MAT750 Project 1

Wei Chieh Chen

2025-01-31

On this project, I refer to the resources on the blackboard, the article by Lisa Sullivan PhD, Boston University bu.edu/sph/profile/lisa-sullivan/, the article of LATEX Mathematical Symbols https://www.cmor-faculty.rice.edu/~heinken/latex/symbols.pdf, R Learning Modules - OARC Stats - UCLA, and using the built-in R Help package. Using ChatGPT on sentence revised and grammar check.

Question on Problem 4.: The use of uniroot with upper bound and lower bound.

Problem 1

Generate a function in R that computes the desired sample size for the confidence interval of one sample population mean given the confidence level 100(1 - α), the population standard deviation σ, and the desired margin of error ϵ.

(a) Provide the R script of your R function with sufficient comments for the steps in the script.

[Sol]: To formulate a function in R for calculating the required sample size for estimating a population mean with a given confidence level, population standard deviation, and desired margin of error, we follow a three-step procedure: First, The significance level α is derived from the given confidence level (expressed as a decimal), where:

$$\alpha = 1 - ConfidenceLevel$$

. Second, I assume that our sample is from a N(μ, σ^2) population. The Z-score corresponds to the given confidence level in a standard normal distribution. The standard normal distribution, denoted as $Z \sim N(0,1)$, is symmetric around zero, and critical values for two-tailed confidence intervals are obtained using:$Z_{\alpha/2}$. Finally, I want the Margin of Error:

$$\epsilon = Z_{\alpha/2} \times (\sigma/\sqrt{n})$$

and compute the required sample size:

$$\sqrt{n} = Z_{\alpha/2} \times (\sigma/\epsilon)$$

We squared both sides:

$$n = (Z_{\alpha/2} \times \frac{\sigma}{\epsilon})^2$$

By the procedure mentioned above, we will get the function for sample size calculation.

```r
samplesize_func <- function(CL, sigma, epsilon){

  # Step 1: Compute alpha (significance level)
  # Confidence level is given as a decimal
  alpha <- 1 - CL

  # Step 2: Determine the critical value (Z-score)
  # The Z-score corresponds to the given confidence level in a standard normal distribution
  # qnorm(1 - alpha / 2) gives the critical value for a two-tailed confidence interval
  z <- qnorm(1 - alpha / 2, mean=0,sd=1)
  # Step 3: Compute the required sample size using the formula:
  # we want margin of error = Z * (standard deviation / sqrt(n))
  # n = (Z * standard deviation / margin of error)^2
  samplesize <- (z * sigma/ epsilon)^2
  samplesize <- ceiling(samplesize) # round up to meet requirement

  # Return the computed sample size
  return(samplesize)
}
```

(b) Using your R function, generate a table of values of the sample sizes for 100(1- α).from 0.80 to 0.99 in the increments of 0.01. For this part, fix $\epsilon/\sigma = 0.1$.

[Sol] I assume ($\sigma = 1$) for simplicity and generate a table of sample sizes for different confidence levels, ranging from 0.80 to 0.99 in increments of 0.01, while keeping the ratio $\epsilon/\sigma = 0.1$. The purpose of this table is to show how the required sample size changes as the confidence level increases. As the confidence level increases, the required sample size also increases. This is because a higher confidence level means we want to be more certain that our sample mean is close to the true population mean, which requires a larger sample. This table helps us to determine the appropriate sample size needed to achieve a desired confidence level while maintaining a specific margin of error relative to the population standard deviation.

```r
set.seed(2023) # Set seed for reproducibility

# Function to calculate sample size for a given confidence level,
# population standard deviation, and margin of error ratio (epsilon/sigma)
samplesize_func2 <- function(CL, sigma, epsilon = sigma * 0.1) {

  # Step 1: Compute alpha (significance level)
  alpha <- 1 - CL

  # Step 2: Determine the critical value (Z-score)
  z <- qnorm(1 - alpha / 2)

  # Step 3: Compute the required sample size
  samplesize <- (z * sigma / epsilon)^2
  samplesize <- ceiling(samplesize) # round up to the nearest integer

  # Return the computed sample size
  return(samplesize)
}

# Constants
sigma <- 1  # Define sigma as some positive number (Assume sigma = 1 for simplicity)
epsilon_sigma_ratio <- 0.1 # Ratio Fixed Value
epsilon <- sigma * epsilon_sigma_ratio # Define epsilon

# Generating a sequence of confidence levels from 0.80 to 0.99 in increments of 0.01
confidence_levels <- seq(0.80, 0.99, by = 0.01)

# First, an empty data frame
Sample_Sizes_DF <- data.frame(
  Confidence_Level = double(),
  Sample_Size = integer(),
  Epsilon_Sigma_Ratio = double()
)
```

```r
# Initialize row counter
RowCounter <- 0

# Loop to populate the data frame
for (CL in confidence_levels) {
  RowCounter <- RowCounter + 1
  sample_size <- samplesize_func2(CL, sigma, epsilon)

  # Populate the data frame
  Sample_Sizes_DF[RowCounter, 1] <- CL
  Sample_Sizes_DF[RowCounter, 2] <- sample_size
  Sample_Sizes_DF[RowCounter, 3] <- epsilon_sigma_ratio
}

# Print the results
print(Sample_Sizes_DF)
```

```
##    Confidence_Level Sample_Size Epsilon_Sigma_Ratio
## 1              0.80         165                 0.1
## 2              0.81         172                 0.1
## 3              0.82         180                 0.1
## 4              0.83         189                 0.1
## 5              0.84         198                 0.1
## 6              0.85         208                 0.1
## 7              0.86         218                 0.1
## 8              0.87         230                 0.1
## 9              0.88         242                 0.1
## 10             0.89         256                 0.1
## 11             0.90         271                 0.1
## 12             0.91         288                 0.1
## 13             0.92         307                 0.1
## 14             0.93         329                 0.1
## 15             0.94         354                 0.1
## 16             0.95         385                 0.1
## 17             0.96         422                 0.1
## 18             0.97         471                 0.1
## 19             0.98         542                 0.1
## 20             0.99         664                 0.1
```

(c) Similar to the previous part, using your R function, generate a table of values of the sample sizes for, $\epsilon/\sigma = 0.1$ to 0.35 in the increments of 0.01. For this part, fix the confidence level $100(1- \alpha) = 0.95$.

[sol] I assume ($\sigma = 1$) for simplicity and generate a table of sample sizes for different ratios of ϵ/σ, ranging from 0.1 to 0.35 in increments of 0.01, while keeping the confidence level fixed at 0.95. The purpose of this table is to show how the required sample size changes as the margin of error (relative to the population standard deviation) changes. As the ratio ϵ/σ increases, the required sample size decreases. This is because a larger margin of error means we are allowing more variability in our estimate, which requires a smaller sample size. This table helps us to determine the appropriate sample size needed to achieve a desired margin of error relative to the population standard deviation, while maintaining a specific confidence level.

```r
set.seed(2023) # Set seed for reproducibility

# Function to calculate sample size
samplesize_func <- function(CL, sigma, epsilon) {
  # Compute alpha (significance level)
  alpha <- 1 - CL

  # Determine the critical value (Z-score)
  z <- qnorm(1 - alpha / 2, mean=0,sd=1)

  # Compute the required sample size
  samplesize <- (z * sigma / epsilon)^2
  samplesize <- ceiling(samplesize) # round up to meet requirement

  # Return the computed sample size
  return(samplesize)
}

# First, an empty data frame
Sample_Sizes_DF <- data.frame(
  Epsilon_Sigma_Ratio = double(),
  Sample_Size = integer()
)

# Constants
sigma <- 1 # Assume sigma = 1 for simplicity
CL <- 0.95 # Fixed confidence level

# Generating a sequence of epsilon/sigma ratios from 0.1 to 0.35 in increments of 0.01
ratio_seq <- seq(0.1, 0.35, by = 0.01)

# Initialize row counter
RowCounter <- 0

# Loop to populate the data frame
for (epsilon_sigma_ratio in ratio_seq) {
  epsilon <- epsilon_sigma_ratio * sigma
  RowCounter <- RowCounter + 1
  sample_size <- samplesize_func(CL, sigma, epsilon)

  # Populate the data frame
  Sample_Sizes_DF[RowCounter, 1] <- epsilon_sigma_ratio
  Sample_Sizes_DF[RowCounter, 2] <- sample_size
}

# Print the results
print(Sample_Sizes_DF)
```

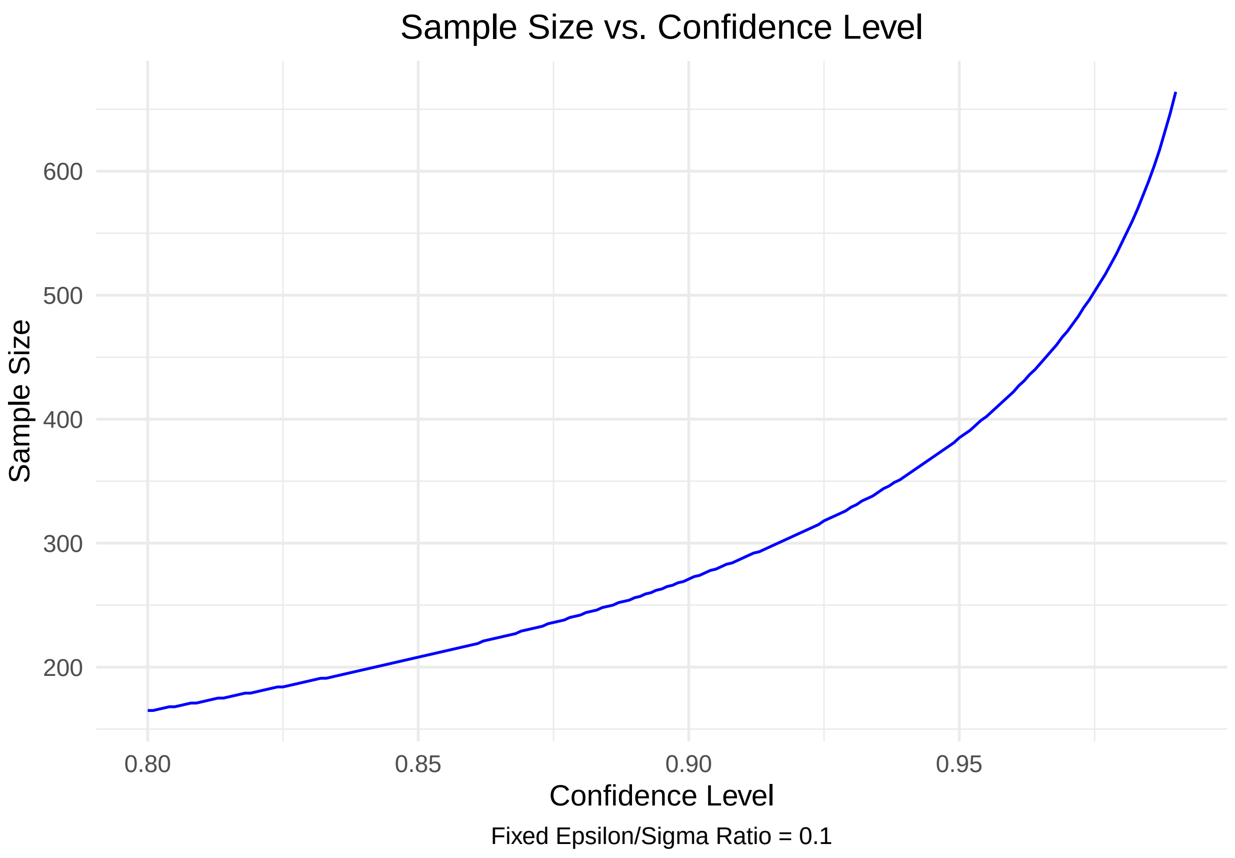
```
##    Epsilon_Sigma_Ratio Sample_Size
## 1                 0.10         385
## 2                 0.11         318
## 3                 0.12         267
## 4                 0.13         228
## 5                 0.14         196
## 6                 0.15         171
## 7                 0.16         151
## 8                 0.17         133
## 9                 0.18         119
## 10                0.19         107
## 11                    0.20           97
## 12                    0.21           88
## 13                    0.22           80
## 14                    0.23           73
## 15                    0.24           67
## 16                    0.25           62
## 17                    0.26           57
## 18                    0.27           53
## 19                    0.28           49
## 20                    0.29           46
## 21                    0.30           43
## 22                    0.31           40
## 23                    0.32           38
## 24                    0.33           36
## 25                    0.34           34
## 26                    0.35           32
```

(d) Change the increments in the two previous parts to 0.001. Then, instead of the corresponding tables, generate corresponding plots as illustrated in class (but cleaned up with legends, proper axes titles, and overall titles).

[sol] The first plot shows the relationship between the sample size required and the confidence level (ranging from 0.80 to 0.99) for a fixed epsilon/sigma ratio of 0.1. As the confidence level increases, the required sample size also increases. This is because a higher confidence level demands more precision, which in turn requires a larger sample size.

```r
set.seed(2023) # Set seed for reproducibility

# Function to calculate sample size for a given confidence level,
# population standard deviation, and margin of error ratio (epsilon/sigma)
samplesize_func2 <- function(CL, sigma, epsilon = sigma * 0.1) {

  # Step 1: Compute alpha (significance level)
  alpha <- 1 - CL

  # Step 2: Determine the critical value (Z-score)
  z <- qnorm(1 - alpha / 2)

  # Step 3: Compute the required sample size
  samplesize <- (z * sigma / epsilon)^2
  samplesize <- ceiling(samplesize) # round up to the nearest integer

  # Return the computed sample size
  return(samplesize)
}

# Constants
sigma <- 1  # Define sigma as some positive number (Assume sigma = 1 for simplicity)
epsilon_sigma_ratio <- 0.1 # Ratio Fixed Value
epsilon <- sigma * epsilon_sigma_ratio # Define epsilon

# Generating a sequence of confidence levels from 0.80 to 0.99 in increments of 0.001
confidence_levels <- seq(0.80, 0.99, by = 0.001)

# First, an empty data frame with the right number of rows
```

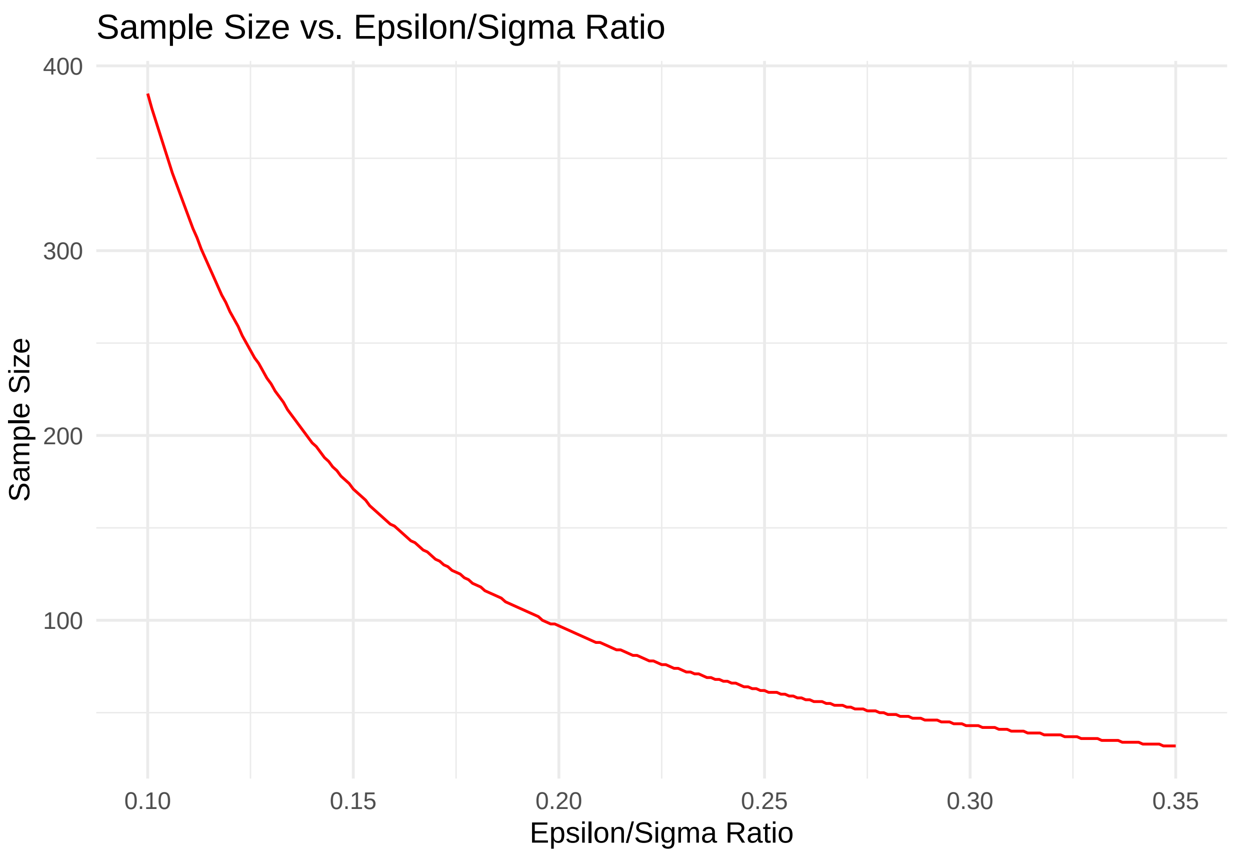
```r
Sample_Sizes_DF <- data.frame(
  Confidence_Level = double(),
  Sample_Size = integer(),
  Epsilon_Sigma_Ratio = double()
)

# Initialize row counter
RowCounter <- 0

# Loop to populate the data frame
for (CL in confidence_levels) {
  sample_size <- samplesize_func2(CL, sigma, epsilon)

  # Populate the data frame
  RowCounter <- RowCounter + 1
  Sample_Sizes_DF[RowCounter, "Confidence_Level"] <- CL
  Sample_Sizes_DF[RowCounter, "Sample_Size"] <- sample_size
  Sample_Sizes_DF[RowCounter, "Epsilon_Sigma_Ratio"] <- epsilon_sigma_ratio
}

# Plot the results
ggplot(Sample_Sizes_DF, aes(x = Confidence_Level, y = Sample_Size)) +
  geom_line(color = "blue") +
  labs(title = "Sample Size vs. Confidence Level",
       x = "Confidence Level",
       y = "Sample Size",
       caption = "Fixed Epsilon/Sigma Ratio = 0.1") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5),
        plot.caption = element_text(hjust = 0.5))
```



The second plot shows the relationship between the sample size required and the epsilon/sigma ratio (ranging from 0.1 to 0.35) for a fixed confidence level of 0.95. As the epsilon/sigma ratio increases, the required sample size decreases. This is because a larger margin of error (relative to the standard deviation) allows for a smaller sample size to achieve the same confidence level.

```
set.seed(2023) # Set seed for reproducibility

# Function to calculate sample size
samplesize_func <- function(CL, sigma, epsilon) {
  # Compute alpha (significance level)
  alpha <- 1 - CL

  # Determine the critical value (Z-score)
  z <- qnorm(1 - alpha / 2, mean=0,sd=1)

  # Compute the required sample size
  samplesize <- (z * sigma / epsilon)^2
  samplesize <- ceiling(samplesize) # round up to meet requirement

  # Return the computed sample size
  return(samplesize)
}

# First, an empty data frame
Sample_Sizes_DF <- data.frame(
  Epsilon_Sigma_Ratio = double(),
  Sample_Size = integer()
```

```r
)

# Constants
sigma <- 1 # Assume sigma = 1 for simplicity
CL <- 0.95 # Fixed confidence level

# Generating a sequence of epsilon/sigma ratios from 0.1 to 0.35 in increments of 0.001
ratio_seq <- seq(0.1, 0.35, by = 0.001)

# Initialize row counter
RowCounter <- 0

# Loop to populate the data frame
for (epsilon_sigma_ratio in ratio_seq) {
  epsilon <- epsilon_sigma_ratio * sigma
  RowCounter <- RowCounter + 1
  sample_size <- samplesize_func(CL, sigma, epsilon)

  # Populate the data frame
  Sample_Sizes_DF[RowCounter, 1] <- epsilon_sigma_ratio
  Sample_Sizes_DF[RowCounter, 2] <- sample_size
}

# Plotting the results
ggplot(Sample_Sizes_DF, aes(x = Epsilon_Sigma_Ratio, y = Sample_Size)) +
  geom_line(color = "red") +
  labs(title = "Sample Size vs. Epsilon/Sigma Ratio",
       x = "Epsilon/Sigma Ratio",
       y = "Sample Size") +
  theme_minimal()
```



Problem 2

Similar to the previous question, generate a function in R that computes the desired sample size for the confidence interval of one sample population proportion given the confidence level 100(1 - α), a prior estimate p* of the population proportion, and the desired margin of error ϵ.

(a) Provide the R script of your R function with sufficient comments for the steps in the script.

[sol] I coupute a three-step fuction procedure: First, The significance level α is derived from the given confidence level (expressed as a decimal), where:

$$\alpha = 1 - ConfidenceLevel$$

. Second, I assume that our sample is from a N(p, $\sqrt{\frac{p(1-p)}{n}}$) population. The Z-score corresponds to the given confidence level in a standard normal distribution. The standard normal distribution, denoted as $Z \sim N(0, 1)$, is symmetric around zero, and critical values for two-tailed confidence intervals are obtained using:$Z_{\alpha/2}$. Finally, I want the Margin of Error:

$$\epsilon = Z_{\alpha/2} \times \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$$

and compute the required sample size:

$$\sqrt{n} = \frac{Z_{\alpha/2}}{\epsilon} \times \sqrt{\hat{p}(1-\hat{p})}$$

We squared both sides:

$$n = (\frac{Z_{\alpha/2}}{\epsilon})^2 \times \hat{p}(1-\hat{p})$$

By the procedure mentioned above, we will get the function for sample size calculation.

```r
# Function to calculate sample size for the confidence interval of
# one sample population proportion
# Inputs:
#   CL: Confidence Level (e.g., 0.95 for 95% confidence level)
#   p_hat: Prior estimate of the population proportion (e.g., 0.5 if no
#   prior estimate is available)
#   epsilon: Desired margin of error (e.g., 0.05 for ±5% margin of error)
# Output:
#   The required sample size
samplesize_prop <- function(CL, p_hat, epsilon) {

  # Step 1: Compute alpha (significance level)
  alpha <- 1 - CL

  # Step 2: Determine the critical value (Z-score)
  z <- qnorm(1 - alpha / 2, mean = 0, sd =1)

  # Step 3: Compute the required sample size
  # Formula: n = (z^2 * p_hat * (1 - p_hat)) / epsilon^2
  samplesize <- (z^2 * p_hat * (1 - p_hat)) / epsilon^2
  samplesize <- ceiling(samplesize) # round up to the nearest integer

  # Return the computed sample size
  return(samplesize)
}
```

(b) Using your R function, generate a table of values of the sample sizes for 100(1- α).from 0.80 to 0.99 in the increments of 0.01. For this part, fix p* = 0.5 and $\epsilon = 0.01$.

[sol] I designed the sample size function for a given confidence level, prior estimate of the population proportion, and desired margin of error. Then I generate a table of sample sizes for different values of confidence levels, ranging from 0.80 to 0.99 in increments of 0.01, with a fixed value epsilon of 0.95 and a prior estimate of the population proportion of 0.5. From the table, I found out that as the confidence level increases, the required sample size also increases. This is because a higher confidence level requires more precision, which in turn requires a larger sample size.

```r
set.seed(2023) # Set seed for reproducibility

sample_size_prop <- function(CL, p_hat, epsilon) {

  # Step 1: Compute alpha (significance level)
  alpha <- 1 - CL

  # Step 2: Determine the critical value (Z-score)
  z <- qnorm(1 - alpha / 2, mean = 0, sd =1)

  # Step 3: Compute the required sample size
  # Formula: n = (z^2 * p_hat * (1 - p_hat)) / epsilon^2
  sample_size <- (z^2 * p_hat * (1 - p_hat)) / epsilon^2
  sample_size <- ceiling(sample_size) # round up to the nearest integer

  # Return the computed sample size
  return(sample_size)
}
```

```r
# Constants
p_hat <- 0.5  # Prior estimate of the population proportion
epsilon <- 0.01  # Desired margin of error

# Generating a sequence of confidence levels from 0.80 to 0.99 in increments of 0.01
confidence_levels <- seq(0.80, 0.99, by = 0.01)

# First, an empty data frame with the right number of rows
Sample_Sizes_DF <- data.frame(
  Confidence_Level = double(),
  Sample_Size = integer()
)

# Initialize row counter
RowCounter <- 0

# Loop to populate the data frame
for (CL in confidence_levels) {
  sample_size <- sample_size_prop(CL, p_hat, epsilon)

  # Populate the data frame
  RowCounter <- RowCounter + 1
  Sample_Sizes_DF[RowCounter, "Confidence_Level"] <- CL
  Sample_Sizes_DF[RowCounter, "Sample_Size"] <- sample_size
}

# Print the results
print(Sample_Sizes_DF)
```

```
##    Confidence_Level Sample_Size
## 1              0.80        4106
## 2              0.81        4295
## 3              0.82        4495
## 4              0.83        4708
## 5              0.84        4936
## 6              0.85        5181
## 7              0.86        5445
## 8              0.87        5732
## 9              0.88        6044
## 10             0.89        6386
## 11             0.90        6764
## 12             0.91        7186
## 13             0.92        7663
## 14             0.93        8208
## 15             0.94        8844
## 16             0.95        9604
## 17             0.96       10545
## 18             0.97       11774
## 19             0.98       13530
## 20             0.99       16588
```

(c) Similar to the previous part, using your R function, generate a table of values of the sample sizes for, ϵ = 0.01 to 0.10 in the increments of 0.005. For this part, fix the confidence level 100(1- α) = 0.95 and p* = 0.5.

[sol] I designed the sample size function for a given confidence level, prior estimate of the population proportion, and desired margin of error. Then I generate a table of sample sizes for different values of epsilon, ranging from 0.01 to 0.10 in increments of 0.005, with a fixed confidence level of 0.95 and a prior estimate of the population proportion of 0.5. From the table, I found out that as the margin of error (ϵ) increases, the required sample size decreases. This is because a larger margin of error allows for more variability in the estimate, requiring fewer samples to achieve the desired confidence level.

```r
set.seed(2023) # Set seed for reproducibility

sample_size_prop <- function(CL, p_hat, epsilon) {

  # Step 1: Compute alpha (significance level)
  alpha <- 1 - CL

  # Step 2: Determine the critical value (Z-score)
  z <- qnorm(1 - alpha / 2, mean = 0, sd =1)

  # Step 3: Compute the required sample size
  # Formula: n = (z^2 * p_hat * (1 - p_hat)) / epsilon^2
  sample_size <- (z^2 * p_hat * (1 - p_hat)) / epsilon^2
  sample_size <- ceiling(sample_size) # round up to the nearest integer

  # Return the computed sample size
  return(sample_size)
}

# Constants
CL <- 0.95  # Fixed confidence level
p_hat <- 0.5  # Prior estimate of the population proportion

# Generating a sequence of epsilon values from 0.01 to 0.10 in increments of 0.005
epsilon_values <- seq(0.01, 0.10, by = 0.005)

# First, an empty data frame with the right number of rows
Sample_Sizes_DF <- data.frame(
  Epsilon = double(),
  Sample_Size = integer()
)

# Initialize row counter
RowCounter <- 0

# Loop to populate the data frame
for (epsilon in epsilon_values) {
  sample_size <- sample_size_prop(CL, p_hat, epsilon)

  # Populate the data frame
  RowCounter <- RowCounter + 1
  Sample_Sizes_DF[RowCounter, "Epsilon"] <- epsilon
  Sample_Sizes_DF[RowCounter, "Sample_Size"] <- sample_size
}

# Print the results
print(Sample_Sizes_DF)
```

```
##    Epsilon Sample_Size
## 1    0.010        9604
## 2    0.015        4269
## 3    0.020        2401
## 4    0.025        1537
## 5    0.030        1068
## 6    0.035         784
## 7    0.040         601
## 8    0.045         475
## 9    0.050         385
## 10   0.055         318
## 11   0.060         267
## 12   0.065         228
## 13   0.070         196
## 14   0.075         171
## 15   0.080         151
## 16   0.085         133
## 17   0.090         119
## 18   0.095         107
## 19   0.100          97
```

(d) Similar to the previous two parts, using your R function, generate a table of values of the sample sizes for p* from 0.05 to 0.95 in the increments of 0.05. For this part, fix the confidence level 100(1- α) = 0.95 and $\epsilon = 0.01$.

[sol] I calculated the required sample size to estimate a population proportion with a 95% confidence level and a margin of error ϵ=0.01, for different values of p* (the assumed population proportion). From the table, I found out that the highest required sample size occurs at p* = 0.50 with 9,604 observations, because variance is maximized at p* =0.50 in a binomial distribution. As p* moves away from 0.50 towards 0.05 or 0.95, the sample size decreases symmetrically.

In Conclusion, The required sample size for estimating a population proportion varies depending on the assumed proportion p* . The maximum sample size is needed when p* = 0.50, as this leads to the highest variability in the estimate. As p* moves closer to 0 or 1, the required sample size decreases due to lower variability.

```r
set.seed(2023) # Set seed for reproducibility

sample_size_prop <- function(CL, p_hat, epsilon) {

  # Step 1: Compute alpha (significance level)
  alpha <- 1 - CL

  # Step 2: Determine the critical value (Z-score)
  z <- qnorm(1 - alpha / 2, mean = 0, sd =1)

  # Step 3: Compute the required sample size
  # Formula: n = (z^2 * p_hat * (1 - p_hat)) / epsilon^2
  sample_size <- (z^2 * p_hat * (1 - p_hat)) / epsilon^2
  sample_size <- ceiling(sample_size) # round up to the nearest integer

  # Return the computed sample size
  return(sample_size)
}

# Constants
```

```
CL <- 0.95  # Fixed confidence level
epsilon <- 0.01  # Desired margin of error

# Generating a sequence of p_hat values from 0.05 to 0.95 in increments of 0.05
p_hat_values <- seq(0.05, 0.95, by = 0.05)

# First, an empty data frame with the right number of rows
Sample_Sizes_DF <- data.frame(
  p_hat = double(),
  Sample_Size = integer()
)

# Initialize row counter
RowCounter <- 0

# Loop to populate the data frame
for (p_hat in p_hat_values) {
  sample_size <- sample_size_prop(CL, p_hat, epsilon)

  # Populate the data frame
  RowCounter <- RowCounter + 1
  Sample_Sizes_DF[RowCounter, "p_hat"] <- p_hat
  Sample_Sizes_DF[RowCounter, "Sample_Size"] <- sample_size
}

# Print the results
print(Sample_Sizes_DF)
```

```
##    p_hat Sample_Size
## 1   0.05        1825
## 2   0.10        3458
## 3   0.15        4898
## 4   0.20        6147
## 5   0.25        7203
## 6   0.30        8068
## 7   0.35        8740
## 8   0.40        9220
## 9   0.45        9508
## 10  0.50        9604
## 11  0.55        9508
## 12  0.60        9220
## 13  0.65        8740
## 14  0.70        8068
## 15  0.75        7203
## 16  0.80        6147
## 17  0.85        4898
## 18  0.90        3458
## 19  0.95        1825
```

(e) Change the increments in the three previous parts to 0.001. Then, instead of the corresponding tables, generate corresponding plots as illustrated in class.

Summary of Graphs

1. Sample Size vs Confidence Level

As the confidence level increases, the required sample size grows exponentially. This occurs because higher confidence levels demand more precise estimates to ensure the population parameter is captured within the interval, making the relationship nonlinear and steep for higher confidence levels.

2. Sample Size vs Margin of Error

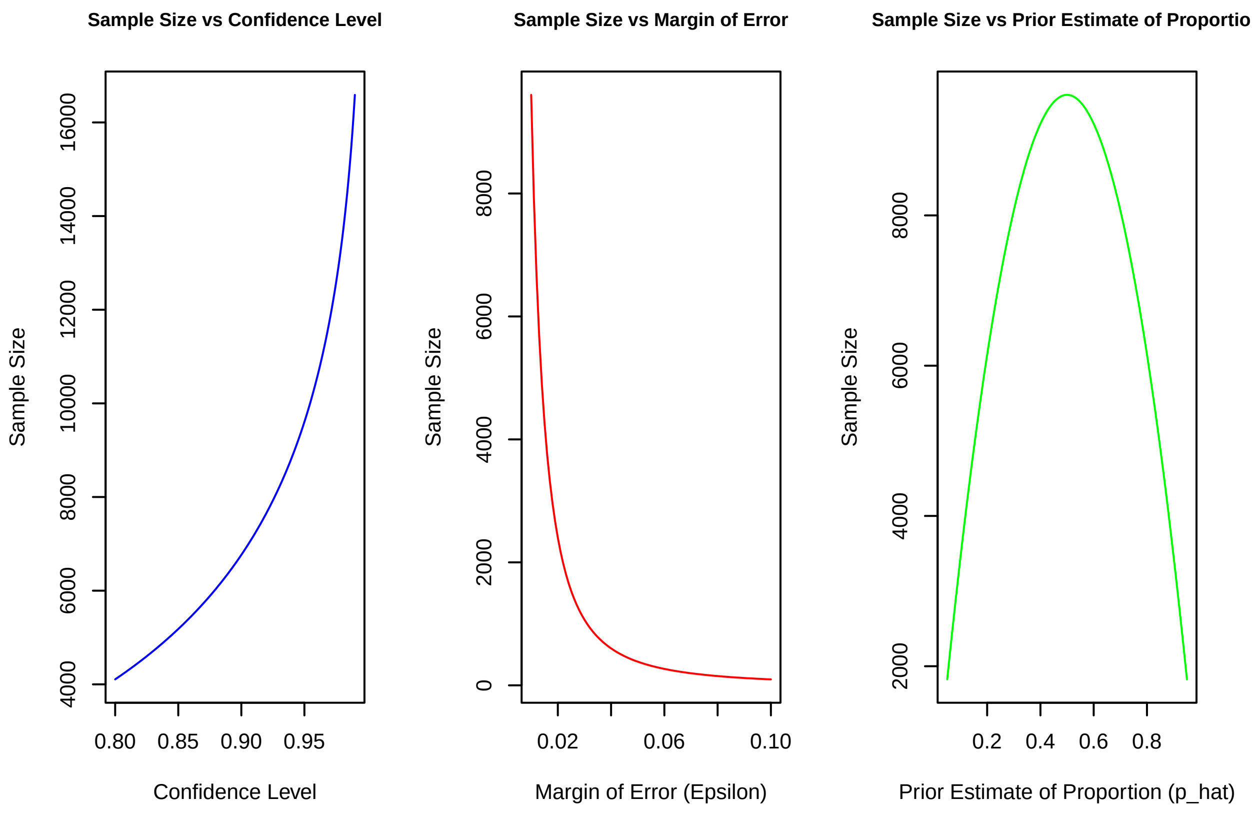
The sample size required increases dramatically as the margin of error decreases. This reflects the need for more data to achieve higher precision, with a steep rise in sample size for very small margins of error, indicating a trade-off between accuracy and feasibility.

3. Sample Size vs Prior Estimate of Proportion

The required sample size is maximized when the prior estimate of the proportion ($\hat{p}$) is 0.5, as variability is highest at this value. As $\hat{p}$ approaches the extremes of 0 or 1, the variability decreases, resulting in a smaller required sample size.

```r
samplesize_prop <- function(CL, p_hat, epsilon) {

  # Step 1: Compute alpha (significance level)
  alpha <- 1 - CL

  # Step 2: Determine the critical value (Z-score)
  z <- qnorm(1 - alpha / 2, mean = 0, sd =1)

  # Step 3: Compute the required sample size
  # Formula: n = (z^2 * p_hat * (1 - p_hat)) / epsilon^2
  samplesize <- (z^2 * p_hat * (1 - p_hat)) / epsilon^2
  samplesize <- ceiling(samplesize) # round up to the nearest integer

  # Return the computed sample size
  return(samplesize)
}

# (b) Generate a table of values of the sample sizes for confidence levels
# from 0.80 to 0.99 in increments of 0.001
p_hat <- 0.5  # Prior estimate of the population proportion
epsilon <- 0.01  # Desired margin of error

confidence_levels <- seq(0.80, 0.99, by = 0.001)

Sample_Sizes_DF_CL <- data.frame(
  Confidence_Level = double(),
  Sample_Size = integer()
)

# Initialize row counter
RowCounter <- 0

for (CL in confidence_levels) {
  sample_size <- samplesize_prop(CL, p_hat, epsilon)
  # Populate the data frame
  RowCounter <- RowCounter + 1
  Sample_Sizes_DF_CL[RowCounter, "Confidence_Level"] <- CL
  Sample_Sizes_DF_CL[RowCounter, "Sample_Size"] <- sample_size
}

# (c) Generate a table of values of the sample sizes for epsilon
# from 0.01 to 0.10 in increments of 0.001
CL <- 0.95  # Fixed confidence level

epsilon_values <- seq(0.01, 0.10, by = 0.001)

Sample_Sizes_DF_Epsilon <- data.frame(
  Epsilon = double(),
  Sample_Size = integer()
)

# Initialize row counter
RowCounter <- 0

for (epsilon in epsilon_values) {
  sample_size <- sample_size_prop(CL, p_hat, epsilon)

  # Populate the data frame
  RowCounter <- RowCounter + 1
  Sample_Sizes_DF_Epsilon[RowCounter, "Epsilon"] <- epsilon
  Sample_Sizes_DF_Epsilon[RowCounter, "Sample_Size"] <- sample_size
}

# (d) Generate a table of values of the sample sizes for p_hat
# from 0.05 to 0.95 in increments of 0.001
epsilon <- 0.01  # Desired margin of error

p_hat_values <- seq(0.05, 0.95, by = 0.001)

Sample_Sizes_DF_p_hat <- data.frame(
  p_hat = double(),
  Sample_Size = integer()
)

# Initialize row counter
RowCounter <- 0

# Loop to populate the data frame
for (p_hat in p_hat_values) {
  sample_size <- sample_size_prop(CL, p_hat, epsilon)

  # Populate the data frame
  RowCounter <- RowCounter + 1
  Sample_Sizes_DF_p_hat[RowCounter, "p_hat"] <- p_hat
  Sample_Sizes_DF_p_hat[RowCounter, "Sample_Size"] <- sample_size
}
```

```r
# Set up a 3x1 plotting layout
par(mfrow = c(1, 3))

# Plot for part (b)
plot(Sample_Sizes_DF_CL$Confidence_Level, Sample_Sizes_DF_CL$Sample_Size, type = "l",
     xlab = "Confidence Level", ylab = "Sample Size",
     main = "Sample Size vs Confidence Level",
     col = "blue", cex.main = 0.9)

# Plot for part (c)
plot(Sample_Sizes_DF_Epsilon$Epsilon, Sample_Sizes_DF_Epsilon$Sample_Size, type = "l",
     xlab = "Margin of Error (Epsilon)", ylab = "Sample Size",
     main = "Sample Size vs Margin of Error",
     col = "red", cex.main = 0.9)

# Plot for part (d)
plot(Sample_Sizes_DF_p_hat$p_hat, Sample_Sizes_DF_p_hat$Sample_Size, type = "l",
     xlab = "Prior Estimate of Proportion (p_hat)", ylab = "Sample Size",
     main = "Sample Size vs Prior Estimate of Proportion",
     col = "green", cex.main = 0.9)
```



```r
# Reset plotting layout
par(mfrow = c(1, 1))
```

Problem 3

Generate a function in R that computes the desired sample size to perform a test of hypothesis involving the mean of a continuous outcome variable in a single population. The function inputs should be α (the size/level of the test - i.e., the maximum P(Type I Error)), β (the maximum P(Type II Error)), and the effect size. Using tables and graphs similar to the previous problems but using appropriate ranges of values of the inputs at your discretion, illustrate the effects of level, power, and the effect size on the needed sample sizes.

[sol] In page 19 of the article by Lisa Sullivan PhD, for sample size for one sample, continuous outcome is to perform a test of hypothesis comparing the mean of a continuous outcome variable in a single population to a known mean: $H_0 : \mu = \mu_0$ vs. $H_1 : \mu \neq \mu_0$ where μ_0 is the known mean. The formula for determining sample size to ensure that the test has a specified power is given below:

$$n = (\frac{Z_{1-\alpha/2} + Z_{1-\beta}}{ES})^2$$

where α is the selected level of significance, $1 - \beta$ is the selected power, ES is the effect size, defined as follows:

$$ES = \frac{\mid \mu_1 - \mu_0 \mid}{\sigma}$$

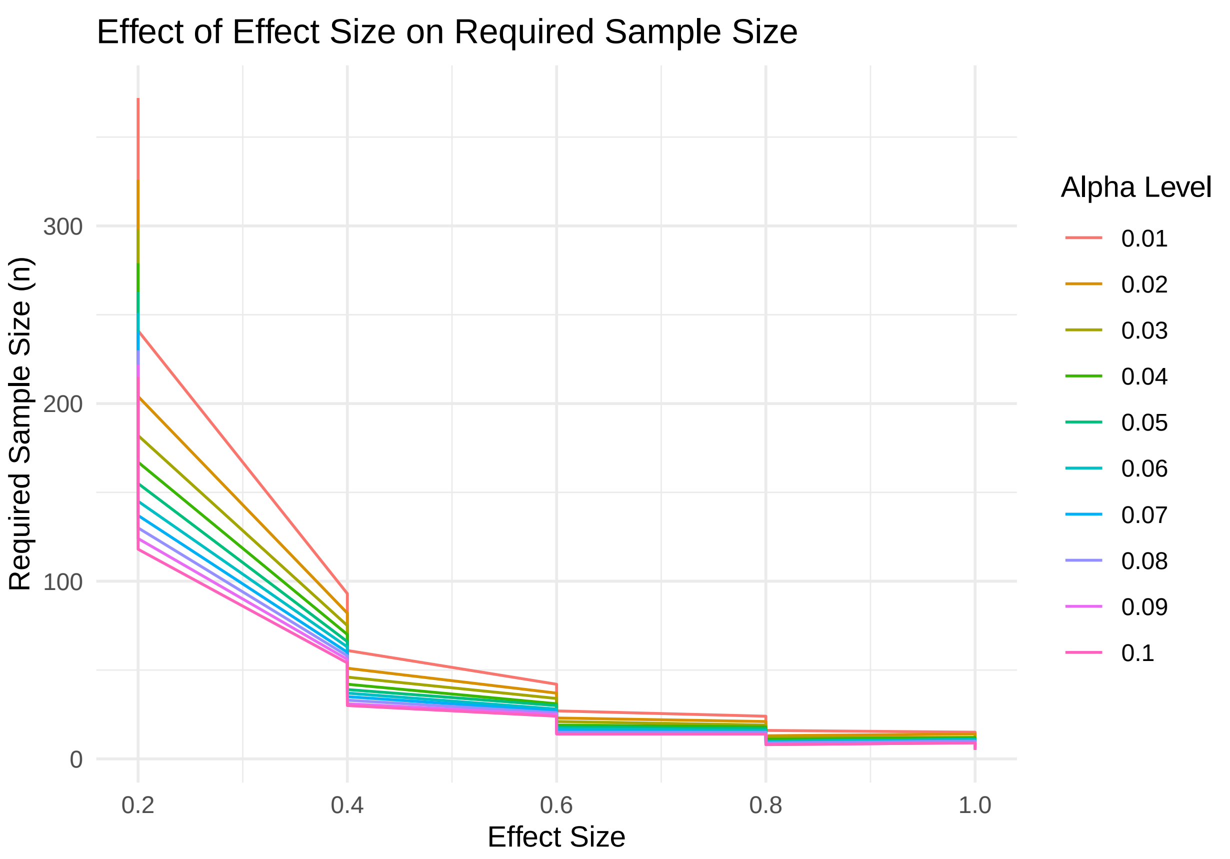
```
# Function to compute sample size
sample_size <- function(alpha, beta, effect_size) {
  z_alpha <- qnorm(1 - alpha / 2)  # Two-tailed test
  z_beta <- qnorm(1 - beta)        # Power calculation
  n <- ((z_alpha + z_beta) / effect_size) ^ 2
  n <- ceiling(n) # Ensure n is an integer
  return(n)
}

# Example usage
sample_size(alpha = 0.05, beta = 0.2, effect_size = 0.5)
```

```
## [1] 32
```

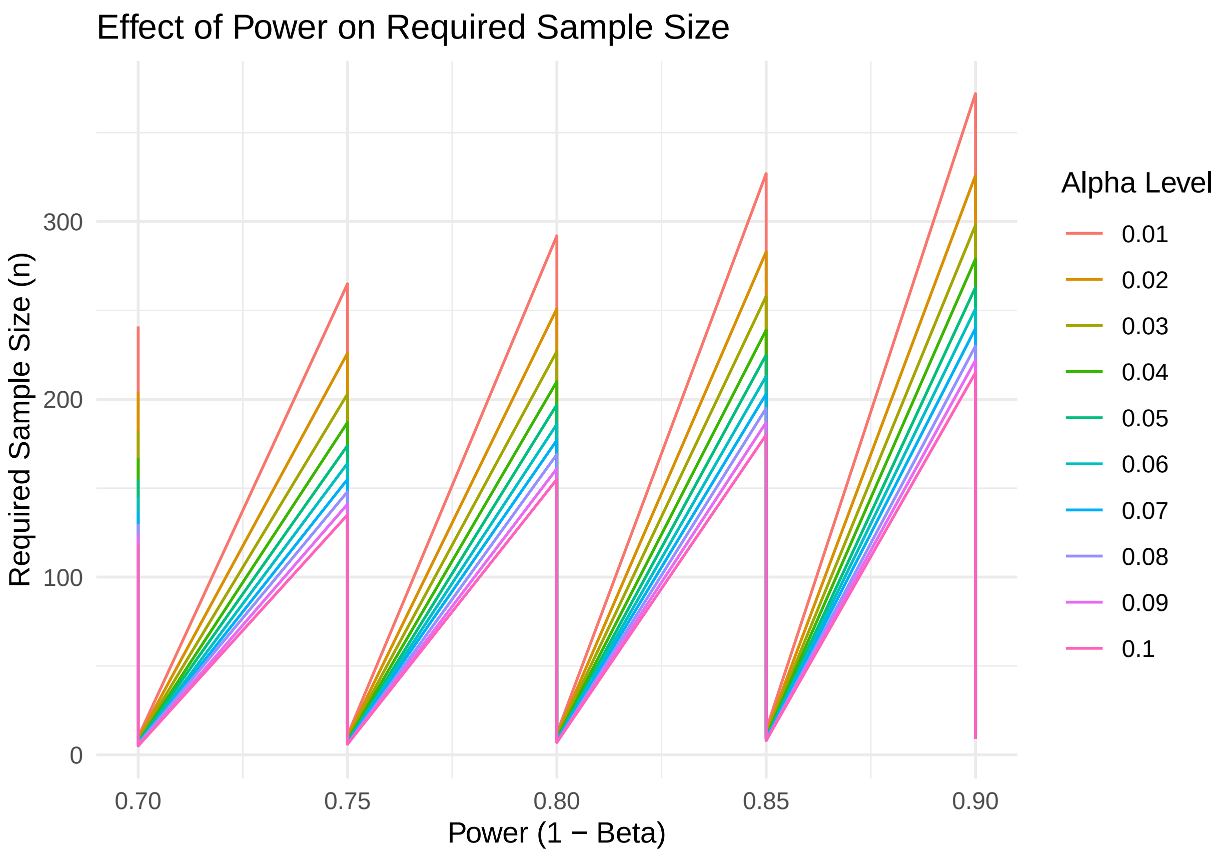
```
# Define sequences for alpha, beta, and effect size values
alpha_values <- seq(0.01, 0.1, by = 0.01)  # Alpha from 0.01 to 0.1
beta_values <- seq(0.1, 0.3, by = 0.05)    # Beta from 0.1 to 0.3
effect_sizes <- seq(0.2, 1.0, by = 0.2)    # Effect sizes from 0.2 to 1.0

# Initialize storage for results
sample_size_results <- data.frame(alpha = double(),
                                  beta = double(),
                                  effect_size = double(),
                                  n = integer())

# Initialize row counter
RowCounter <- 0

# Compute sample sizes using nested loops
for (alpha in alpha_values) {
  for (beta in beta_values) {
    for (effect_size in effect_sizes) {
      n <- sample_size(alpha, beta, effect_size)  # Compute required sample size

      # Populate the data frame
      RowCounter <- RowCounter + 1
      sample_size_results[RowCounter, "alpha"] <- alpha
      sample_size_results[RowCounter, "beta"] <- beta
      sample_size_results[RowCounter, "effect_size"] <- effect_size
      sample_size_results[RowCounter, "n"] <- n
    }
  }
}

# Print the computed sample sizes
print(sample_size_results)
```

```
##      alpha beta effect_size   n
## 1     0.01 0.10         0.2 372
## 2     0.01 0.10         0.4  93
## 3     0.01 0.10         0.6  42
## 4     0.01 0.10         0.8  24
## 5     0.01 0.10         1.0  15
## 6     0.01 0.15         0.2 327
## 7     0.01 0.15         0.4  82
## 8     0.01 0.15         0.6  37
## 9     0.01 0.15         0.8  21
## 10    0.01 0.15         1.0  14
## 11    0.01 0.20         0.2 292
## 12    0.01 0.20         0.4  73
## 13    0.01 0.20         0.6  33
## 14    0.01 0.20         0.8  19
## 15    0.01 0.20         1.0  12
## 16    0.01 0.25         0.2 265
## 17    0.01 0.25         0.4  67
## 18    0.01 0.25         0.6  30
## 19    0.01 0.25         0.8  17
## 20    0.01 0.25         1.0  11
## 21    0.01 0.30         0.2 241
## 22    0.01 0.30         0.4  61
## 23    0.01 0.30         0.6  27
## 24    0.01 0.30         0.8  16
## 25    0.01 0.30         1.0  10
## 26    0.02 0.10         0.2 326
## 27    0.02 0.10         0.4  82
## 28    0.02 0.10         0.6  37
## 29    0.02 0.10         0.8  21
## 30    0.02 0.10         1.0  14
## 31    0.02 0.15         0.2 283
## 32    0.02 0.15         0.4  71
## 33    0.02 0.15         0.6  32
## 34    0.02 0.15         0.8  18
## 35    0.02 0.15         1.0  12
## 36    0.02 0.20         0.2 251
## 37    0.02 0.20         0.4  63
## 38    0.02 0.20         0.6  28
## 39    0.02 0.20         0.8  16
## 40    0.02 0.20         1.0  11
## 41    0.02 0.25         0.2 226
## 42    0.02 0.25         0.4  57
## 43    0.02 0.25         0.6  26
## 44    0.02 0.25         0.8  15
## 45    0.02 0.25         1.0  10
## 46    0.02 0.30         0.2 204
## 47   0.02 0.30          0.4  51
## 48   0.02 0.30          0.6  23
## 49   0.02 0.30          0.8  13
## 50   0.02 0.30          1.0   9
## 51   0.03 0.10          0.2 298
## 52   0.03 0.10          0.4  75
## 53   0.03 0.10          0.6  34
## 54   0.03 0.10          0.8  19
## 55   0.03 0.10          1.0  12
## 56   0.03 0.15          0.2 258
## 57   0.03 0.15          0.4  65
## 58   0.03 0.15          0.6  29
## 59   0.03 0.15          0.8  17
## 60   0.03 0.15          1.0  11
## 61   0.03 0.20          0.2 227
## 62   0.03 0.20          0.4  57
## 63   0.03 0.20          0.6  26
## 64   0.03 0.20          0.8  15
## 65   0.03 0.20          1.0  10
## 66   0.03 0.25          0.2 203
## 67   0.03 0.25          0.4  51
## 68   0.03 0.25          0.6  23
## 69   0.03 0.25          0.8  13
## 70   0.03 0.25          1.0   9
## 71   0.03 0.30          0.2 182
## 72   0.03 0.30          0.4  46
## 73   0.03 0.30          0.6  21
## 74   0.03 0.30          0.8  12
## 75   0.03 0.30          1.0   8
## 76   0.04 0.10          0.2 279
## 77   0.04 0.10          0.4  70
## 78   0.04 0.10          0.6  31
## 79   0.04 0.10          0.8  18
## 80   0.04 0.10          1.0  12
## 81   0.04 0.15          0.2 239
## 82   0.04 0.15          0.4  60
## 83   0.04 0.15          0.6  27
## 84   0.04 0.15          0.8  15
## 85   0.04 0.15          1.0  10
## 86   0.04 0.20          0.2 210
## 87   0.04 0.20          0.4  53
## 88   0.04 0.20          0.6  24
## 89   0.04 0.20          0.8  14
## 90   0.04 0.20          1.0   9
## 91   0.04 0.25          0.2 187
## 92   0.04 0.25          0.4  47
## 93   0.04 0.25          0.6  21
## 94   0.04 0.25          0.8  12
## 95   0.04 0.25          1.0   8
## 96   0.04 0.30          0.2 167
## 97   0.04 0.30          0.4  42
## 98   0.04 0.30          0.6  19
## 99   0.04 0.30          0.8  11
## 100  0.04 0.30          1.0   7
## 101  0.05 0.10         0.2 263
## 102  0.05 0.10         0.4  66
## 103  0.05 0.10         0.6  30
## 104  0.05 0.10         0.8  17
## 105  0.05 0.10         1.0  11
## 106  0.05 0.15         0.2 225
## 107  0.05 0.15         0.4  57
## 108  0.05 0.15         0.6  25
## 109  0.05 0.15         0.8  15
## 110  0.05 0.15         1.0   9
## 111  0.05 0.20         0.2 197
## 112  0.05 0.20         0.4  50
## 113  0.05 0.20         0.6  22
## 114  0.05 0.20         0.8  13
## 115  0.05 0.20         1.0   8
## 116  0.05 0.25         0.2 174
## 117  0.05 0.25         0.4  44
## 118  0.05 0.25         0.6  20
## 119  0.05 0.25         0.8  11
## 120  0.05 0.25         1.0   7
## 121  0.05 0.30         0.2 155
## 122  0.05 0.30         0.4  39
## 123  0.05 0.30         0.6  18
## 124  0.05 0.30         0.8  10
## 125  0.05 0.30         1.0   7
## 126  0.06 0.10         0.2 251
## 127  0.06 0.10         0.4  63
## 128  0.06 0.10         0.6  28
## 129  0.06 0.10         0.8  16
## 130  0.06 0.10         1.0  11
## 131  0.06 0.15         0.2 213
## 132  0.06 0.15         0.4  54
## 133  0.06 0.15         0.6  24
## 134  0.06 0.15         0.8  14
## 135  0.06 0.15         1.0   9
## 136  0.06 0.20         0.2 186
## 137  0.06 0.20         0.4  47
## 138  0.06 0.20         0.6  21
## 139  0.06 0.20         0.8  12
## 140  0.06 0.20         1.0   8
## 141  0.06 0.25         0.2 164
## 142  0.06 0.25         0.4  41
## 143  0.06 0.25         0.6  19
## 144  0.06 0.25         0.8  11
## 145  0.06 0.25         1.0   7
## 146  0.06 0.30         0.2 145
## 147  0.06 0.30         0.4  37
## 148  0.06 0.30         0.6  17
## 149  0.06 0.30         0.8  10
## 150  0.06 0.30         1.0   6
## 151  0.07 0.10         0.2 240
## 152  0.07 0.10         0.4  60
## 153  0.07 0.10         0.6  27
## 154  0.07 0.10         0.8  15
## 155  0.07 0.10          1.0  10
## 156  0.07 0.15          0.2 203
## 157  0.07 0.15          0.4  51
## 158  0.07 0.15          0.6  23
## 159  0.07 0.15          0.8  13
## 160  0.07 0.15          1.0   9
## 161  0.07 0.20          0.2 177
## 162  0.07 0.20          0.4  45
## 163  0.07 0.20          0.6  20
## 164  0.07 0.20          0.8  12
## 165  0.07 0.20          1.0   8
## 166  0.07 0.25          0.2 155
## 167  0.07 0.25          0.4  39
## 168  0.07 0.25          0.6  18
## 169  0.07 0.25          0.8  10
## 170  0.07 0.25          1.0   7
## 171  0.07 0.30          0.2 137
## 172  0.07 0.30          0.4  35
## 173  0.07 0.30          0.6  16
## 174  0.07 0.30          0.8   9
## 175  0.07 0.30          1.0   6
## 176  0.08 0.10          0.2 230
## 177  0.08 0.10          0.4  58
## 178  0.08 0.10          0.6  26
## 179  0.08 0.10          0.8  15
## 180  0.08 0.10          1.0  10
## 181  0.08 0.15          0.2 195
## 182  0.08 0.15          0.4  49
## 183  0.08 0.15          0.6  22
## 184  0.08 0.15          0.8  13
## 185  0.08 0.15          1.0   8
## 186  0.08 0.20          0.2 169
## 187  0.08 0.20          0.4  43
## 188  0.08 0.20          0.6  19
## 189  0.08 0.20          0.8  11
## 190  0.08 0.20          1.0   7
## 191  0.08 0.25          0.2 148
## 192  0.08 0.25          0.4  37
## 193  0.08 0.25          0.6  17
## 194  0.08 0.25          0.8  10
## 195  0.08 0.25          1.0   6
## 196  0.08 0.30          0.2 130
## 197  0.08 0.30          0.4  33
## 198  0.08 0.30          0.6  15
## 199  0.08 0.30          0.8   9
## 200  0.08 0.30          1.0   6
## 201  0.09 0.10          0.2 222
## 202  0.09 0.10          0.4  56
## 203  0.09 0.10          0.6  25
## 204  0.09 0.10          0.8  14
## 205  0.09 0.10          1.0   9
## 206  0.09 0.15          0.2 187
## 207  0.09 0.15          0.4  47
## 208  0.09 0.15          0.6  21
## 209  0.09 0.15         0.8  12
## 210  0.09 0.15         1.0   8
## 211  0.09 0.20         0.2 161
## 212  0.09 0.20         0.4  41
## 213  0.09 0.20         0.6  18
## 214  0.09 0.20         0.8  11
## 215  0.09 0.20         1.0   7
## 216  0.09 0.25         0.2 141
## 217  0.09 0.25         0.4  36
## 218  0.09 0.25         0.6  16
## 219  0.09 0.25         0.8   9
## 220  0.09 0.25         1.0   6
## 221  0.09 0.30         0.2 124
## 222  0.09 0.30         0.4  31
## 223  0.09 0.30         0.6  14
## 224  0.09 0.30         0.8   8
## 225  0.09 0.30         1.0   5
## 226  0.10 0.10         0.2 215
## 227  0.10 0.10         0.4  54
## 228  0.10 0.10         0.6  24
## 229  0.10 0.10         0.8  14
## 230  0.10 0.10         1.0   9
## 231  0.10 0.15         0.2 180
## 232  0.10 0.15         0.4  45
## 233  0.10 0.15         0.6  20
## 234  0.10 0.15         0.8  12
## 235  0.10 0.15         1.0   8
## 236  0.10 0.20         0.2 155
## 237  0.10 0.20         0.4  39
## 238  0.10 0.20         0.6  18
## 239  0.10 0.20         0.8  10
## 240  0.10 0.20         1.0   7
## 241  0.10 0.25         0.2 135
## 242  0.10 0.25         0.4  34
## 243  0.10 0.25         0.6  15
## 244  0.10 0.25         0.8   9
## 245  0.10 0.25         1.0   6
## 246  0.10 0.30         0.2 118
## 247  0.10 0.30         0.4  30
## 248  0.10 0.30         0.6  14
## 249  0.10 0.30         0.8   8
## 250  0.10 0.30         1.0   5
```

```r
# Plot: Effect of Effect Size on Required Sample Size for different alpha levels
ggplot(sample_size_results, aes(x = effect_size, y = n, color = as.factor(alpha))) +
  geom_line() +
  labs(title = "Effect of Effect Size on Required Sample Size",
       x = "Effect Size",
       y = "Required Sample Size (n)",
       color = "Alpha Level") +
  theme_minimal()
```



```r
# Plot: Effect of Power (1 - Beta) on Required Sample Size for different alpha levels
ggplot(sample_size_results, aes(x = 1 - beta, y = n, color = as.factor(alpha))) +
  geom_line() +
  labs(title = "Effect of Power on Required Sample Size",
       x = "Power (1 - Beta)",
       y = "Required Sample Size (n)",
       color = "Alpha Level") +
  theme_minimal()
```



The required sample size n is primarily influenced by the **effect size (ES), significance level (α), and power** $(1-\beta)$.

- A **larger effect size** significantly reduces the required sample size due to the inverse square relationship:

$$n \propto \frac{1}{ES^2}.$$

  This means that detecting smaller differences requires much **larger samples**.

- Increasing the **significance level** (α) decreases n, as a higher α allows for a greater chance of Type I errors, making the test less conservative.

- Conversely, increasing **power** $\mathbf{(1-\beta)}$ raises n, ensuring a lower chance of Type II errors but requiring a **larger sample size** to achieve higher certainty.

Problem 4

Consider a random sample $(X_1, \ldots, X_n$ i.i.d.) from a $N(\mu, \sigma^2)$ population where both parameters are unknown and we would like to estimate σ^2 using S^2. Note that

$$\frac{\sum_{i=1}^{n}(X_i-\bar{X})^2}{\sigma^2} \sim \chi^2 \text{ with d.f. } = n-1.$$

Using this, we want to determine what n should be (i.e., find the smallest integer n) such that

$$P\left(1-\epsilon \leq \frac{S^2}{\sigma^2} \leq 1+\epsilon\right) \geq \gamma$$

where

$$S^2 = \frac{\sum_{i=1}^{n}(X_i-\bar{X})^2}{n-1}.$$

Generate a function in R that computes the desired sample size for a given ϵ and γ. Using tables and graphs illustrate the effects of ϵ and γ on the needed sample sizes.

[sol] Consider a random sample $X_1, X_2, \ldots, X_n$ from a normal population $N(\mu, \sigma^2)$ where both μ and σ^2 are unknown. The sample variance is defined as:

$$S^2 = \frac{\sum_{i=1}^n (X_i - \bar{X})^2}{n-1}.$$

From statistical theory, we know that:

$$\frac{(n-1)S^2}{\sigma^2} \sim \chi^2_{(n-1)}.$$

This implies that the ratio S^2/σ^2 follows:

$$\frac{S^2}{\sigma^2} = \frac{\chi^2_{n-1}}{n-1}.$$

We want to determine the smallest n such that:

$$P\left(1 - \epsilon \leq \frac{S^2}{\sigma^2} \leq 1 + \epsilon\right) \geq \gamma.$$

Substituting S^2/σ^2, we get:

$$P\left((1-\epsilon)(n-1) \leq \chi^2_{n-1} \leq (1+\epsilon)(n-1)\right) \geq \gamma.$$

This means that the probability mass of a chi-square distribution with $n-1$ degrees of freedom must be concentrated between $(1-\epsilon)(n-1)$ and $(1+\epsilon)(n-1)$.

Let $F_{\chi^2(k)}(x)$ denote the cumulative distribution function (CDF) of a chi-square distribution with k degrees of freedom.

Define:

$$Q_L = F^{-1}_{\chi^2(n-1)}\left(\frac{1-\gamma}{2}\right),$$

$$Q_U = F^{-1}_{\chi^2(n-1)}\left(\frac{1+\gamma}{2}\right).$$

The required probability condition can now be rewritten as:

$$F_{\chi^2(n-1)}\left((1+\epsilon)(n-1)\right) - F_{\chi^2(n-1)}\left((1-\epsilon)(n-1)\right) \geq \gamma.$$

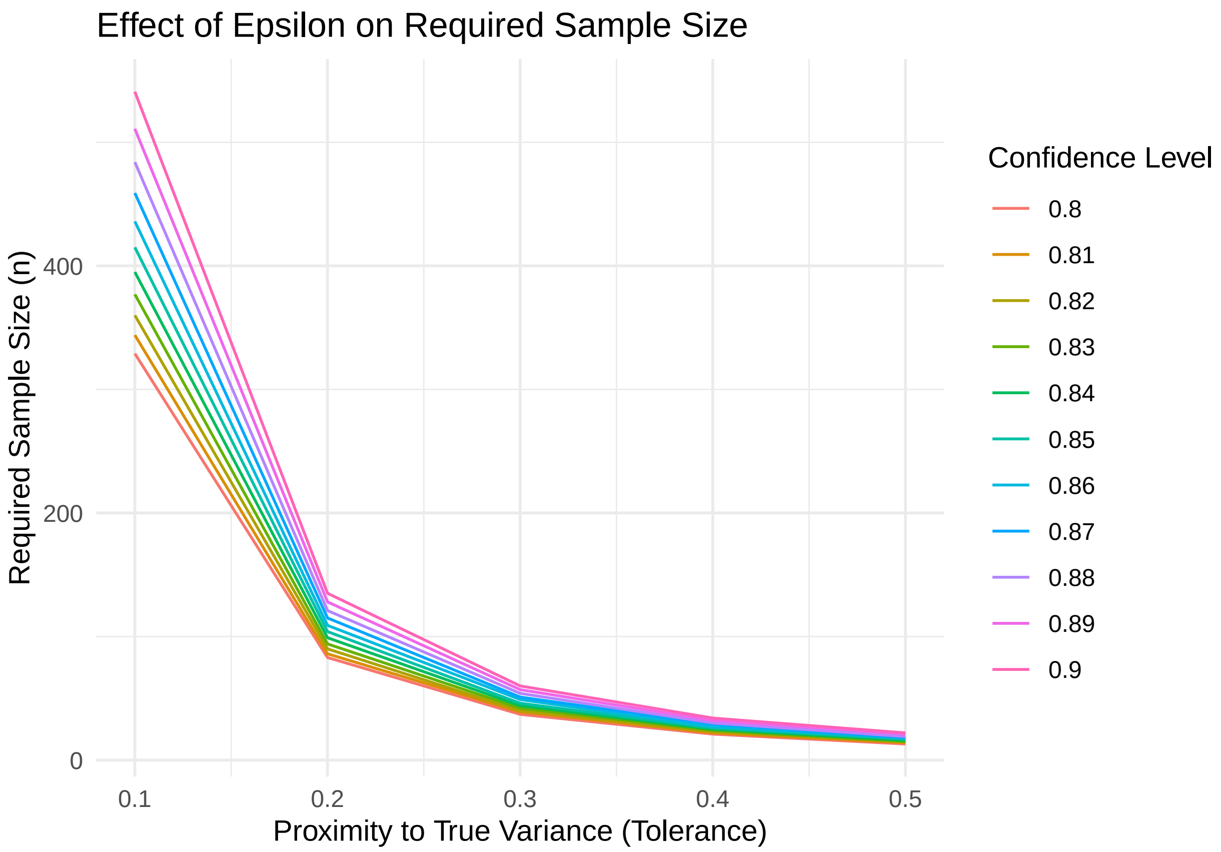
To determine n, we solve:

$$F_{\chi^2(n-1)}\left((1+\epsilon)(n-1)\right) - F_{\chi^2(n-1)}\left((1-\epsilon)(n-1)\right) - \gamma = 0.$$

```r
samplesize <- function(epsilon, gamma) {
  objective_function <- function(n) {
    lower_bound <- (1 - epsilon) * (n - 1)
    upper_bound <- (1 + epsilon) * (n - 1)

    # Compute probability mass between chi-square quantiles
    probability <- pchisq(upper_bound, df = n - 1) - pchisq(lower_bound, df = n - 1)

    return(probability - gamma)  # Find where probability meets gamma
  }

  # Start searching from n = 2
  n <- 2
  while (objective_function(n) < 0) {
    n <- n + 1  # Increment n until condition is met
  }

  return(n)  # Return the smallest valid sample size
}

# Define ranges for epsilon and gamma
epsilon_values <- seq(0.1, 0.5, by = 0.1) # Tolerance levels
gamma_values <- seq(0.8, 0.9, by = 0.01) # Confidence levels

# Initialize storage for results
chi_results <- data.frame(epsilon = double(),
                           gamma = double(),
                           n = integer())

# Initialize row counter
RowCounter <- 0

# Compute sample sizes using nested loops
for (epsilon in epsilon_values) {
  for (gamma in gamma_values) {
    n <- samplesize(epsilon, gamma)  # Compute required sample size

    # Populate the data frame
    RowCounter <- RowCounter + 1
    chi_results[RowCounter, "epsilon"] <- epsilon
    chi_results[RowCounter, "gamma"] <- gamma
    chi_results[RowCounter, "n"] <- n
  }
}

print(chi_results)
```

```
##    epsilon gamma   n
## 1      0.1  0.80 329
## 2      0.1  0.81 344
## 3      0.1  0.82 360
## 4      0.1  0.83 377
## 5      0.1  0.84 395
## 6      0.1  0.85 415
## 7       0.1  0.86 436
## 8       0.1  0.87 459
## 9       0.1  0.88 484
## 10      0.1  0.89 511
## 11      0.1  0.90 541
## 12      0.2  0.80  83
## 13      0.2  0.81  86
## 14      0.2  0.82  90
## 15      0.2  0.83  94
## 16      0.2  0.84  99
## 17      0.2  0.85 104
## 18      0.2  0.86 109
## 19      0.2  0.87 115
## 20      0.2  0.88 121
## 21      0.2  0.89 128
## 22      0.2  0.90 135
## 23      0.3  0.80  37
## 24      0.3  0.81  39
## 25      0.3  0.82  40
## 26      0.3  0.83  42
## 27      0.3  0.84  44
## 28      0.3  0.85  46
## 29      0.3  0.86  49
## 30      0.3  0.87  51
## 31      0.3  0.88  54
## 32      0.3  0.89  57
## 33      0.3  0.90  60
## 34      0.4  0.80  21
## 35      0.4  0.81  22
## 36      0.4  0.82  23
## 37      0.4  0.83  24
## 38      0.4  0.84  25
## 39      0.4  0.85  26
## 40      0.4  0.86  27
## 41      0.4  0.87  29
## 42      0.4  0.88  30
## 43      0.4  0.89  32
## 44      0.4  0.90  34
## 45      0.5  0.80  13
## 46      0.5  0.81  14
## 47      0.5  0.82  15
## 48      0.5  0.83  15
## 49      0.5  0.84  16
## 50      0.5  0.85  17
## 51      0.5  0.86  18
## 52      0.5  0.87  18
## 53      0.5  0.88  19
## 54      0.5  0.89  21
## 55      0.5  0.90  22
```

```r
ggplot(chi_results, aes(x = epsilon, y = n, color = as.factor(gamma))) +
  geom_line() +
  labs(title = "Effect of Epsilon on Required Sample Size",
       x = "Proximity to True Variance (Tolerance)",
       y = "Required Sample Size (n)",
       color = "Confidence Level") +
  theme_minimal()
```



From the table and graph, I found out the following observations.

- **Higher confidence levels (γ) require larger sample sizes.** Increasing the probability coverage (e.g., from 80% to 90%) significantly increases the required sample size.
- **Larger tolerances (ϵ) allow smaller sample sizes.** A wider acceptable range around σ^2 means variance can be estimated with fewer samples.
- **The relationship between n, γ, and ϵ is non-linear.** Sample size increases rapidly as ϵ decreases or γ increases.

Problem 5

Suppose that a random sample (i.i.d. observations) $X_1, \ldots, X_n$ of size n is to be taken from a random variable $X \sim N(\mu, \sigma^2)$ for which the population mean μ and the population variance $\sigma^2 < \infty$ are unknown.

Suppose the study would like to make sure that the probability is at least γ that the sample mean computed from the sample will be within $\epsilon\sigma$ of the unknown population mean μ. That is, the researcher would like the probability requirement

$$P\left(|\overline{X} - \mu| \leq \epsilon\sigma\right) \geq \gamma$$

to be satisfied.

Using the fact that $\overline{X} \sim N(\mu, \sigma^2/n)$, one would like to compute the sample size n (i.e., find the smallest integer n) that will satisfy the probability requirement. Generate a function in R that computes the desired sample size for a given ϵ and γ. Using tables and graphs illustrate the effects of ϵ and γ on the needed sample sizes.

[sol] Derivation of Sample Size Requirement: We need to determine the smallest sample size n that ensures:

$$P\left(|\overline{X} - \mu| \leq \epsilon\sigma\right) \geq \gamma.$$

Since $X_1, \ldots, X_n$ are i.i.d. observations from $X \sim N(\mu, \sigma^2)$, the sample mean $\overline{X}$ follows:

$$\overline{X} \sim N\left(\mu, \frac{\sigma^2}{n}\right).$$

Define the standardized variable:

$$Z = \frac{\overline{X} - \mu}{\sigma/\sqrt{n}} \sim N(0, 1).$$

Rewriting the probability condition using Z:

$$P\left(-\epsilon\sigma \leq \overline{X} - \mu \leq \epsilon\sigma\right)$$

$$P\left(-\epsilon \leq \frac{\overline{X} - \mu}{\sigma/\sqrt{n}} \leq \epsilon\right).$$

Using the standard normal variable Z, this becomes:

$$P\left(-\epsilon\sqrt{n} \leq Z \leq \epsilon\sqrt{n}\right) \geq \gamma.$$

The probability statement translates to:

$$P\left(Z \leq \epsilon\sqrt{n}\right) - P\left(Z \leq -\epsilon\sqrt{n}\right) \geq \gamma.$$

Using the cumulative distribution function $\Phi(z)$ of the standard normal distribution:

$$\Phi(\epsilon\sqrt{n}) - \Phi(-\epsilon\sqrt{n}) \geq \gamma.$$

Using symmetry of the normal distribution: $\Phi(-x) = 1 - \Phi(x)$,

$$\Phi(\epsilon\sqrt{n}) - (1 - \Phi(\epsilon\sqrt{n})) \geq \gamma.$$

$$2\Phi(\epsilon\sqrt{n}) - 1 \geq \gamma.$$

$$\Phi(\epsilon\sqrt{n}) \geq \frac{1 + \gamma}{2}.$$

Taking the inverse normal (quantile function):

$$\epsilon\sqrt{n} \geq z_{\frac{1+\gamma}{2}},$$

where $z_{\frac{1+\gamma}{2}}$ is the critical value from the standard normal table.

Solving for n:

$$n \geq \left(\frac{z_{\frac{1+\gamma}{2}}}{\epsilon}\right)^2.$$

Since n must be an integer, we take the **ceiling** function:

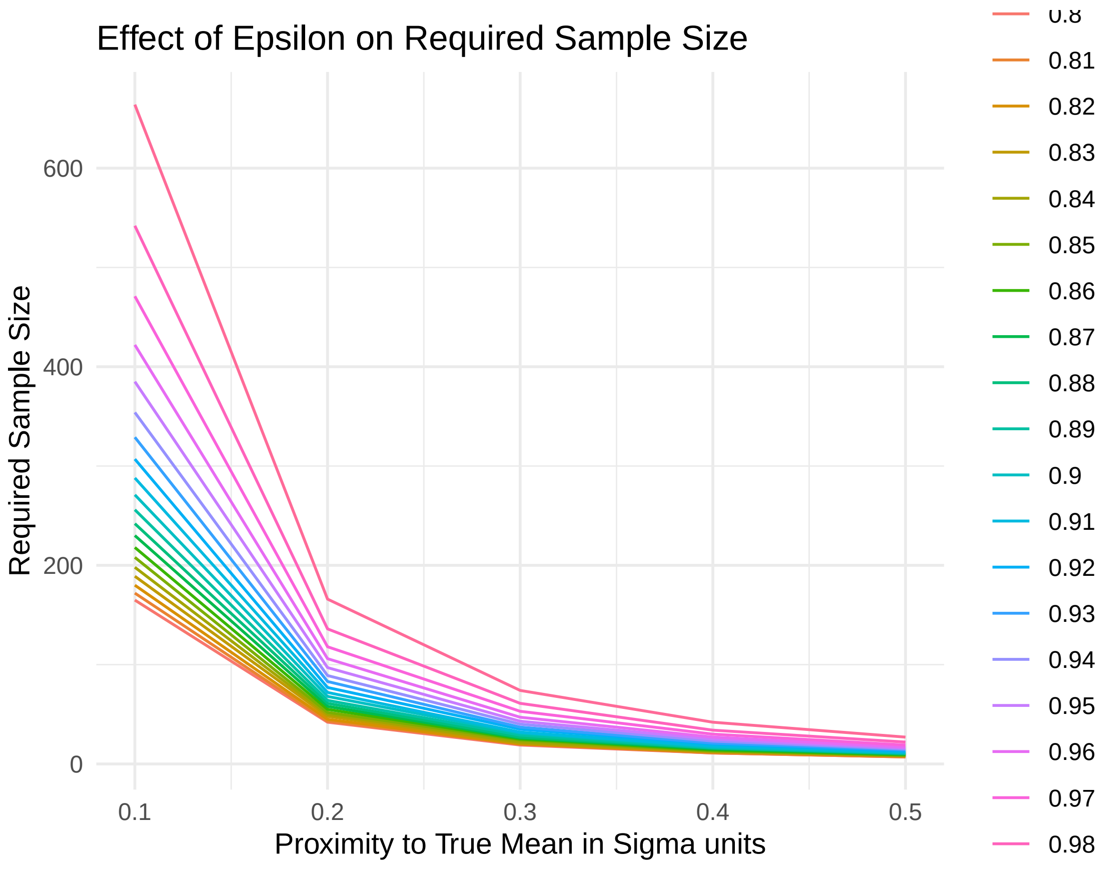
$$n = \left\lceil \left(\frac{z_{\frac{1+\gamma}{2}}}{\epsilon}\right)^2 \right\rceil.$$

This formula gives the minimum required sample size for a given confidence level γ and precision parameter ϵ.

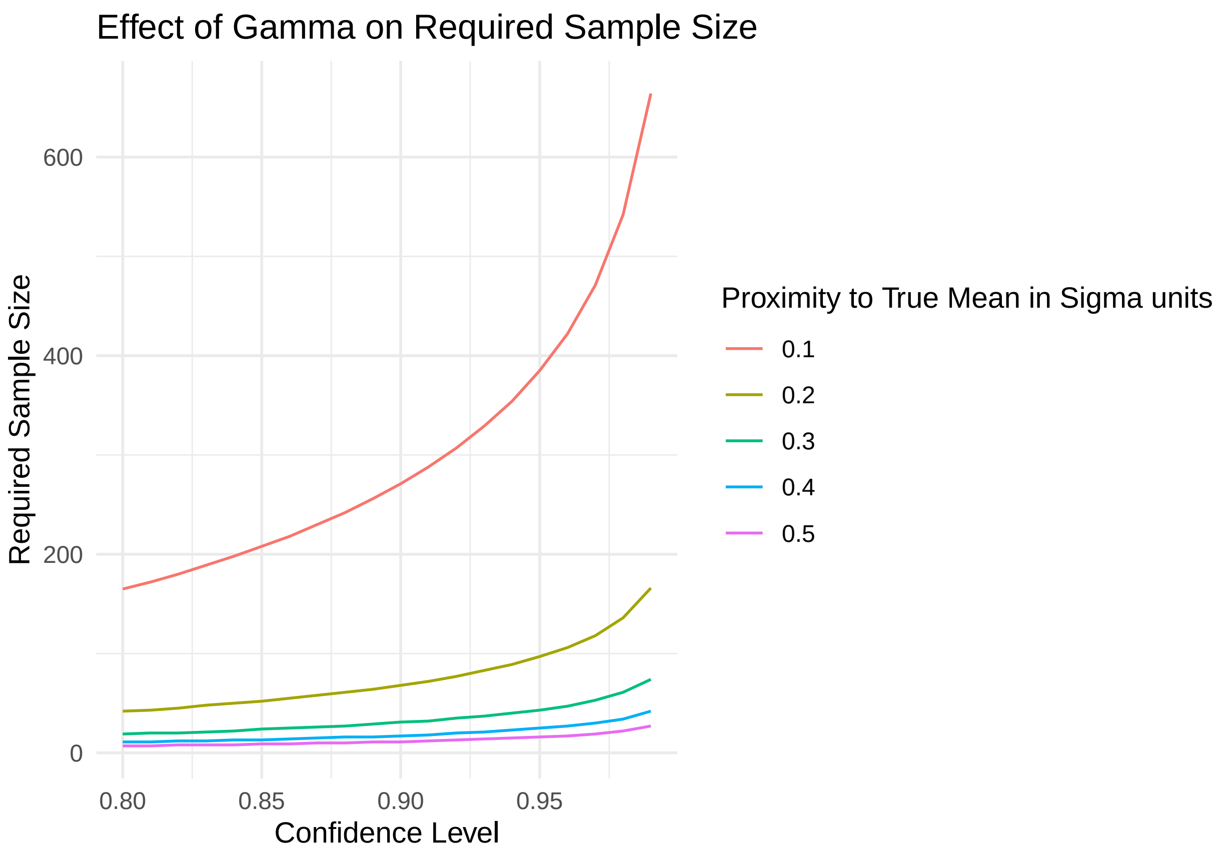
```r
norm_sample_size <- function(epsilon, gamma) {
  # Find the critical z-score corresponding to the given probability
  z_critical <- qnorm((1 + gamma) / 2)

  # Compute the required sample size using the formula
  n <- (z_critical / epsilon)^2

  n <- ceiling(n)

  return(n)
}


# Define ranges for epsilon and gamma
epsilon_values <- seq(0.1, 0.5, by = 0.1)
gamma_values <- seq(0.8, 0.99, by = 0.01)

# Initialize storage for results
norm_results <- data.frame(epsilon = double(),
                           gamma = double(),
                           n = integer())

# Initialize row counter
RowCounter <- 0

# Compute sample sizes using nested loops
for (epsilon in epsilon_values) {
  for (gamma in gamma_values) {
    n <- norm_sample_size(epsilon, gamma)

    # Populate the data frame
  RowCounter <- RowCounter + 1
  norm_results[RowCounter, "epsilon"] <- epsilon
  norm_results[RowCounter, "gamma"] <- gamma
  norm_results[RowCounter, "n"] <- n
  }
}

print(norm_results)
```

```
##     epsilon gamma   n
## 1       0.1  0.80 165
## 2       0.1  0.81 172
## 3       0.1  0.82 180
## 4       0.1  0.83 189
## 5       0.1  0.84 198
## 6       0.1  0.85 208
## 7       0.1  0.86 218
## 8       0.1  0.87 230
## 9       0.1  0.88 242
## 10      0.1  0.89 256
## 11      0.1  0.90 271
## 12      0.1  0.91 288
## 13      0.1  0.92 307
## 14      0.1  0.93 329
## 15      0.1  0.94 354
## 16      0.1  0.95 385
## 17      0.1  0.96 422
## 18      0.1  0.97 471
## 19      0.1  0.98 542
## 20      0.1  0.99 664
## 21      0.2  0.80  42
## 22      0.2  0.81  43
## 23      0.2  0.82  45
## 24      0.2  0.83  48
## 25      0.2  0.84  50
## 26      0.2  0.85  52
## 27      0.2  0.86  55
## 28      0.2  0.87  58
## 29      0.2  0.88  61
## 30      0.2  0.89  64
## 31      0.2  0.90  68
## 32      0.2  0.91  72
## 33      0.2  0.92  77
## 34      0.2  0.93  83
## 35      0.2  0.94  89
## 36      0.2  0.95  97
## 37      0.2  0.96 106
## 38      0.2  0.97 118
## 39      0.2  0.98 136
## 40      0.2  0.99 166
## 41      0.3  0.80  19
## 42      0.3  0.81  20
## 43      0.3  0.82  20
## 44      0.3  0.83  21
## 45      0.3  0.84  22
## 46      0.3  0.85  24
## 47      0.3  0.86  25
## 48      0.3  0.87  26
## 49      0.3  0.88  27
## 50      0.3  0.89  29
## 51      0.3  0.90  31
## 52      0.3  0.91  32
## 53      0.3  0.92  35
## 54      0.3  0.93  37
## 55      0.3  0.94  40
## 56      0.3  0.95  43
## 57      0.3  0.96  47
## 58      0.3  0.97  53
## 59      0.3  0.98  61
## 60      0.3  0.99  74
## 61      0.4  0.80  11
## 62      0.4  0.81  11
## 63      0.4  0.82  12
## 64      0.4  0.83  12
## 65      0.4  0.84  13
## 66      0.4  0.85  13
## 67      0.4  0.86  14
## 68      0.4  0.87  15
## 69      0.4  0.88  16
## 70      0.4  0.89  16
## 71      0.4  0.90  17
## 72      0.4  0.91  18
## 73      0.4  0.92  20
## 74      0.4  0.93  21
## 75      0.4  0.94  23
## 76      0.4  0.95  25
## 77      0.4  0.96  27
## 78      0.4  0.97  30
## 79      0.4  0.98  34
## 80      0.4  0.99  42
## 81      0.5  0.80   7
## 82      0.5  0.81   7
## 83      0.5  0.82   8
## 84      0.5  0.83   8
## 85      0.5  0.84   8
## 86      0.5  0.85   9
## 87      0.5  0.86   9
## 88      0.5  0.87  10
## 89      0.5  0.88  10
## 90      0.5  0.89  11
## 91      0.5  0.90  11
## 92      0.5  0.91  12
## 93      0.5  0.92  13
## 94      0.5  0.93  14
## 95      0.5  0.94  15
## 96      0.5  0.95  16
## 97      0.5  0.96  17
## 98      0.5  0.97  19
## 99      0.5  0.98  22
## 100     0.5  0.99  27
```

```r
# Plot the effects of epsilon on required sample size
ggplot(norm_results, aes(x = epsilon, y = n, color = factor(gamma))) +
  geom_line() +
  labs(title = "Effect of Epsilon on Required Sample Size",
       x = "Proximity to True Mean in Sigma units",
       y = "Required Sample Size",
       color = "Confidence Level") +
  theme_minimal()
```



```
# Plot the effects of gamma on required sample size
ggplot(norm_results, aes(x = gamma, y = n, color = factor(epsilon))) +
  geom_line() +
  labs(title = "Effect of Gamma on Required Sample Size",
       x = "Confidence Level",
       y = "Required Sample Size",
       color = "Proximity to True Mean in Sigma units") +
  theme_minimal()
```



The results show that the required sample size n increases significantly as the desired probability γ approaches 1, ensuring a higher probability that the sample mean is close to the true mean. Mathematically, this follows from the derived formula:

$$n \geq \left(\frac{z_{(1+\gamma)/2}}{\epsilon}\right)^2.$$

As $\gamma \to 1$, the critical value $z_{(1+\gamma)/2}$ increases, leading to a larger n. Conversely, increasing the tolerance level ϵ reduces the required sample size due to the inverse quadratic relationship:

$$n \propto \frac{1}{\epsilon^2}.$$

This highlights the trade-off between precision and feasibility: smaller margins of error ($\epsilon \to 0$) require significantly larger sample sizes, while allowing for greater tolerance ($\epsilon \to 0.5$) reduces n. These insights guide researchers in determining the optimal sample size based on their accuracy and confidence requirements.

Problem 6

Suppose that a random sample (i.i.d. observations) $X_1, \ldots, X_n$ of size n is to be taken from the uniform distribution on the interval $[0, \theta]$ with p.d.f. $f(x) = 1/\theta$ and c.d.f. $F(x) = x/\theta$, for $0 \leq x \leq \theta$, for an unknown θ.

Let $Y = \max\{X_1, \ldots, X_n\}$. As a side note, Y is known to be the Maximum Likelihood Estimator (MLE) of θ. Note that if $X_1, \ldots, X_n$ are i.i.d. where each X_i has c.d.f. $F(x)$, then the c.d.f. of $V = \max(X_1, \ldots, X_n)$ is

$$P(V \leq v) = [F(v)]^n.$$

Using this, first determine $P(Y \leq \alpha\theta)$ as a function of n and α, for $0 < \alpha < 1$.

Then, one would like to compute how large the sample size n should be (i.e., find the smallest integer n) such that it satisfies the probability requirement

$$P\left(|\max\{X_1,\ldots,X_n\}-\theta|\le\epsilon\theta\right)\ge\gamma.$$

Generate a function in R that computes the desired sample size for a given ϵ and γ. Using tables and graphs illustrate the effects of ϵ and γ on the needed sample sizes.

[sol] Since $Y=\max(X_1,\ldots,X_n)$, its cumulative distribution function (CDF) is given by:

$$P(Y\le v)=[F(v)]^n$$

For our uniform distribution, $F(v)=v/\theta$, so:

$$P(Y\le\alpha\theta)=\left(\frac{\alpha\theta}{\theta}\right)^n=\alpha^n.$$

Solving for n in the Probability Requirement, we need to satisfy:

$$P\left(|Y-\theta|\le\epsilon\theta\right)\ge\gamma.$$

This expands to:

$$P(\theta-\epsilon\theta\le Y\le\theta)\ge\gamma.$$

Since $P(Y\le\theta)=1$, we focus on:

$$P(Y\ge(1-\epsilon)\theta)=1-P(Y\le(1-\epsilon)\theta).$$

Using our CDF:

$$1-(1-\epsilon)^n\ge\gamma.$$

Rearrange:

$$(1-\epsilon)^n\le1-\gamma.$$

Taking the natural logarithm:

$$n\log(1-\epsilon)\le\log(1-\gamma).$$

Solving for n:

$$n\ge\frac{\log(1-\gamma)}{\log(1-\epsilon)}.$$

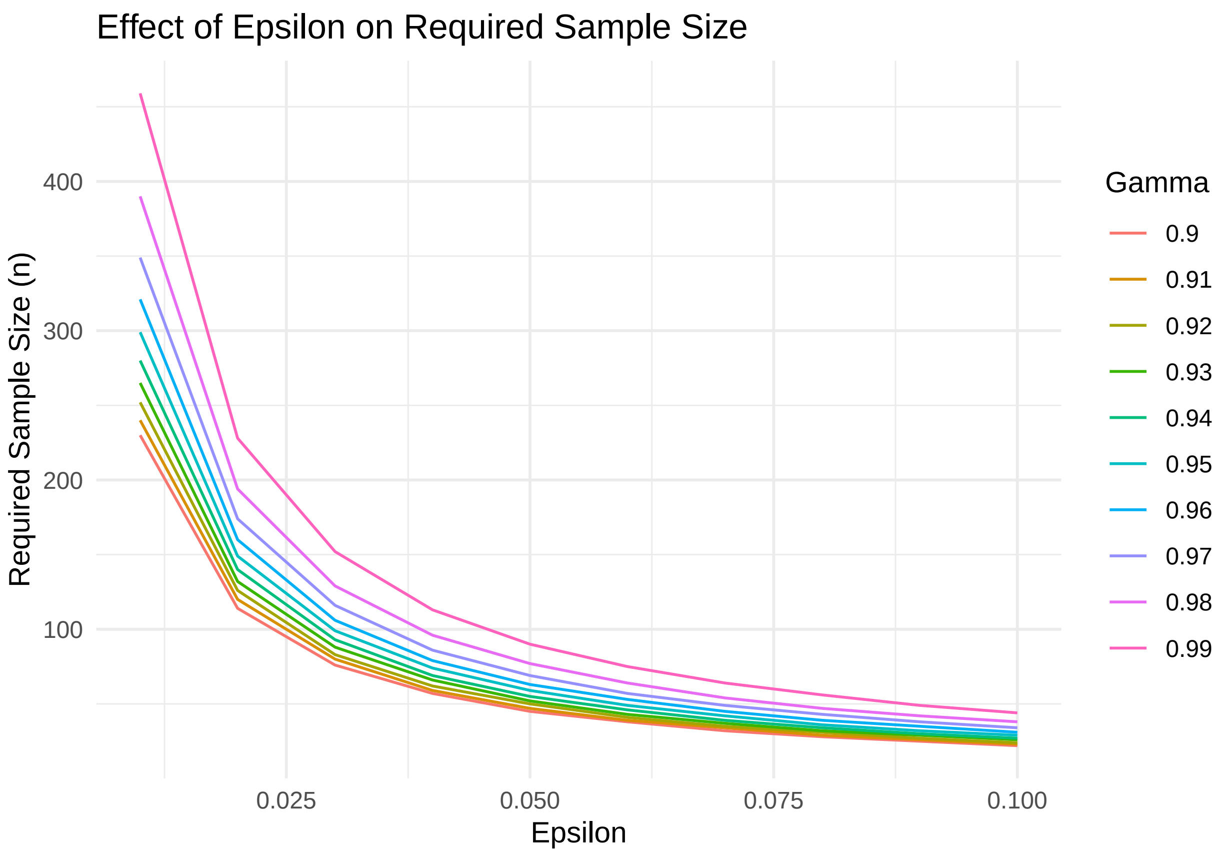
Since n must be an integer, we take the ceiling:

$$n=\left\lceil\frac{\log(1-\gamma)}{\log(1-\epsilon)}\right\rceil.$$

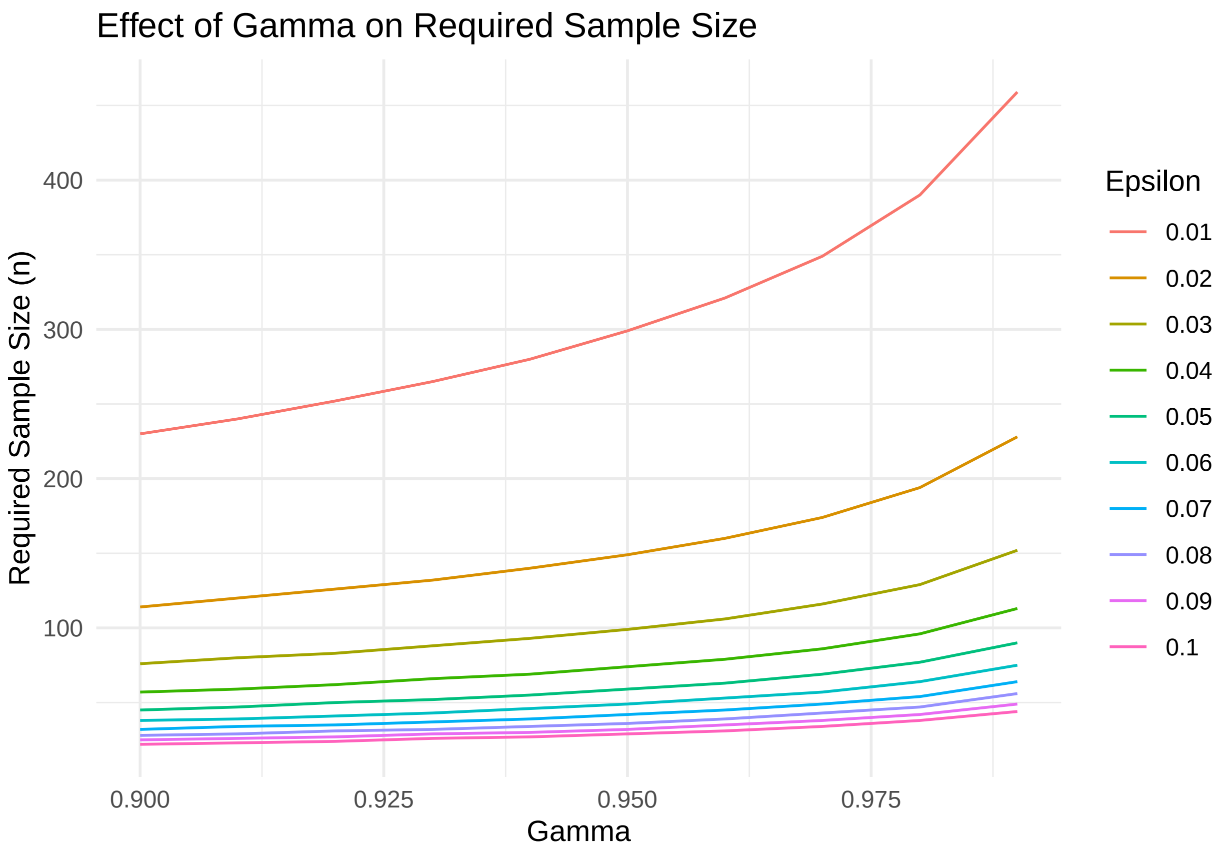
```r
# Function to compute the required sample size
sample_size <- function(epsilon, gamma) {
  n <- log(1 - gamma) / log(1 - epsilon)
  n <- ceiling(n)
  return(n)
}

# Create a grid of epsilon and gamma values
epsilon_values <- seq(0.01, 0.1, by=0.01)
gamma_values <- seq(0.9, 0.99, by=0.01)

# Compute sample sizes for different epsilon and gamma combinations
uni_results <- data.frame(epsilon = numeric(),
                              gamma = numeric(),
                              sample_size = integer())

# Initialize row counter
RowCounter <- 0

for (epsilon in epsilon_values) {
  for (gamma in gamma_values) {
    pareto_sample_size <- sample_size(epsilon, gamma)

    # Populate the data frame
    RowCounter <- RowCounter + 1
    uni_results[RowCounter, "epsilon"] <- epsilon
    uni_results[RowCounter, "gamma"] <- gamma
    uni_results[RowCounter, "sample_size"] <- pareto_sample_size
  }
}

# Print the results
print(uni_results)
```

```
##     epsilon gamma sample_size
## 1      0.01  0.90         230
## 2      0.01  0.91         240
## 3      0.01  0.92         252
## 4      0.01  0.93         265
## 5      0.01  0.94         280
## 6      0.01  0.95         299
## 7      0.01  0.96         321
## 8      0.01  0.97         349
## 9      0.01  0.98         390
## 10     0.01  0.99         459
## 11     0.02  0.90         114
## 12     0.02  0.91         120
## 13     0.02  0.92         126
## 14     0.02  0.93         132
## 15     0.02  0.94         140
## 16     0.02  0.95         149
## 17     0.02  0.96         160
## 18     0.02  0.97         174
## 19     0.02  0.98         194
## 20     0.02  0.99         228
## 21     0.03  0.90          76
## 22     0.03  0.91          80
## 23     0.03  0.92          83
## 24     0.03  0.93          88
## 25     0.03  0.94          93
## 26     0.03  0.95          99
## 27     0.03  0.96         106
## 28     0.03  0.97         116
## 29     0.03  0.98         129
## 30     0.03  0.99         152
## 31     0.04  0.90          57
## 32     0.04  0.91          59
## 33      0.04  0.92            62
## 34      0.04  0.93            66
## 35      0.04  0.94            69
## 36      0.04  0.95            74
## 37      0.04  0.96            79
## 38      0.04  0.97            86
## 39      0.04  0.98            96
## 40      0.04  0.99           113
## 41      0.05  0.90            45
## 42      0.05  0.91            47
## 43      0.05  0.92            50
## 44      0.05  0.93            52
## 45      0.05  0.94            55
## 46      0.05  0.95            59
## 47      0.05  0.96            63
## 48      0.05  0.97            69
## 49      0.05  0.98            77
## 50      0.05  0.99            90
## 51      0.06  0.90            38
## 52      0.06  0.91            39
## 53      0.06  0.92            41
## 54      0.06  0.93            43
## 55      0.06  0.94            46
## 56      0.06  0.95            49
## 57      0.06  0.96            53
## 58      0.06  0.97            57
## 59      0.06  0.98            64
## 60      0.06  0.99            75
## 61      0.07  0.90            32
## 62      0.07  0.91            34
## 63      0.07  0.92            35
## 64      0.07  0.93            37
## 65      0.07  0.94            39
## 66      0.07  0.95            42
## 67      0.07  0.96            45
## 68      0.07  0.97            49
## 69      0.07  0.98            54
## 70      0.07  0.99            64
## 71      0.08  0.90            28
## 72      0.08  0.91            29
## 73      0.08  0.92            31
## 74      0.08  0.93            32
## 75      0.08  0.94            34
## 76      0.08  0.95            36
## 77      0.08  0.96            39
## 78      0.08  0.97            43
## 79      0.08  0.98            47
## 80      0.08  0.99            56
## 81      0.09  0.90            25
## 82      0.09  0.91            26
## 83      0.09  0.92            27
## 84      0.09  0.93            29
## 85      0.09  0.94            30
## 86      0.09  0.95            32
## 87     0.09  0.96           35
## 88     0.09  0.97           38
## 89     0.09  0.98           42
## 90     0.09  0.99           49
## 91     0.10  0.90           22
## 92     0.10  0.91           23
## 93     0.10  0.92           24
## 94     0.10  0.93           26
## 95     0.10  0.94           27
## 96     0.10  0.95           29
## 97     0.10  0.96           31
## 98     0.10  0.97           34
## 99     0.10  0.98           38
## 100    0.10  0.99           44
```

```r
# Plot n as a function of epsilon for different gamma values
ggplot(uni_results, aes(x = epsilon, y = sample_size, color = as.factor(gamma))) +
  geom_line() +
  labs(title = "Effect of Epsilon on Required Sample Size",
       x = "Epsilon",
       y = "Required Sample Size (n)",
       color = "Gamma") +
  theme_minimal()
```



```r
# Plot n as a function of gamma for different epsilon values
ggplot(uni_results, aes(x = gamma, y = sample_size, color = as.factor(epsilon))) +
  geom_line() +
  labs(title = "Effect of Gamma on Required Sample Size",
       x = "Gamma",
       y = "Required Sample Size (n)",
       color = "Epsilon") +
  theme_minimal()
```



The required sample size, n, is determined by the formula:

$$n = \left\lceil \frac{\log(1-\gamma)}{\log(1-\epsilon)} \right\rceil,$$

where ϵ is the tolerance level and γ is the confidence level.

As ϵ increases, the denominator, $\log(1-\epsilon)$, becomes less negative, leading to a smaller sample size. For example, when $\gamma = 0.90$, increasing ϵ from 0.01 to 0.04 reduces the sample size from 230 to 57. Conversely, as γ increases, the numerator, $\log(1-\gamma)$, becomes more negative, requiring a larger sample size. For instance, when $\epsilon = 0.01$, increasing γ from 0.90 to 0.99 raises the sample size from 230 to 459.

This formula highlights the trade-off between precision and confidence. Higher precision (smaller ϵ) and higher confidence (larger γ) require significantly larger sample sizes. Allowing more tolerance and lower confidence reduces the sample size. Achieving both high precision and confidence comes at the cost of more extensive sampling.

Problem 7

Suppose that the random variable X has a Pareto(θ, β). Its pdf $f(x)$ and cdf $F(x)$, respectively, are given below:

$$f(x) = \begin{cases} \theta\beta^\theta x^{-(\theta+1)} & \text{for } x > \beta \\ 0 & \text{else} \end{cases}$$

$$F(x) = 1 - \left(\frac{\beta}{x}\right)^{\theta}$$

Suppose that a random sample $X_1, \ldots, X_n$ of size n is to be taken where $X_i \overset{i.i.d.}{\sim} \text{Pareto}(\theta, \beta)$. Further suppose that θ is known and β is unknown.

Let $Y = \min(X_1, \ldots, X_n)$. As a side note, Y is known to be the Maximum Likelihood Estimator (MLE) of β. Note that if $X_1, \ldots, X_n$ are i.i.d. where each X_i has c.d.f. $F(x)$, then the c.d.f. of $U = \min(X_1, \ldots, X_n)$ is

$$P(U \leq u) = 1 - [1 - F(u)]^n.$$

Using this, first determine $P(Y \geq \alpha\beta)$ as a function of n and α, for $\alpha > 0$.

Then, one would like to compute how large the sample size n should be (i.e., find the smallest integer n) such that $P(Y \geq (1+\epsilon)\beta) \leq 1 - \gamma$. For example, if $\epsilon = 0.01$ and $\gamma = 0.95$, we are looking for the smallest sample size so that the probability of Y overestimating β by more than 1% is no more than 5%.

Generate a function in R that computes the desired sample size for a given ϵ and γ. Using tables and graphs illustrate the effects of ϵ and γ on the needed sample sizes.

[sol] We derive the fuction with given information that $X \sim \text{Pareto}(\theta, \beta)$ with probability density function (pdf):

$$f(x) = \begin{cases} \theta\beta^{\theta}x^{-(\theta+1)}, & x > \beta, \\ 0, & \text{otherwise.} \end{cases}$$

The cumulative distribution function (CDF) is:

$$F(x) = 1 - \left(\frac{\beta}{x}\right)^{\theta}, \quad x > \beta.$$

Suppose we take a random sample $X_1, \ldots, X_n$, where X_i are i.i.d. $\text{Pareto}(\theta, \beta)$. The minimum of the sample is:

$$Y = \min(X_1, \ldots, X_n).$$

Using the formula for the CDF of the minimum:

$$P(Y \leq y) = 1 - [1 - F(y)]^n,$$

we substitute $F(y)$:

$$P(Y \leq y) = 1 - \left[\left(\frac{\beta}{y}\right)^{\theta}\right]^n = 1 - \left(\frac{\beta}{y}\right)^{\theta n}, \quad y > \beta.$$

Thus, the probability that Y is at least $\alpha\beta$ (where $\alpha > 0$) is:

$$P(Y \geq \alpha\beta) = 1 - P(Y \leq \alpha\beta).$$

Substituting the CDF:

$$P(Y \geq \alpha\beta) = \left(\frac{\beta}{\alpha\beta}\right)^{\theta n} = \left(\frac{1}{\alpha}\right)^{\theta n}.$$

Since $\alpha = 1 + \epsilon$, we get:

$$P(Y \geq (1+\epsilon)\beta) = \left(\frac{1}{1+\epsilon}\right)^{\theta n}.$$

Now, we solve for n such that:

$$P(Y \geq (1+\epsilon)\beta) \leq 1 - \gamma.$$

Substituting our result:

$$\left(\frac{1}{1+\epsilon}\right)^{\theta n} \leq 1 - \gamma.$$

Taking the natural logarithm on both sides:

$$\theta n \log\left(\frac{1}{1+\epsilon}\right) \leq \log(1-\gamma).$$

Since $\log(1/(1+\epsilon)) = -\log(1+\epsilon)$, we rewrite:

$$-\theta n \log(1+\epsilon) \leq \log(1-\gamma).$$

Dividing both sides by $-\theta \log(1+\epsilon)$ (which is positive because $\theta > 0$ and $\epsilon > 0$):

$$n \geq \frac{\log(1-\gamma)}{-\theta \log(1+\epsilon)}.$$

Since n must be an integer, we take the ceiling:

$$n = \left\lceil \frac{\log(1-\gamma)}{-\theta \log(1+\epsilon)} \right\rceil.$$

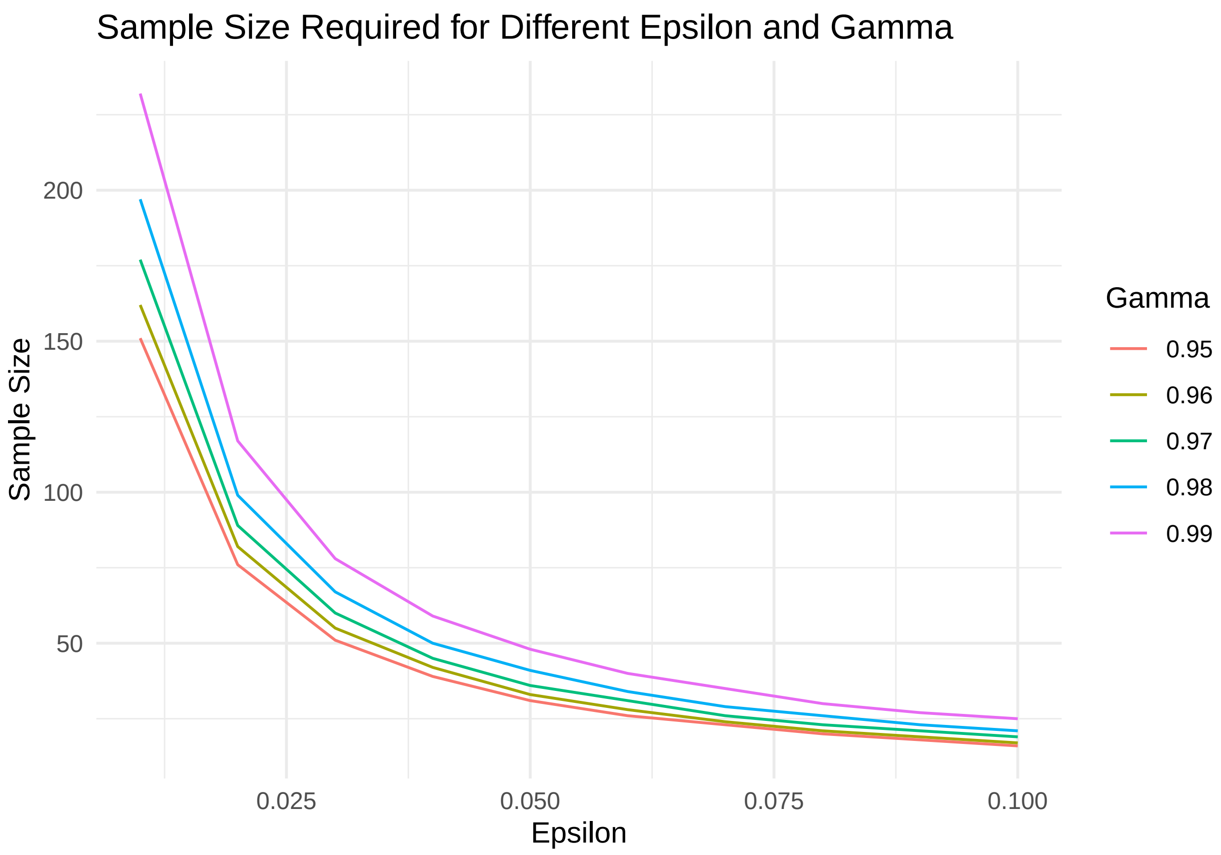
```r
# Function to compute the required sample size
sample_size <- function(epsilon, gamma, theta) {
  n <- (log(1 - gamma)) / (-theta * log(1 + epsilon))
  n <- ceiling(n)
  return(n)
}

# Example computation for epsilon = 0.01 and gamma = 0.95
epsilon_example <- 0.01
gamma_example <- 0.95
theta <- 2  # Assume the given theta is 2

sample_size_example <- sample_size(epsilon_example, gamma_example, theta)
cat("The required sample size for epsilon =", epsilon_example,
    "and gamma =", gamma_example, "is", sample_size_example, "\n")
```

```
## The required sample size for epsilon = 0.01 and gamma = 0.95 is 151
```

```r
# Create a grid of epsilon and gamma values
epsilon_values <- seq(0.01, 0.1, by=0.01)
gamma_values <- seq(0.95, 0.99, by=0.01)

# Compute sample sizes for different epsilon and gamma combinations
pareto_results <- data.frame(epsilon = double(),
                             gamma = double(),
                             sample_size = integer())

# Initialize row counter
RowCounter <- 0

for (epsilon in epsilon_values) {
  for (gamma in gamma_values) {
    pareto_sample_size <- sample_size(epsilon, gamma, theta)

    # Populate the data frame
    RowCounter <- RowCounter + 1
    pareto_results[RowCounter, "epsilon"] <- epsilon
    pareto_results[RowCounter, "gamma"] <- gamma
    pareto_results[RowCounter, "sample_size"] <- pareto_sample_size
  }
}

# Print the results
print(pareto_results)
```

```
##    epsilon gamma sample_size
## 1     0.01  0.95         151
## 2     0.01  0.96         162
## 3     0.01  0.97         177
## 4     0.01  0.98         197
## 5     0.01  0.99         232
## 6     0.02  0.95          76
## 7     0.02  0.96          82
## 8     0.02  0.97          89
## 9     0.02  0.98          99
## 10    0.02  0.99         117
## 11    0.03  0.95          51
## 12    0.03  0.96          55
## 13    0.03  0.97          60
## 14    0.03  0.98          67
## 15    0.03  0.99          78
## 16    0.04  0.95          39
## 17    0.04  0.96          42
## 18    0.04  0.97          45
## 19    0.04  0.98          50
## 20    0.04  0.99          59
## 21    0.05  0.95          31
## 22    0.05  0.96          33
## 23    0.05  0.97          36
## 24    0.05  0.98          41
## 25    0.05  0.99          48
## 26    0.06  0.95           26
## 27    0.06  0.96           28
## 28    0.06  0.97           31
## 29    0.06  0.98           34
## 30    0.06  0.99           40
## 31    0.07  0.95           23
## 32    0.07  0.96           24
## 33    0.07  0.97           26
## 34    0.07  0.98           29
## 35    0.07  0.99           35
## 36    0.08  0.95           20
## 37    0.08  0.96           21
## 38    0.08  0.97           23
## 39    0.08  0.98           26
## 40    0.08  0.99           30
## 41    0.09  0.95           18
## 42    0.09  0.96           19
## 43    0.09  0.97           21
## 44    0.09  0.98           23
## 45    0.09  0.99           27
## 46    0.10  0.95           16
## 47    0.10  0.96           17
## 48    0.10  0.97           19
## 49    0.10  0.98           21
## 50    0.10  0.99           25
```

```
# Plot the results
library(ggplot2)
ggplot(pareto_results, aes(x = epsilon, y = sample_size, color = as.factor(gamma))) +
  geom_line() +
  labs(title = "Sample Size Required for Different Epsilon and Gamma",
       x = "Epsilon",
       y = "Sample Size",
       color = "Gamma") +
  theme_minimal()
```



The results show that the required sample size for estimating β in a Pareto distribution depends on two parameters:

- ϵ (the acceptable overestimation threshold)
- γ (the confidence level)

Effects of ϵ and γ

1. **Larger ϵ (more tolerance for overestimation) leads to a smaller sample size.**
   - Example: When $\epsilon = 0.01$, the required sample size is 151 for $\gamma = 0.95$, but when $\epsilon = 0.08$, it drops to 20.
2. **Higher γ (greater confidence requirement) leads to a larger sample size.**
   - Example: For $\epsilon = 0.02$, increasing γ from 0.95 to 0.99 raises the required sample size from 76 to 117.

Key Takeaways

- **Higher precision** (small ϵ) requires more data.
- **Stronger confidence** (high γ) increases the sample size.

Problem 8

With clear details and mathematics, describe how problems 1 and 5 are related.

[sol] Problems 1 and 5 both focus on determining the required sample size for estimating a population mean with a specified level of certainty, but they differ in their approach.

Problem 1 deals with constructing a confidence interval, ensuring that the true mean μ lies within a margin of error ϵ with a confidence level $100(1-\alpha)$. Given that the sample mean follows $\bar{X} \sim N(\mu, \sigma^2/n)$, the margin of error is given by:

$$\epsilon = Z_{\alpha/2} \times \frac{\sigma}{\sqrt{n}}$$

Solving for n gives the required sample size:

$$n = \left(\frac{Z_{\alpha/2}\sigma}{\epsilon}\right)^2.$$

On the other hand, **Problem 5** focuses on ensuring that the probability of the sample mean staying within $\epsilon\sigma$ of the true mean is at least γ. This translates to the probability condition:

$$P\left(-\epsilon \leq \frac{\bar{X}-\mu}{\sigma/\sqrt{n}} \leq \epsilon\right) \geq \gamma.$$

Using the cumulative distribution function Φ of the standard normal distribution, this simplifies to:

$$\Phi(\epsilon\sqrt{n}) - \Phi(-\epsilon\sqrt{n}) \geq \gamma.$$

By symmetry, this results in:

$$2\Phi(\epsilon\sqrt{n}) - 1 \geq \gamma.$$

Solving for n, we obtain:

$$n \geq \left(\frac{z_{(1+\gamma)/2}}{\epsilon}\right)^2.$$

Both problems use normal approximations and yield a quadratic relationship between sample size and the critical value, but Problem 1 is centered on **interval estimation**, while Problem 5 guarantees **probability-based proximity** to μ. The key difference lies in the choice of critical values: **Problem 1 uses $Z_{\alpha/2}$ (two-tailed confidence level), while Problem 5 uses $Z_{(1+\gamma)/2}$ (central limit theorem probability bound)**.

In practical applications, selecting between these approaches depends on whether one requires a **confidence interval interpretation** or a **probability guarantee for deviation from the true mean**.

Problem 9

With clear details and mathematics, show the derivation behind the sample size in problem 3. You may start with parts of what was discussed in lecture, but the steps must be shown clearly.

[sol]

Step 1: Defining the Hypothesis Test

We consider the two-sided hypothesis test:

$$H_0 : \mu = \mu_0 \quad \text{(Null Hypothesis)}$$

$$H_1 : \mu \neq \mu_0 \quad \text{(Alternative Hypothesis)}$$

where:

- μ_0 is a known population mean under the null hypothesis.
- μ_1 is the true population mean under the alternative hypothesis.
- σ is the population standard deviation.

Step 2: Test Statistic and Sampling Distribution

For a sample of size n, the sample mean $\bar{X}$ follows a normal distribution:

$$\bar{X} \sim N\left(\mu, \frac{\sigma^2}{n}\right).$$

Under H_0, the standardized test statistic follows:

$$Z = \frac{\bar{X} - \mu_0}{\sigma/\sqrt{n}} \sim N(0, 1).$$

The rejection region for a two-tailed test is:

$$Z \leq -Z_{\alpha/2} \quad \text{or} \quad Z \geq Z_{\alpha/2}.$$

where $Z_{\alpha/2}$ is the critical value from the standard normal distribution.

Step 3: Defining Type I and Type II Errors

Type I Error (α): Occurs when H_0 is rejected when it is true:

$$P\left(Z \leq -Z_{\alpha/2}\right) + P\left(Z \geq Z_{\alpha/2}\right) = \alpha.$$

Type II Error (β): Occurs when we fail to reject H_0 when $\mu = \mu_1$. The probability of correctly rejecting H_0 is:

$$P\left(\text{Reject } H_0 \mid H_1 \text{ is true}\right) = 1 - \beta.$$

Under H_1, the sample mean follows:

$$\bar{X} \sim N\left(\mu_1, \frac{\sigma^2}{n}\right).$$

The standardized test statistic under H_1 is:

$$Z = \frac{\bar{X} - \mu_0}{\sigma/\sqrt{n}} \sim N\left(\frac{\mu_1 - \mu_0}{\sigma/\sqrt{n}}, 1\right).$$

For the test to have **power** $1 - \beta$, the test statistic must fall in the rejection region:

$$P\left(Z \leq -Z_{\alpha/2} + \frac{\mu_1 - \mu_0}{\sigma/\sqrt{n}}\right) + P\left(Z \geq Z_{\alpha/2} + \frac{\mu_1 - \mu_0}{\sigma/\sqrt{n}}\right) = 1 - \beta.$$

Since we want to minimize β, we set:

$$Z_{1-\beta} = Z_{\alpha/2} + \frac{\mu_1 - \mu_0}{\sigma/\sqrt{n}}.$$

Rearranging:

$$\frac{\mu_1 - \mu_0}{\sigma/\sqrt{n}} = Z_{1-\beta} - Z_{\alpha/2}.$$

$$\frac{\mu_1 - \mu_0}{\sigma} = \sqrt{n} \times (Z_{1-\beta} - Z_{\alpha/2}).$$

Step 4: Expressing in Terms of Effect Size

The **effect size (ES)** is defined as:

$$ES = \frac{|\mu_1 - \mu_0|}{\sigma}.$$

Substituting this into our equation:

$$ES = \sqrt{n} \times (Z_{\alpha/2} + Z_{1-\beta}).$$

Solving for n:

$$n = \left(\frac{Z_{\alpha/2} + Z_{1-\beta}}{ES}\right)^2.$$

This is the required **sample size formula**:

$$n = \left(\frac{Z_{\alpha/2} + Z_{1-\beta}}{ES}\right)^2.$$