Assignment 3

Rory Sarten 301005654

25 September, 2020

Question 1

a)

A Runs Test tests a set of binary variables $X_1, ..., X_n$ to verify if the variables occur randomly.

H_0: variables occur randomly, i.e. knowing $X_1, ..., X_n$ does not help predict X_{n+1}.

H_A: variables are not random, i.e. knowing some part of the sequence can help predict subsequent variables.

As the variables are binary, they will take the value 0 or 1. The number of 0s is n_0 and the number of 1s is n_1, where:

$$n_0 = n - \sum_{i=1}^{n} X_i$$

$$n_1 = \sum_{i=1}^{n} X_i$$

To perform a Runs Test the observations are combined into one collection of $n = n_0 + n_1$ observations and arranged in increasing order of magnitude or observation. They are labeled according to which set they originally came from. A run is a group of two or more sequential values of 0 or 1.

Let R denote the number of runs in the combined ordered sample of $X \in \{0, 1\}$. Under H_0, R can be approximated as a normally distributed random variable, assuming both n_0 and n_1 are sufficiently large.

$R = 1 + \sum_{i=2}^{n} I_{(X_i, X_{i-1})}$, where $I_{(X_i, X_{i-1})} = 0$ if $X_i = X_{i-1}$ and $I_{(X_i, X_{i-1})} = 1$ if $X_i \neq X_{i-1}$

$$\bar{R} = \frac{2n_0n_1}{n} + 1$$

$$Var(\bar{R}) = \frac{2n_0n_1(2n_0n_1 - n)}{n^2(n-1)}$$

With test statistic $Z = \dfrac{R - \bar{R}}{\sqrt{Var(\bar{R})}}$ where $Z \sim N(0, 1)$

b)

A small number of runs (a small value for R) would indicate that X_i is more likely to be the same as X_{i-1}. A large number means that X is fluctuating regularly between values and X is less likely to be the same as X_{i-1}.

c)

```r
## 0 healthy, 1 diseased
X <- "HHHDDDHDDHHHHHDHHHHHHHHHHHDDDHHHHDDDHHHHDHHHDHH" %>%
  stringr::str_split("") %>% unlist() %>% `==`("D") %>% as.integer()

n <- length(X)
n_0 <- n - sum(X)
n_1 <- sum(X)
```

```r
R <- 1 + sum(X[2:n] != X[1:n-1])
R_est <- 1 + 2*n_0*n_1/n
R_var <- (2*n_0*n_1*(2*n_0*n_1 - n))/(n^2*(n - 1))
Z <- (R - R_est)/sqrt(R_var)
p <- pnorm(Z)
cbind(R, R_est, R_var, Z, p)
```

```
##       R    R_est    R_var         Z          p
## [1,] 15 20.65957 7.974767 -2.004125 0.02252834
```

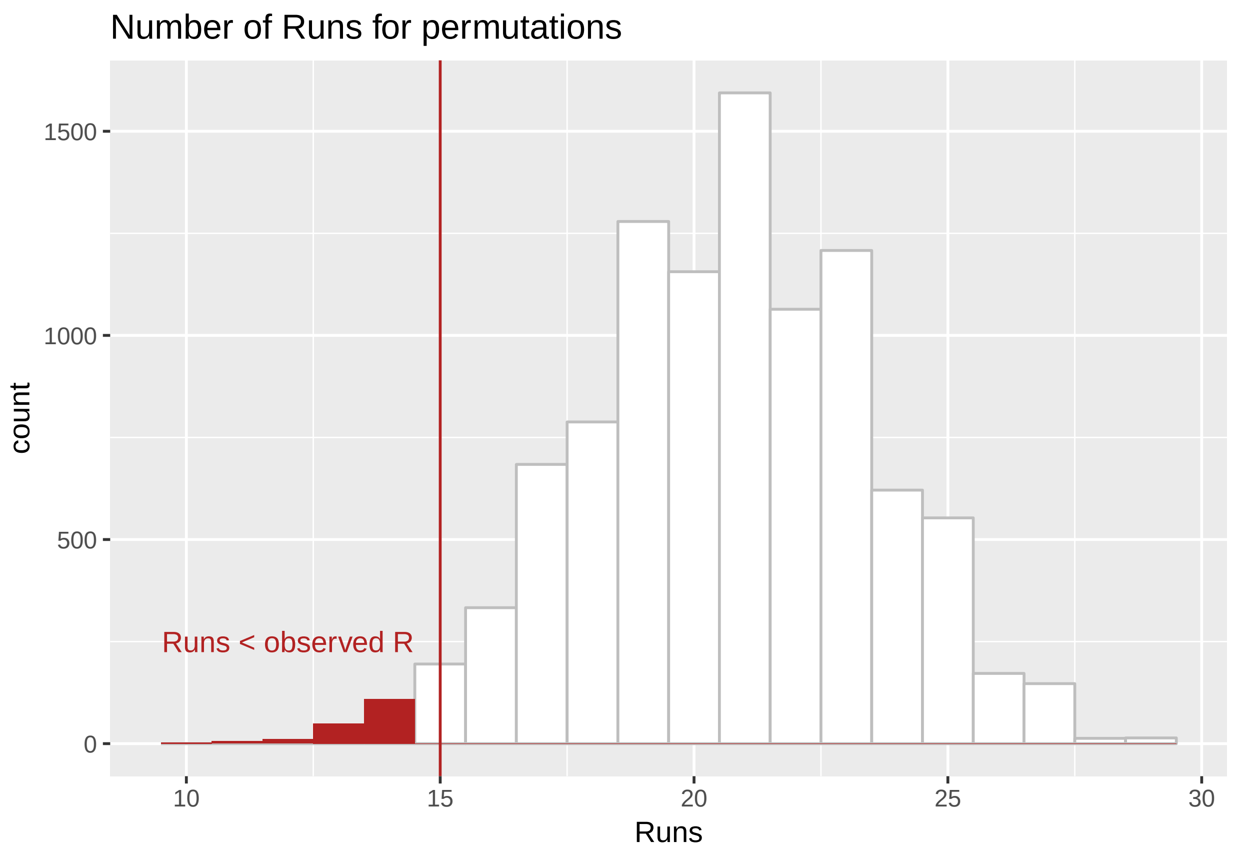
With p-value of 0.0225 we reject H_0 at the 5% level. We conclude that values are not randomly ordered.

d)

```r
set.seed(101)

calc_R <- function(input)
  1 + sum(input[2:length(input)] != input[1:length(input)-1])

sample_R <- function(iter, input)
  input %>% sample() %>% calc_R()

obs_R <- calc_R(X)

N <- 1e4
perm_R <- 1:N %>% sapply(sample_R, input = X)

p <- sum(perm_R < obs_R)/length(perm_R)
p
```

```
## [1] 0.0179
```

```r
ggplot() +
  geom_histogram(data = data.frame(x = perm_R[perm_R >= obs_R]),
                 aes(x = x), binwidth = 1, fill = "white", colour = "grey") +
  geom_histogram(data = data.frame(x = perm_R[perm_R < obs_R]),
                 aes(x = x), binwidth = 1, fill = "firebrick") +
  geom_vline(xintercept = obs_R, colour = "firebrick") +
  ggtitle("Number of Runs for permutations") +
  xlab("Runs") +
  geom_text(aes(x = 12, y = 250), label = "Runs < observed R", colour = "firebrick")
```



Because the number of runs is not a continuous variable, there is some ambiguity around whether the p-value should be calculated comparing values $<$ or $<=$ or even using some half point. I have decided to use $<$ as it gives the test the highest power.

Given this we find a p-value of 0.0179. We reject H_0 at the 5% level. We conclude that values are not randomly ordered.

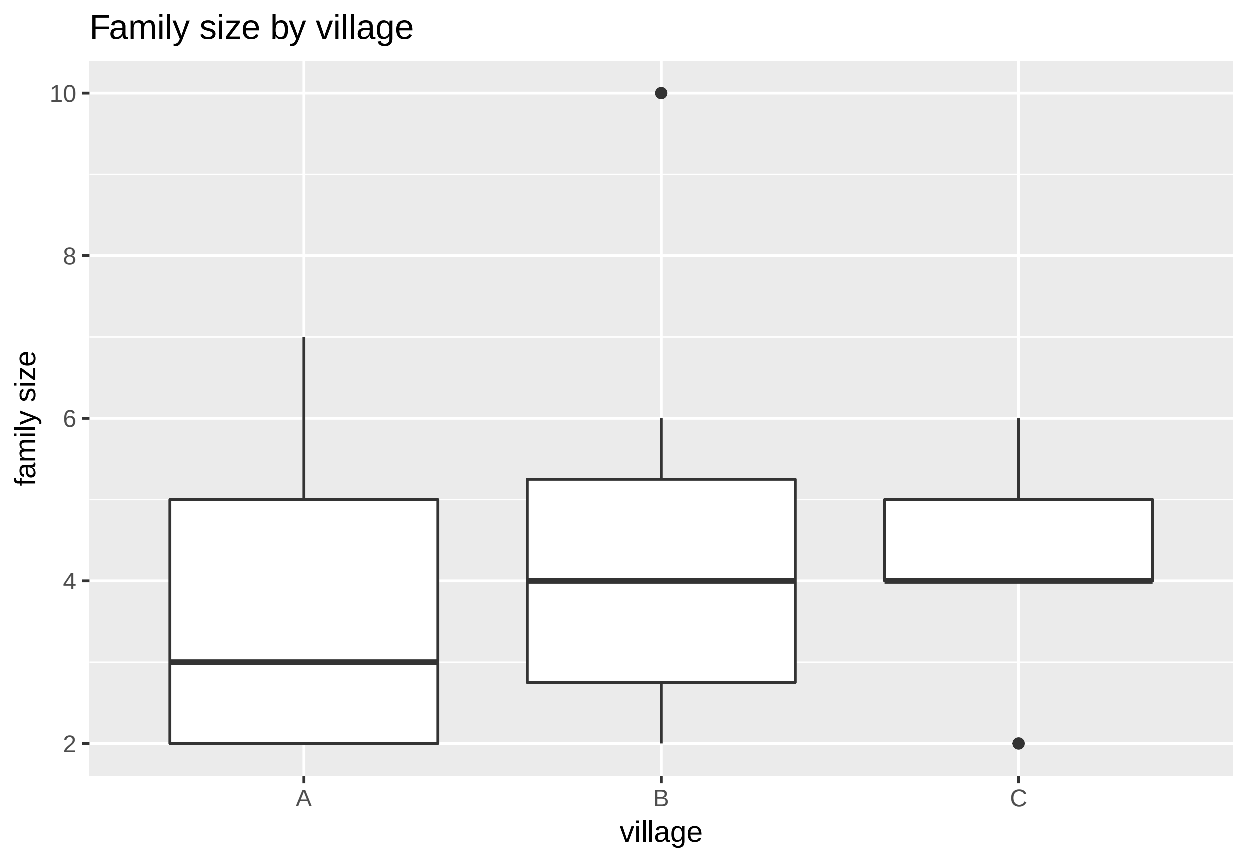
Question 2

a)

We could use a Poisson distribution

b)

```r
dataset <- data.frame(
  size = c(2, 3, 2, 7, 5, 5, 3, 2, 6, 10, 3, 2, 2, 5, 6, 4, 4, 5, 4, 4, 6, 5, 4, 2),
  village = rep(c("A", "B", "C"), c(9, 8, 7))
)

ggplot(dataset, aes(x = village, y = size)) +
  geom_boxplot() +
  ggtitle("Family size by village") +
  ylab("family size")
```



```
## village means
tapply(dataset$size, dataset$village, mean) %>% round(digits = 4)
```

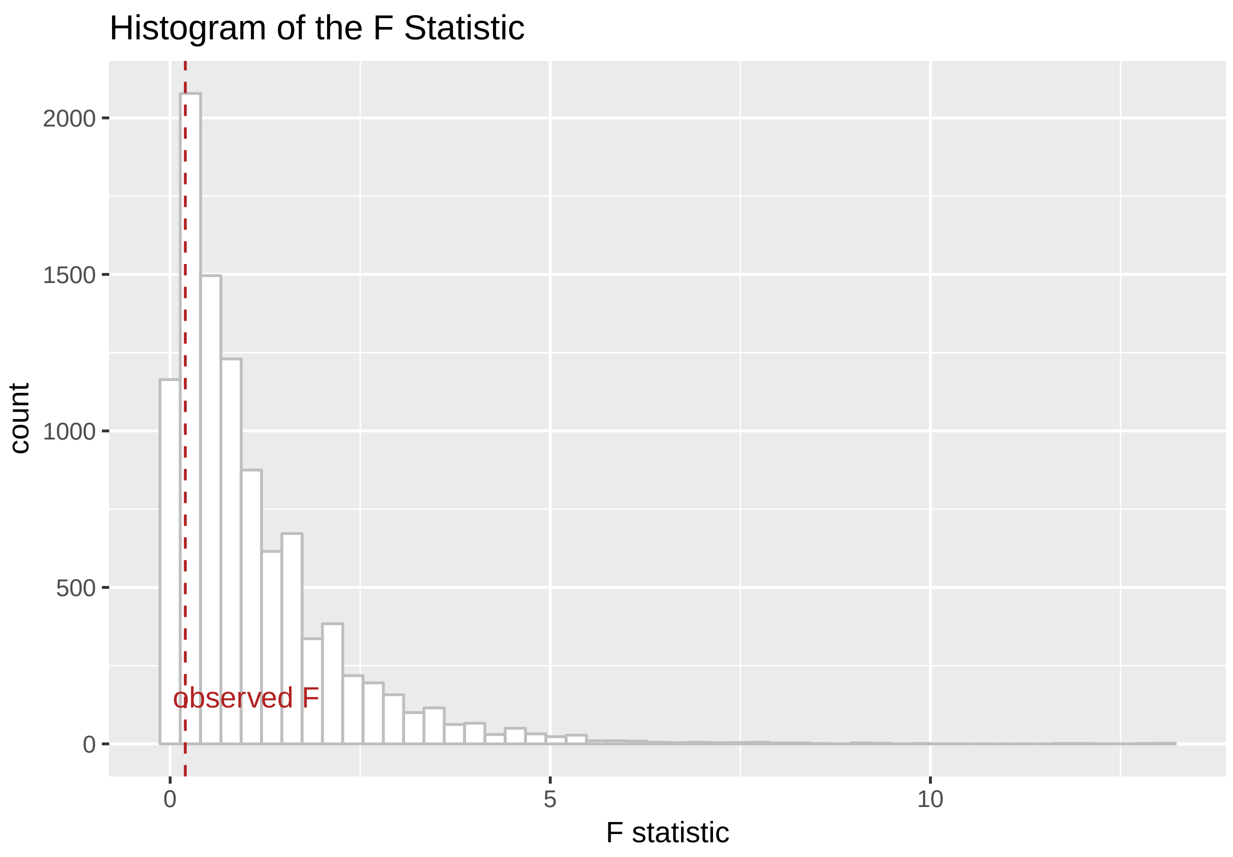
```
##      A      B      C
## 3.8889 4.5000 4.2857
```

c)

```
set.seed(101)

calc_F <- function(input)
  lm(size ~ village, data = input) %>% anova() %>% .[["F value"]] %>% .[1]

sample_F <- function(iter, input)
  input %>% dplyr::mutate(village = sample(village)) %>% calc_F()

obs_F <- calc_F(dataset) %>% round(digits = 4)

N <- 1e4
perm_F <- 1:N %>% sapply(sample_F, input = dataset)

p <- sum(perm_F > obs_F)/length(perm_F) %>% round(digits = 4)
p
```

```
## [1] 0.8275
```

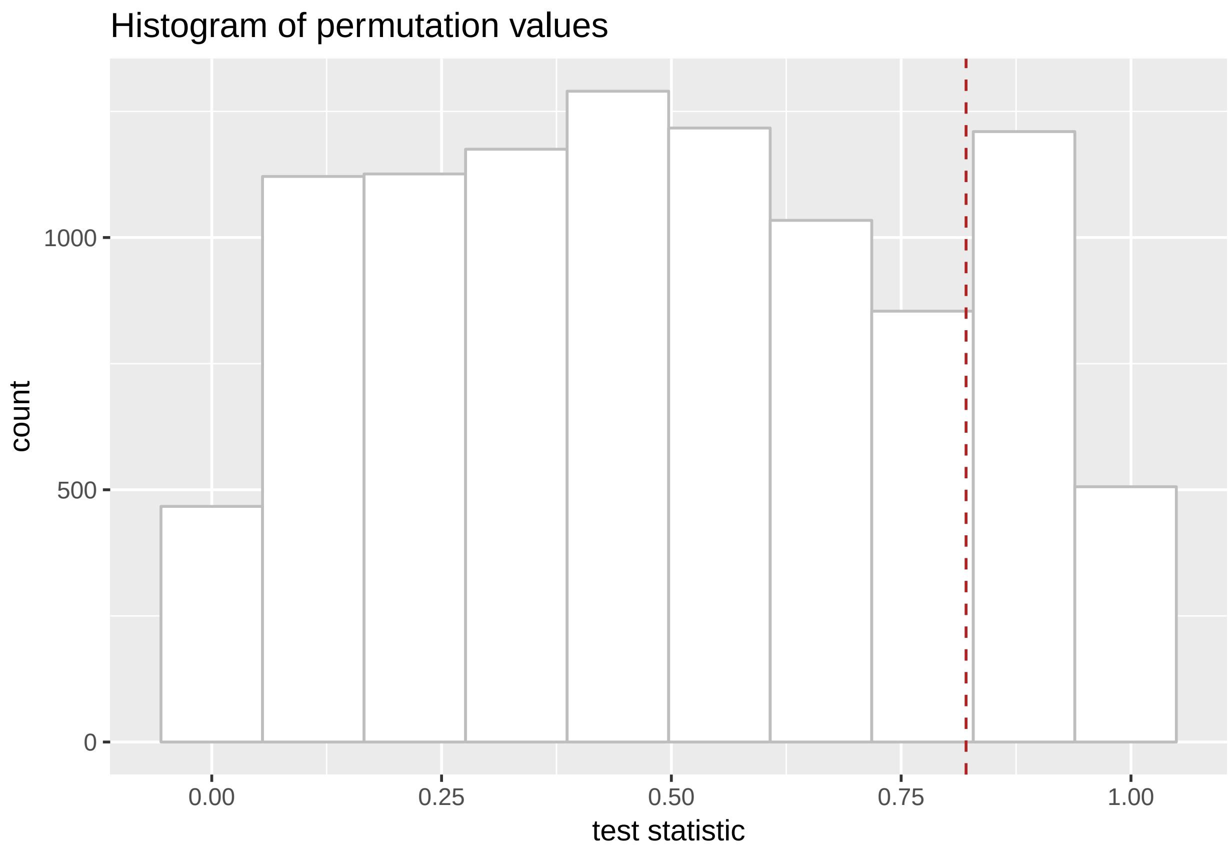
```
ggplot() +
  geom_histogram(data = data.frame(x = perm_F),
                 aes(x = x), bins = 50, fill = "white", colour = "grey") +
  geom_vline(xintercept = obs_F, colour = "firebrick", linetype = "dashed") +
  ggtitle("Histogram of the F Statistic") +
  xlab("F statistic") +
  geom_text(aes(x = 1, y = 150), label = "observed F", colour = "firebrick")
```



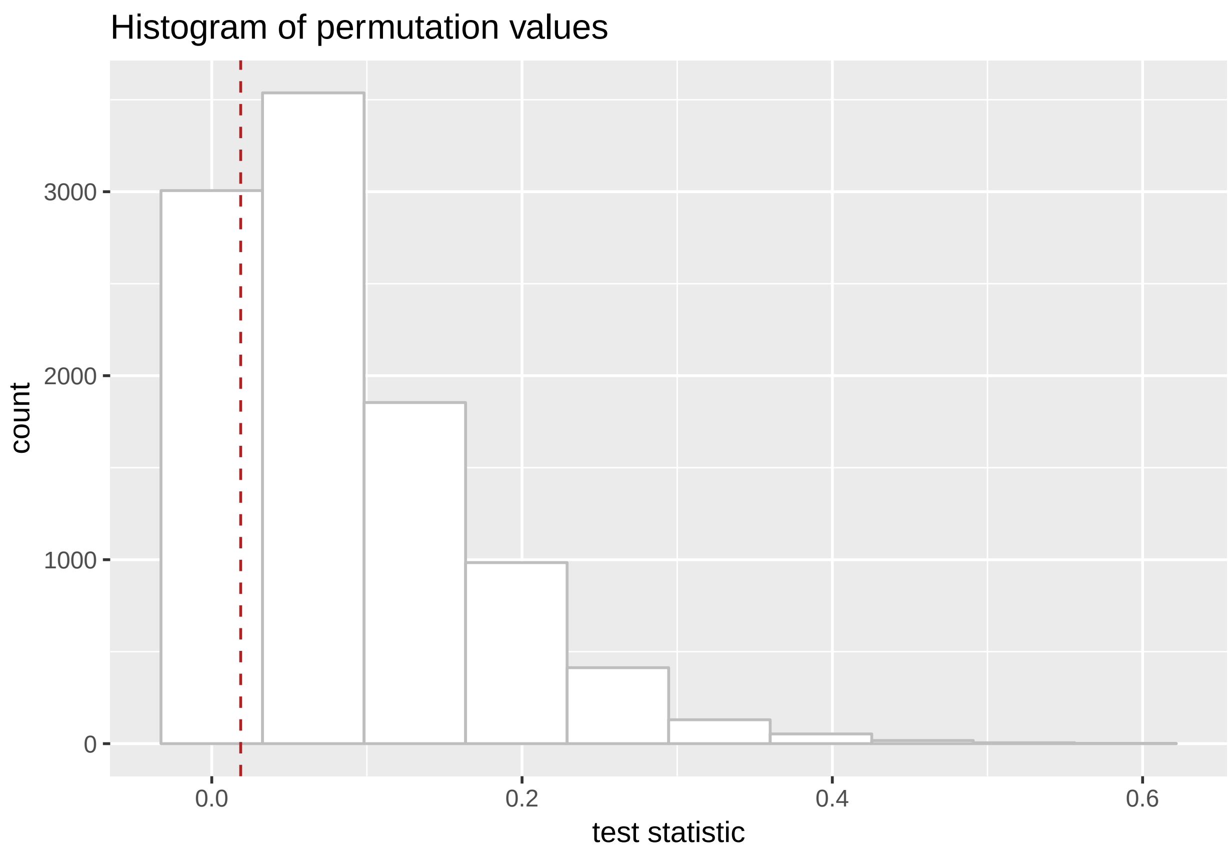
Given a p-value of 0.8275 we fail to reject H_0 and find that there is not difference in the village mean family sizes.

d)

```
test_p <- function(i = 1, input, perm = FALSE) {
  if (perm)
    input <- input %>% dplyr::mutate(village = sample(village))
  lm(size ~ village, data = input) %>% anova %>% .[["Pr(>F)"]] %>% .[1]
}

test_r2 <- function(i = 1, input, perm = FALSE) {
  if (perm)
    input <- input %>% dplyr::mutate(village = sample(village))
  lm(size ~ village, data = input) %>% summary %>% .[["r.squared"]]
}

draw_hist <- function(perm_values, observed) {
  h <- ggplot() +
  geom_histogram(data = data.frame(x = perm_values),
                 aes(x = x), bins = 10, fill = "white", colour = "grey") +
  geom_vline(xintercept = observed, colour = "firebrick", linetype = "dashed") +
  ggtitle("Histogram of permutation values") +
  xlab("test statistic")
  print(h)
}

permutation_test = function(test_func, dataset, niter, comparator) {
  test_stat <- test_func(input = dataset)
  perms <- 1:niter %>% sapply(test_func, input = dataset, perm = TRUE)

  draw_hist(perms, test_stat)

  sum(comparator(test_stat, perms))/niter
}

set.seed(202)
p <- permutation_test(test_p, dataset, 1e4, `>`)
```



```
set.seed(303)
p_r2 <- permutation_test(test_r2, dataset, 1e4, `<`)
```


Histogram of permutation values
count
3000
2000
1000
0
0.0
0.2
0.4
0.6
test statistic