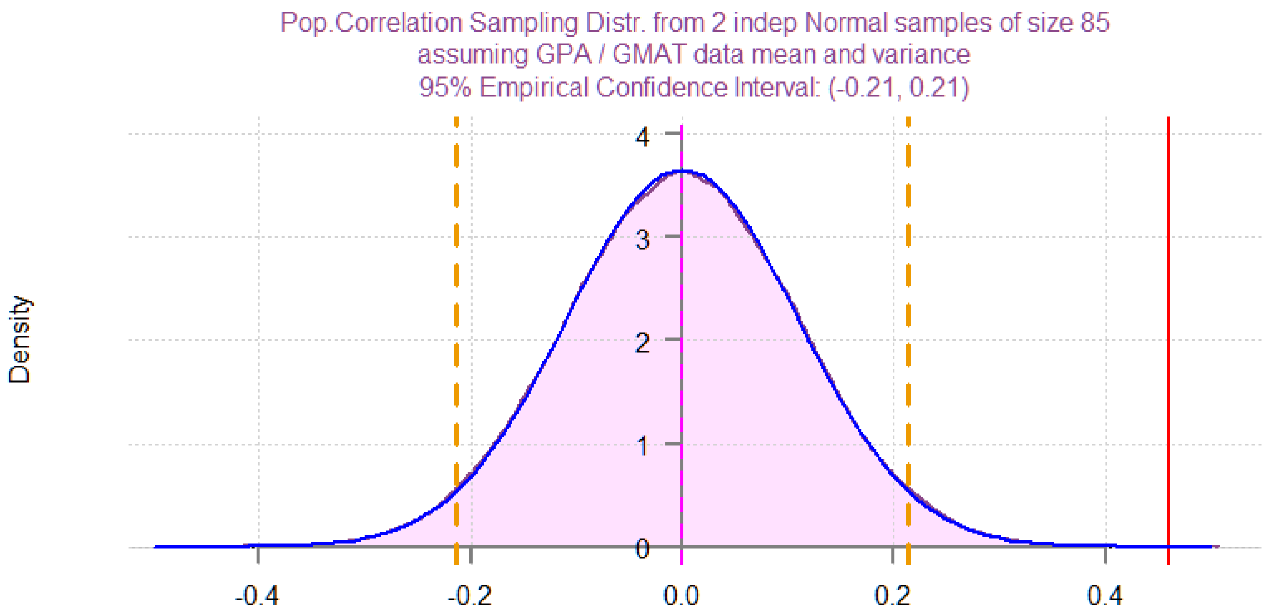


First approach: Assume normality of GPA and GMAT data

```
# Find the confidence interval for the correlation of two 'independent' vectors
# from normal distributions N(sm1,ssd1) and N(sm2,ssd2), respectively, where
# sm1, ssd1, sm2, and ssd2 are sample variance and sample standard deviation
# of GPA and GMAT vectors from the admission data set.

# Question: if GPA and GMAT vectors were independent (more precisely, uncorrelated),
#           how likely it is that their correlation is approx. 0.46?
# We can answer this if we know the distribution of the 'sample' correlation of
# two independent normal random vectors of same length as GPA and GMAT vectors

  n = length(GPA)
  cor(rnorm(n,sm1,ssd1),rnorm(n,sm2,ssd2))

# The sampling distribution is created by replicating this 100 thousand times
  m = 100000
  sdv4a = replicate(m,cor(rnorm(n,sm1,ssd1),rnorm(n,sm2,ssd2)))

  mysdci(sdv4a,ylim=c(0,4),pcol="thistle1",dcol="orchid4",
         main=paste("Pop.Correlation Sampling Distr. from 2 indep Normal samples of size",n),
         sub="assuming GPA / GMAT data mean and variance")

# Where does the correlation of our GPA and GMAT vectors (0.46) fall in?
  abline(v=Scor,lwd=2,col="red")
```



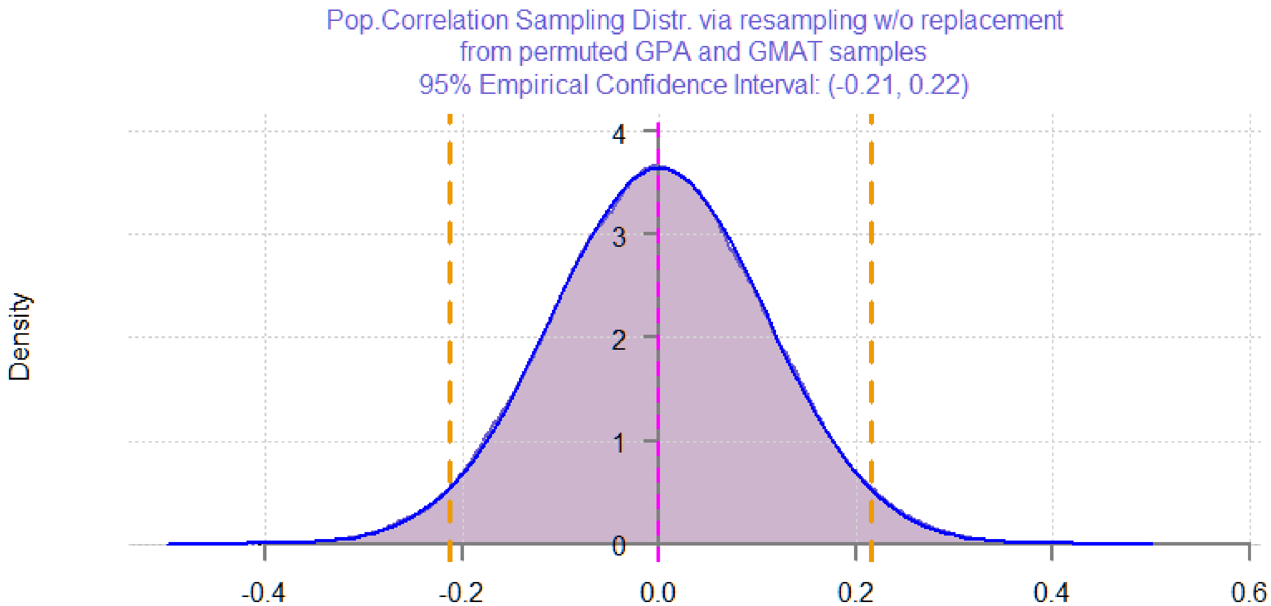

```
# Extremly far in the tail.  Check the maximum of the sampling distribution vector
  max(sdv4a)

# Do you accept or reject H0: GPA and GMAT scores are independent?

# Related question: how close is the sampling  distr. to the normal distribution:
  x = seq(-0.5,0.5,0.01)
  lines(x,dnorm(x,0,sd(sdv4a)),lwd=2,col="blue")
```



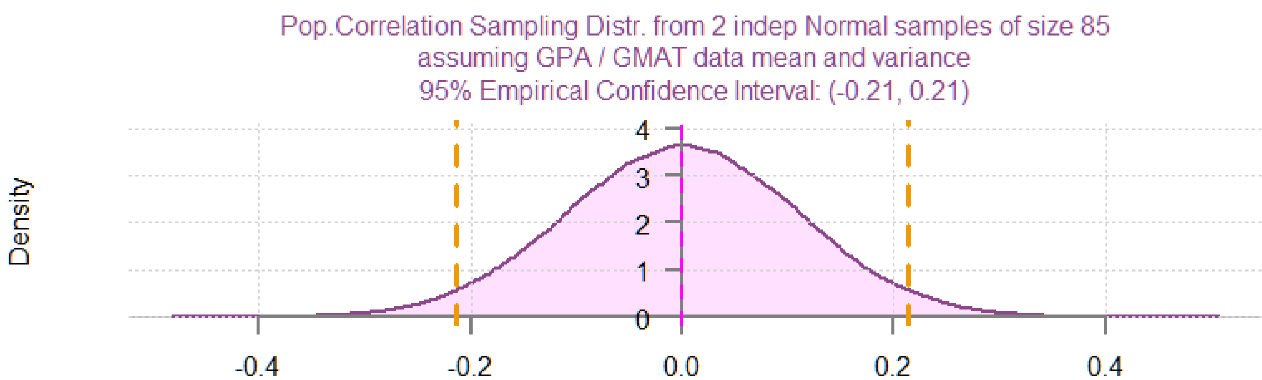
Second approach: Resampling

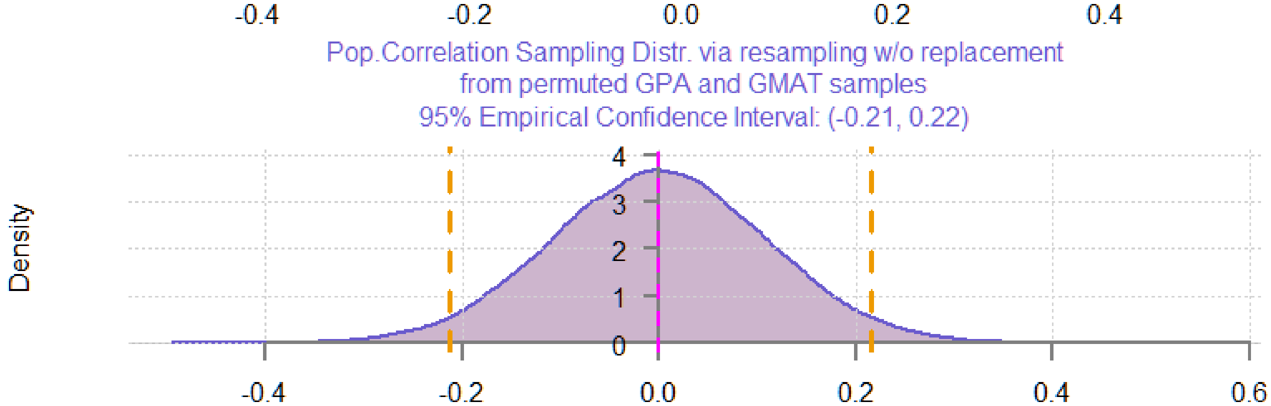
```
# Recall: resampling (a full length of vector) without replacement is same as permuting it
  v = seq(6)
  sample(v,6,replace=FALSE)

# Resampling without replacement (permutations of data vectors)
  m = 100000
  sdv4b = rpl(m,cor(sample(GPA,n,F),sample(GMAT,n,F)))

# It is not hard to see that it is enough to permute only one of the vectors
```

```
 sdv4b = rpl(m,cor(GPA,sample(GMAT,n,F)))
 mysdci(sdv4b,ylim=c(0,4),pcol="thistle3",dcol="slateblue",
        main=paste("Pop.Correlation Sampling Distr. via resampling w/o replacement"),
        sub="from permuted GPA and GMAT samples")
 x = seq(-0.5,0.5,0.01)
 lines(x,dnorm(x,0,sd(sdv4b)),lwd=2,col="blue")
```

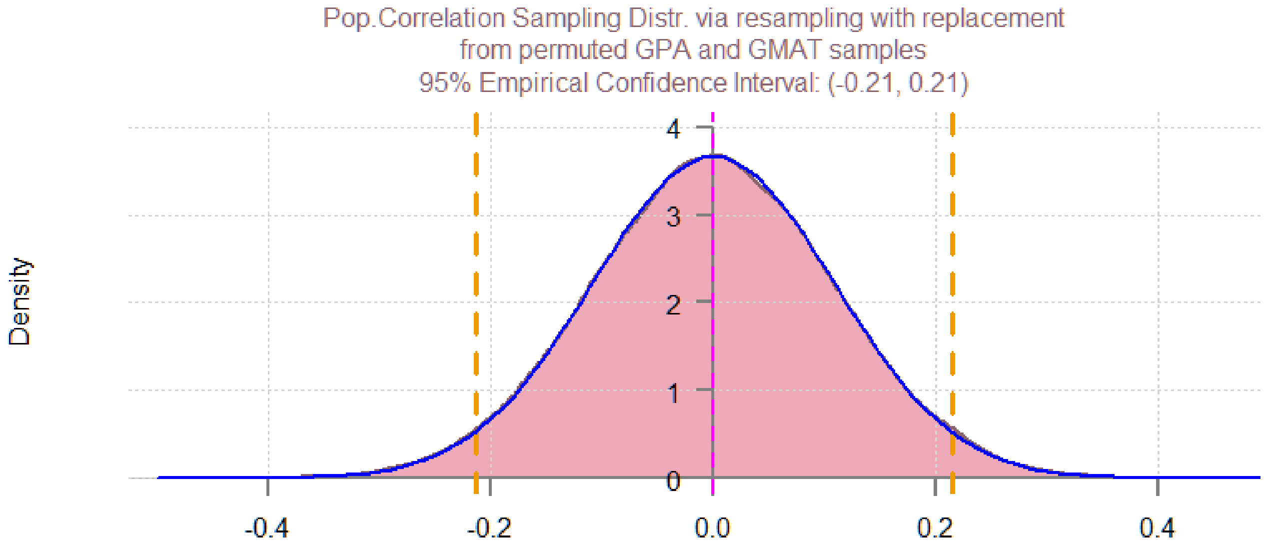



```
# Do you accept or reject H0: GPA and GMAT scores are independent?

# Plot both sample distributions from 4a and 4b
 par(mfrow=c(2,1))                        # set 2x1 plots
 mysdci(sdv4a,ylim=c(0,4),pcol="thistle1",dcol="orchid4",
        main=paste("Pop.Correlation Sampling Distr. from 2 indep Normal samples of size",n),
        sub="assuming GPA / GMAT data mean and variance")
 mysdci(sdv4b,ylim=c(0,4),pcol="thistle3",dcol="slateblue",
        main=paste("Pop.Correlation Sampling Distr. via resampling w/o replacement"),
        sub="from permuted GPA and GMAT samples")
 par(mfrow=c(1,1))                        # set 1x1 plot
```

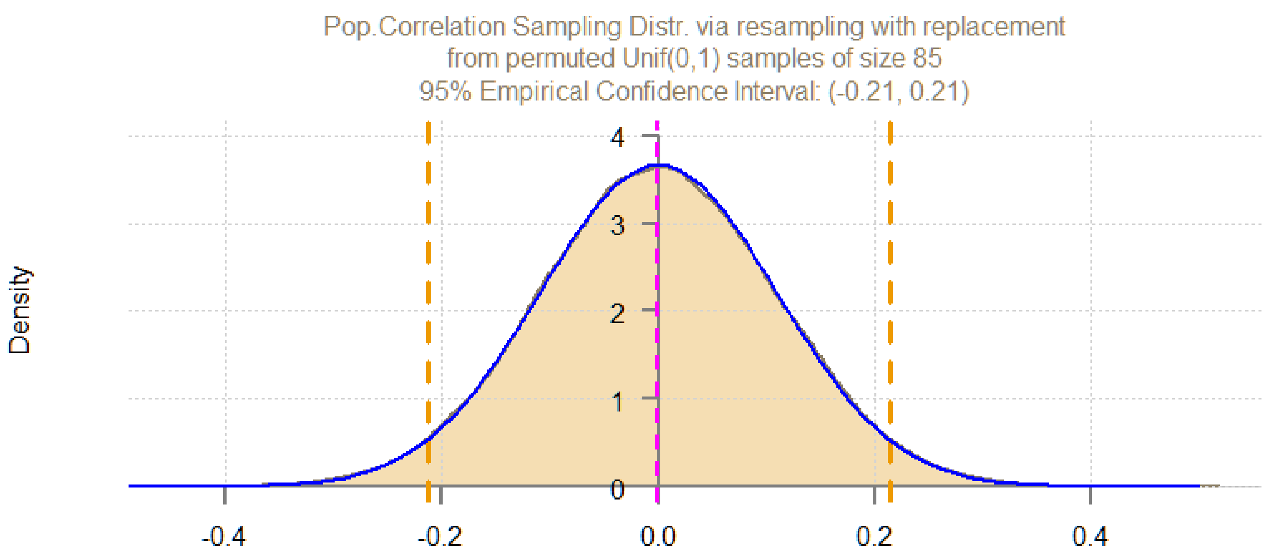




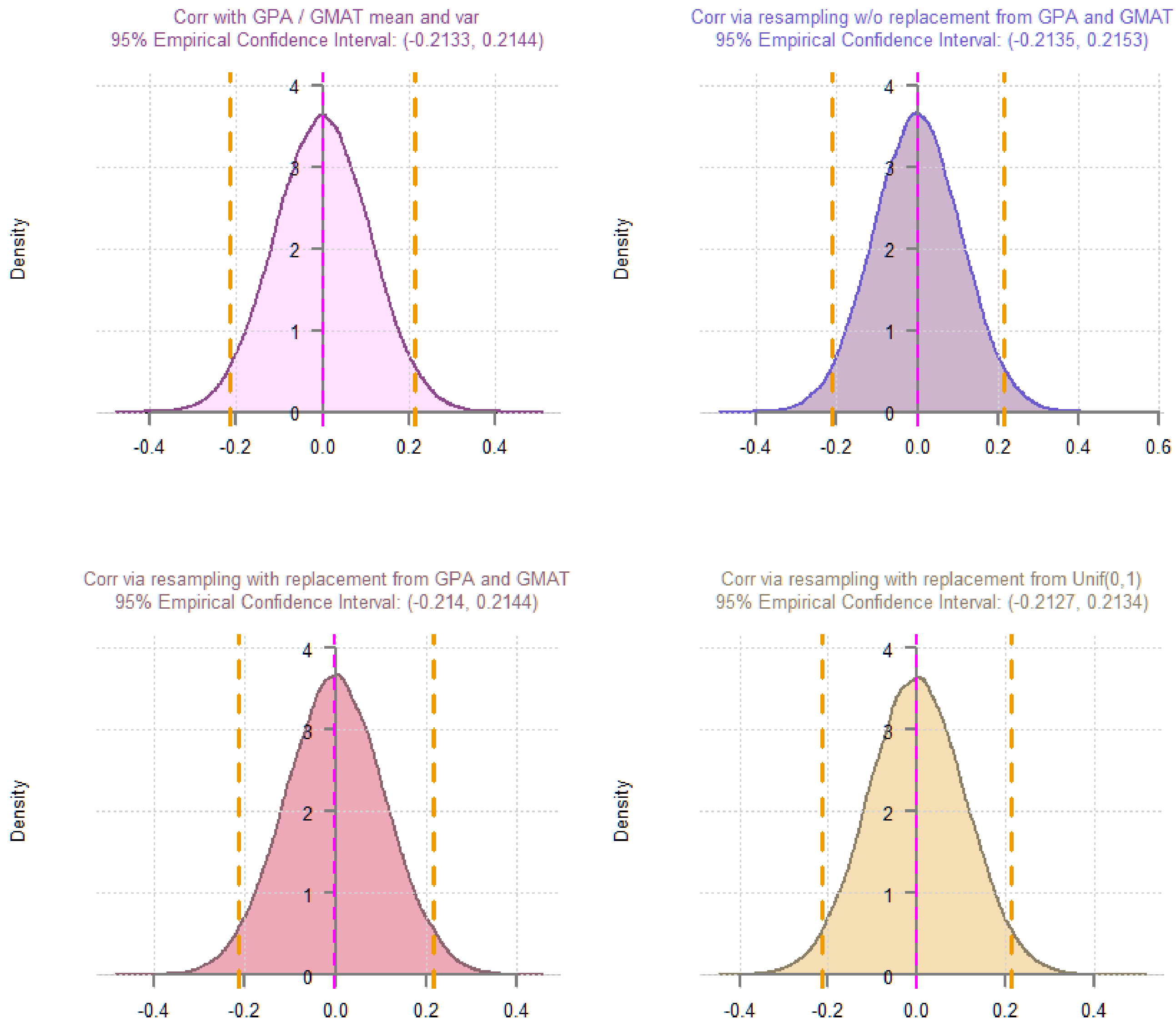
```
# Interesting observations:

# Try resampling WITH replacement - the sampling distr. look pretty much the same

  sdv4c = rpl(m,cor(sample(GPA,n,T),sample(GMAT,n,T)))
  mysdci(sdv4c,ylim=c(0,4),pcol="pink2",dcol="pink4",
         main=paste("Pop.Correlation Sampling Distr. via resampling with replacement"),
         sub="from permuted GPA and GMAT samples")
  x = seq(-0.5,0.5,0.01)
  lines(x,dnorm(x,0,sd(sdv4c)),lwd=2,col="blue")
```




```
# In fact, replace the GPA and GMAT vectors by any two 'independent' generated vectors
# of length 85. The sampling distribution will look identical

  sdv4d = rpl(m,cor(runif(n),runif(n)))
  mysdci(sdv4d,ylim=c(0,4),pcol="wheat",dcol="wheat4",
         main=paste("Pop.Correlation Sampling Distr. via resampling with replacement"),
         sub=paste("from permuted Unif(0,1) samples of size",n))
  x = seq(-0.5,0.5,0.01)
  lines(x,dnorm(x,0,sd(sdv4d)),lwd=2,col="blue")
```



```
# All-star lineup
  par(mfrow=c(2,2))                              # set 2x2 plots
  mysdci(sdv4a,ylim=c(0,4),pcol="thistle1",dcol="orchid4",rnd=4,
         main="Corr with GPA / GMAT mean and var")
  mysdci(sdv4b,ylim=c(0,4),pcol="thistle3",dcol="slateblue",rnd=4,
         main="Corr via resampling w/o replacement from GPA and GMAT")
  mysdci(sdv4c,ylim=c(0,4),pcol="pink2",dcol="pink4",rnd=4,
         main="Corr via resampling with replacement from GPA and GMAT")
  mysdci(sdv4d,ylim=c(0,4),pcol="wheat",dcol="wheat4",rnd=4,
         main="Corr via resampling with replacement from Unif(0,1)")
  par(mfrow=c(1,1))                              # set 1x1 plot
```




```
# Hence the distributions of the vectors play no role here - only the fact that
# the vectors were independently generated (via R random number generator: the
# vectors are just uncorrelated) matters in shaping the sampling distribution.
```