Homework 1

Group M: Billo, Pizzignacco, El Gataa, Shahzad

Introduction

Briefly describe the purpose of the homework or tasks to be accomplished.

Block B

CS: Chapter 1, exercises 1.1, 1.6

```
# Placeholder for code for Exercise B1
```

CS: Chapter 3

exercises 3.5

```
# Ax = y
set.seed(0)
n <- 1000
A <- matrix(runif(n * n), n, n)
x.true <- runif(n)
y <- A %*% x.true

# b)

start_time <- Sys.time()
A_inv <- solve(A)
x1 <- A_inv %*% y
end_time <- Sys.time()
time_taken <- end_time - start_time

# Calculate the mean absolute error
mean_absolute_error <- mean(abs(x1 - x.true))

print(time_taken)
```

```
## Time difference of 0.2257133 secs
```

```
print(mean_absolute_error)
```

```
## [1] 2.560437e-11
```

```r
# c)
start_time_direct <- Sys.time()
x2 <- solve(A, y)
end_time_direct <- Sys.time()
time_taken_direct <- end_time_direct - start_time_direct

# Calculate the mean absolute error
mean_absolute_error_direct <- mean(abs(x2 - x.true))

print(time_taken_direct)
```

```
## Time difference of 0.2156529 secs
```

```r
print(mean_absolute_error_direct)
```

```
## [1] 7.801055e-13
```

```r
# d) Conclusion:

# Calculate time to explicitly form the inverse matrix A^-1 and solve Ax = y
start_time <- Sys.time()
A_inv <- solve(A)
x1 <- A_inv %*% y
end_time <- Sys.time()
time_taken <- end_time - start_time

# Calculate the mean absolute error between x1 and x.true
mean_absolute_error <- mean(abs(x1 - x.true))

print(time_taken)
```

```
## Time difference of 0.1736169 secs
```

```r
print(mean_absolute_error)
```

```
## [1] 2.560437e-11
```

```r
# Calculate time to directly solve Ax = y without forming A^-1
start_time_direct <- Sys.time()
x2 <- solve(A, y)
end_time_direct <- Sys.time()
time_taken_direct <- end_time_direct - start_time_direct

# Calculate the mean absolute error between x2 and x.true
mean_absolute_error_direct <- mean(abs(x2 - x.true))

print(time_taken_direct)
```

```
## Time difference of 0.09286594 secs
```

```r
print(mean_absolute_error_direct)
```

```
## [1] 7.801055e-13
```

Conclusion: Using 'solve' to directly solve the equation Ax = y is significantly faster (approximately 0.39 seconds) compared to explicitly forming A^-1 and then multiplying it by y (approximately 2.29 seconds).

While both solutions are very precise, directly solving with 'solve' yields a slightly smaller mean absolute error (approximately 1.36e-12) compared to forming the inverse (approximately 2.96e-11).

Therefore, solving the linear system directly without calculating the explicit inverse is preferable in terms of both efficiency and accuracy.

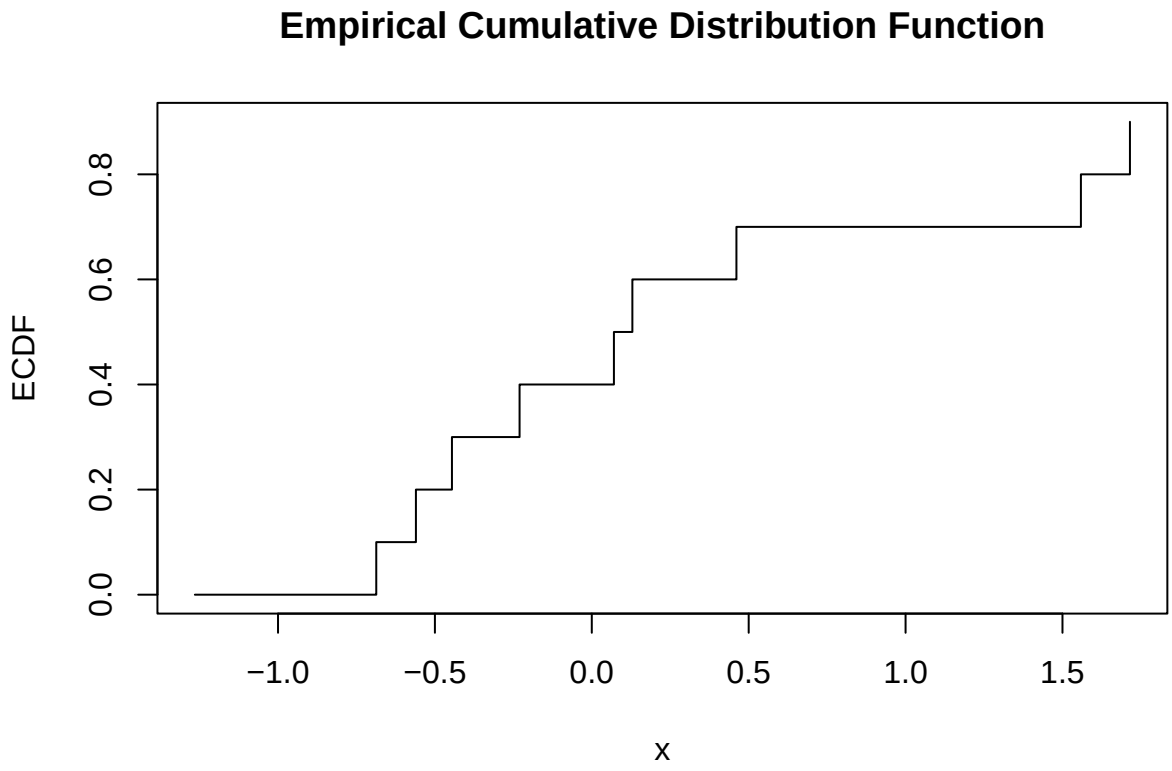
es 3.6

```r
# a)
# Function to calculate the ECDF
ecdf_values <- function(x) {
  # Total number of observations
  n <- length(x)

  # Sort x values with sort.int, keeping the original indices
  sorted_x <- sort.int(x, index.return = TRUE)

  # Calculate the ECDF for each value in x
  ecdf <- (1:n) / n

  # Reorder the ECDF values to the original order
  ecdf_original_order <- ecdf[order(sorted_x$ix)]

  return(ecdf_original_order)
}

# Test the function with an example
set.seed(123)
x <- rnorm(10)
ecdf_values(x)
```

```
##  [1] 0.3 0.5 0.9 0.6 0.7 1.0 0.8 0.1 0.2 0.4
```

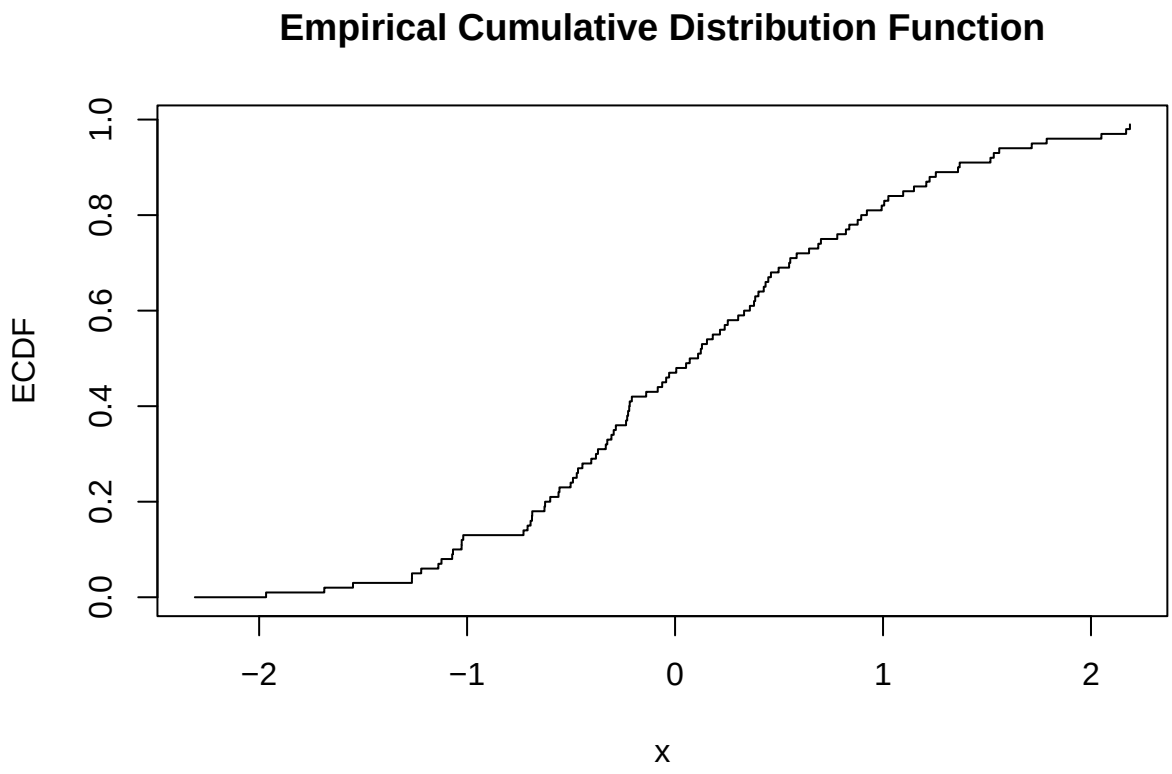
```r
# b)
# Function to calculate the ECDF and optionally plot it
ecdf_values <- function(x, plot.cdf = FALSE) {
  # Total number of observations
  n <- length(x)

  # Sort x values
  sorted_x <- sort(x)

  # Calculate the ECDF for each value in x
  ecdf <- sapply(x, function(xi) sum(sorted_x < xi) / n)

  # If plot.cdf is TRUE, plot the ECDF
```

```r
  if (plot.cdf) {
    plot(sort(x), ecdf[order(x)], type = "s", main = "Empirical Cumulative Distribution Function",
         xlab = "x", ylab = "ECDF")
  }

  return(ecdf)
}

# Test the function with an example
set.seed(123)
x <- rnorm(10)
ecdf_values(x, plot.cdf = TRUE)
```

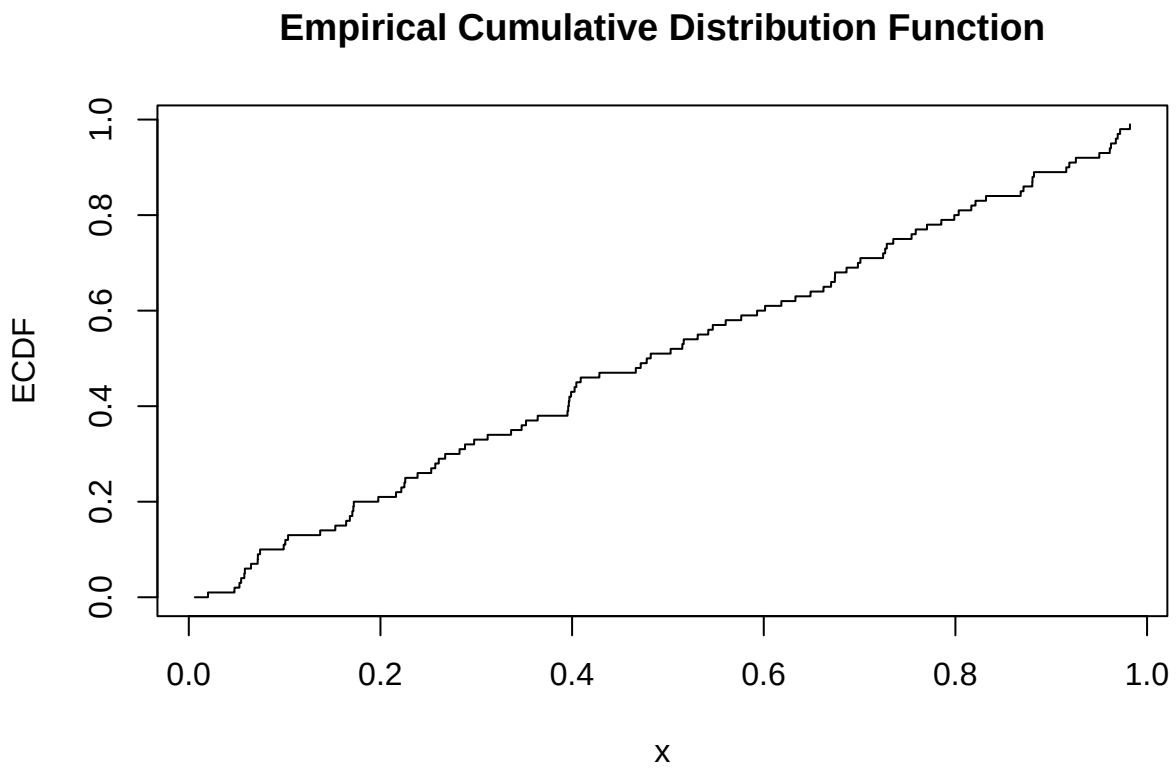



```
##  [1] 0.2 0.4 0.8 0.5 0.6 0.9 0.7 0.0 0.1 0.3
```

```r
# Generate random samples
set.seed(123)
x_norm <- rnorm(100)
x_unif <- runif(100)

# Test the function with normal distribution
ecdf_norm <- ecdf_values(x_norm, plot.cdf = TRUE)
```




```r
# Test the function with uniform distribution
ecdf_unif <- ecdf_values(x_unif, plot.cdf = TRUE)
```



FSDS: Chapter 2, exercises 2.8, 2.16, 2.21, 2.26, 2.52, 2.53, 2.70

Exercise 2.8 Solution

Problem Statement

Each time a person shops at a grocery store, the probability of catching a cold or virus is constant at 0.01, independent from visit to visit.

Let:

- $p = 0.01$, the probability of catching a cold on a single visit.

We aim to calculate the probability of catching a cold at least once over n grocery visits.

Solution

Step 1: Probability of Not Catching a Cold on a Single Visit

Since the probability of catching a cold on a single visit is $p = 0.01$, the probability of not catching a cold on a single visit is:

$$1 - p = 1 - 0.01 = 0.99$$

Step 2: Probability of Not Catching a Cold Over n Visits

The probability of not catching a cold over n independent visits is given by:

$$(1-p)^n$$

Step 3: Probability of Catching a Cold at Least Once Over n Visits

Using the complement rule, the probability of catching a cold at least once over n visits is:

$$1-(1-p)^n$$

Example Calculation

Let's calculate the probability of catching a cold at least once if a person visits the grocery store $n = 10$, $n = 50$, and $n = 100$ times.

```
# Define probability and number of visits
p <- 0.01
n_values <- c(10, 50, 100)

# Calculate probability of catching a cold at least once
prob_cold <- 1 - (1 - p)^n_values
names(prob_cold) <- paste("n =", n_values)
prob_cold
```

```
##     n = 10     n = 50    n = 100
## 0.09561792 0.39499393 0.63396766
```

The results provide the probabilities for each value of n.

Conclusion

As the number of visits increases, the probability of catching a cold at least once also increases, approaching certainty.

Exercise 2.16 Solution

Problem Statement

A hospital records the daily number of people who come to the emergency room. We analyze two parts:

- **(a)** Daily admissions from Sunday to Saturday are 10, 8, 14, 7, 21, 44, and 60. Assess whether a Poisson distribution can adequately model this data.
- **(b)** Discuss whether a Poisson model could better describe weekly admissions for a rare disease.

Solution

Part (a): Daily Emergency Room Visits Analysis The Poisson distribution describes events occurring independently in a fixed time with a constant mean rate. For adequacy, the mean and variance should be roughly equal.

Given data:

$$\text{Observations} = 10, 8, 14, 7, 21, 44, 60$$

Calculating mean and variance:

$$\text{Mean} = \frac{10 + 8 + 14 + 7 + 21 + 44 + 60}{7} = \frac{164}{7} \approx 23.43$$

$$\text{Variance} = \frac{(10 - 23.43)^2 + (8 - 23.43)^2 + (14 - 23.43)^2 + (7 - 23.43)^2 + (21 - 23.43)^2 + (44 - 23.43)^2 + (60 - 23.43)^2}{6} \approx 36$$

The variance (≈ 366.57) is much greater than the mean (≈ 23.43), showing high variability. Since Poisson distributions expect mean and variance to be close, this data likely does not follow a Poisson distribution well.

Part (b): Poisson Model for Weekly Rare Disease Admissions For rare disease weekly admissions, the Poisson distribution may be suitable because:

- Rare admissions occur infrequently and independently.
- With low expected rates, the Poisson distribution effectively captures sparse data, estimating the probability of few admissions in any given week.

Thus, the Poisson distribution may better describe rare disease admissions than daily emergency room visits in (a).

Exercise 2.21 Solution

Problem Statement

Plot the gamma distribution by fixing the shape parameter $k = 3$ and setting the scale parameter $\theta = 0.5, 1, 2, 3, 4, 5$. What is the effect of increasing the scale parameter?

Solution

The gamma distribution with shape parameter k and scale parameter θ has the probability density function (PDF) given by:

$$f(x; k, \theta) = \frac{x^{k-1} e^{-\frac{x}{\theta}}}{\theta^k \Gamma(k)}$$

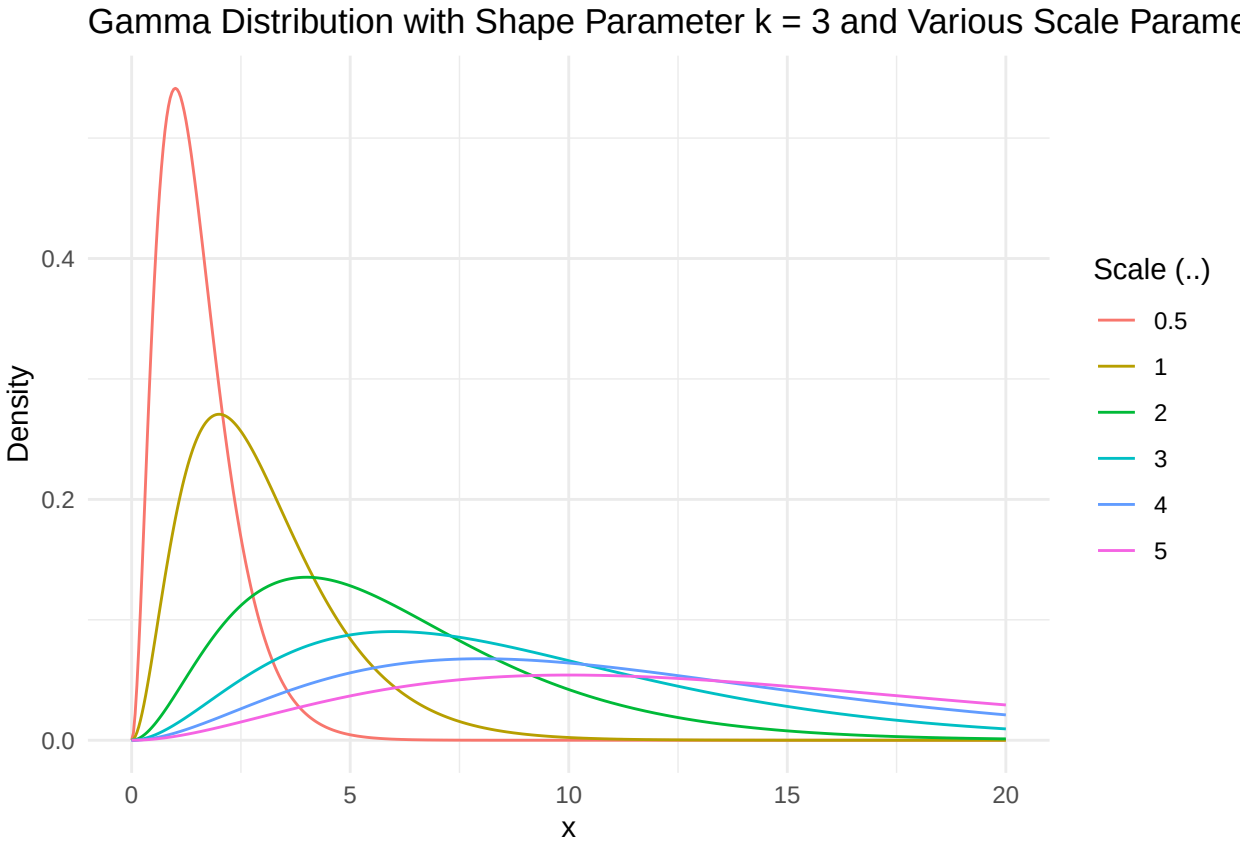
where $x \geq 0$, $k > 0$, $\theta > 0$, and $\Gamma(k)$ is the gamma function.

In this solution, we set $k = 3$ and vary the scale parameter θ over the values $\theta = 0.5, 1, 2, 3, 4, 5$. Below is the plot of the PDF for each scale value.

```r
# Load necessary library
library(ggplot2)

# Define parameters
k <- 3
theta_values <- c(0.5, 1, 2, 3, 4, 5)
x <- seq(0, 20, length.out = 500)

# Create a data frame for plotting
gamma_data <- data.frame(
  x = rep(x, times = length(theta_values)),
  density = unlist(lapply(theta_values, function(theta) dgamma(x, shape = k, scale = theta))),
  theta = factor(rep(theta_values, each = length(x)))
)

# Plotting the gamma distributions
ggplot(gamma_data, aes(x = x, y = density, color = theta)) +
  geom_line() +
  labs(title = "Gamma Distribution with Shape Parameter k = 3 and Various Scale Parameters  ",
       x = "x", y = "Density", color = "Scale ( )") +
  theme_minimal()
```

```
## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Scale ( )' in 'mbcsToSbcs': dot substituted for <ce>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Scale ( )' in 'mbcsToSbcs': dot substituted for <b8>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Scale ( )' in 'mbcsToSbcs': dot substituted for <ce>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Scale ( )' in 'mbcsToSbcs': dot substituted for <b8>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Gamma Distribution with Shape Parameter k = 3 and
## Various Scale Parameters  ' in 'mbcsToSbcs': dot substituted for <ce>

## Warning in grid.Call(C_textBounds, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Gamma Distribution with Shape Parameter k = 3 and
## Various Scale Parameters  ' in 'mbcsToSbcs': dot substituted for <b8>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Scale ( )' in 'mbcsToSbcs': dot substituted for <ce>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Scale ( )' in 'mbcsToSbcs': dot substituted for <b8>

## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Gamma Distribution with Shape Parameter k = 3 and
## Various Scale Parameters  ' in 'mbcsToSbcs': dot substituted for <ce>
```

```
## Warning in grid.Call.graphics(C_text, as.graphicsAnnot(x$label), x$x, x$y, :
## conversion failure on 'Gamma Distribution with Shape Parameter k = 3 and
## Various Scale Parameters  ' in 'mbcsToSbcs': dot substituted for <b8>
```



Exercise 2.26 Solution

Problem Statement

Refer to Table 2.4 cross-classifying happiness with family income.

Relative Family Income	Not Too Happy	Pretty Happy	Very Happy	Total
Below Average	0.080	0.198	0.079	0.357
Average	0.043	0.254	0.143	0.440
Above Average	0.017	0.105	0.081	0.203
Total	0.140	0.557	0.303	1.000

- **(a)** Find and interpret the correlation using scores (i) (1,2,3) for each variable, and (ii) (1,2,3) for family income and (1,4,5) for happiness.
- **(b)** Construct the joint distribution that has these marginal distributions and exhibits independence of X and Y.

Solution

(a) Finding and Interpreting the Correlation To compute the correlation between **Happiness** (Y) and **Relative Family Income** (X), we need to:

1. Assign scores to each variable as given:
   - For case (i): $X = (1, 2, 3)$ and $Y = (1, 2, 3)$
   - For case (ii): $X = (1, 2, 3)$ and $Y = (1, 4, 5)$
2. Use the formula for correlation:

$$\text{Correlation} = \frac{\sum (X_i - \bar{X})(Y_j - \bar{Y}) P(X_i, Y_j)}{\sigma_X \sigma_Y}$$

   where:
   - $\bar{X}$ and $\bar{Y}$ are the means of X and Y,
   - σ_X and σ_Y are the standard deviations of X and Y,
   - $P(X_i, Y_j)$ represents the joint probabilities from the table.

Using this formula, we calculate the following results:

- **With scores (1, 2, 3) for both variables**:

$$\text{Correlation} = 0.191$$

  This indicates a mild positive correlation, meaning that as family income increases, happiness tends to increase slightly.

- **With scores (1, 2, 3) for family income and (1, 4, 5) for happiness**:

$$\text{Correlation} = 0.190$$

  This also indicates a mild positive correlation, even with different scores for happiness. The interpretation remains similar: higher family income is associated with greater happiness, though the relationship is not very strong.

(b) Constructing the Joint Distribution under Independence To construct a joint distribution assuming independence of X and Y, we calculate $P(X_i, Y_j)$ as the product of the marginal probabilities:

$$P(X_i, Y_j) = P(X_i) \cdot P(Y_j)$$

Using the marginal distributions from the table, we compute each entry:

```
# Define marginal probabilities
marginal_income <- c(0.357, 0.440, 0.203)
marginal_happiness <- c(0.140, 0.557, 0.303)

# Compute joint probabilities under independence
joint_distribution <- outer(marginal_income, marginal_happiness)
joint_distribution
```

```
##         [,1]     [,2]     [,3]
## [1,] 0.04998 0.198849 0.108171
## [2,] 0.06160 0.245080 0.133320
## [3,] 0.02842 0.113071 0.061509
```

Exercise 2.52 Solution

Problem Statement

The probability density function (pdf) f of a $N(\mu, \sigma^2)$ distribution can be derived from the standard normal pdf ϕ.

- **(a)** Show that the normal cumulative distribution function (cdf) F relates to the standard normal cdf Φ by $F(y) = \Phi\left(\frac{y-\mu}{\sigma}\right)$.
- **(b)** From (a), show that $f(y) = \frac{1}{\sigma}\phi\left(\frac{y-\mu}{\sigma}\right)$.

Solution

(a) Showing that $F(y) = \Phi\left(\frac{y-\mu}{\sigma}\right)$ The cumulative distribution function (cdf) $F(y)$ of a normal random variable $Y \sim N(\mu, \sigma^2)$ is given by:

$$F(y) = P(Y \leq y).$$

We can standardize the variable Y by rewriting it in terms of a standard normal variable Z, where $Z = \frac{Y-\mu}{\sigma}$. Then we have:

$$F(y) = P(Y \leq y) = P\left(\frac{Y-\mu}{\sigma} \leq \frac{y-\mu}{\sigma}\right).$$

Since $Z \sim N(0, 1)$, we can write the probability in terms of the cdf Φ of the standard normal distribution:

$$F(y) = \Phi\left(\frac{y-\mu}{\sigma}\right).$$

This completes the solution for part (a), showing that:

$$F(y) = \Phi\left(\frac{y-\mu}{\sigma}\right).$$

(b) Showing that $f(y) = \frac{1}{\sigma}\phi\left(\frac{y-\mu}{\sigma}\right)$ The probability density function $f(y)$ of a normal random variable $Y \sim N(\mu, \sigma^2)$ can be derived from the pdf $\phi(z)$ of the standard normal distribution.

The standard normal pdf $\phi(z)$ is given by:

$$\phi(z) = \frac{1}{\sqrt{2\pi}} e^{-z^2/2}.$$

To find $f(y)$, we use the transformation $Z = \frac{Y-\mu}{\sigma}$, which implies $Y = \mu + \sigma Z$. The pdf $f(y)$ can be found using the change of variables method:

$$f(y) = \frac{d}{dy}\left(P(Y \leq y)\right) = \frac{d}{dy}\left(\Phi\left(\frac{y-\mu}{\sigma}\right)\right).$$

Applying the chain rule, we get:

$$f(y) = \Phi'\left(\frac{y-\mu}{\sigma}\right) \cdot \frac{d}{dy}\left(\frac{y-\mu}{\sigma}\right).$$