Assignment 4

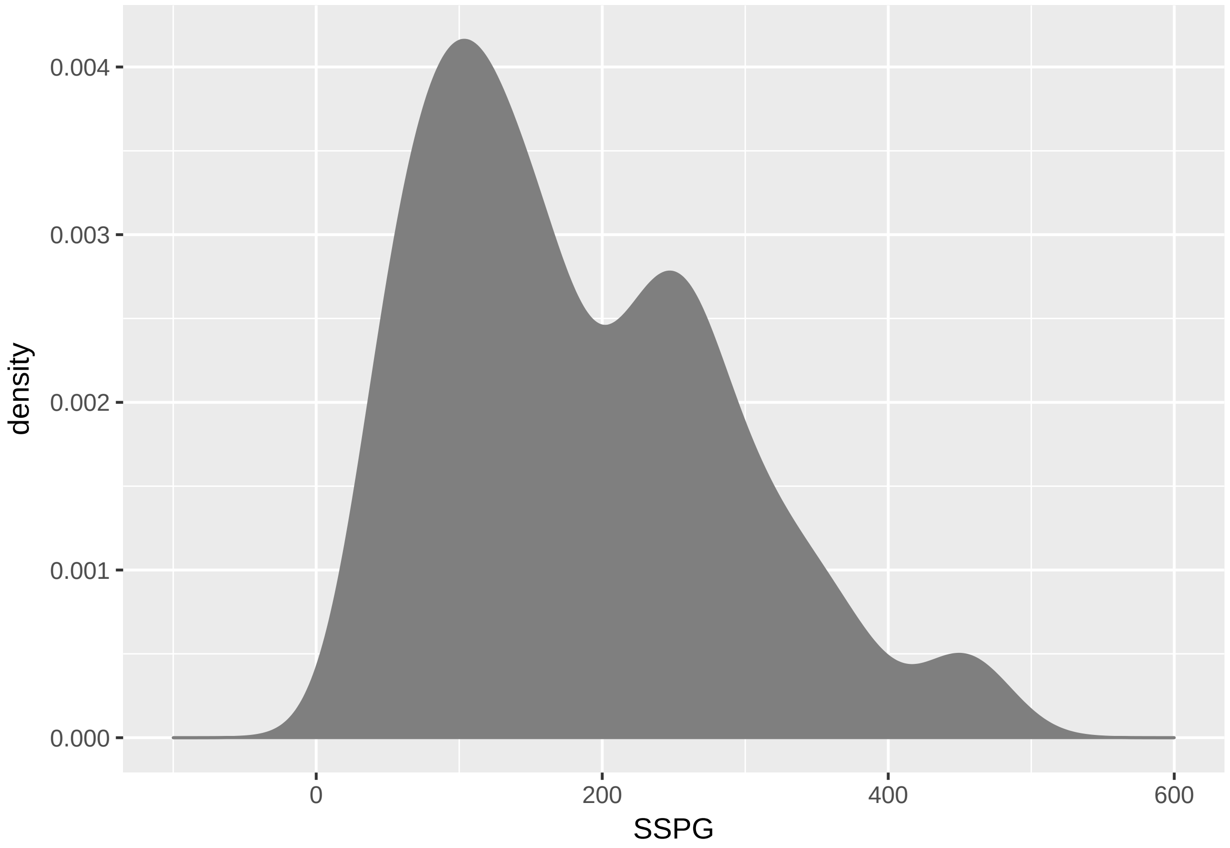
Azoacha Forcheh, 20558994

1. Download the `diabetes` data from the course website. In that file, there is a dataset on various measurements of 145 patients. Once you load this file into your R session (or equivalently, execute its contents there) there will be a data set called `diabetes`.

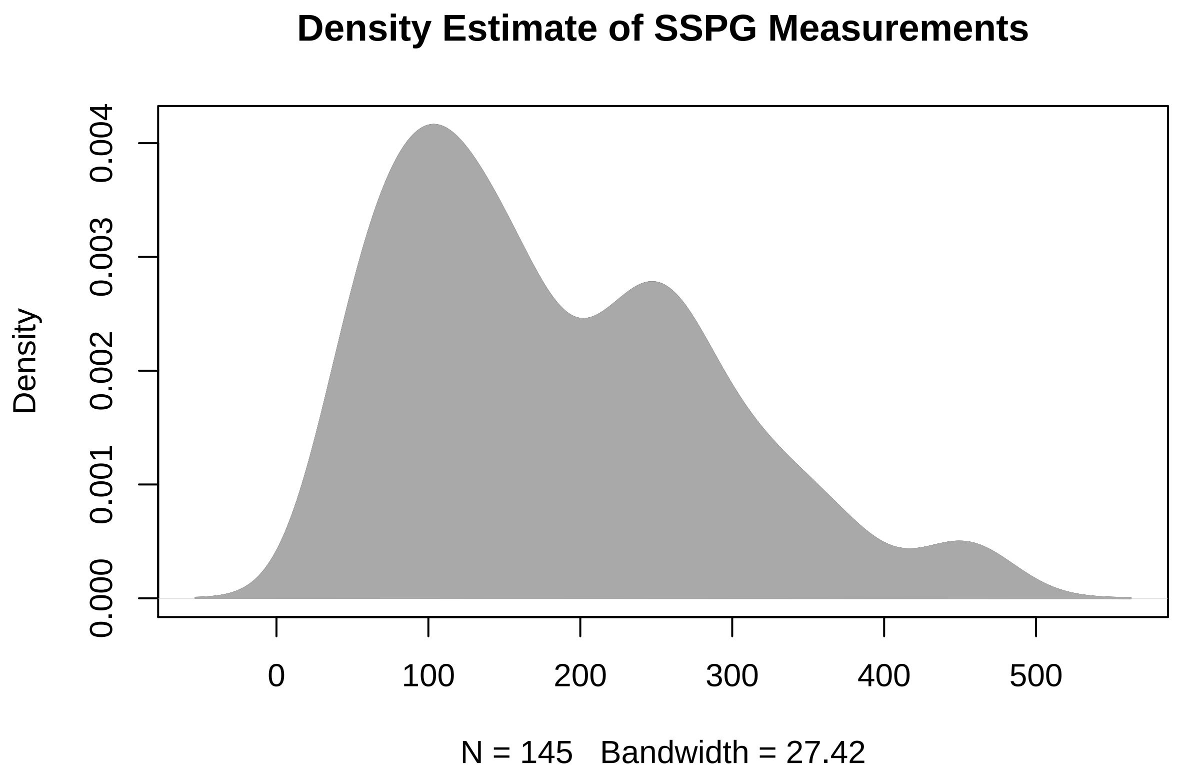
   The variate `SSPG` stands for steady state plasma glucose which measures the patient's insulin resistance, a pathological condition where the body's cells fail to respond to the hormone insulin.

   (a) (3 marks) Produce a plot of a density estimate of `SSPG` and comment on what you see.

```r
# Should I stretch out the limits so that this doesn't look cut off?
library(ggplot2)
sspg_plot = ggplot(data = diabetes, mapping = aes(x=SSPG))
sspg_plot +
  geom_density(colour="grey50", fill="grey50", bw="SJ") +
  scale_x_continuous(limits=c(-100, 600))
```



```r
sspg_dens = density(diabetes$SSPG, bw="SJ")
plot(sspg_dens, main="Density Estimate of SSPG Measurements")
polygon(sspg_dens, col="dark grey", border="dark grey", xlab="SSPG")
```

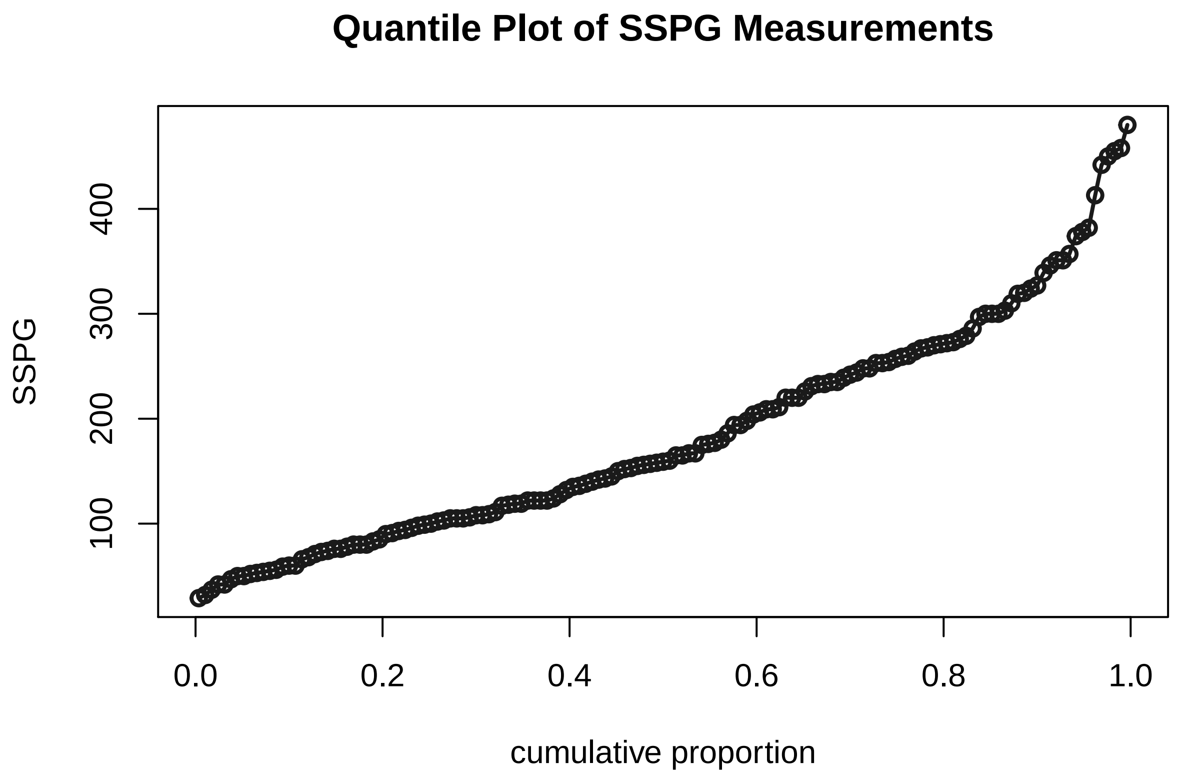


The density is right-skewed and trimodal, with modes at about 100, 250 and 450. The curve rises quickly to the first mode, then gradually decreases. In addition, the majority of patients have an SSG measurement below 250. Lastly, the SSPG seems lacks normality - i.e. it does not seem to have been sampled from the normal - as its density does not resemble the symmetric bell curve of a normal density.

(b) Construct a quantile plot of `SSPG` and comment on the shape of its distribution.

```r
sspg_vals = sort(diabetes$SSPG)
n = length(sspg_vals)
p = ppoints(n) # the proportions

plot(x = p, y = sspg_vals, type="o", lwd=2, col="grey10",
    xlab="cumulative proportion", xlim=c(0,1),
    ylab="SSPG", main="Quantile Plot of SSPG Measurements")
```

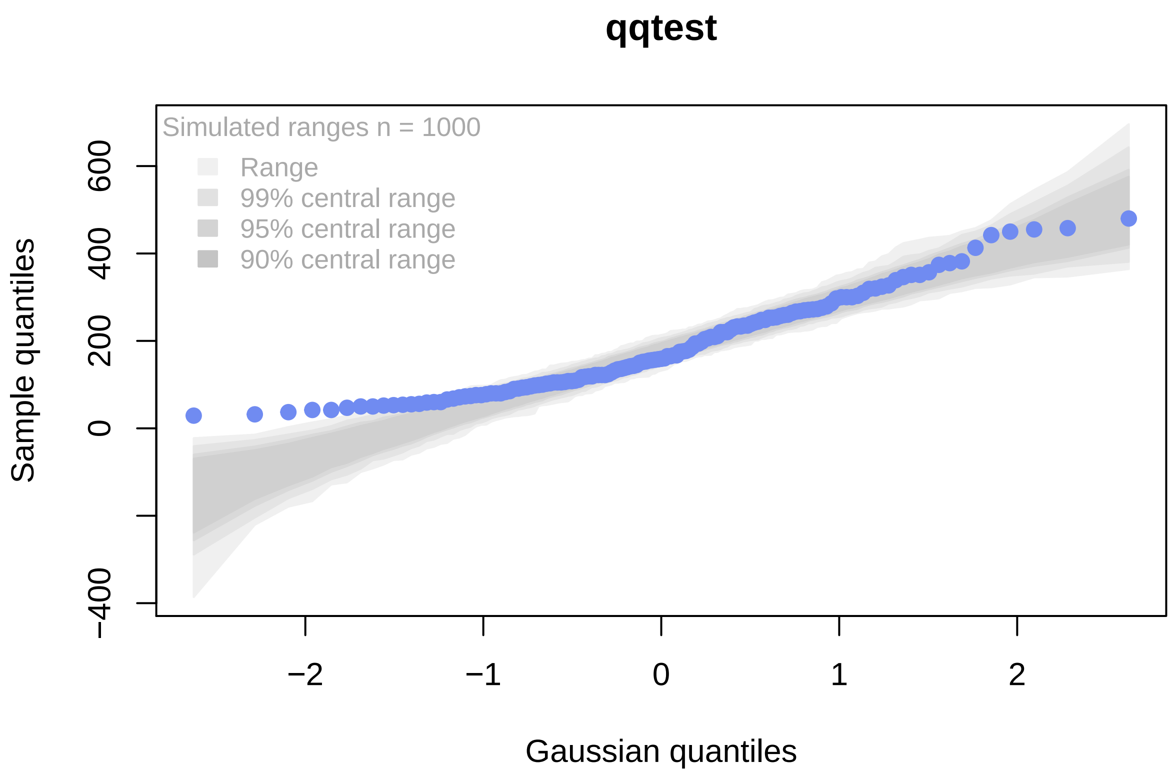


The shape of the quantile plot is nearly linear up until about the 60th percentile, at which point it begins to curve upwards. The data seems to be evenly concentraed throughout the plot, with sparsity at the right tail. This indicates a lot of variation between the extremes as the concentration of points at the left tail of the plot is relatively higher. Overall, the plot has a convex, trough-down shape.

(c) (3 marks) Use `qqtest` to construct a qqplot that compares `SSPG` to a standard normal distribution. Include envelopes in the plot. Comment on the distribution of `SSPG` and whether it might reasonably be regarded as a sample from some normal distribution. Explain your reasoning.

Important: Before every `qqtest` execute `set.seed(3124159)` so that we are all seeing the same plots.

```r
library(qqtest)
# Setting the seed
# See page 57 of Part 3 of Comparing Distributions for commentary info
set.seed(3124159)
qqtest(data=diabetes$SSPG)
```



(d) The last variate, `ClinClass`, represents the classification of each patient according to the 1979 medical criteria into one of three groups: 1 = "Overt Diabetic", 2 = "Chemical Diabetic", and 3 = "Normal".

i. (4 marks) Construct a back to back density line-up plot to assess whether the normal and diabetic (chemical and overt combined) `SSPG` values come from the same distribution. Use `set.seed(3124159)` and show your code. What conclusions do you draw?

```
normal_SSPG = diabetes[diabetes$ClinClass == 3,]$SSPG
diabetic_SSPG = diabetes[diabetes$ClinClass != 3,]$SSPG

back2back <- function(data, suspectNo) {
  ylim = extendrange(c(data$x, data$y)) #c(0,600)
  Xdensity = density(data$x, bw="SJ")
  Ydensity = density(data$y, bw="SJ")
  Ydensity$y = -Ydensity$y
  xlim = extendrange(c(Xdensity$y, Ydensity$y))

  xyswitch <- function(xy_plot) {
    yx_plot = xy_plot
    yx_plot$x = xy_plot$y
    yx_plot$y = xy_plot$x
    yx_plot
  }

  plot(xyswitch(Xdensity), col="firebrick",
       xlab="", ylab="", xaxt="n", yaxt="n",
       main=paste("i = ", suspectNo),# display suspect number
       cex.main = 2, # increase suspect number size
       xlim=xlim, ylim=ylim)
  polygon(xyswitch(Xdensity), col=adjustcolor("firebrick", 0.4))
  lines(xyswitch(Ydensity), col="steelblue")
  polygon(xyswitch(Ydensity), col=adjustcolor("steelblue",0.4))
}
```

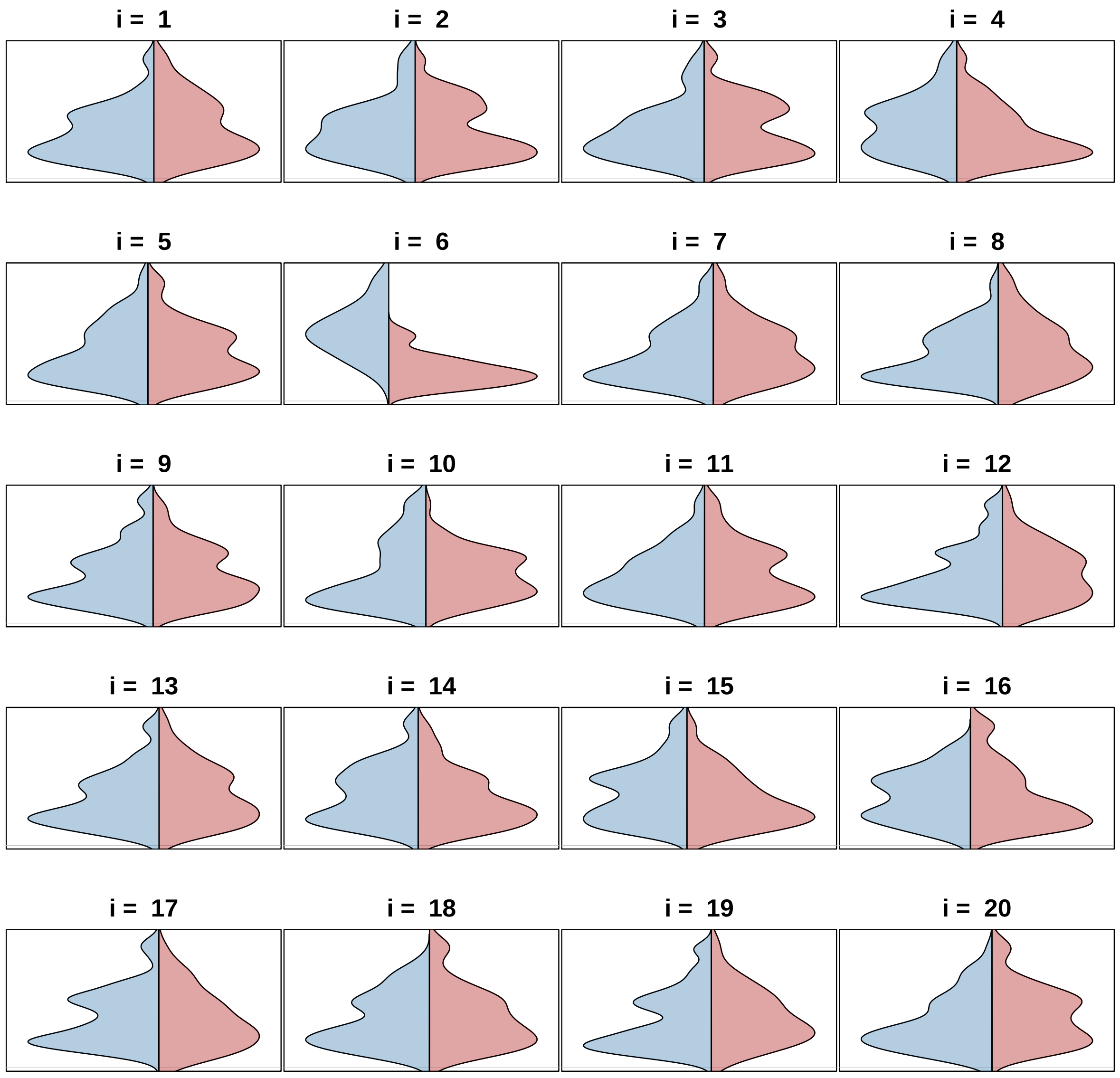
```r
mixRandomly <- function(data) {
  # Note that data need not be a data frame
  # It is expected to be a list with an x and a y component
  # (possibly of different lengths)
  x <- data$x
  y <- data$y
  n_x <-length(x)
  n_y <-length(y)
  mix <-c(x,y)
  select4x <-sample(1:(n_x+n_y),n_x,replace = FALSE)
  new_x <- mix[select4x] # The mixing occurs
  new_y <- mix[-select4x]
  list(x=new_x, y=new_y)
}

lineup <- function(data, showSuspect=NULL, generateSuspect=NULL,
                   trueLoc=NULL, layout =c(5,4)) {
  # Get the number of suspects in total
  nSuspects <- layout[1] * layout[2]
  if (is.null(trueLoc)) {trueLoc <-sample(1:nSuspects, 1)}
  if (is.null(showSuspect)) {stop("need a plot function for the suspect")}
  if (is.null(generateSuspect)) {stop("need a function to generate suspect")}
  # Need to decide which subject to present
  presentSuspect <- function(suspectNo) {
    if(suspectNo != trueLoc) {data <-generateSuspect(data)}
    showSuspect(data, suspectNo)
  }
  # This does the plotting
  savePar <-par(mfrow=layout,mar=c(2.5, 0.1, 3, 0.1), oma=rep(0,4))
  sapply(1:nSuspects, FUN = presentSuspect)
  par(savePar)

  # Obfuscate location to keep us honest
  possibleBaseVals <- 3:min(2*nSuspects, 50) # remove easy base values
  possibleBaseVals <- possibleBaseVals[possibleBaseVals != 10 & possibleBaseVals != 5]
  base <-sample(possibleBaseVals, 1)
  offset <-sample(5:min(5*nSuspects, 125),1)

  # return obfuscated location
  list(trueLoc =paste0("log(",base^(trueLoc + offset),
                       ", base=",base,") - ", offset))
}

data = list(x=normal_SSPG, y=diabetic_SSPG)
set.seed(3124159)
lineup(data,
       generateSuspect = mixRandomly,
       showSuspect = back2back)
```

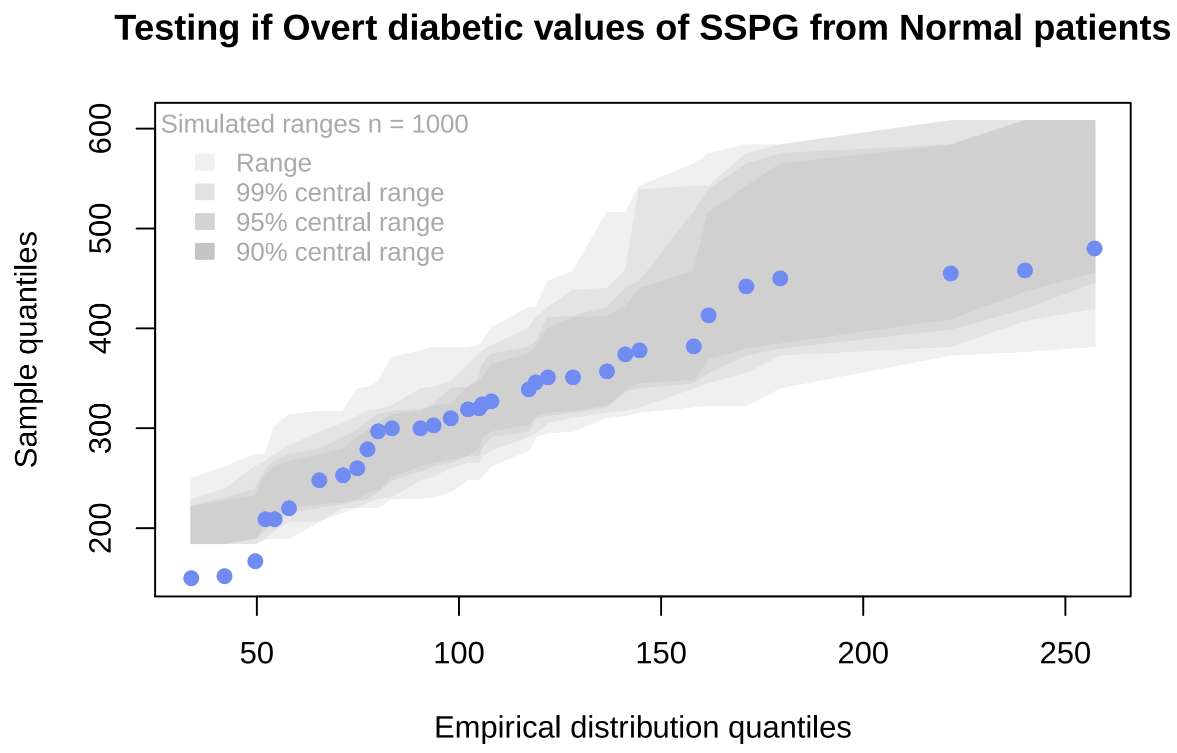


```
## $trueLoc
## [1] "log(7.51141330201283e+30, base=22) - 17"
```

From the lineup test, it is visually clear that the true plot is at $i = 6$ - which is indeed equal to the generated value of `trueLoc` - as this plot stands out more than any of the others. Hence, using this lineup test, there is enough evidence to reject the hypothesis that the two sets come from the same-shaped distribution.

ii. (4 marks) Use `qqtest` to construct a lineup plot making the same assessment, but this time assess whether the overt diabetic values of `SSPG` might have been generated from the same distribution as the normal patients. Use `set.seed(3124159)` and show your code. What conclusions do you draw?

```r
# See page 57 of Part 3 of Comparing Distributions for commentary info
set.seed(3124159)
overt_SSPG = diabetes[diabetes$ClinClass == 1,]$SSPG
qqtest(data = overt_SSPG, dataTest=normal_SSPG,
       main="Testing if Overt diabetic values of SSPG from Normal patients")
```



Most of the points in the plot lie within the envelepe, with a few at the left tail sitting outside the envelope. Hence, there is not enough evidence to reject the hypothesis that the overt diabetic values of `SSPG` might have been generated from the same distribution as the normal patients.

iii. **Grad students, bonus undergraduates** (8 marks) Consider the following code:

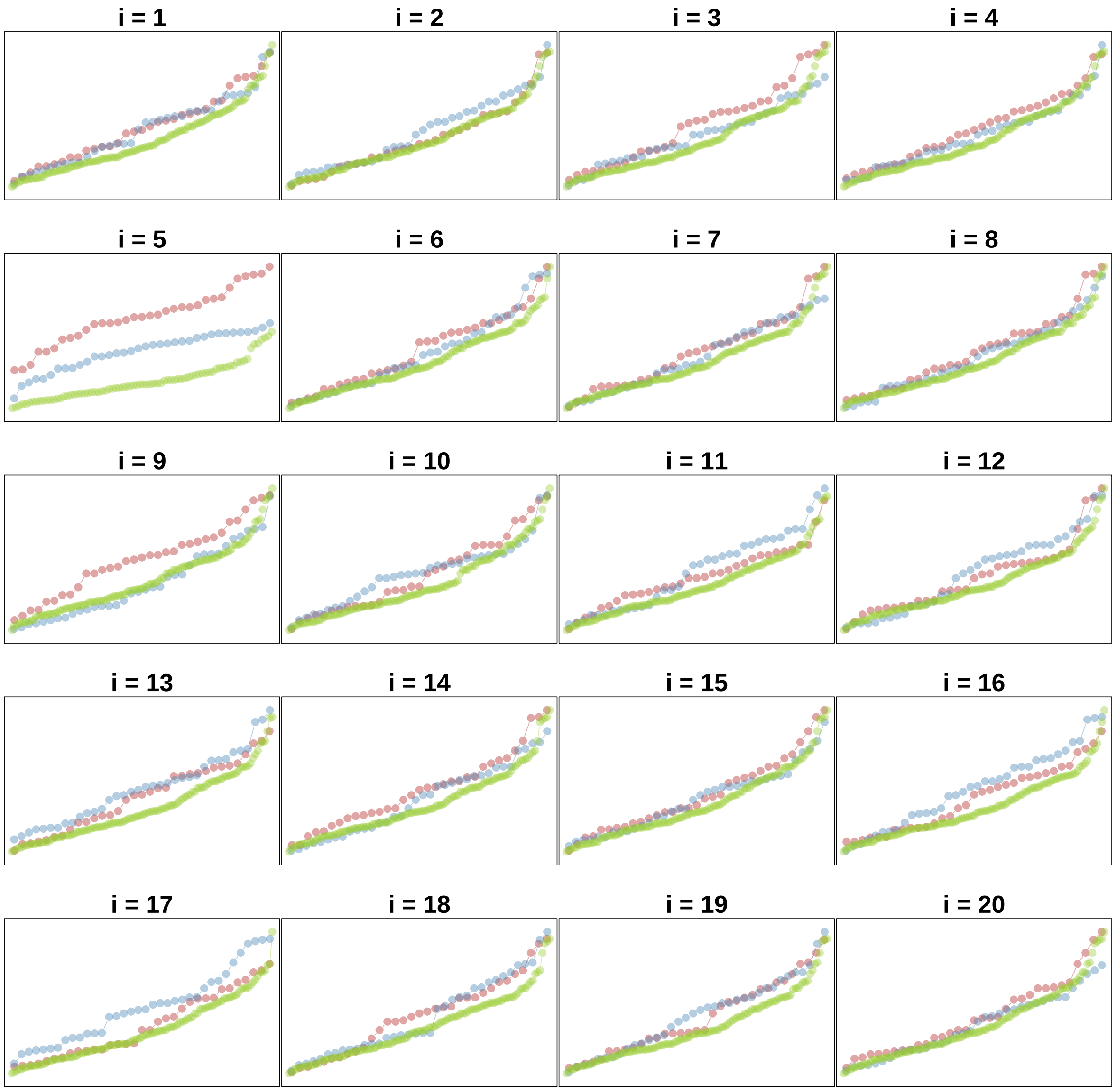
```
data <- list(x=x, y=y, z=z)
lineup(data,
generateSuspect = mixRandomly,
showSuspect = myQuantilePlot,
layout=c(5,4))
```

The function `mixRandomly` will need to be rewritten to handle `data` being a list of three samples. Write the function `myQuantilePlot` so that it overlays the sample quantile functions of each of `x`, `y`, and `z` in the same display using different colours. Hand in your code for these two functions and illustrate the outcome (using `set.seed(314159)`) on `SSPG` for the three different clinical classes. Comment on your findings.

```
mixRandomly <- function(data) {
  # Note that data need not be a data frame
  # It is expected to be a list with an x, a y and a z component
  # (possibly of different lengths)
  x = data$x
  y = data$y
  z = data$z
  n_x = length(x)
  n_y = length(y)
  n_z = length(y)
  mix = c(x,y,z)
  full_range = 1:(n_x+n_y+n_z)
  select4x = sample(full_range,n_x,replace = FALSE)
  # remove indices in select4x from sampling pool and sample indices for y
  select4y = sample(full_range[-select4x],n_y,replace = FALSE)
  new_x = mix[select4x] # The mixing occurs
  new_y = mix[select4y]
  new_z = mix[-select4y]
  list(x=new_x, y=new_y, z=new_z)
}

# should there be an option to specify colors or can we choose any?
myQuantilePlot <- function(data, suspectNo) {
  ylim = extendrange(c(data$x, data$y, data$z))
  n_x = length(data$x)
  n_y = length(data$y)
  n_z = length(data$z)
  p_x = ppoints(n_x)
  p_y = ppoints(n_y)
  p_z = ppoints(n_z)
  plot(p_x, sort(data$x), type="b", col=adjustcolor("firebrick", 0.4),
       pch=19, cex=2, ylim = ylim,
       main=paste("i =", suspectNo), # display suspect number
       cex.main = 3,# increase suspect number size
       ylab="", xlab="", xaxt="n", yaxt="n")
  points(p_y,sort(data$y), type="b",
         col=adjustcolor("steelblue", 0.4),  pch=19, cex=2)
  points(p_z,sort(data$z), type="b",
         col=adjustcolor("yellowgreen", 0.4),  pch=19, cex=2)
}

chemical_SSPG = diabetes[diabetes$ClinClass == 2,]$SSPG
sspg_data = list(x=overt_SSPG, y=chemical_SSPG, z=normal_SSPG)
set.seed(314159)
lineup(sspg_data,
        generateSuspect = mixRandomly,
        showSuspect = myQuantilePlot,
        layout=c(5,4))
```



```
## $trueLoc
## [1] "log(6.6790315523026e+66, base=39) - 37"
```

From the lineup test, it is visually clear that the true plot is at $i = 5$ - which is indeed equal to the generated value of `trueLoc` - as in this plot, none of the graphs never overlap unlike every other plot. Hence, using this lineup test, there is enough evidence to reject the hypothesis that the three sets come from the same-shaped distribution.