R Portfolio

Knit this R markdown file, read and understand the html file, then edit and knit the R markdown file to showcase your evidence. Save the html file for submission.

Please ensure that the evidence covers all of the following learning objectives:

LO1: Can apply methods concerning estimation of parameters in standard statistical models; in particular the method of moments and the maximum likelihood method.

LO2: Can perform statistical hypotheses tests using data from studies (such as t and F-tests, comparison of models and parameter values).

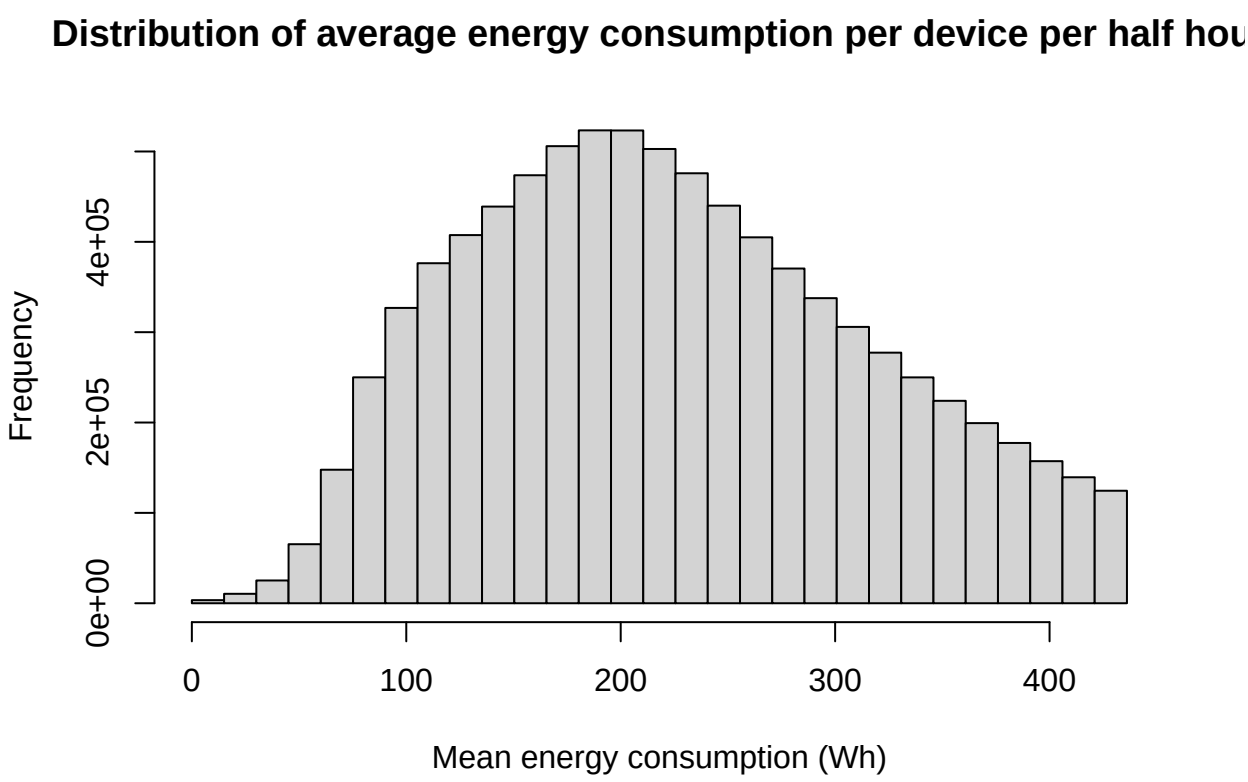
LO3: Can apply methods for interval estimation; in particular, exact and approximate confidence intervals based on asymptotic theory.

LO4: Can apply methods for analysing categorical data.

Evidence 1 - Distribution of customer energy consumption

Learning objective(s) assessed: **LO1 | LO3**

```
# For each row, calculate the power consumed per device
dat1$import_per_device <- dat1$Total_consumption_active_import / dat1$aggregated_device_count_Active

# Filter the data set to remove erroneous data values
filt <- remove_outliers(dat1, dat1$import_per_device)

# Plot histogram of the values observed in power consumed per device
hist(filt$import_per_device,
     breaks = seq(min(filt$import_per_device), max(filt$import_per_device), length.out = 30),
     main = "Distribution of average energy consumption per device per half hour",
     xlab = "Mean energy consumption (Wh)")
```



By dividing the total energy consumption by the number of active devices for each row we can find the the average (mean) energy consumption for each device for each half-hour interval. After removing outliers (greater than 1.5x the interquartile range) and plotting a histogram for the distribution of these values we can see a fairly normal graph.

To find good estimates for the mean and standard deviation of this distribution we can use the maximum likelihood method:

```
# Fit normal distribution to the filtered data (find the maximum likelihood estimators for mean and sta
fit_norm <- fitdistr(filt$import_per_device, densfun = "normal")
print(fit_norm)
```

```
##        mean             sd
##   223.60121613    92.20016834
##  (  0.03168924) (  0.02240768)
```

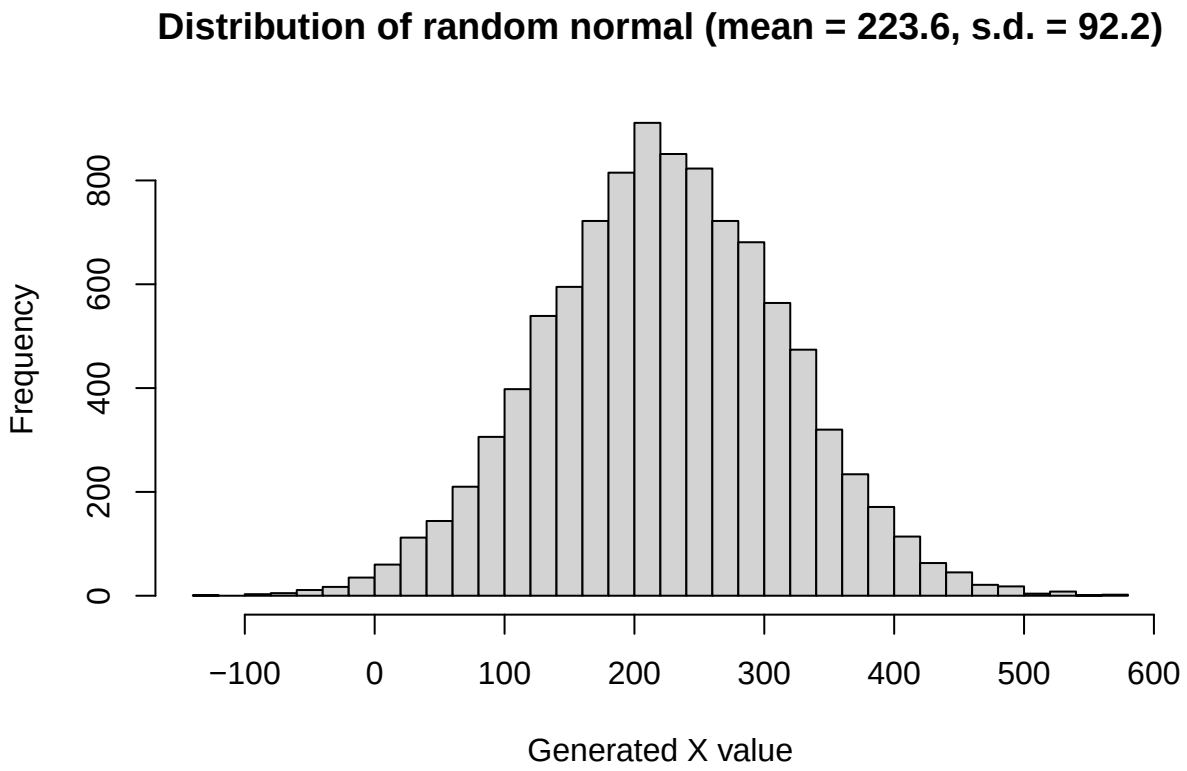
The maximum likelihood estimators for the mean ($\hat{\mu}$) and the standard deviation ($\hat{\sigma}$) are shown above with the standard error shown below them. A 95% confidence interval can be calculated for each estimator to help quantify the uncertainty of the estimations for the population parameters derived from the sample:

```
# Finds the 95% confidence intervals for the MLEs of both the mean and standard deviation
conf_intervals <- confint(fit_norm, level=0.95)
print(conf_intervals)
```

```
##          2.5 %    97.5 %
## mean 223.53911 223.66333
## sd    92.15625  92.24409
```

By inserting these estimates into a random normal distribution generator we can compare the estimated shape of the population with the shape we observe in the sample.

```
# Plot a random normal distribution using the MLEs to compare against actual distribution
test_data <- rnorm(10000,mean=fit_norm$estimate['mean'],sd=fit_norm$estimate['sd'])
hist(test_data, breaks = 30,
     main = "Distribution of random normal (mean = 223.6, s.d. = 92.2)",
     xlab = "Generated X value")
```



Evidence 2 - Smart meter uptake in different regions

Learning objective(s) assessed: **LO 2 | LO4**

```
# Grabs the names of all 4 different licence areas
LAs = unique(dat2$Licence_Area)

# Loops through each row in the data set and finds the number of smart and non-smart meters for each re
tab_dat = c()
for (LA in LAs) {
  filtered <- dat2[dat2$Licence_Area == LA,]

  S1 = sum(filtered$SMETS1)
  S2 = sum(filtered$SMETS2)
  nonS = sum(filtered$NON_SMART)
```

```
  row_dat = c(S1, S2, nonS)

  tab_dat = append(tab_dat, row_dat)
}

# This data is then stored in a 3x4 matrix (contingency table)
tab <- matrix(tab_dat, nrow=3, ncol=length(LAs), dimnames=list(c("SMETS1","SMETS2","NON SMART"),LAs))

print(tab)
```

```
##           South Wales South West East Midlands West Midlands
## SMETS1         252766     296476        748336        413451
## SMETS2         437953     556289       1249099        659224
## NON SMART      553615     710112       1358099        889501
```

Using this table we can test for the independence of the variables 'Licence Area' and 'Meter Type'.

H_0 : The meter type is independent of which licence area it resides in. *vs.* H_1 : The meter type depends on which licence area the meter resides in.

Under H_0 for large n (which is applicable here as n = 8,124,921),

$-2\log(\Lambda) \sim \chi^2_{(J-1)(K-1)} approximately,$

$$-2\log(\Lambda) = 2\sum_{j=1}^{J}\sum_{k=1}^{K} y_{jk}\log\left(\frac{y_{jk}}{\hat{e}_{jk}}\right)$$

$\hat{e}_{jk} = \frac{y_{j\bullet}y_{\bullet k}}{n}$

```
# Uses a G-Test (with no correction) to test the independence of the licence area and the type of meter
results <- GTest(tab,correct="none")

print(results)
```

```
##
##  Log likelihood ratio (G-test) test of independence without correction
##
## data:  tab
## G = 21012, X-squared df = 6, p-value < 2.2e-16
```

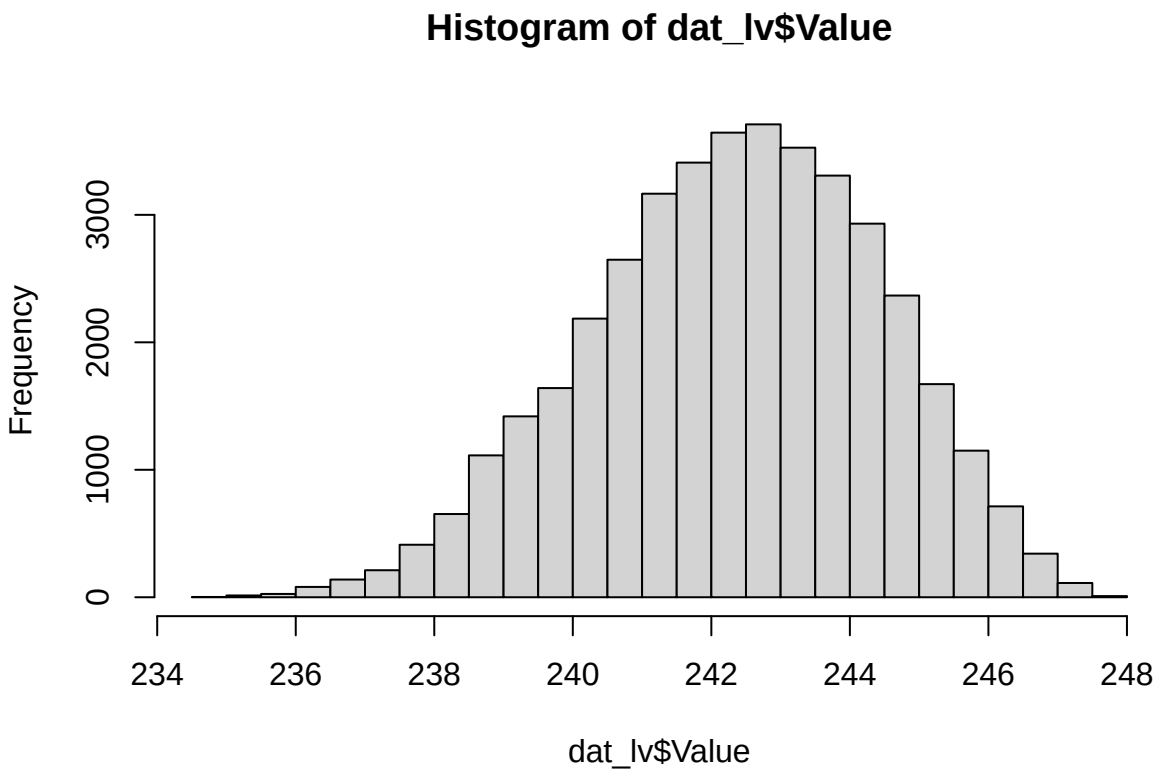
The p-value for this test is <2.2e-16 which is much less than 0.01. This suggests there is overwhelming evidence to reject the null hypothesis (H_0) and we can say the variables are not independent. This suggests that different areas have different levels of uptake of smart meters compared to others.

Evidence 3

Learning objective(s) assessed: **???**

Please add new evidence and subsections as necessary.

```
# Filter to only voltage values
dat_lv <- dat3[dat3$Units == 'V',]

# Plot initial histogram
hist(dat_lv$Value, breaks=30)
```



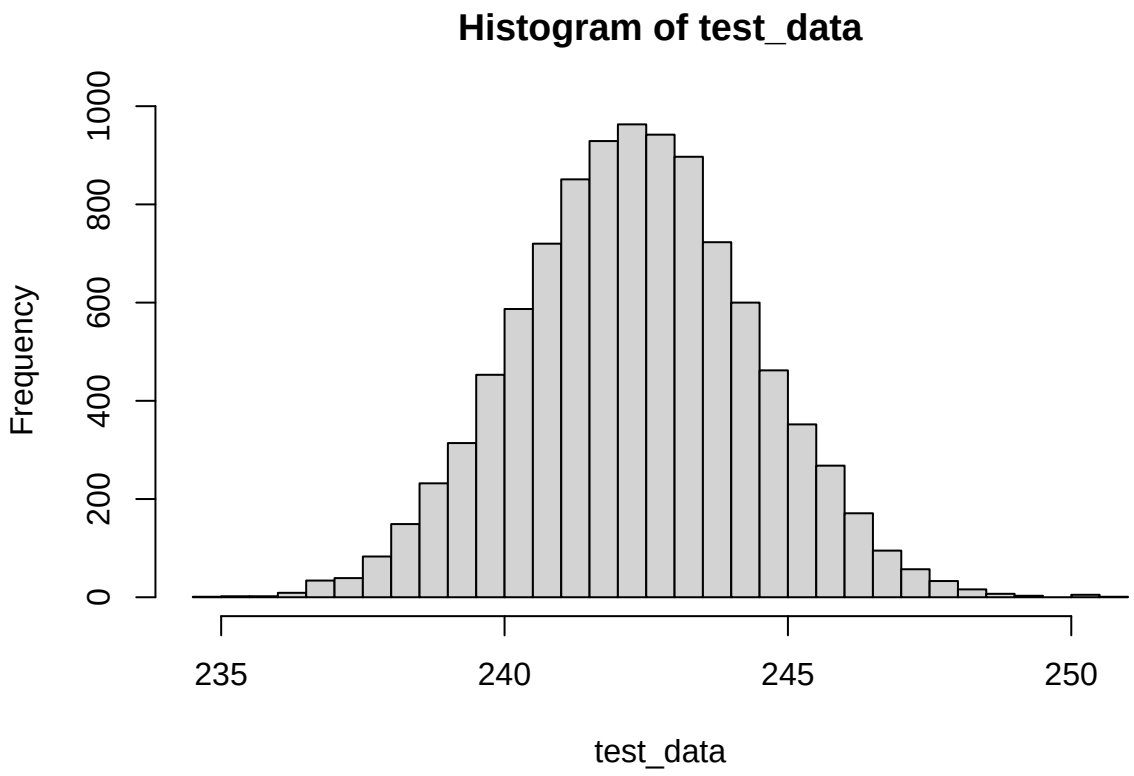
SOMETHING SOMEHTING SOMETHING THIS LOOKS LIKE GAMMA...

```
# Fit gamma distribution to the subset data
fit_gamma <- fitdistr(dat_lv$Value, densfun = "gamma")
print(fit_gamma)
```

```
##       shape           rate
##   1.352729e+04   5.581807e+01
##  (1.233701e+01) (5.096219e-02)
```

TALK ABOUT FITTED SHAPE AND RATE

```
test_data <- rgamma(10000,shape=fit_gamma$estimate['shape'],rate=fit_gamma$estimate['rate'])
hist(test_data, breaks = 30)
```



LOOKS DECENTLY SIMILAR

```
# Calculates the fisher information for shape using trigamma function
shape_trigam <- trigamma(fit_gamma$estimate['shape'])

variance_shape <- 1/shape_trigam

z_score <- 1.96  # Approximate standardised value for 95% conf int

lwrb <- fit_gamma$estimate['shape'] - z_score * sqrt(variance_shape)
uprb <- fit_gamma$estimate['shape'] + z_score * sqrt(variance_shape)

shape_conf_int <- c(lwrb, uprb)

print(c('2.5%', '97.5%'))
```

```
## [1] "2.5%"  "97.5%"
```

```
print(shape_conf_int)
```

```
##    shape    shape
## 13299.34 13755.25
```

```r
rate_fisher <- fit_gamma$estimate['shape'] / (fit_gamma$estimate['rate']^2)

variance_rate <- 1/rate_fisher

lwrb2 <- fit_gamma$estimate['rate'] - z_score * sqrt(variance_rate)
uprb2 <- fit_gamma$estimate['rate'] + z_score * sqrt(variance_rate)

shape_conf_int <- c(lwrb2, uprb2)

print(shape_conf_int)
```

```
##     rate     rate
## 54.87742 56.75871
```

EXPLAIN HERE, NOT EXACT, 95%, ETC

Evidence 4 - Time until next fault

Learning objective(s) assessed: **LO1**

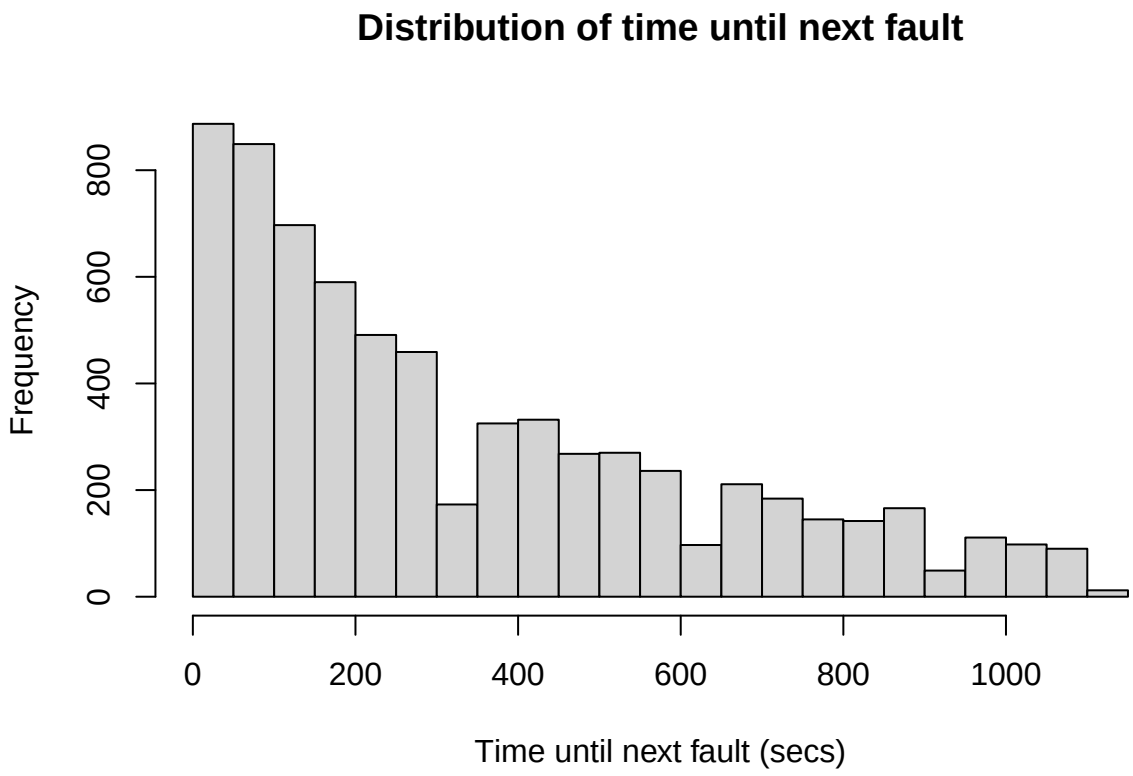
Another important area of the business is fault prediction.

```r
# Aggregate by fault
faults <- dat4 %>%
  group_by(incident_ref) %>%
  summarise(licence_area = first(licence_area), # Should all be the same anyway
            interruption_date = min(interruption_date), # Gets earliest interruption date (will be from
            restoration_date = max(restoration_date), # Gets latest restoration date (will be from last
            voltage = first(voltage), # Should all be the same anyway
            report_ci = max(report_ci)) # Gets the largest value (the max number of customers off simul
```

By using historic fault records, we can calculate the time intervals (in seconds) between faults occurring on our network. Here we could split the data into seperate parts for each licence area or GSP/BSP but to keep things simple we will use the whole network.

```r
# Convert datetime column to POSIXct
faults$interruption_date <- as.POSIXct(faults$interruption_date, format = "%d/%m/%Y %H:%M:%S")

# Order data by datetime column
faults <- faults[order(faults$interruption_date), ]

faults$fault_interval <- NA  # Initialize the new column with NA

for (i in 2:nrow(faults)) {
  # Calculate time difference in minutes between current row and previous row
  interval <- difftime(faults$interruption_date[i], faults$interruption_date[i - 1], units = "secs")

  # Assign the calculated interval to the corresponding row in the new column
  faults$fault_interval[i] <- as.numeric(interval)
}
```

```r
# Remove the initial row (where interval = NA)
faults <- na.omit(faults)

# Remove any outliers from the fault interval column
faults <- remove_outliers(faults, faults$fault_interval)

# Plot distribution of time until
hist(faults$fault_interval, breaks=20, main="Distribution of time until next fault", xlab="Time until ne
```



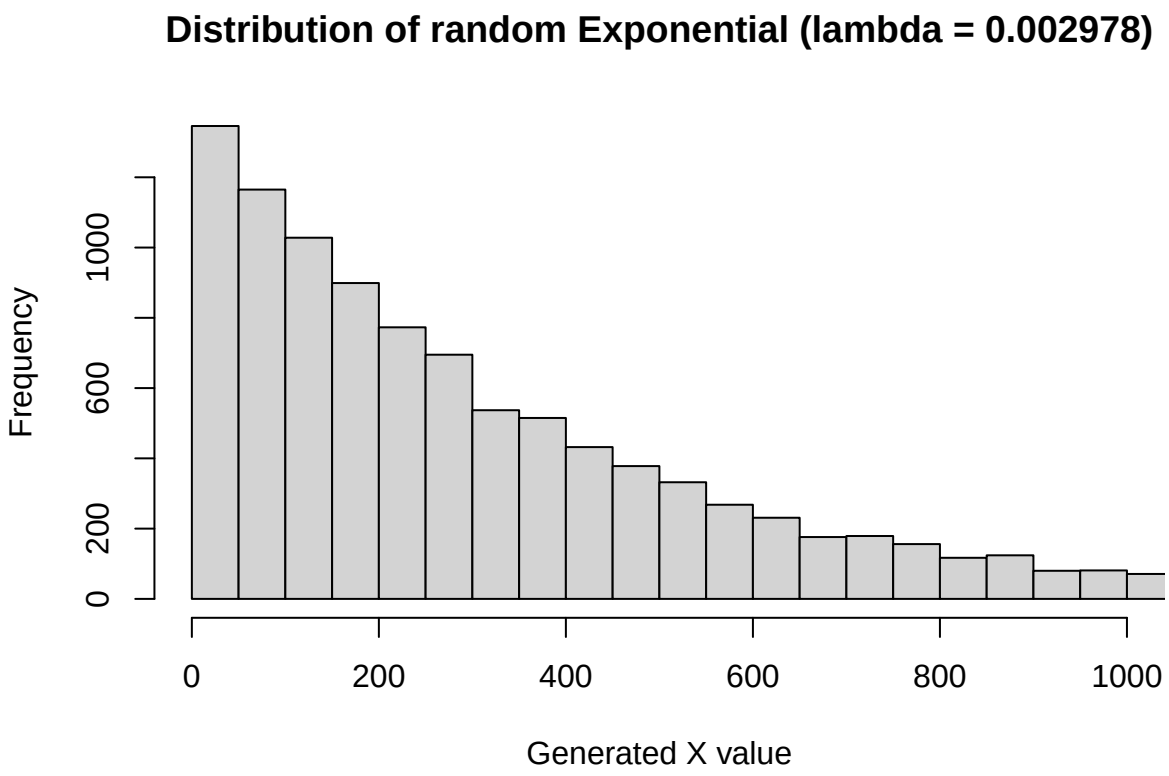
Plotting the distribution of intervals on a histogram we can see a clear exponential decay. With this knowledge, we can now try to fit our data to an exponential distribution to find an estimate for the rate (λ) parameter.

```r
fit_exp <- fitdistr(faults$fault_interval, densfun="exponential")

print(fit_exp)
```

```
##        rate
##   2.977981e-03
##  (3.589753e-05)
```

Once we have our estimate we can check it against a randomly generated exponential distribution with the same rate.

```r
test_dat <- rexp(10000, rate=fit_exp$estimate)

hist(test_dat, xlim = c(0, 1000), breaks = 50, main = "Distribution of random Exponential (lambda = 0.0
```



The shape of the distribution is very similar which indicates the value found is a good estimation for λ. An alternative method for parameter estimation is the 'Method of Moments'. This method simply equates theoretical and sample moments - choosing to ensure the distribution 'matches' the data in the sense of equating the moments. For a single parameter this means solving: $E(X) = \bar{X}$

The expected value of X where X $\sim Exp(\lambda)$ is defined as: $E(X) = \frac{1}{\lambda}$

We can find $\bar{X}$ (sample mean) by summing the time intervals and then dividing the result by the number of intervals in our data set.

```r
sum <- sum(faults$fault_interval)
count <- length(faults$fault_interval)

sample_mean <- mean(faults$fault_interval)

lambda_estimate <- 1/sample_mean
```

$\bar{X} = 2.310962 \times 10^6 \ / \ 6882 = 335.7980238$

Using the equations above we can say

$335.7980238 = \frac{1}{\lambda}$

$\therefore \hat{\lambda} = 1 \ / \ 335.7980238 = 0.002978$

This estimate is coincidentally the same as the estimate derived using the maximum likelihood estimator methods above.