Project 2
STAT 355

Sabbir Ahmed

April 12, 2017

1 Part 1

1000 random samples of size 40 were generated from normal distribution with mean $\mu = 3$ and standard deviation $\sigma = 2$.

```r
# initialize parameters for normal distribution
N <- 40 # size
mu <- 3 # mean
sigma <- 2 # standard deviation

sampMeans <- randDist(N, mu, sigma, "normal", "part1.tex")
plotHist(sampMeans, "hist1.png", 0.1)
```

1.1 Output

The first sample mean and standard deviation were computed:

$$E(\overline{X}) = 3.110, \ \ \sigma_{\overline{X}} = 0.316$$

All the samples were then used to find the sample mean and standard deviation. The theoretical values were also computed based on the relationships:

$$\mu = \mu$$
$$E(\overline{X}) = \mu$$
$$\sigma = \sigma$$
$$\sigma_{\overline{X}} = \frac{\sigma}{\sqrt{n}}$$

	Computed	**Theoretical**
μ	3.000	3.000
E($\overline{X}$)	3.007	3.000
σ	2.000	2.000
$\sigma_{\overline{X}}$	0.309	0.316

1.2 Distribution

Distribution of the data was plotted with a histogram using ggplot2 in Figure 1.

```
ggplot() + aes(sampMeans) +
    geom_histogram(binwidth=0.1, color="black", fill="white") +
    labs(y="Count", x="Sample Means")
```

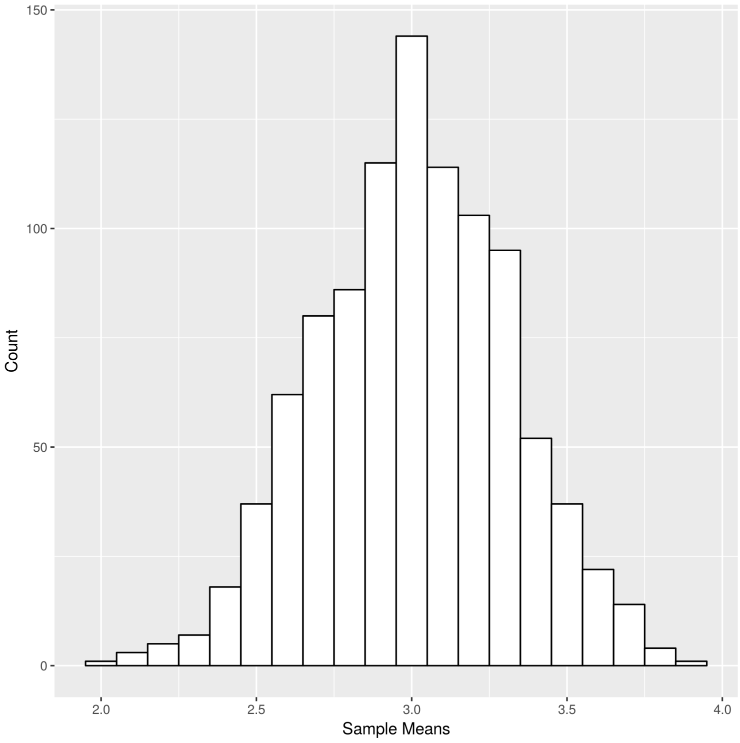

Figure 1: Histogram of the Generated Data

2 Part 2

1000 random samples of size 15 were generated from a binomial distribution with n = 10 and standard deviation p = 0.15.

```
# initialize parameters for binomial distribution
N <- 15
n <- 10
p <- 0.15

sampMeans <- randDist(N, n, p, "binomial", "part2.tex")
plotHist(sampMeans, "hist2.png", 0.1)
```

2.1 Output

The first sample mean and standard deviation were computed:

$$E(\overline{X}) = 1.667,\ \sigma_{\overline{X}} = 0.292$$

All the samples were then used to find the sample mean and standard deviation. The theoretical values were also computed based on the relationships:

$$\mu = np$$
$$E(\overline{X}) = np$$
$$\sigma = \sqrt{np(1-p)}$$
$$\sigma_{\overline{X}} = \sqrt{\frac{np(1-p)}{N}}$$

	Computed	**Theoretical**
μ	1.500	1.500
E($\overline{X}$)	1.492	1.500
σ	1.129	1.129
$\sigma_{\overline{X}}$	0.289	0.292

2.2 Distribution

Distribution of the data was plotted with a histogram using ggplot2 in Figure 2.

```
# plot a histogram of the data
ggplot() + aes(sampMeans) +
    geom_histogram(binwidth=0.1, color="black", fill="white") +
    labs(y="Count", x="Sample Means")
```

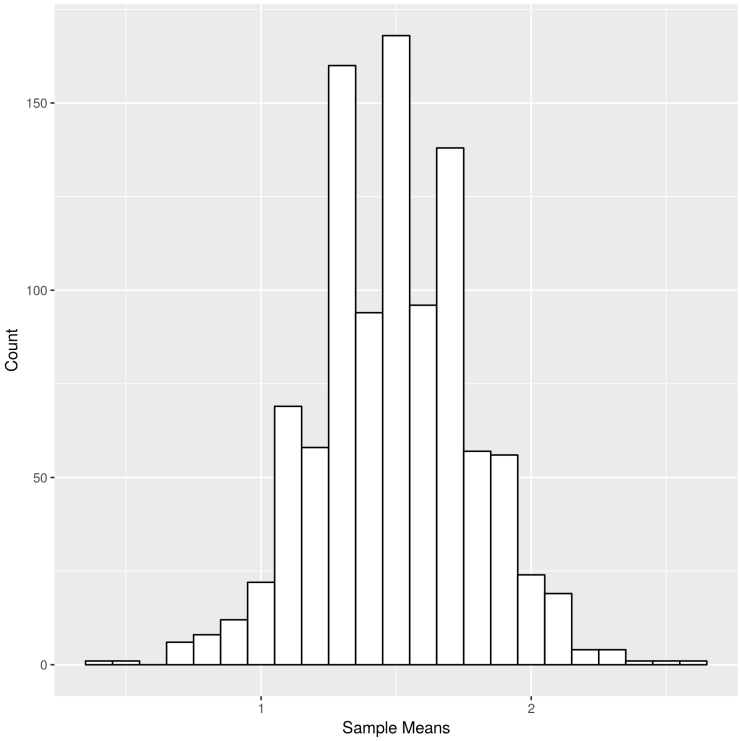

Figure 2: Histogram of the Generated Data

3 Part 3

1000 random samples of size 120 were generated from a binomial distribution with n = 10 and standard deviation p = 0.15.

```
# initialize parameters for binomial distribution
N <- 120
n <- 10
p <- 0.15

sampMeans <- randDist(N, n, p, "binomial", "part3.tex")
plotHist(sampMeans, "hist3.png", 0.025)
```

3.1 Output

The first sample mean and standard deviation were computed:

$$E(\overline{X}) = 1.658, \ \sigma_{\overline{X}} = 0.103$$

All the samples were then used to find the sample mean and standard deviation. The theoretical values were also computed based on the relationships:

$$\mu = np$$
$$E(\overline{X}) = np$$
$$\sigma = \sqrt{np(1-p)}$$
$$\sigma_{\overline{X}} = \sqrt{\frac{np(1-p)}{N}}$$

	Computed	**Theoretical**
μ	1.500	1.500
E($\overline{X}$)	1.500	1.500
σ	1.129	1.129
$\sigma_{\overline{X}}$	0.104	0.103

3.2 Distribution

Distribution of the data was plotted with a histogram using ggplot2 in Figure 3.

```
# plot a histogram of the data
ggplot() + aes(sampMeans) +
    geom_histogram(binwidth=0.025, color="black", fill="white") +
    labs(y="Count", x="Sample Means")
```

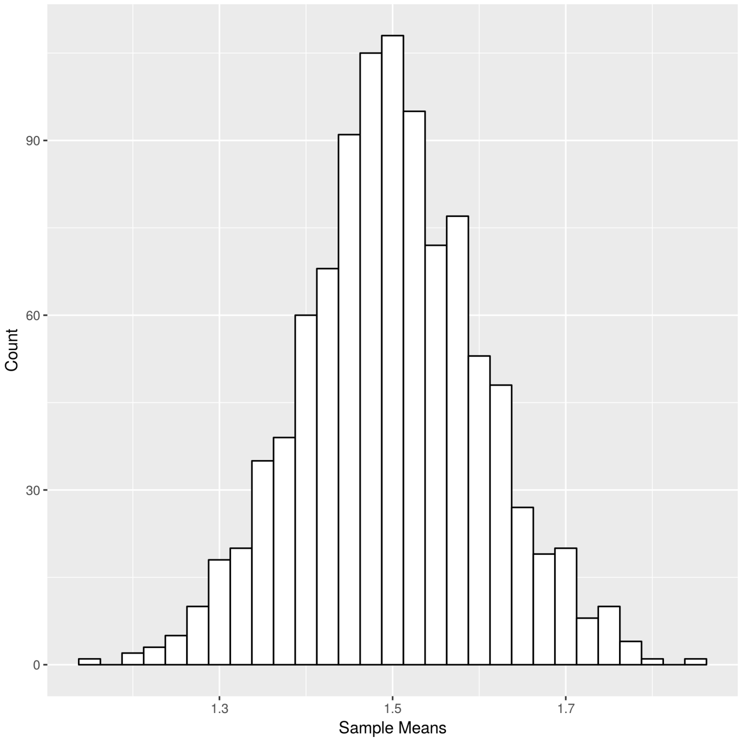

Figure 3: Histogram of the Generated Data

4 Conclusion

The sample means for the normal and binomial distributions all generated normal distributions. The Part 1 distribution (Figure 1) demonstrates a normal sample mean distribution. The theoretical sample mean and standard deviation appear very close to the computed values.

For the binomial distributions, it can be observed that the distribution with the larger sample size of 150 per random sample yielded sample mean and standard deviation values much closer to their theoretical values. The larger sample size also generated a normal distribution of sample means for the binomial distribution.

The normal distributions generated from the various distributions essentially demonstrated the central limit theorem, which implies a given distribution with a mean μ and standard deviation σ, the sampling distribution of the mean approaches a normal distribution with a mean $(E(\overline{X}))$ and a standard deviation $\sqrt{\frac{\sigma}{N}}$ as N, the sample size, increases. [1]

References

[1] Central Limit Theorem
http://mathworld.wolfram.com/CentralLimitTheorem.html

main.R

```
# main.R
# This file contains the implementation of the functions in the Project 2
# NOTE: THIS SCRIPT WAS COMPILED ON A LINUX MACHINE - SOME STATEMENTS MAY THROW
# WARNINGS OR ERRORS IN OTHER SYSTEMS

library(ggplot2)  # for generating high quality plots
set.seed(0)  # seed the random generators

# LaTex template for the output
outputTemplate <- "\\subsection{Output}

    The first sample mean and standard deviation were computed:

    \\[ E(\\overline{X}) = %.3f, \\ \\sigma_{\\overline{X}} = %.3f \\]

    All the samples were then used to find the sample mean and standard
    deviation. The theoretical values were also computed based on the
    relationships:

    \\[ \\mu = %s \\]
    \\[ E(\\overline{X}) = %s \\]
    \\[ \\sigma = %s \\]
    \\[ \\sigma_{\\overline{X}} = %s \\]

    \\begin{table}[h]
        \\centering
        \\begin{tabular*}{200pt}{@{\\extracolsep{\\fill}} c c c}

        & \\textbf{Computed} & \\textbf{Theoretical} \\\\
        \\hline
        $\\mu$ & %.3f  & %.3f \\\\
        E($\\overline{X}$) & %.3f & %.3f \\\\
        $\\sigma$ & %.3f & %.3f \\\\
        $\\sigma$\\textsubscript{$\\overline{X}$} & %.3f & %.3f \\\\

        \\end{tabular*}
    \\end{table}
"

# global variables
NUMSAMPS <- 1000  # number of random samples per distribution

randDist <- function(N, a, b, distType, outputFile) {
    # Generates a random normal or binomial distribution.
    #
    # Args:
    #   N: size of sample
    #   a: First distribution parameter.
    #       a = mu for normal distribution
    #       a = n for binomial distribution
    #   b: Second distribution parameter.
    #       b = sigma for normal distribution
    #       b = p for binomial distribution
    #   distType: Type of distribution.
    #       Options: "normal", "binomial"
    #   outputFile: Name of LaTex output file

    # initialize variables to hold data for the first sample
    firstMean <- firstStd <- 0

    # initialize distribution variables
    mu <- sigma <- n <- p <- 0

    # initialize empty arrays
    sampMeans <- generatedData <- rep(0, times=NUMSAMPS)
```

```
# rename parameter values to distribution parameters for convenience
if (distType == "normal") {
    mu <- a
    sigma <- b
} else if (distType == "binomial") {
    n <- a
    p <- b
}

# generate 1000 samples
for (i in 1:NUMSAMPS) {

    # generate distribution based on type chosen
    if (distType == "normal") {
        generatedData <- rnorm(N, mu, sigma)
    } else if (distType == "binomial") {
        generatedData <- rbinom(N, n, p)
    }

    # store the sample means in vector
    sampMeans[i] = sum(generatedData)/N

    if (i == 1) {

        # store the first sample mean
        firstMean = sum(generatedData)/N

        # store the first sample standard deviation
        if (distType == "normal") {
            # sigma/sqrt(N) if normal
            firstStd = sigma/sqrt(N)
        } else if (distType == "binomial") {
            # sqrt(n*p*(1-p)/N) if binomial
            firstStd = sqrt(n*p*(1-p)/N)
        }

    }

}

# generate templates based on the distribution type and computed values
outputData <- ’’
if (distType == "normal") {
    outputData <- sprintf(
        outputTemplate,
        firstMean, firstStd,
        "\\mu", "\\mu", "\\sigma", "\\frac{\\sigma}{\\sqrt{n}}",
        mu, mu,
        mean(sampMeans), mu,
        sigma, sigma,
        sd(sampMeans), sigma/sqrt(N)
    )
} else if (distType == "binomial") {
    outputData <- sprintf(
        outputTemplate,
        firstMean, firstStd,
        "np", "np", "\\sqrt{np(1-p)}", "\\sqrt{\\frac{np(1-p)}{N}}",
        n*p, n*p,
        mean(sampMeans), n*p,
        sqrt(n*p*(1-p)), sqrt(n*p*(1-p)),
        sd(sampMeans), sqrt(n*p*(1-p)/N)
    )
}

# dump output to LaTex modules
sink(outputFile, append=FALSE, split=FALSE)
cat(outputData)
```

```
    sink()  # return stdout to console

    return(sampMeans)

}


plotHist <- function(sampMeans, figureFile, binwidth) {
    # Plot a histogram of the data
    #
    # Args:
    #   sampMeans: the sample means generated from the random distributions
    #   figureFile: file name of the output plot
    #   binwidth: width of the bins of the histogram

    histPlot <- ggplot() + aes(sampMeans) +
        geom_histogram(binwidth=binwidth, color="black", fill="white") +
        labs(y="Count", x="Sample Means")

    # save plot to filename
    ggsave(filename=paste0("figures/", figureFile), plot=histPlot)

}


# ------------------------------ Part 1 ------------------------------

# initialize parameters for normal distribution
N <- 40  # size
mu <- 3  # mean
sigma <- 2  # standard deviation

sampMeans <- randDist(N, mu, sigma, "normal", "part1.tex")
plotHist(sampMeans, "hist1.png", 0.1)


# ------------------------------ Part 2 ------------------------------

# initialize parameters for binomial distribution
N <- 15
n <- 10
p <- 0.15

sampMeans <- randDist(N, n, p, "binomial", "part2.tex")
plotHist(sampMeans, "hist2.png", 0.1)


# ------------------------------ Part 3 ------------------------------

# initialize parameters for binomial distribution
N <- 120
n <- 10
p <- 0.15

sampMeans <- randDist(N, n, p, "binomial", "part3.tex")
plotHist(sampMeans, "hist3.png", 0.025)
```