W2_HW

110078509

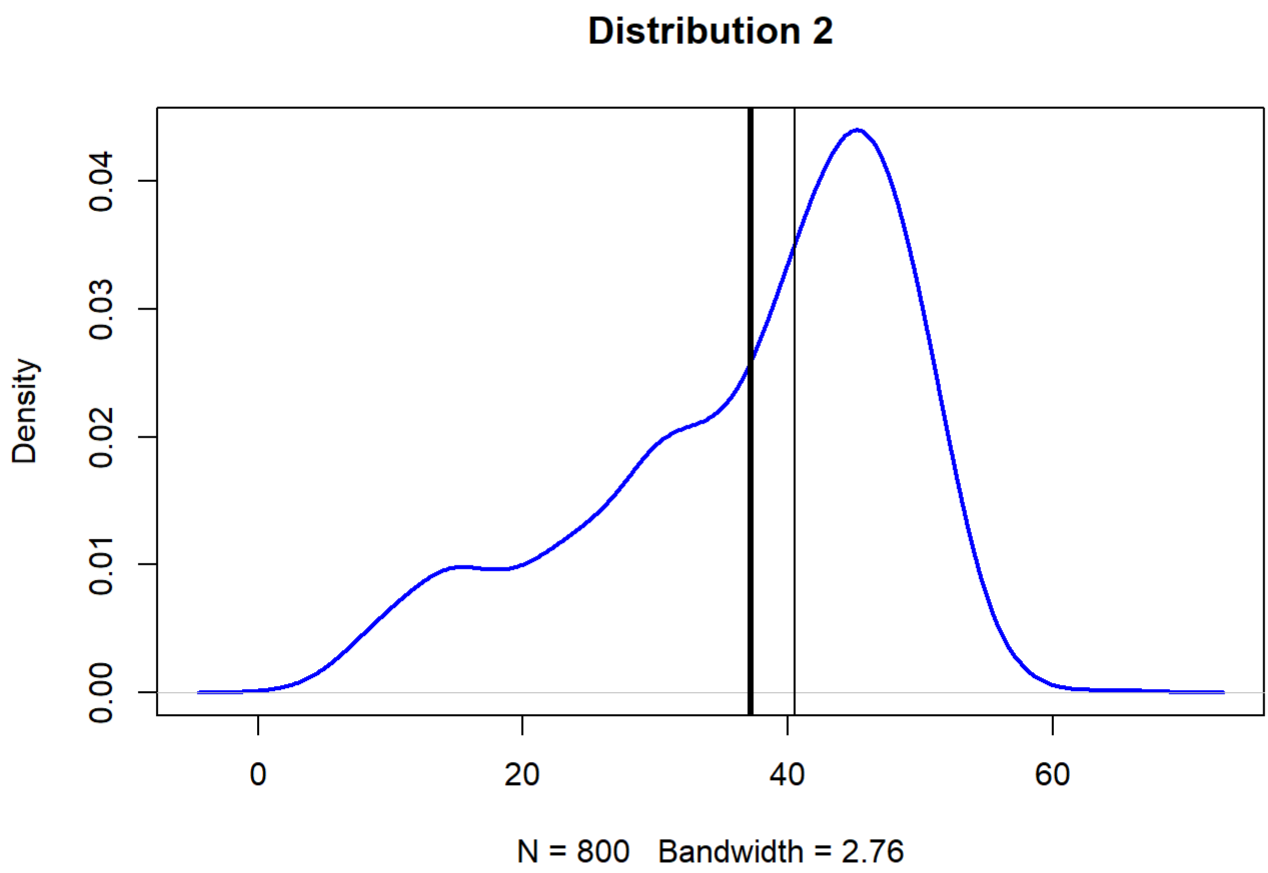
2022/2/25

```
rm(list=ls()) #remove the random variable to fresh the working environment
ls() #suppose to be nothing 'character(0)'
```

```
## character(0)
```

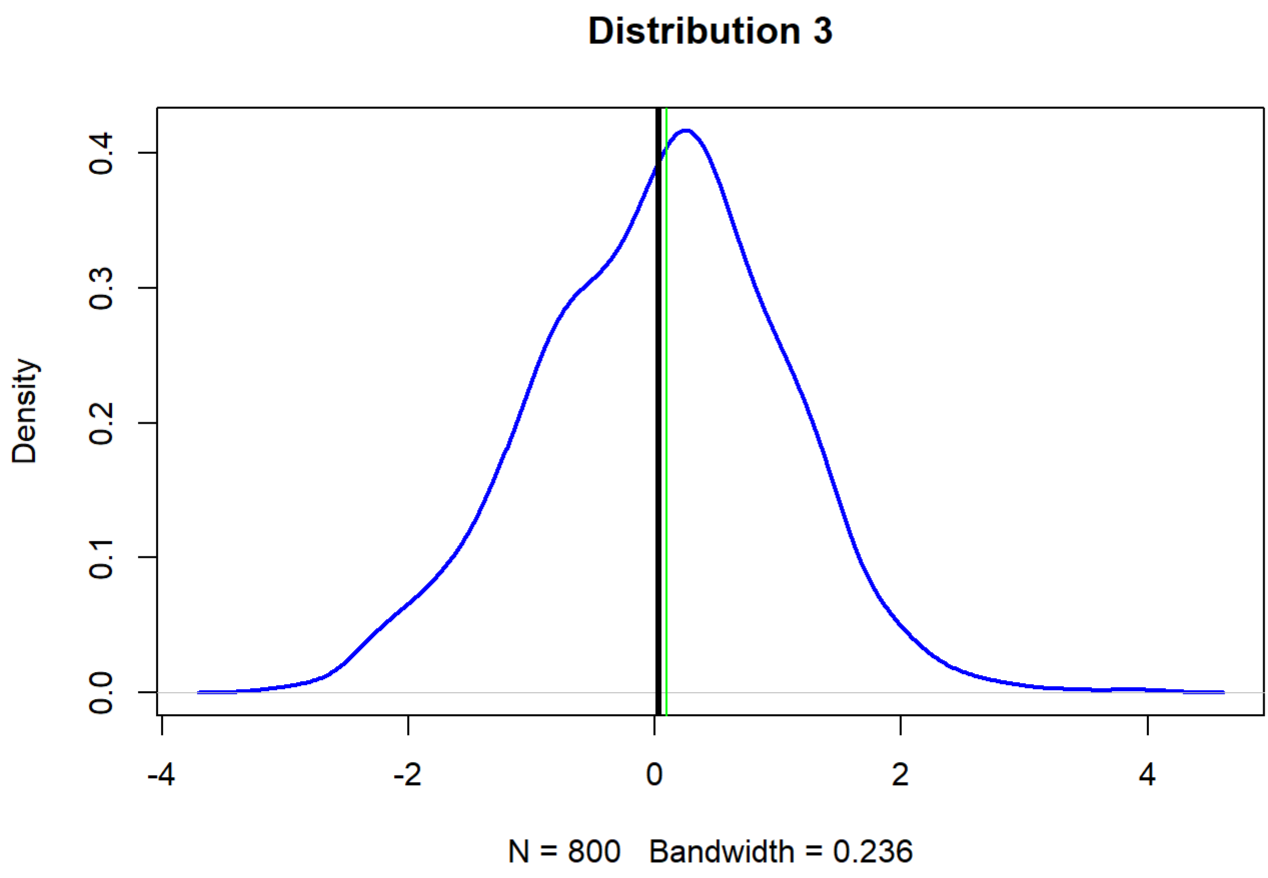
- Section1 *(a) Three normally distributed data sets* ****

```
# Change the mean and standard deviation of d1, d2, and d3 to achieve this new distribution.
set.seed(110078509)
d1 <- rnorm(n=500, mean=45, sd=5) #Change mean from 15 to  45

set.seed(110078509)
d2 <- rnorm(n=200, mean=30, sd=4.99999) #Change sd from 5 to  4.99999

set.seed(110078509)
d3 <- rnorm(n=100,mean=15, sd=5) #Change mean from 15 to  45

d123 <- c(d1, d2, d3) # Combining them into a composite dataset

plot(density(d123), col="blue", lwd=2,
     main = "Distribution 2")

#mean (thick line) and median (thin line)
#mean
abline(v=mean(d123), lwd=3)

#median
abline(v=median(d123),  lwd=1) #lty="dashed"
```



(b) Create a “Distribution 3”: to create a single large dataset (n=800).

```
b123 <- rnorm(800, mean= 0, sd = 1);
plot(density(b123), col="blue", lwd=2,
     main = "Distribution 3")
#mean (thick one)
abline(v=mean(b123), lwd=3)
#median with green color (thin one)
abline(v=median(b123),  lwd=1, col = "green")
```



Explain: the line of median and the line of mean are overlapped.

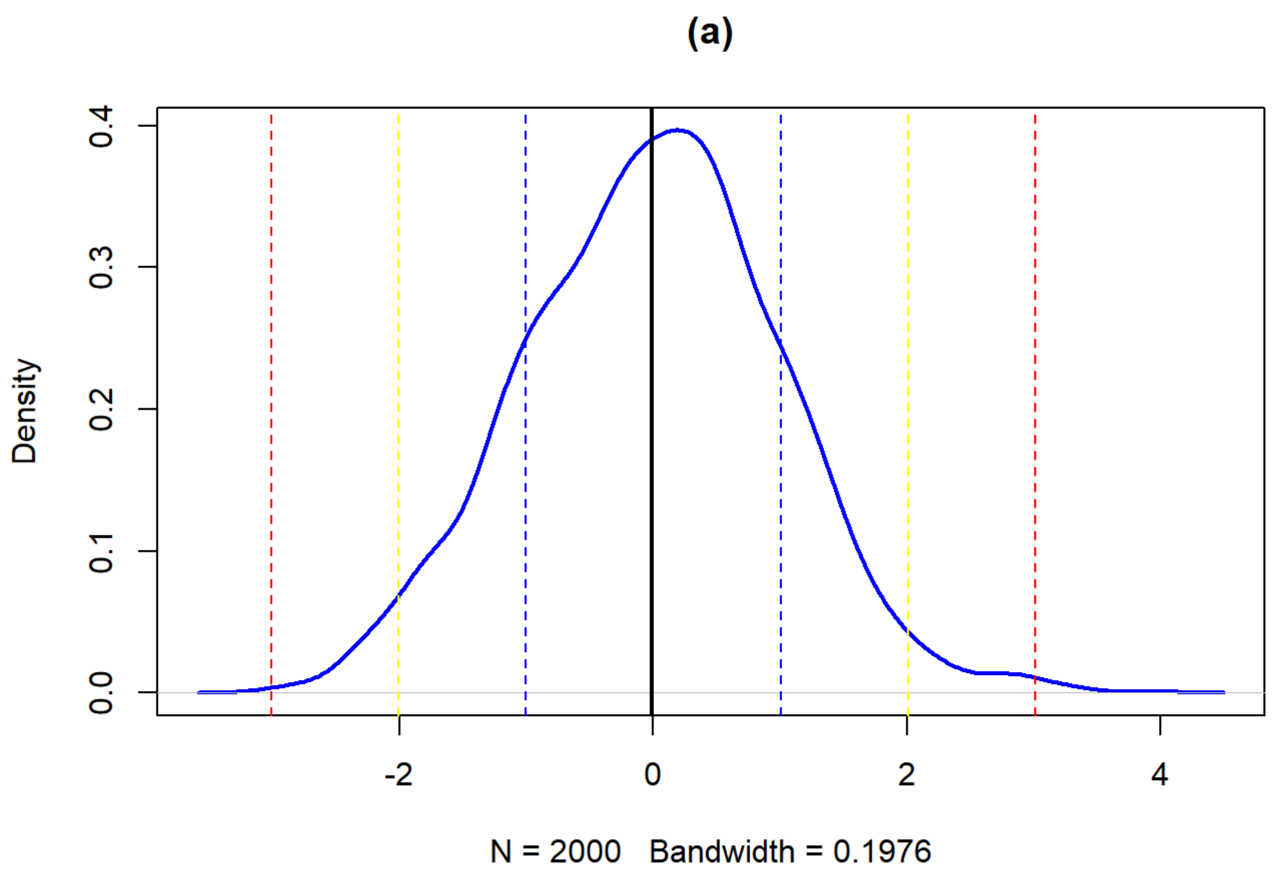
(c) Which measure of central tendency (mean or median) do you think will be more sensitive (will change more) to outliers?

Ans: mean is more sensitive

-Section 2

2-(a)

```
set.seed(110078509)
rdata <-rnorm(n=2000, mean=0, sd=1);
plot(density(rdata), col="blue", lwd=2,
     main = "(a)")
abline(v=mean(rdata), lwd=2)

rdata_std = sd(rdata)

#1 standard deviations (68%)

abline(v=0+1*rdata_std,  lwd=1,  lty="dashed",  col="blue")
abline(v=0-1*rdata_std,  lwd=1,  lty="dashed",  col="blue")
#2 standard deviations (95%)

abline(v=0+2*rdata_std,  lwd=1,  lty="dashed",  col="yellow")
abline(v=0-2*rdata_std,  lwd=1,  lty="dashed",  col="yellow")
#3 standard deviations (99.7%)

abline(v=0+3*rdata_std,  lwd=1,  lty="dashed",  col="red")
abline(v=0-3*rdata_std,  lwd=1,  lty="dashed",  col="red")
```



**** *2-(b)*

```
#Q1
Q1 <- unname(quantile(rdata)['25%'] );Q1 #-0.7121004
```

```
## [1] -0.7121004
```

```
rdata_Q1std_dist <- (Q1-mean(rdata) )/ rdata_std ; rdata_Q1std_dist#[1] -0.6965661
```

```
## [1] -0.6965661
```

```
#Q2
Q2 <- unname(quantile(rdata)['50%']); Q2 #0.01494514
```

```
## [1] 0.01494514
```

```
rdata_Q2std_dist <- (Q2-mean(rdata) )/ rdata_std ; rdata_Q2std_dist # 0.02772567
```

```
## [1] 0.02772567
```

```
#Q3
Q3 <- unname(quantile(rdata)['75%']); Q3#[1] 0.6405586
```

```
## [1] 0.6429939
```

```
rdata_Q3std_dist <- (Q3-mean(rdata) )/ rdata_std ; rdata_Q3std_dist #0.6533957
```

```
## [1] 0.6533957
```

Explain: For Q1, the data point is -0.7121004, it’s -0.6965661 std away from the mean For Q2, the data point is 0.01494514, it’s 0.02772567 std away from the mean For Q3, the data point is 0.6405586, it’s 0.6533957 std away from the mean ****

2-(C) New random dataset that is normally distributed with: n=2000, mean=35, sd=3.5

```
set.seed(110078509)
cdata <-rnorm(n=2000, mean=35, sd=3.5);
cdata_std = sd(cdata)
cdata_Q1 <- unname(quantile(cdata)['25%'] );cdata_Q1 #[1] 32.50765
```

```
## [1] 32.50765
```

```
cdata_Q3 <- unname(quantile(cdata)['75%'] );cdata_Q3 #[1] 37.25048
```

```
## [1] 37.25048
```

```
cdata_Q1std_dist <- (cdata_Q1 - mean(cdata))/cdata_std ; cdata_Q1std_dist#[1] -0.6965661
```

```
## [1] -0.6965661
```

```
cdata_Q3std_dist <- (cdata_Q3 - mean(cdata))/cdata_std ; cdata_Q3std_dist#[1]  0.6533957
```

```
## [1] 0.6533957
```

Former part Explaination Q1 in the new dataset (cdata) is 32.50765, it's -0.6965661 std away from mean Q2 in the new dataset (cdata) is 37.25048, it's 0.6533957 std away from mean

Later part Comparsion (c)cdata Q1, Q3 to (b)rdata Q1, Q3

```
#print('----rdata Q1-----')
rdata_Q1std_dist;
```

```
## [1] -0.6965661
```

```
#print('----cdata Q1-----')
cdata_Q1std_dist;
```

```
## [1] -0.6965661
```

```
#print('----rdata Q3-----')
rdata_Q3std_dist;
```

```
## [1] 0.6533957
```

```
#print('----cdata Q3-----')
cdata_Q3std_dist; #cdata Q3
```

```
## [1] 0.6533957
```

Q1 in the new dataset (cdata) is 32.50765, it's -0.6965661 std away from mean Q3 in the new dataset (cdata) is 37.25048, it's 0.6533957 std away from mean Q1 in the new dataset (rdata) is -0.7121004, it's -0.6965661 std away from the mean Q3 in the new dataset (rdata) is 0.6405586, it's 0.6533957 std away from the mean

Even though the data point is not the same point,the points are as many standard deviations from the mean of it's own.

(d)

```
d123.std <- sd(d123)
d123_Q1 <- unname(quantile(d123)['25%'] );d123_Q1
```

```
## [1] 29.84721
```

```
d123_Q3 <- unname(quantile(d123)['75%'] );d123_Q3
```

```
## [1] 46.01444
```

```
d123_Q1std_dist <- (d123_Q1 - mean(d123))/d123.std ; d123_Q1std_dist
```

```
## [1] -0.6288669
```

```
d123_Q3std_dist <- (d123_Q3 - mean(d123))/d123.std ; d123_Q3std_dist
```

```
## [1] 0.7559303
```

Explaination & Comparsion (d)d123 Q1, Q3 to (b)rdata Q1, Q3 Q1 in the dataset (d123) is 29.84721, it's -0.6288669 std away from the mean of d123 Q3 in the dataset (d123) is 46.01444, it's 0.7559303 std away from the mean of d123

Q1 in the dataset (rdata) is -0.7121004, it's -0.6965661 std away from the mean of rdata Q3 in the dataset (rdata) is 0.6405586, it's 0.6533957 std away from the mean of rdata

Well,unsurprisingly, none of their data points or the relative distance(unit is their own standard deviation) are the same. The reason why it have a huge different is that d123 dataset is left-sknewed. However, rdata is much more like normal distrubution(bell shape)

Section 3

(a) Which formula does Rob Hyndman's answer (1st answer) suggest to use for bin widths/number? What does the Wikipedia article say is the benefit of that formula?

ANS (a)- 1 Freedman-Diaconis It replaces 3.5σ of Scott's rule with 2 IQR, which is less sensitive than the standard deviation to outliers in data.

(b) Compute the bin widths (h) and number of bins (k) according to each of the following formula:

```
set.seed(110078509)
rand_data <- rnorm(800, mean=20, sd = 5)
rand.size <- length(rand_data)
rand.range <- max(rand_data)- min(rand_data)
```

- (b-i). Sturges' formula

```
rand.sturge.bin_num <- ceiling(log(rand.size, base = 2))+1;
sprintf("rand.sturge.bin_num: %d",rand.sturge.bin_num )
```

```
## [1] "rand.sturge.bin_num: 11"
```

```
rand.sturge.bin_width <- ceiling(rand.range/rand.sturge.bin_num) ;
sprintf("rand.sturge.bin_width: %f",rand.sturge.bin_width )
```

```
## [1] "rand.sturge.bin_width: 4.000000"
```

- (b-ii). Scott's normal reference rule (uses standard deviation)

```
Scott.std <- sd(rand_data) #4
#thouht the formula is sample standard deviation(s) instead of standard deviation.
#In order to follow the the requirement above, we use std to replace the s

rand.Scott.bin_width <- 3.49*Scott.std/ (rand.size^(1/3));  #1.852806
rand.Scott.bin_num <- ceiling(rand.range/rand.Scott.bin_width);  # 18
sprintf("rand.Scott.bin_width: %f", rand.Scott.bin_width)
```

```
## [1] "rand.Scott.bin_width: 1.852806"
```

```
sprintf("rand.Scott.bin_num: %d",rand.Scott.bin_num )
```

```
## [1] "rand.Scott.bin_num: 19"
```

-(b-iii). Freedman-Diaconis' choice (uses IQR)

```
rand.Freedman.bin_width <- 2 * IQR(rand_data)/(rand.size^(1/3))
rand.Freedman.bin_num <- ceiling(rand.range/rand.Freedman.bin_width)
sprintf("rand.Freedman.bin_width: %f", rand.Freedman.bin_width)
```

```
## [1] "rand.Freedman.bin_width: 1.467264"
```

```
sprintf("rand.Freedman.bin_num: %d",rand.Freedman.bin_num )
```

```
## [1] "rand.Freedman.bin_num: 24"
```

(c) Repeat part (b) but extend the rand_data dataset with some outliers

```
set.seed(110078509)
out_data <- c(rand_data, runif(10, min=40, max=60))
out.size <- length(out_data);out.size #810
```

```
## [1] 810
```

```
out.range <- max(out_data)- min(out_data); out.range #54.24097
```

```
## [1] 54.24097
```

- c-i. Sturges' formula

```
outdata.sturge.bin_num <- ceiling(log(out.size, base = 2))+1
outdata.sturge.bin_width <-out.range/outdata.sturge.bin_num

sprintf("outdata.sturge.bin_width: %f", outdata.sturge.bin_width)
```

```
## [1] "outdata.sturge.bin_width: 4.930998"
```

```
sprintf("outdata.sturge.bin_num: %d",outdata.sturge.bin_num )
```

```
## [1] "outdata.sturge.bin_num: 11"
```

- c-ii. Scott’s normal reference rule

```
outdata.Scott.std <- sd(out_data)
outdata.Scott.bin_width <- 3.49*outdata.Scott.std/ (out.size^(1/3))

#In order to follow the the requirement above, we use std to replace the s
outdata.Scott.bin_num <- ceiling(out.range/outdata.Scott.bin_width)

sprintf("outdata.Scott.bin_width: %f", outdata.Scott.bin_width)
```

```
## [1] "outdata.Scott.bin_width: 2.236739"
```

```
sprintf("outdata.Scott.bin_num: %d",outdata.Scott.bin_num )
```

```
## [1] "outdata.Scott.bin_num: 25"
```

```
#c-iii. Freedman-Diaconis’ choice (uses IQR)
outdata.Freedman.bin_width <- 2 * IQR(out_data)/(out.size^(1/3))
outdata.Freedman.bin_num <- ceiling(out.range/outdata.Freedman.bin_width)
sprintf("outdata.Freedman.bin_width: %f", outdata.Freedman.bin_width)
```

```
## [1] "outdata.Freedman.bin_width: 1.483496"
```

```
sprintf("outdata.Freedman.bin_num: %d",outdata.Freedman.bin_num )
```

```
## [1] "outdata.Freedman.bin_num: 37"
```

Summary of bin width(h)

The width before outlier(rdata):

```
sprintf("rand.sturge.bin_width: %f",rand.sturge.bin_width )
```

```
## [1] "rand.sturge.bin_width: 4.000000"
```

```
sprintf("rand.Scott.bin_width: %f", rand.Scott.bin_width)
```

```
## [1] "rand.Scott.bin_width: 1.852806"
```

```
sprintf("rand.Freedman.bin_width: %f", rand.Freedman.bin_width)
```

```
## [1] "rand.Freedman.bin_width: 1.467264"
```

The width after outlier (outdata):

```
sprintf("outdata.sturge.bin_width: %f", outdata.sturge.bin_width)
```

```
## [1] "outdata.sturge.bin_width: 4.930998"
```

```
sprintf("outdata.Scott.bin_width: %f", outdata.Scott.bin_width)
```

```
## [1] "outdata.Scott.bin_width: 2.236739"
```

```
sprintf("outdata.Freedman.bin_width: %f", outdata.Freedman.bin_width)
```

```
## [1] "outdata.Freedman.bin_width: 1.483496"
```

The percentage they changed separately

```
Sturge.change <- abs(outdata.sturge.bin_width-rand.sturge.bin_width)/rand.sturge.bin_width
Scott.change <- abs(outdata.Scott.bin_width-rand.Scott.bin_width)/rand.Scott.bin_width
Freedman.change<- abs(outdata.Freedman.bin_width-rand.Freedman.bin_width)/rand.Freedman.bin_w
idth

Change_list <- c(Sturge.change, Scott.change, Freedman.change)
names(Change_list) <- c('Sturge', 'Scott', 'Freedman'); Change_list
```

```
##     Sturge      Scott   Freedman
## 0.23274942 0.20721726 0.01106278
```

Ans: it show that the proportion of Freedman changed the least after the outliers was added

Explain: Because Sturges is based on sample size and Scott is based on standard deviation. However, Freedman method is based on IQR, which is the idea of median(Q2) instead of mean. Hence, it’s less sensitive among the three as the sample size rising due to the adding of outliers.

- Thanks for your review and efforts *