**Exercise 4.1**

1. Store the following vector of 15 values as an object in your workspace: c(6,9,7,3,6,7,9,6,3,6,6,7,1,9,1). Identify the following elements:

i. Those equal to 6

ii. Those greater than or equal to 6

iii. Those less than 6 + 2

iv. Those not equal to 6

1. Create a new vector from the one used in (a) by deleting its first three elements. With this new vector, fill a 2 × 2 × 3 array. Examine the array for the following entries:

i. Those less than or equal to 6 divided by 2, plus 4

ii. Those less than or equal to 6 divided by 2, plus 4, after increasing every element in the array by 2

1. Confirm the specific locations of elements equal to 0 in the 10 × 10 identity matrix I10 (see Section 3.3).
2. Check whether any of the values of the logical arrays created in (b) are TRUE. If they are, check whether they are all TRUE.
3. By extracting the diagonal elements of the logical matrix created in (c), use any to confirm there are no TRUE entries.

**Solution**

**Part (a) – Store and Filter Data**

Part a can be completed by using the following code:



Running the code gives the following results:



**Part (b) – Matrix**

Deleting elements from a vector can be done in a number of ways, I choose to use slicing and store the result into a new variable, as shown below:



This new vector can then be converted into a 2x2x3 matrix by modifying the dim variable of the vector:



We can then use the same syntax as in part a to show which elements are greater than another value:



Part ii can be done by simply adding 2 to the stored matrix, and then we run the same comparison as before:





**Part (c) – Identity Matrix**

An identity matrix can be constructed via the diag(size) function displayed below:



Values equal to 0 can be identified using the same comparison operators used previously:



**Part (d) – Any/All**

Checking whether any value or all values in a Boolean array/matrix are true can be done utilizing the helpfully named functions any() and all() as shown below for subpart i:



And for subpart ii:



**Part (e) – Diagonal Extraction**

Extracting the diagonal from the original identity matrix can be done by separating out the lower and upper triangles from the matrix using the lower.tri() and upper.tri() functions. Then, we use the any() function on both of these to confirm that there are no true entries left:



**Exercise 16.2**

Every Saturday, at the same time, an individual stands by the side of a road and tallies the number of cars going by within a 120-minute window. Based on previous knowledge, she believes that the mean number of cars going by during this time is exactly 107. Let X represent the appropriate Poisson random variable of the number of cars passing her position in each Saturday session.

1. What is the probability that more than 100 cars pass her on any given Saturday?
2. Determine the probability that no cars pass.
3. Plot the relevant Poisson mass function over the values in 60 ≤ x ≤ 150.

**Solution**

**Part (a) – Any Given Saturday**

The probability of more than 100 cars passing can be found using the Poisson distribution and subtract the probability that less or equal 100 cars pass through from 1. This done using the ppois function as shown below:



**Part (b) – Zero Cars**

Finding the probability for a single value is done using the same distribution, but this time we will utilize the dpois function as shown below:



**Part (c) – 150 to 60 Cars**

This can be done by subtracting the value for 60 cars from the value for 150 cars, once again using the ppois function as shown below:



**Summary**

This assignment covered one exercise which went over simple data manipulation and comparison in R and another which involved utilizing both the cdf and pmf of the Poisson distribution to discover the probabilities associated with specific events.