Homework 7. Based on Chapter 7 of Trosset's textbook. Due Friday, October 15th. All of these questions have computational solutions. Your

Preamble

In this HW you will do two kinds of things. The first is compute some population parameters and compare them to sample plug-in estimates. The second is that you will simply compute some sample plug in estimators for samples that are taken from populations that are not available for direct scrutiny. I have created a file `hw7.RData` that contains five populations and five data vectors. The five populations come from two different kinds of sources. The first source is random generation; I have randomly generated two populations using random variable generators. The second source is large datasets; I have found (and modified where appropriate) three large datasets and we will treat them like a population (instead of like a sample). The five samples are all taken from datasets that are available in different `R` packages.

You need to download the `hw7.RData` file and store it in some directory (folder) on you computer. I have mine stored in the directory

```
"/Users/ajwomack/OneDrive - Indiana University/stat520Fa2021/Assignments/HW7"
```

At the top of your homework `.R` file, you need to have the three following lines.

```
setwd("/Users/ajwomack/OneDrive - Indiana University/stat520Fa2021/Assignments/HW7")
load(file = "hw7.RData")
set.seed(0123456789)
```

where you replace the directory in quote marks in `setwd` with the directory where you have saved `hw7.RData` and the integer `0123456789` in `set.seed` with your 10-digit Student ID. This will load the data in the file `hw7.RData` and set the seed of `R`'s random number generator.

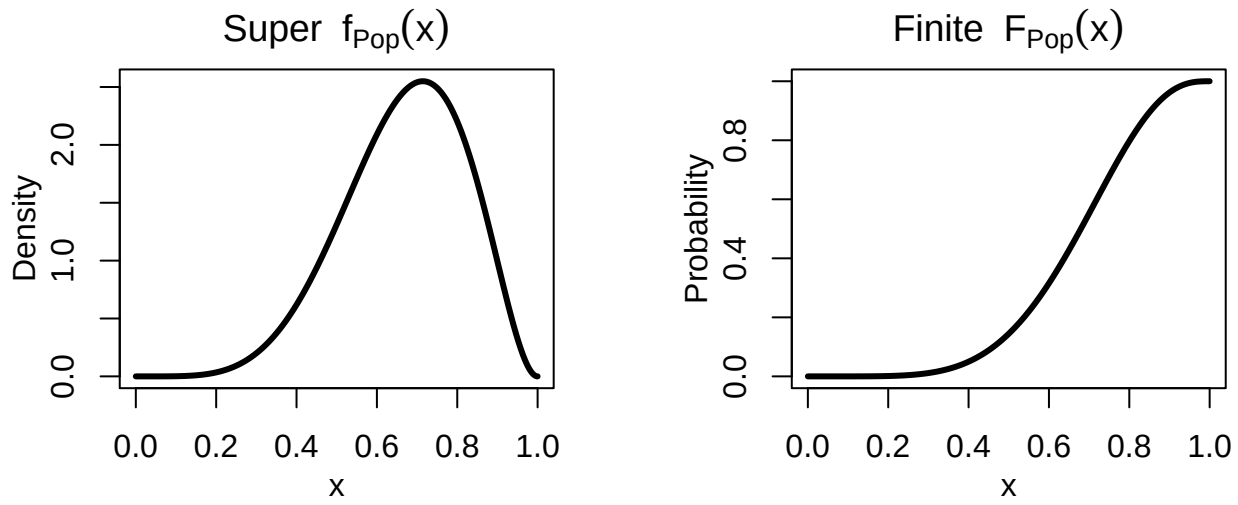
Example Population Parameter Calculations

I am going to generate a finite population X values from a super-population. The super-population X value distribution is a Beta distribution with left shape 6 and right shape 3. The density and distribution functions of the super-population are given by the following functions on the interval $(0, 1)$

$$f_{Pop}(x) = 168x^{6-1}(1-x)^{3-1} \quad \text{and} \quad F_{Pop}(x) = 21x^8 - 48x^7 + 28x^6$$

and the graphs

```
par(mfrow=c(1,2))
x = seq(0,1,length.out=1000)
plot(dbeta(x,6,3)~x,main='',ylab='',xlab='',type='l',lwd=3)
title(main=expression('Super '~f[Pop](x)),line=1)
title(ylab='Density',line = 2)
title(xlab='x',line = 2)
plot(pbeta(x,6,3)~x,main='',ylab='',xlab='',type='l',lwd=3)
title(main=expression('Finite '~F[Pop](x)),line=1)
title(ylab='Probability',line = 2)
title(xlab='x',line = 2)
```

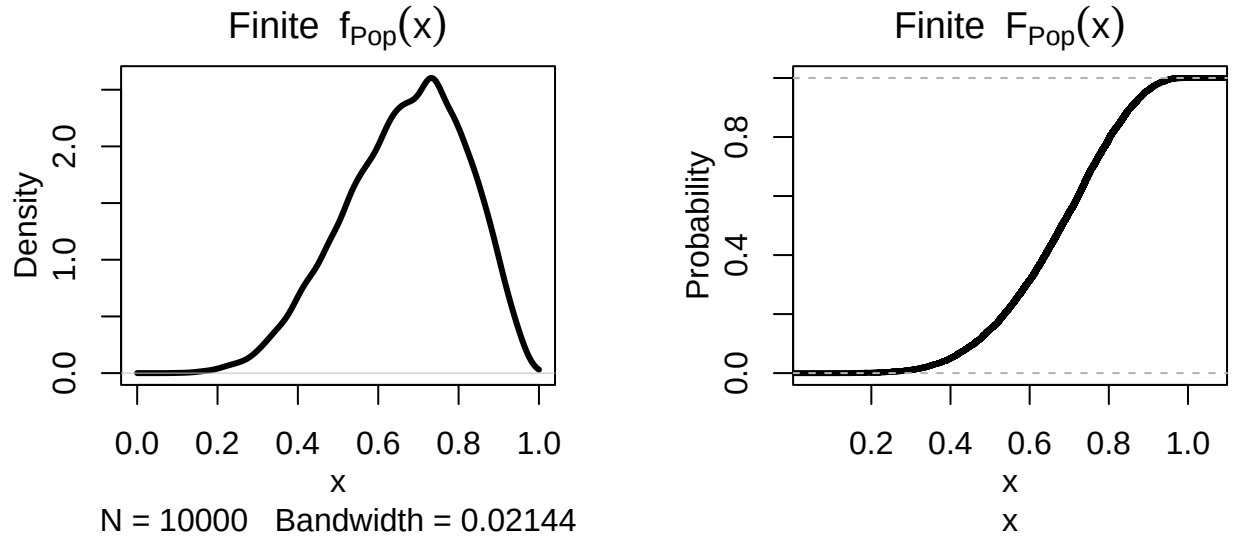


I am going to draw a finite population of X values of size $N = 10,000$ from this super-population. So the finite population X values are going to be close to the super-population X values but not absurdly close. (I am using 10,000 instead of 1,000,000 or more mostly to keep the time required for rendering of the distribution function from getting out of hand.) First, I draw the finite population

```
x_pop = rbeta(1e4,6,3)
```

Now we look at the density and distribution plots for the finite population

```
par(mfrow=c(1,2))
plot(density(x_pop,from=0,to=1),main='',ylab='',lwd=3)
title(main=expression('Finite '~f[Pop](x)),line=1)
title(ylab='Density',line = 2)
title(xlab='x',line = 2)
plot(ecdf(x_pop),main='',ylab='',lwd=3,pch='.')
title(main=expression('Finite '~F[Pop](x)),line=1)
title(ylab='Probability',line = 2)
title(xlab='x',line = 2)
```



Let X be the random variable defined on the population that maps unit i with the corresponding value x_i and let the underlying distribution on $\{1, \ldots, N = 10000\}$ have probability $1/N$ assigned to each $\{i\}$.

Common population parameters of interest include probabilities of sets, quantiles, expectations of functions,

and combinations of such quantities. Below is some commented R code for computing such quantities for the population.

```
# probability that X is below some value x  between 0 and 1
x_value = 0.7
mean(x_pop < x_value)
```

```
## [1] 0.5474
```

```
# probability that X is above some value x  between 0 and 1
x_value = 0.23
mean(x_pop > x_value)
```

```
## [1] 0.997
```

```
# probability that X is between two values between 0 and 1
x_value_1 = 0.23
x_value_2 = 0.7
mean(x_pop>x_value_1 & x_pop<x_value_2)
```

```
## [1] 0.5444
```

```
# median of population
median(x_pop)
```

```
## [1] 0.6796783
```

```
# a vector quantiles (x values corresponding to the probabilities input)
quantile(x_pop,c(0.1,0.3,0.7,0.9))
```

```
##       10%       30%       70%       90%
## 0.4590630 0.5902930 0.7590894 0.8548961
```

```
# inter-quartile range
quantile(x_pop,0.75) - quantile(x_pop,0.25)
```

```
##       75%
## 0.2160788
```

```
# quartile based skewness (Yule)
upper_length = quantile(x_pop,0.75) - quantile(x_pop,0.5)
lower_length = quantile(x_pop,0.5) - quantile(x_pop,0.25)
iqr = quantile(x_pop,0.75) - quantile(x_pop,0.25)
(upper_length - lower_length) / iqr
```

```
##         75%
## -0.06912377
```

```
# mean absolute deviation
mean(abs(x_pop-median(x_pop)))
```

```
## [1] 0.1223293
```

```
# absolute deviation based skewness (i just made this up)
upper = mean((x_pop-median(x_pop))*(x_pop>median(x_pop)))
lower = mean((median(x_pop)-x_pop)*(x_pop<median(x_pop)))
mad = mean(abs(x_pop-median(x_pop)))
```

```
(upper - lower) / mad
```

```
## [1] -0.106849
```

```
# expectation
mean(x_pop)
```

```
## [1] 0.6666075
```

```
# variance (two formulas)
mean((x_pop-mean(x_pop))^2) # from definition
```

```
## [1] 0.02259785
```

```
mean(x_pop^2) - mean(x_pop)^2 # (see Theorem 4.5 in Trosset)
```

```
## [1] 0.02259785
```

```
# population standard deviation
sqrt(mean(x_pop^2) - mean(x_pop)^2)
```

```
## [1] 0.1503258
```

```
# expectations based skewness (Pearson)
mean((x_pop-mean(x_pop))^3) / (mean((x_pop-mean(x_pop))^2))^(3/2)
```

```
## [1] -0.4022115
```

```
# expectation of some functions
mean(sqrt(x_pop))
```

```
## [1] 0.8107289
```

```
mean(1/x_pop)
```

```
## [1] 1.601706
```

```
mean(cos(pi*x_pop))
```

```
## [1] -0.4524144
```

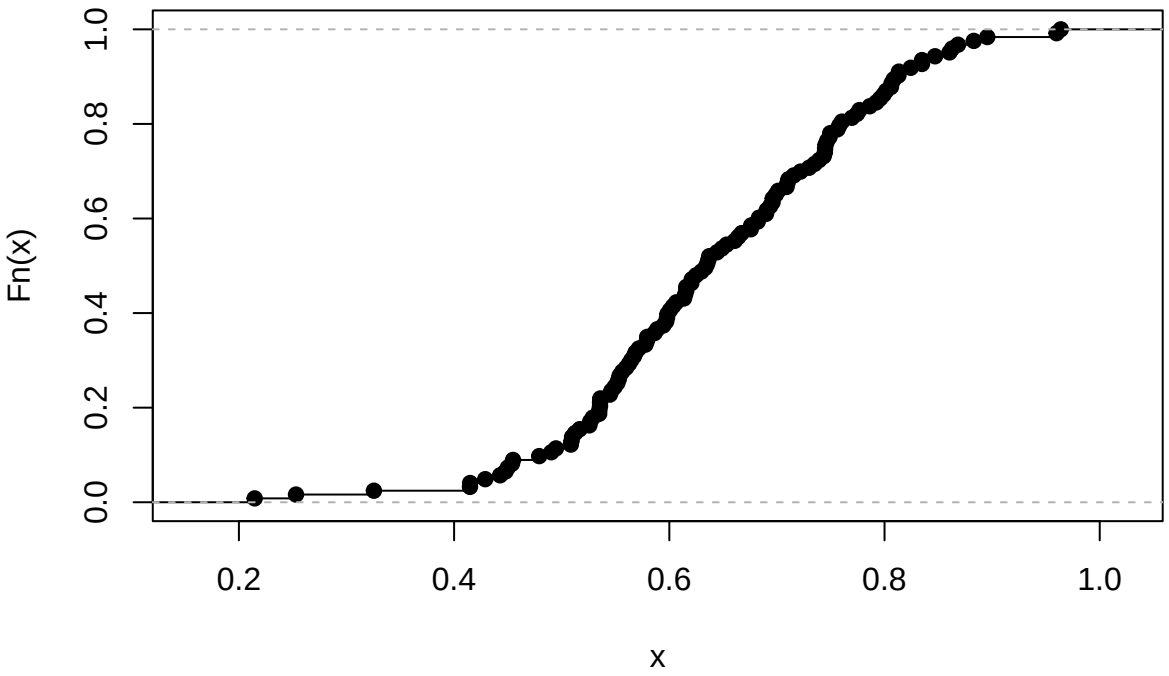
```
mean(sin(pi*x_pop))
```

```
## [1] 0.7709502
```

Example Plug-in Estimator Calculations

The idea of a plug-in estimator is to us the exact same operation on a sample that was used on the population to get the parameter value. Instead of doing the operation on the distribution of the population, we do the operation on the empirical distribution of the sample. So, in order to compute a plug-in estimator, we first need to have taken a sample. Let's do that now. I will take a random sample of size 123 from the population using the `sample` function. I will then plot the empirical distribution function of the sample.

```
x_samp = sample(x_pop,123)
plot(ecdf(x_samp))
```



The plug-in estimators corresponding to all of the population parameters we computed before are obtained by simply replacing `x_pop` with `x_samp` in our code.

```
# probability that X is below some value x  between 0 and 1
x_value = 0.7
mean(x_samp < x_value)
```

```
## [1] 0.6504065
```

```
# probability that X is above some value x  between 0 and 1
x_value = 0.23
mean(x_samp > x_value)
```

```
## [1] 0.9918699
```

```
# probability that X is between two values between 0 and 1
x_value_1 = 0.23
x_value_2 = 0.7
mean(x_samp>x_value_1 & x_samp<x_value_2)
```

```
## [1] 0.6422764
```

```
# median of sample
median(x_samp)
```

```
## [1] 0.6351804
```

```
# a vector quantiles (x values corresponding to the probabilities input)
quantile(x_samp,c(0.1,0.3,0.7,0.9))
```

```
##       10%       30%       70%       90%
## 0.4912125 0.5662050 0.7250970 0.8120299
```

```r
# inter-quartile range
quantile(x_samp,0.75) - quantile(x_samp,0.25)
```

```
##       75%
## 0.1926448
```

```r
# quartile based skewness (Yule)
upper_length = quantile(x_samp,0.75) - quantile(x_samp,0.5)
lower_length = quantile(x_samp,0.5) - quantile(x_samp,0.25)
iqr = quantile(x_samp,0.75) - quantile(x_samp,0.25)
(upper_length - lower_length) / iqr
```

```
##       75%
## 0.1398839
```

```r
# mean absolute deviation
mean(abs(x_samp-median(x_samp)))
```

```
## [1] 0.109865
```

```r
# absolute deviation based skewness (i just made this up)
upper = mean((x_samp-median(x_samp))*(x_samp>median(x_samp)))
lower = mean((median(x_samp)-x_samp)*(x_samp<median(x_samp)))
mad = mean(abs(x_samp-median(x_samp)))
(upper - lower) / mad
```

```
## [1] 0.07979908
```

```r
# expectation
mean(x_samp)
```

```
## [1] 0.6439475
```

```r
# variance (two formulas)
mean((x_samp-mean(x_samp))^2) # from definition
```

```
## [1] 0.01851506
```

```r
mean(x_samp^2) - mean(x_samp)^2 # (see Theorem 4.5 in Trosset)
```

```
## [1] 0.01851506
```

```r
# sample standard deviation
sqrt(mean(x_samp^2) - mean(x_samp)^2)
```

```
## [1] 0.1360701
```

```r
# expectations based skewness (Pearson)
mean((x_samp-mean(x_samp))^3) / (mean((x_samp-mean(x_samp))^2))^(3/2)
```

```
## [1] -0.2229475
```

```r
# expectation of some functions
mean(sqrt(x_samp))
```

```
## [1] 0.7975979
```

```
mean(1/x_samp)
```

```
## [1] 1.643593
```

```
mean(cos(pi*x_samp))
```

```
## [1] -0.4012264
```

```
mean(sin(pi*x_samp))
```

```
## [1] 0.8201071
```

1 Population and plug-in estimator problems

Question 1.a

If you have loaded the file `hw7.RData` correctly, there should be a population in your global environment called `negative_binomial_pop`. This population was drawn from a negative binomial distribution. Compute the population parameters of the standard deviation, mean absolute deviation, and inter-quartile range. Take a random sample of size 243 from the population. Compute the plug-in estimators of the parameters you just computed. What do you think about the differences between the parameters' values and the corresponding values of the plug-in estimators?

Question 1.b

If you have loaded the file `hw7.RData` correctly, there should be a population in your global environment called `laplace_pop`. This population was drawn from a Laplace distribution. Compute the population parameters of the quartile based skewness, absolute deviation based skewness, and expectations based skewness. Take a random sample of size 85 from the population. Compute the plug-in estimators of the parameters you just computed. Do you think the super-population is skewness 0 based on these finite population measures? Why or why not? Do you think you see the same information in the sample estimates you obtained? Why or why not?

Question 1.c

If you have loaded the file `hw7.RData` correctly, there should be a population in your global environment called `singh_pop`. This population is the gene expression data from 10+ individuals over 6000+ genes with gene and individual information stripped away. Compute the population parameters of the mean, the median, and the average of the 0.25 and 0.75 quantiles. Take a random sample of size 64 from the population. Compute the plug-in estimators of the parameters you just computed. Do the plug-in estimators correspond well to the population parameters? As measures of center, what are the estimates telling you about the center of the population.

Question 1.d

If you have loaded the file `hw7.RData` correctly, there should be a population in your global environment called `treering_pop`. This population is normalized treering measurements over about 8000 years with the smallest value (normalized to 0) removed. Compute the population parameters of the deciles (the

quantiles corresponding to tenths). Take a random sample of size 202 from the population. Compute the plug-in estimators of the parameters you just computed. Do the plug-in estimators correspond well to the population parameters? What do the estimators tell you about the distribution of the population? For which deciles would you feel most comfortable making statements about the population using the estimates?

Question 1.e

If you have loaded the file `hw7.RData` correctly, there should be a population in your global environment called `speed_pop`. This population is normalized speed measurements before and after warning sign erection with location and warning sign location removed. Compute the population parameters of the expected values of the functions `log(x)`, `exp(x)`, and `sqrt(x)` applied to the random variable X. Take a random sample of size 49 from the population. Compute the plug-in estimators of the parameters you just computed. Do the plug-in estimators correspond well to the population parameters? Would you trust the estimate of one function's expected value more than others? Why or why not?

2 Plug-in estimator problems

Question 2.a

An alternative to standard deviation (tied to the mean) and mean absolute deviation (tied to the mean) to try to quantify spread is to use a different idea of distance and define the spread as the minimal value of that distance over possible values of the action we could take. We talked about this in class and I thought it would be useful to have you compute an estimator of one such measure. The measure we will think about in this problem is spread defined by finding the a that minimizes

$$S(a) = \left(\mathrm{E}\left[|X-a|^{3/2}\right]\right)^{2/3}$$

This is trying to compromise in some way between the power 1 in the expectation for mean absolute deviation and the power 2 in the expectation for the mean squared error. Below, I have provided a function that returns both the measure of center (best action a) and the measure of spread (corresponding value of the function $S(a)$).

```
S_fun_opt = function(x){
  f = function(a) (mean(abs(x-a)^(3/2)))^(2/3)
  opt = optimize(f,range(x),tol=1e-8)
  return(c( center = opt$minimum, spread = opt$objective))
}
```

add this function to your code.

If you have loaded the file `hw7.RData` correctly, there should be a data vector in your global environment called `acme_samp`, which is the excess returns for the Acme Cleveland Corporation over a five year period. Compute plug-in estimates of the mean, standard deviation, median, mean absolute deviation, and the measures of center and spread that come from `S_fun_opt`. What do the different measures of center tell you about the population? What is the ordering of the sizes of the different measures of spread?

Question 2.b

There are alternative measures of spread based on quantiles also. Instead of using the inter-quartile range, we could take the difference in any quantiles that are symmetric (have the same probability above the upper

quantile and below the lower quantile). The inter-quartile range is based on having 0.25 probability above the upper quantile and 0.25 probability below the lower quantile. Below, I have provided a function that returns the quantile based spread in a data vector `x` for a given probability value (between 0 and 0.5).

```
quantile_spread_fun = function(x,prob){
  if(prob<=0 || prob>=0.5){
    stop('probability must be between 0 and 0.5')
  }
  return(quantile(x,1-prob) - quantile(x,prob))
}
```

add this function to your code.

If you have loaded the file `hw7.RData` correctly, there should be a data vector in your global environment called `melanoma_samp`, which are measures of melanoma thickness for around 200 individuals. Compute plug-in estimates of the quantile based spread for this data for `prob` being 0.15, 0.2, 0.25, and 0.3. What do the different measures of spread tell you about the population? What is the ordering of the sizes of the different measures of spread?

Question 2.c

An alternative popular way to compute a measure of center is the clipped mean. The clipped mean excludes data points outside of some symmetric quantiles and computes the mean of the remaining data. Below, I have provided a function that returns the clipped mean of a data vector `x` for a given probability value (between 0 and 0.5).

```
clipped_mean_fun = function(x,prob){
  if(prob<=0 || prob>=0.5){
    stop('probability must be between 0 and 0.5')
  }
  x_clipped = x[x<=quantile(x,1-prob) & x>=quantile(x,prob)]
  return(mean(x_clipped))
}
```

add this function to your code.

If you have loaded the file `hw7.RData` correctly, there should be a data vector in your global environment called `poisons_samp`, which are survival times in units of 10 hours for around 500 poisoned animals. Compute plug-in estimates of the clipped mean for this data for `prob` being 0, 0.05, 0.1, and 0.15. What do the different measures of center tell you about the population? Would you feel comfortable throwing away data when computing the measure of center for a sample of size about 50?

Question 2.d

Skewness tries to measure asymmetry in a population. You already computed three measures of skewness in problem 1.b, whose calculations were given in the copious examples before the first set of questions. Pearson's skewness is related to expectations and quartile-based skewness is related to medians. There are two other definitions of skewness that look at the difference between the mean and the median and make this nubmer unitless by dividing by some measure of spread. If we let $\mu = \mathrm{E}[X]$ and $\nu = \mathrm{median}[X]$, then two such measures of skew are given by

$$\mathrm{Skew}_{\mathrm{SD}} = \frac{\mu - \nu}{\sqrt{\mathrm{E}[(X-\mu)^2]}} \quad \text{and} \quad \mathrm{Skew}_{\mathrm{MAD}} = \frac{\mu - \nu}{\mathrm{E}[|X-\mu|]}$$

These are in addition to the moment based skewness given by

$$\text{Skew}_{\text{Mom}} = \frac{\text{E}[(X-\mu)^3]}{(\text{E}[(X-\mu)^2])^{3/2}}$$

If you have loaded the file `hw7.RData` correctly, there should be a data vector in your global environment called `tree_samp`, which are heights of around 30 felled black cherry trees. Compute plug-in estimates of these three measures of skewness. Also, produce a plot of a kernel density estimate of the sample using the code `plot(density(tree_samp))`. Based on the results of this analysis, do you think that the data are skewed? If so, is the skew positive (towards the right) or negative (towards the left)?

Question 2.e

In addition to measures of skewness based on means, meadians, spread, and moments are measures of skewness based on quantiles. These are based on the differences between some quantiles and the median and is made unitless by dividing by the "spread" defined by the same quantiles. For $0 < p < 0.5$, the p-th quantile based measure of skewness is given by

$$Skew_p = \frac{\text{Quantile}(X,p) + \text{Quantile}(X,1-p) - 2 \times \text{Quantile}(X,0.5)}{\text{Quantile}(X,1-p) - \text{Quantile}(X,p)}$$

Below, I have provided a function that returns the quantile based skewness of a data vector `x` for a given probability value (between 0 and 0.5).

```
skew_quantile_fun = function(x,prob){
  if(prob<=0 || prob>=0.5){
    stop('probability must be between 0 and 0.5')
  }
  upper_quant = quantile(x,1-prob)
  lower_quant = quantile(x,prob)
  median_x = quantile(x,0.5)
  numerator = upper_quant+lower_quant - 2*median_x
  denominator = upper_quant - lower_quant
  return(numerator / denominator)
}
```

add this function to your code.

If you have loaded the file `hw7.RData` correctly, there should be a data vector in your global environment called `gravity_samp`, which are measurements of accelaration due to gravity in around 80 experiments. Compute plug-in estimates of the quantile based skewness for this data for `prob` being 0.15, 0.2, 0.25, and 0.3. What do the different measures of skewness tell you about the population? Are you more comfortable with one value of `prob` to use in this function versus another? Why or why not?