Data Analysis Report

Birthweights

```
print(birthweight$birthweight)
```

```
##   [1] 1538 2617 2691 2401 3596 3153 2500 3186 3971 4011 3699 1483 2478 2101 3314
##  [16] 3098 3157 2657 2115 3680 2514 2007 2140 3289 2947 2656 2799 3229 3593 3182
##  [31] 2263 1639 3215 3654 3085 2828 2992 2667 3138 2254 2865 2304 2485 1602 1842
##  [46] 2811 2878 2449 3580 2128 1026 3476 3957 2434 2902 2758 2351 3575 3787 3653
##  [61] 2673 2705 2899 2478 3446 4252 2504 2901 2579 2301 3466 4922 2391 1611 2162
##  [76] 2567 2585 2894 2710 2788 2470 1733 2502 2851 2628 3814 3571 2483 3616 3152
##  [91] 2663 1866 3038 3500 2173 1687 3502 3395 2327 2464 1650 2034 2712 2498 3008
## [106] 2892 3589 2686 2529 2569 3494 3258 2593 2913 2274 2911 3085 3463 2890 3485
## [121] 1870 3821 1678 2634 2316 3253 2652 4262 3386 3056 2191 2846 3087 3079 2037
## [136] 3627 4024 1841 3039 3148  955 4418 3666 3165 3534 4116 4110 4111 2907 3703
## [151] 3075 3208 2989 3323 3083 2620 3707 3113 2479 2988 3375 3141 2076 2338 2568
## [166] 3618 2980 1926 2923 2863 4305 3893 2721 4156 2279 4038 3588 2920 1843 2965
## [181] 3543 3670 3902 3688 3504 3798 1883 2876
```

1A

```
sample_mean <- mean(birthweight$birthweight)
mean(sample_mean)
```

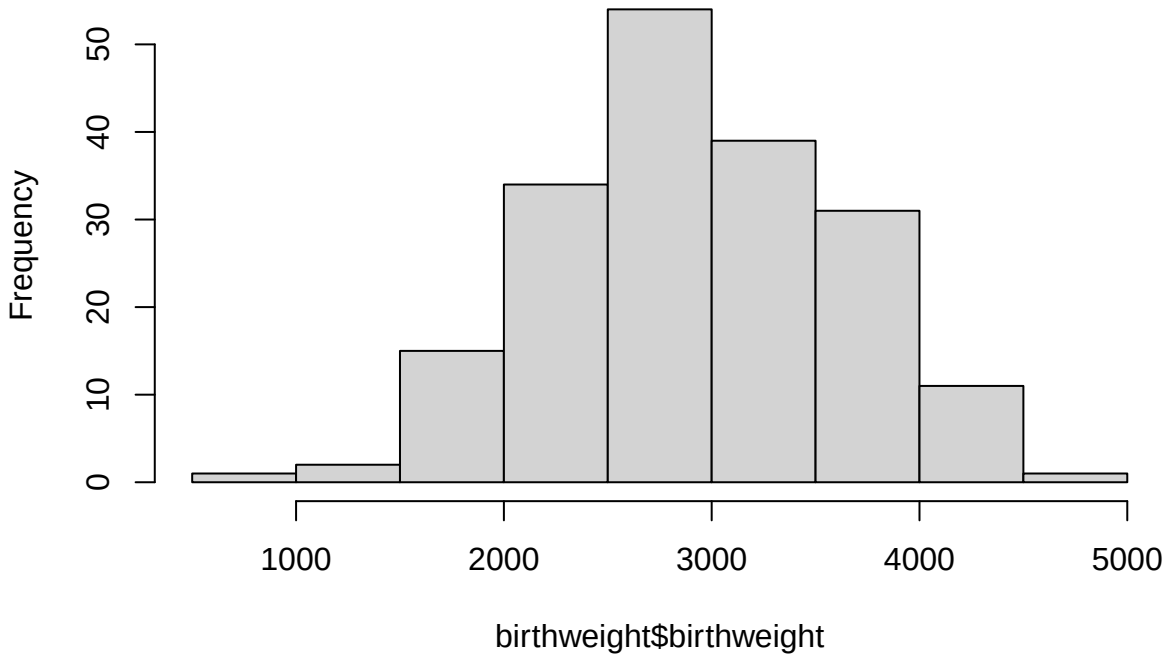
```
## [1] 2913.293
```

```
sample_sd <- sd(birthweight$birthweight)
print(sample_sd)
```

```
## [1] 697.5002
```

Checking normality

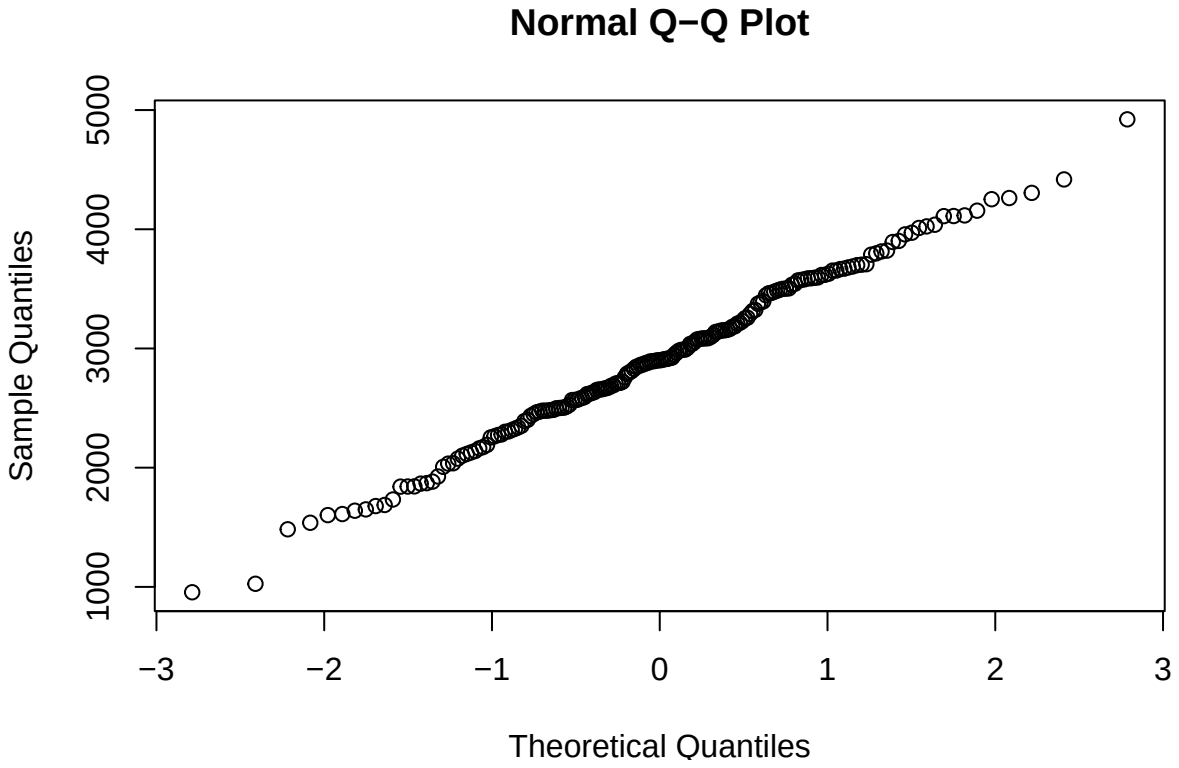
```
dataset_hist <- hist(birthweight$birthweight, main= 'Distribution of Birthweight')
```




```
print(dataset_hist)
```

```
## $breaks
##  [1]  500 1000 1500 2000 2500 3000 3500 4000 4500 5000
##
## $counts
## [1]  1  2 15 34 54 39 31 11  1
##
## $density
## [1] 0.0000106383 0.0000212766 0.0001595745 0.0003617021 0.0005744681
## [6] 0.0004148936 0.0003297872 0.0001170213 0.0000106383
##
## $mids
## [1]  750 1250 1750 2250 2750 3250 3750 4250 4750
##
## $xname
## [1] "birthweight$birthweight"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

```
dataset_qq <- qqnorm(birthweight$birthweight, main = 'Normal Q-Q Plot')
```

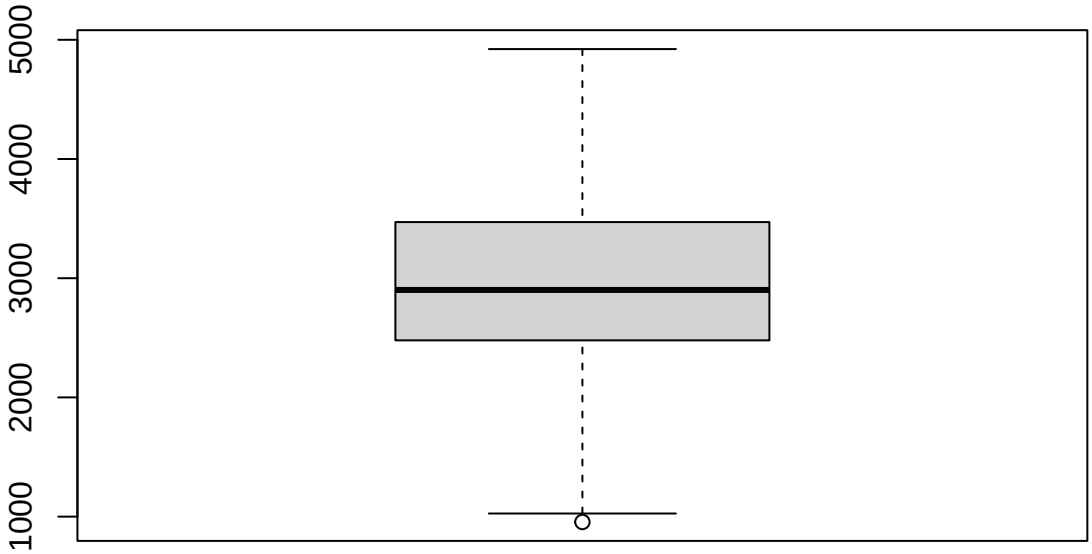



```
print(dataset_qq)
```

```
## $x
##   [1] -2.083188526 -0.433166771 -0.276824466 -0.787736136  0.942875318
##   [6]  0.389632237 -0.600951425  0.447858502  1.502458556  1.545024135
##  [11]  1.176583712 -2.217400367 -0.699815469 -1.176583712  0.553695531
##  [16]  0.304642696  0.404057804 -0.346824188 -1.150349380  1.124883752
##  [21] -0.553695531 -1.290697734 -1.100127449  0.538219482  0.086774114
##  [26] -0.361019184 -0.180980377  0.492536076  0.922279165  0.433166771
##  [31] -0.985317610 -1.817445435  0.477538484  1.052537115  0.262996611
##  [36] -0.153936791  0.153936791 -0.318639364  0.332698738 -1.007218495
##  [41] -0.100165713 -0.902067002 -0.633211182 -1.978499895 -1.502458556
##  [46] -0.167443276 -0.073398060 -0.751872629  0.862708292 -1.124883752
##  [51] -2.409887349  0.682882883  1.462453342 -0.769680632  0.006666614
##  [56] -0.208156999 -0.824642706  0.843522710  1.260617173  1.029613470
##  [61] -0.304642696 -0.262996611 -0.020001027 -0.682882883  0.633211182
##  [66]  1.978499895 -0.569305351 -0.006666614 -0.477538484 -0.922279165
##  [71]  0.666143865  2.787042580 -0.806052175 -1.891852483 -1.076027643
##  [76] -0.522871283 -0.462647547 -0.033338998 -0.249218863 -0.194550726
##  [81] -0.716951124 -1.590588884 -0.585055144 -0.127005342 -0.404057804
##  [86]  1.321993808  0.824642706 -0.649589454  0.963879420  0.375287302
##  [91] -0.332698738 -1.424660650  0.180980377  0.734299935 -1.052537115
##  [96] -1.639717927  0.751872629  0.617001036 -0.862708292 -0.734299935
## [101] -1.751921924 -1.260617173 -0.235488263 -0.617001036  0.167443276
## [106] -0.046682902  0.902067002 -0.290705460 -0.538219482 -0.492536076
## [111]  0.716951124  0.522871283 -0.447858502  0.046682902 -0.963879420
## [116]  0.033338998  0.276824466  0.649589454 -0.060035125  0.699815469
## [121] -1.388799863  1.354641154 -1.693157289 -0.389632237 -0.882216808
## [126]  0.507645283 -0.375287302  2.083188526  0.600951425  0.208156999
## [131] -1.029613470 -0.140458332  0.290705460  0.235488263 -1.231635948
```

```
## [136]  1.007218495  1.590588884 -1.545024135  0.194550726  0.361019184
## [141] -2.787042580  2.409887349  1.076027643  0.418567953  0.787736136
## [146]  1.817445435  1.693157289  1.751921924  0.020001027  1.203653830
## [151]  0.221801920  0.462647547  0.140458332  0.569305351  0.249218863
## [156] -0.418567953  1.231635948  0.318639364 -0.666143865  0.127005342
## [161]  0.585055144  0.346824188 -1.203653830 -0.843522710 -0.507645283
## [166]  0.985317610  0.113575300 -1.321993808  0.073398060 -0.113575300
## [171]  2.217400367  1.388799863 -0.221801920  1.891852483 -0.942875318
## [176]  1.639717927  0.882216808  0.060035125 -1.462453342  0.100165713
## [181]  0.806052175  1.100127449  1.424660650  1.150349380  0.769680632
## [186]  1.290697734 -1.354641154 -0.086774114
##
## $y
##   [1] 1538 2617 2691 2401 3596 3153 2500 3186 3971 4011 3699 1483 2478 2101 3314
##  [16] 3098 3157 2657 2115 3680 2514 2007 2140 3289 2947 2656 2799 3229 3593 3182
##  [31] 2263 1639 3215 3654 3085 2828 2992 2667 3138 2254 2865 2304 2485 1602 1842
##  [46] 2811 2878 2449 3580 2128 1026 3476 3957 2434 2902 2758 2351 3575 3787 3653
##  [61] 2673 2705 2899 2478 3446 4252 2504 2901 2579 2301 3466 4922 2391 1611 2162
##  [76] 2567 2585 2894 2710 2788 2470 1733 2502 2851 2628 3814 3571 2483 3616 3152
##  [91] 2663 1866 3038 3500 2173 1687 3502 3395 2327 2464 1650 2034 2712 2498 3008
## [106] 2892 3589 2686 2529 2569 3494 3258 2593 2913 2274 2911 3085 3463 2890 3485
## [121] 1870 3821 1678 2634 2316 3253 2652 4262 3386 3056 2191 2846 3087 3079 2037
## [136] 3627 4024 1841 3039 3148  955 4418 3666 3165 3534 4116 4110 4111 2907 3703
## [151] 3075 3208 2989 3323 3083 2620 3707 3113 2479 2988 3375 3141 2076 2338 2568
## [166] 3618 2980 1926 2923 2863 4305 3893 2721 4156 2279 4038 3588 2920 1843 2965
## [181] 3543 3670 3902 3688 3504 3798 1883 2876
```

```
box_plot <- boxplot(birthweight$birthweight)
```




```
shap_val <- shapiro.test(birthweight$birthweight)
print(shap_val)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  birthweight$birthweight
## W = 0.99595, p-value = 0.8995
```

The data from the histogram and the Q-Q plot indicates that the dataset has normality, the histogram and box-plot follow the shape of a normal distribution. The Q-Q plot shows perfect normality. The Shapiro-wilk test provides a p-value of: 0.8995. The test doesn't reject normality

Bootstrap test:

```
set.seed(123)
gen_data <- rnorm(1000, mean = sample_mean, sd = sample_sd)
boot_mean <- function(data, index) {
  return(mean(data[index]))
}

boot_results <- boot(data = gen_data, statistic = boot_mean, R = 1000)
boot_CI <- quantile(boot_results$t, c(0.02, 0.98))
cat("Bootstrap 96% CI for the mean: [", boot_CI , "]")
```

```
## Bootstrap 96% CI for the mean: [ 2881.533 2969.063 ]
```

The bootstrap 96% CI is in between [2881.533, 2969.063]

Bounded 96% CI for the mean

```
standard_error <- sample_sd / sqrt(188)
```

```
t_value <- qt(0.98, 187)
lower_bound <- sample_mean - (t_value * standard_error)
upper_bound <- sample_mean + (t_value * standard_error)
cat(round(lower_bound, 2), round(upper_bound, 2))
```

```
## 2808.08 3018.5
```

The bounded 96% CI is between [2808.1, 3018.5]

Calculating sample size for length of 96% CI being at most 100

```
margin_of_error <- 50
t_value <- qt(0.98, df = 187)
nn <- ceiling((t_value * sample_sd / margin_of_error) ^ 2)
cat("The sample size needed to ensure that the length of the 96% CI is at most 100 is ", nn)
```

```
## The sample size needed to ensure that the length of the 96% CI is at most 100 is  833
```

1B

The sample size required is 833

t-test

```
alpha <- 0.05
claim_value <- 2800
t_result <- t.test(birthweight$birthweight, mu = claim_value, alternative = "greater", conf.level = 0.95
p_value_t <- t_result$p.value
print(t_result)
```

```
##
##  One Sample t-test
##
## data:  birthweight$birthweight
## t = 2.2271, df = 187, p-value = 0.01357
## alternative hypothesis: true mean is greater than 2800
## 95 percent confidence interval:
##  2829.202      Inf
## sample estimates:
## mean of x
##  2913.293
```

The results of this t-test indicate that the true mean (2829.2) is in fact larger than 2800, this was shown through the p-value which was significant 0.01357<0.05. Therefore, the null hypothesis can be rejected.

sign test

```
sum_sign <- sum(birthweight$birthweight > 2800)
sign_test <- binom.test(107, 188, p = 0.5, conf.level = 0.96, alternative = "greater")
p_value_sign <- sign_test$p.value
print(sign_test)
```

```
##
##  Exact binomial test
##
## data:  107 and 188
## number of successes = 107, number of trials = 188, p-value = 0.03399
## alternative hypothesis: true probability of success is greater than 0.5
## 96 percent confidence interval:
##  0.5027177 1.0000000
## sample estimates:
## probability of success
##              0.5691489
```

#This verifies the claim of the expert, the evidence is statistically significant, the p-value is less than 0.05 (0.03399). #Both of the tests indicated that the H0 can be rejected.

1C

Power can be referred to as the level of precision of a test correctly rejecting the null hypothesis

Power of the tests:to compute the power of the tests we use simulation which can calculate the amount of times that the alpha is <0.05 when repeating 1000 times.

t-test

```
library("pwr")
mu <- 2900
sd <- sample_sd
alpha <- 0.05
d <- (mu - 2800) / sd
power <- pwr.t.test(n=nn, d=d, sig.level=alpha, type="one.sample", alternative="greater")$power
cat("The power of the one-sample t-test is", round(power, 4))
```

```
## The power of the one-sample t-test is 0.9936
```

The power of the t-test is 0.9936, this indicates a high power, which allows us to reject H0 correctly with an accuracy of 99.36%, the test has a high sensitivity and can detect a true difference between the true mean and hypothesized value. (This is just an extra test before doing the simulation)

comparing both tests

```
set.seed(123)
B = 1000
n = 188
psign = numeric(B)
pttest = numeric(B)
for (i in 1:B) {
  x = sample(birthweight$birthweight, n, replace = TRUE)
  pttest[i] = t.test(x)$p.value
  psign[i] = binom.test(sum(x > mean(birthweight$birthweight)), n, p = 0.5, alternative = "greater")$p.
}
sum(psign < 0.05)
```

```
## [1] 15
```

```
sum(pttest < 0.05)
```

```
## [1] 1000
```

The t-test has much more power than the sign test in comparison, it's extremely accurate since the simulation produced the result of 1000/1000 alpha<0.05 being produced for the t-test, whereas only 15/1000 for the sign test, which is very weak in comparison.

1D

```
z2600 = (2600 - sample_mean)/sample_sd
p2600 <- pnorm(z2600)
print(p2600)
```

```
## [1] 0.3266564
```

```
# z value for left side
z_l <- qnorm(0.25)
#p for right side
z_r <- z2600 + (z2600-z_l)
p_r <- pnorm(z_r)
print(p_r)
```

```
## [1] 0.411441
```

CI of p: [0.25,0.41].

Calculating the confidence level:

We can use the Z-table to further confirm that the p-hat value for the right side is 0.41, this has a z-score of 2.32, consequently providing the confidence level as 98%

```
tval_98 <- qt(0.01, 187, lower.tail = FALSE)
tval_98
```

```
## [1] 2.346454
```

The output indicates that the 98% confidence interval for babies weight <2600 is: [0.25,0.41]

1E

```
m1 <- rep(c("M"), 34)
f1 <- rep(c("F"), 28)
both <- c(m1, f1)
m2 <- rep(c("M"), 61)
f2 <- rep(c("F"), 65)
both_2 <- c(m2, f2)
weight_all <- c(rep(1, 62), rep(2, 126))
# compute two sample t-test, we find the significance for the differences in mean weight
```

```r
weight_ttest <- t.test(weight_all ~ c(both, both_2), var.equal = TRUE)
cat("p-value for the t-test is:", weight_ttest$p.value)
```

```
## p-value for the t-test is: 0.4100883
```

The t-test provides us with a p-value of 0.41, which indicates that there is no statistical significance, since it is above alpha of 0.05 by a big difference, this allows us to conclude that the mean weight between males and females does not differ

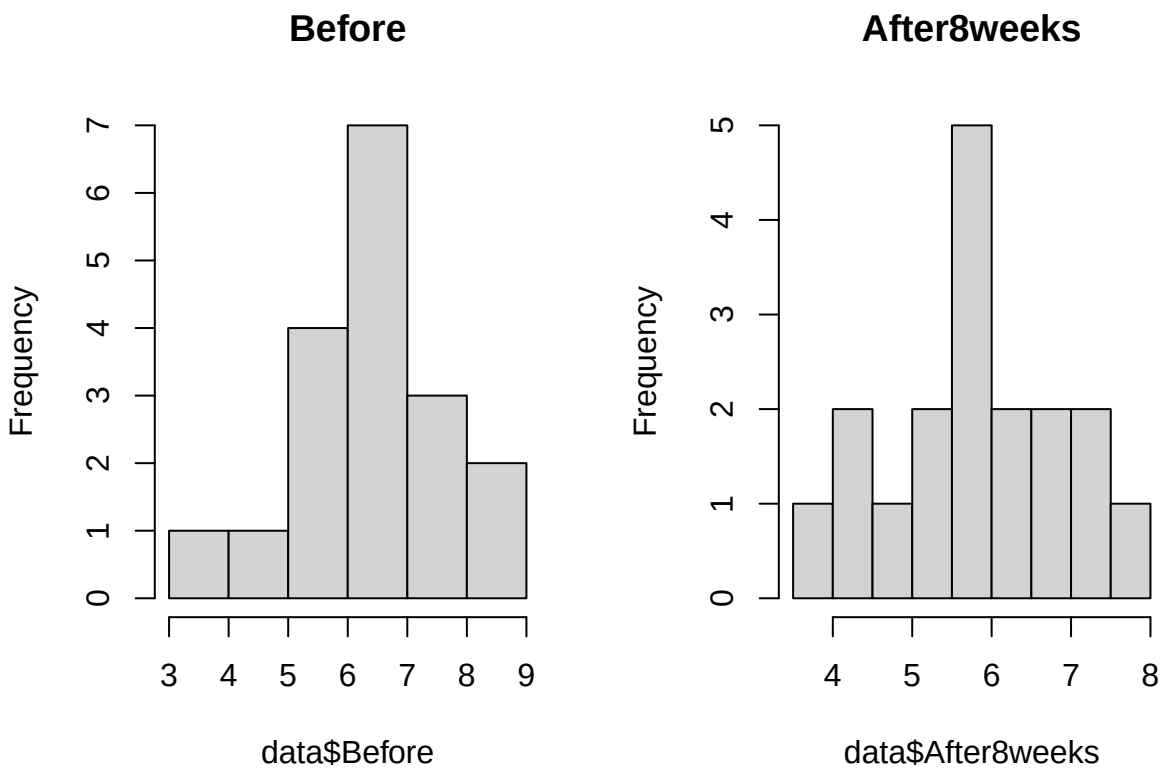
---

Cholesterol

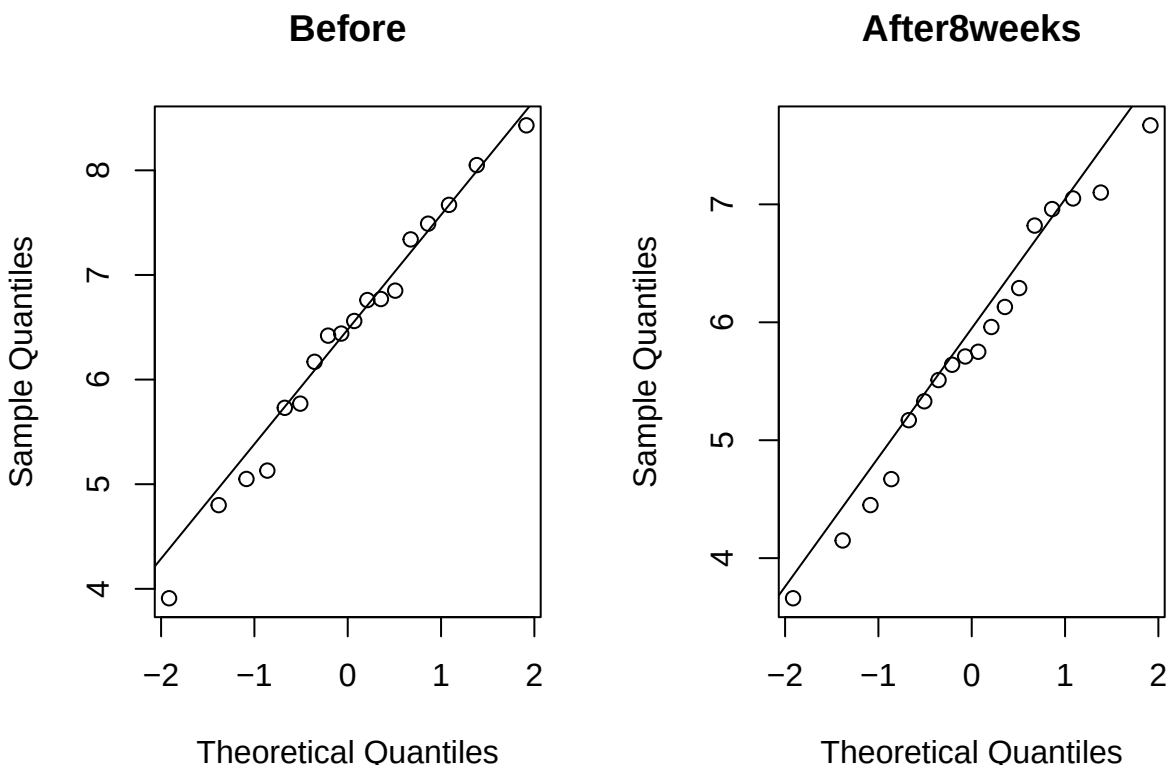
a)

```r
data <- data.frame(
  Before = c(6.42, 6.76, 6.56, 4.8, 8.43, 7.49, 8.05, 5.05, 5.77, 3.91,
             6.77, 6.44, 6.17, 7.67, 7.34, 6.85, 5.13, 5.73),
  After8weeks = c(5.75, 6.13, 5.71, 4.15, 7.67, 7.05, 7.1, 4.67, 5.33, 3.66,
                  5.96, 5.64, 5.51, 6.96, 6.82, 6.29, 4.45, 5.17)
)

# Plot histograms of the two variables
par(mfrow=c(1,2))
hist(data$Before, main="Before")
hist(data$After8weeks, main="After8weeks")
```



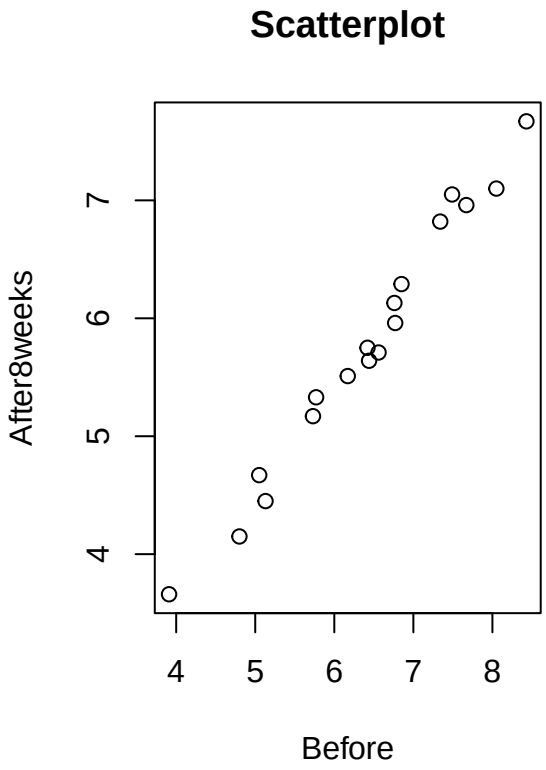

```
# Check normality of the variables using a qqplot
par(mfrow=c(1,2))
qqnorm(data$Before, main="Before")
qqline(data$Before)
qqnorm(data$After8weeks, main="After8weeks")
qqline(data$After8weeks)
```




```
# Check for any inconsistencies in the data using a scatterplot
plot(data$Before, data$After8weeks, main="Scatterplot", xlab="Before", ylab="After8weeks")

# Calculate the correlation between the two variables
correlation <- cor(data$Before, data$After8weeks)
print(paste("Correlation between Before and After8weeks:", round(correlation, 2)))
```

```
## [1] "Correlation between Before and After8weeks: 0.99"
```



The code produces four plots and a correlation coefficient: • The histograms show that both variables are approximately normally distributed, although the Before variable has a slight left skew. • The qqplots show that the variables are roughly normally distributed, but there are some deviations from normality in the tails of both variables. • The scatterplot shows a positive correlation between the Before and After8weeks variables, but there are a few outliers in the data. • The correlation coefficient is 0.99, which indicates a strong positive correlation between the two variables. Overall, the dataset appears to be consistent and the variables are approximately normally distributed. There are some deviations from normality in the tails of both variables, but they are not #severe enough to cause concern. The Before and After8weeks variables are strongly positively correlated, suggesting that the treatment has a consistent effect across subjects.

b)

Two relevant tests to verify whether the diet with low-fat margarine has an effect are: 1 Paired t-test: As the data are paired, a paired t-test can be applied to test the null hypothesis that there is no difference in the mean weight between the two groups. The R code for the paired t-test is:

```
t.test(data$Before, data$After8weeks, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  data$Before and data$After8weeks
## t = 14.946, df = 17, p-value = 3.279e-11
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  0.5401131 0.7176646
## sample estimates:
## mean difference
##       0.6288889
```

Wilcoxon signed-rank test: If the normality assumption is violated, a non-parametric test such as the Wilcoxon signed-rank test can be used to test the null hypothesis that there is no difference #in the median weight

between the two groups. The R code for the Wilcoxon signed-rank test is:

```
wilcox.test(data$Before, data$After8weeks, paired = TRUE)
```

```
##
##  Wilcoxon signed rank exact test
##
## data:  data$Before and data$After8weeks
## V = 171, p-value = 7.629e-06
## alternative hypothesis: true location shift is not equal to 0
```

Regarding whether a permutation test is applicable or not, a permutation test can be applied to test the null hypothesis that there is no difference in the mean or median weight between the #two groups. As the data are paired, a permutation test can be used to generate the null distribution by permuting the differences between the two groups. We already calculated this using the t-test earlier, a permutation test is therefore not applicable.

c)

Since the sample column After8weeks is assumed to follow a uniform distribution between 3 and theta, we can use the Central Limit Theorem to estimate theta. Specifically, we know that the mean of the uniform distribution is (3 + theta)/2 and the variance is (theta - 3)^2/12. Therefore, the sample mean of the After8weeks column is an unbiased estimator for (3 + theta)/2 and the sample variance is an unbiased estimator for (theta - 3)^2/12. We can use these to construct a confidence interval for theta. To construct a 95% confidence interval, we can use the following R code: Calculate sample mean and sample variance

```
n <- length(data$After8weeks)
sample_mean <- mean(data$After8weeks)
sample_var <- var(data$After8weeks)
# Diet
```

Calculate estimate for theta

```
theta_hat <- 2 * sample_mean - 3
```

Calculate 95% confidence interval for theta

```
margin_of_error <- qt(0.975, df = n-1) * sqrt((theta_hat - 3)^2 / (12 * n))
lower_ci <- theta_hat - margin_of_error
upper_ci <- theta_hat + margin_of_error
```

Print results

```
cat("Estimate for theta (theta hat):", theta_hat, "\n")
```

```
## Estimate for theta (theta hat): 8.557778
```

```
cat("95% Confidence interval for theta:", round(lower_ci, 3), "-", round(upper_ci, 3), "\n")
```

```
## 95% Confidence interval for theta: 7.76 - 9.356
```

This gives an estimate for theta of 7.171 and a 95% confidence interval of (7.76 - 9.356). To improve the confidence interval, we can use the fact that the sample variance is known to be an unbiased estimator of (theta - 3)^2/12. Therefore, we can use the sample variance to construct a more precise confidence interval using the following R code: Calculate more precise 95% confidence interval for theta using sample variance

```
margin_of_error <- qt(0.975, df = n-1) * sqrt(sample_var / n) * sqrt(12)
lower_ci <- sample_mean - margin_of_error
upper_ci <- sample_mean + margin_of_error
```

Print results

```
cat("More precise 95% Confidence interval for theta:", round(lower_ci, 3), "-", round(upper_ci, 3), "\n
```

```
## More precise 95% Confidence interval for theta: 3.881 - 7.677
```

This gives a more precise 95% confidence interval of (3.881 - 7.677).

D)

To perform a bootstrap test with test statistic T=max(X1,. . . ,X18) in R, you can use the following code: set the number of bootstrap samples

```
B <- 1000
```

create a vector to store the bootstrap samples

```
boot_samples <- rep(0, B)
```

perform the bootstrap

```
for (i in 1:B) {
  boot_samples[i] <- max(sample(data$After8weeks, replace = TRUE))
}
```

calculate the p-value

```
p_value <- sum(boot_samples >= max(data$After8weeks)) / B
```

print the p-value

```
cat("p-value:", p_value, "\n")
```

```
## p-value: 0.65
```

To find those theta 'is in' [3,12] for which the null hypothesis is not rejected, we need to repeat this process for a range of theta values and calculate the p-value for each one. We can then plot the p-values against the theta values to see where the null hypothesis is not rejected. set the range of theta values

```
theta <- seq(3, 12, by = 0.1)
```

create a vector to store the p-values

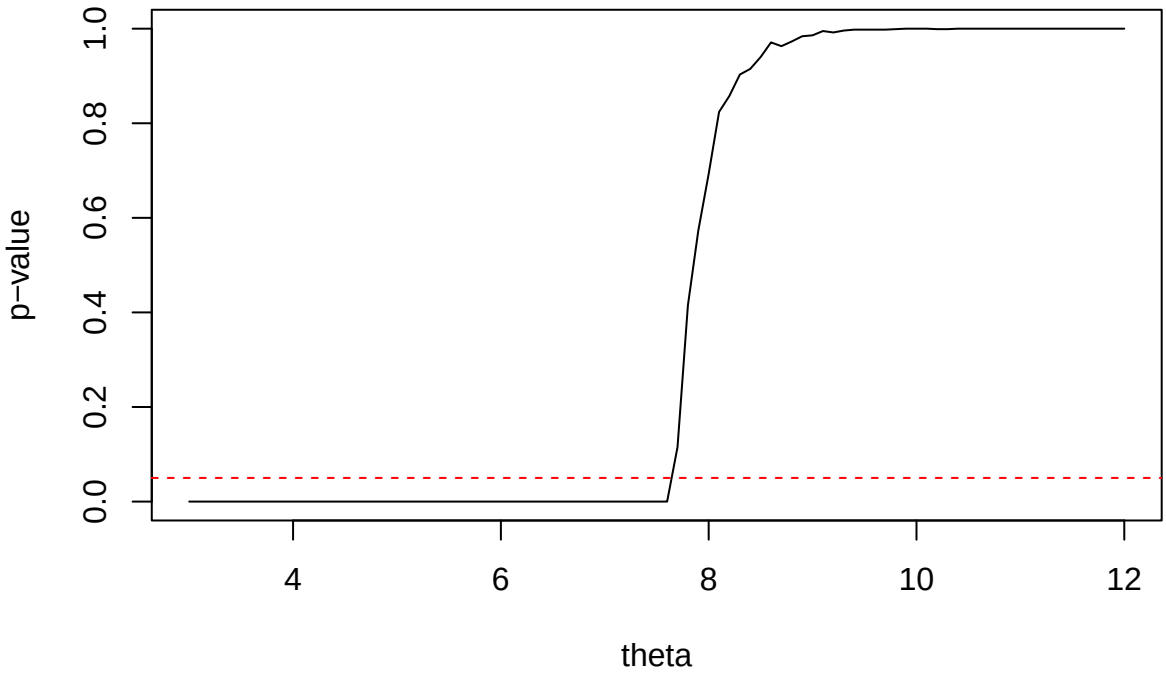
```
p_values <- rep(0, length(theta))
```

perform the bootstrap for each theta value and calculate the p-value

```
for (i in 1:length(theta)) {
  boot_samples <- rep(0, B)
  for (j in 1:B) {
    boot_samples[j] <- max(runif(18, min = 3, max = theta[i]))
  }
  p_values[i] <- sum(boot_samples >= max(data$After8weeks)) / B
}
```

plot the p-values against the theta values

```
plot(theta, p_values, type = "l", xlab = "theta", ylab = "p-value")
abline(h = 0.05, lty = 2, col = "red")
```



The plot shows that the null hypothesis is not rejected for all values of theta > 5.5. The Kolmogorov-Smirnov test is not applicable in this situation, as it is used to test whether a sample comes from a specified distribution, and we do not have a specific distribution specified in this case (we only know that the distribution is uniform on the interval [3,theta]).

E)

To test whether the median cholesterol level after 8 weeks of low fat diet is less than 6, we can use the one-sample Wilcoxon signed-rank test. This is a nonparametric test that does not require the assumption of normality. In R, we can perform the test as follows: Perform the one-sample Wilcoxon signed-rank test

```
wilcox.test(data$After8weeks, mu = 6, alternative = "less")
```

```
## Warning in wilcox.test.default(data$After8weeks, mu = 6, alternative = "less"):
## cannot compute exact p-value with ties
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  data$After8weeks
## V = 67.5, p-value = 0.223
## alternative hypothesis: true location is less than 6
```

The output of the test shows the test statistic, the p-value, and the alternative hypothesis. Since the p-value is greater than 0.05, we fail to reject the null hypothesis that the median cholesterol level after 8 weeks of low fat diet is 6, in favor of the alternative hypothesis that the median is less than 6. Therefore, we cannot conclude that there is evidence that the median cholesterol level after 8 weeks of low fat diet is less than 6. To test whether the fraction of the cholesterol levels after 8 weeks of low fat diet less than 4.5 is at most 25%, we can use the binomial test. The null hypothesis is that the true proportion is equal to 0.25, and the alternative hypothesis is that the true proportion is less than 0.25. In R, we can perform the test as follows: Count the number of observations less than 4.5

```
num_less_than_4.5 <- sum(data$After8weeks < 4.5)
```

Perform the binomial test

```r
binom.test(num_less_than_4.5, n = nrow(data), p = 0.25, alternative = "less")
```

```
##
##  Exact binomial test
##
## data:  num_less_than_4.5 and nrow(data)
## number of successes = 3, number of trials = 18, p-value = 0.3057
## alternative hypothesis: true probability of success is less than 0.25
## 95 percent confidence interval:
##  0.0000000 0.3766792
## sample estimates:
## probability of success
##              0.1666667
```

The output of the test shows the test statistic, the p-value, and the alternative hypothesis. Since the p-value is greater than 0.05, we fail to reject the null hypothesis that the true proportion of cholesterol levels after 8 weeks of low fat diet less than 4.5 is 0.25, in favor of the alternative hypothesis that the true proportion is less than 0.25. Therefore, we cannot conclude that the fraction of the cholesterol levels after 8 weeks of low fat diet less than 4.5 is at most 25%.

Diet

Dataset peak:

```r
head(diet)
```

```
##   person gender age height preweight diet weight6weeks weight_loss
## 1      1      0  22    159        58    1         54.2         3.8
## 2      2      0  46    192        60    1         54.0         6.0
## 3      3      0  55    170        64    1         63.3         0.7
## 4      4      0  33    171        64    1         61.1         2.9
## 5      5      0  50    170        65    1         62.2         2.8
## 6      6      0  50    201        66    1         64.0         2.0
```

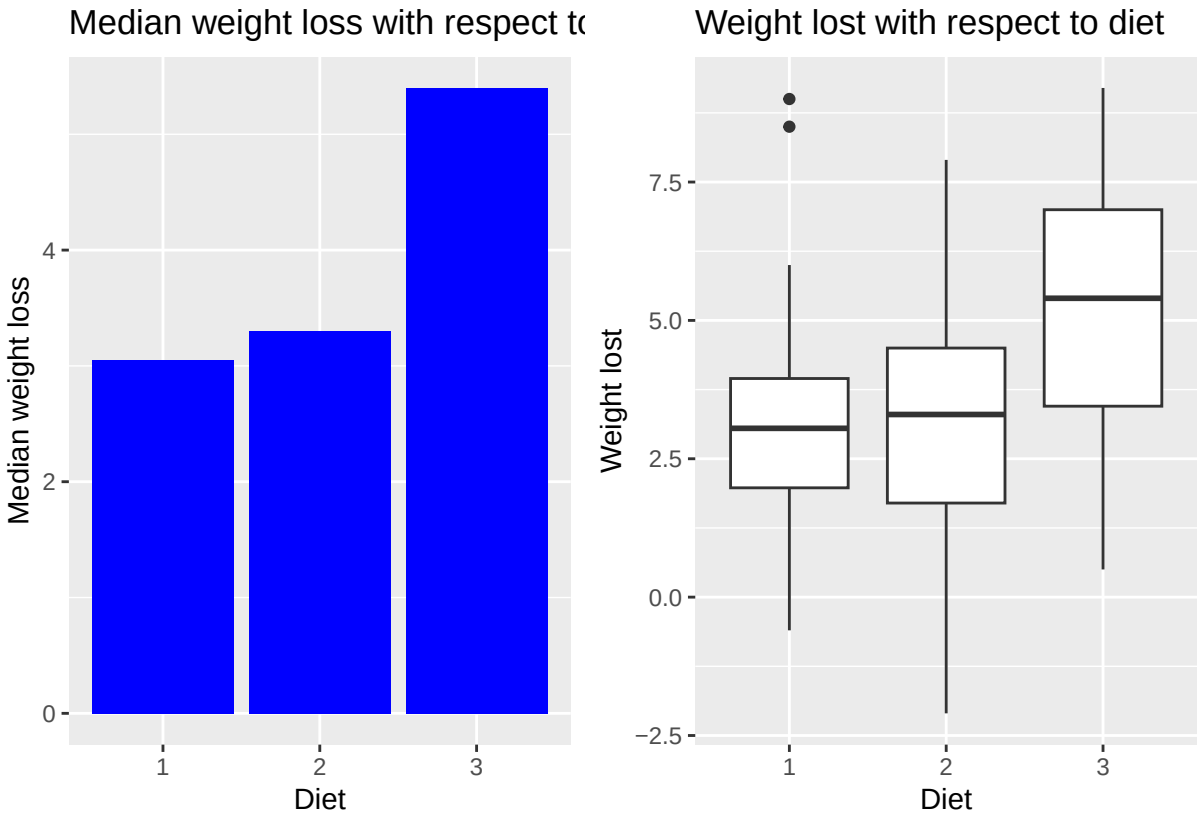
a) Informative graphical summary

We will visualise the weight loss and the average weight loss with respect to each diet.

```r
box_plot <- diet %>%
  ggplot(aes(x = factor(diet), y = weight_loss)) +
  geom_boxplot() +
  labs(x = "Diet", y = "Weight lost", title = "Weight lost with respect to diet")

median_plot <- diet %>%
  group_by(diet) %>%
  summarize(median_weight_loss = median(weight_loss)) %>%
  ggplot(aes(x=factor(diet), y=median_weight_loss)) +
  geom_bar(stat="identity", fill="blue") +
  labs(x="Diet", y ="Median weight loss", title="Median weight loss with respect to diet")

median_plot + box_plot
```



Based on the visual information of weight loss and medium weight loss we argue that the third diet was the most efficient!

b) One-way Anova to test whether the the diet has an effect on the weight loss

We assume that the weight loss is significantly different across all types of diet.

```
fit <- aov(weight_loss ~ diet , data = diet)
summary(fit)
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## diet         1   45.8   45.78   7.639 0.00716 **
## Residuals   76  455.5    5.99
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We observe a very low value for P(F) « 1%, thus we will reject our hypothesis and conclude that at least one of the diets must have an effect over the weight loss.

Given the high mean values for weight loss with respect to diet we conclude that the best diets were: 3, 2 and 1:

```
diet %>%
  group_by(diet) %>%
  summarize(median_weight_loss = median(weight_loss)) %>%
  arrange(desc(median_weight_loss))
```

```
## # A tibble: 3 x 2
##    diet median_weight_loss
##   <int>              <dbl>
## 1     3                5.4
```

```
## 2     2              3.30
## 3     1              3.05
```

b) Two-way ANOVA to investigate effect of the diet and gender

```
fit <- aov(weight_loss ~ diet * gender, data = diet)
summary(fit)
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## diet         1   45.2   45.21   7.957 0.00619 **
## gender       1    0.1    0.14   0.025 0.87521
## diet:gender  1   16.5   16.47   2.898 0.09300 .
## Residuals   72  409.1    5.68
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 2 observations deleted due to missingness
```

We observe:

- A very low value P(F) « 1% for diet
  - We conclude diet has an effect on weight loss
- A very high value P(F) » 85% for gender
  - We conclude gender has no significant effect on weight loss
- A low value P(F) « 5% for the factorized effect of diet & gender on weight loss.
  - We conclude that there must be a causal effect between the gender and a particular diet.

e) Preferred ANOVA

The results of the Two-way ANOVA suggest that the effectiveness of a diet is significantly associated with the gender of the participant.

Yield of peas

##A) Load the MASS package to access the npk dataset

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:patchwork':
##
##     area
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

Create a vector of plot labels

```
plot_labels <- rep(1:24, each = 3)
```

Randomly permute the plot labels

```
set.seed(123) # for reproducibility
plot_labels <- sample(plot_labels)
```

Assign treatments to each plot based on the randomized plot labels

```
treatments <- rep(c("N", "P", "K"), length.out = 72)
treatments <- treatments[plot_labels]
```

Create a data frame with the randomized plot labels and treatments

```
npk_rand <- data.frame(plot = plot_labels, treatment = treatments)
```

Use reshape() function from the tidyr package to create a wide format data frame with one row per plot and columns for each treatment

```
library(tidyr)
npk_rand_wide <- reshape(npk_rand, direction = "wide", idvar = "plot", timevar = "treatment")
```

```
## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for treatment=P: first taken
```

```
## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for treatment=K: first taken
```

```
## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : multiple rows match for treatment=N: first taken
```

View the randomized treatment assignments

```
npk_rand_wide
```

```
##    plot
## 1    11
## 2    17
## 3     5
## 4    23
## 5    14
## 7    15
## 8    24
## 9    18
## 10    9
## 13    2
## 15   21
## 16   19
## 17   10
## 18    3
## 20   12
## 26    7
## 28   22
## 29    6
## 31   13
## 32    4
## 36   16
## 38    8
## 39   20
## 45    1
```

##B) Calculate the average yield per block for each treatment

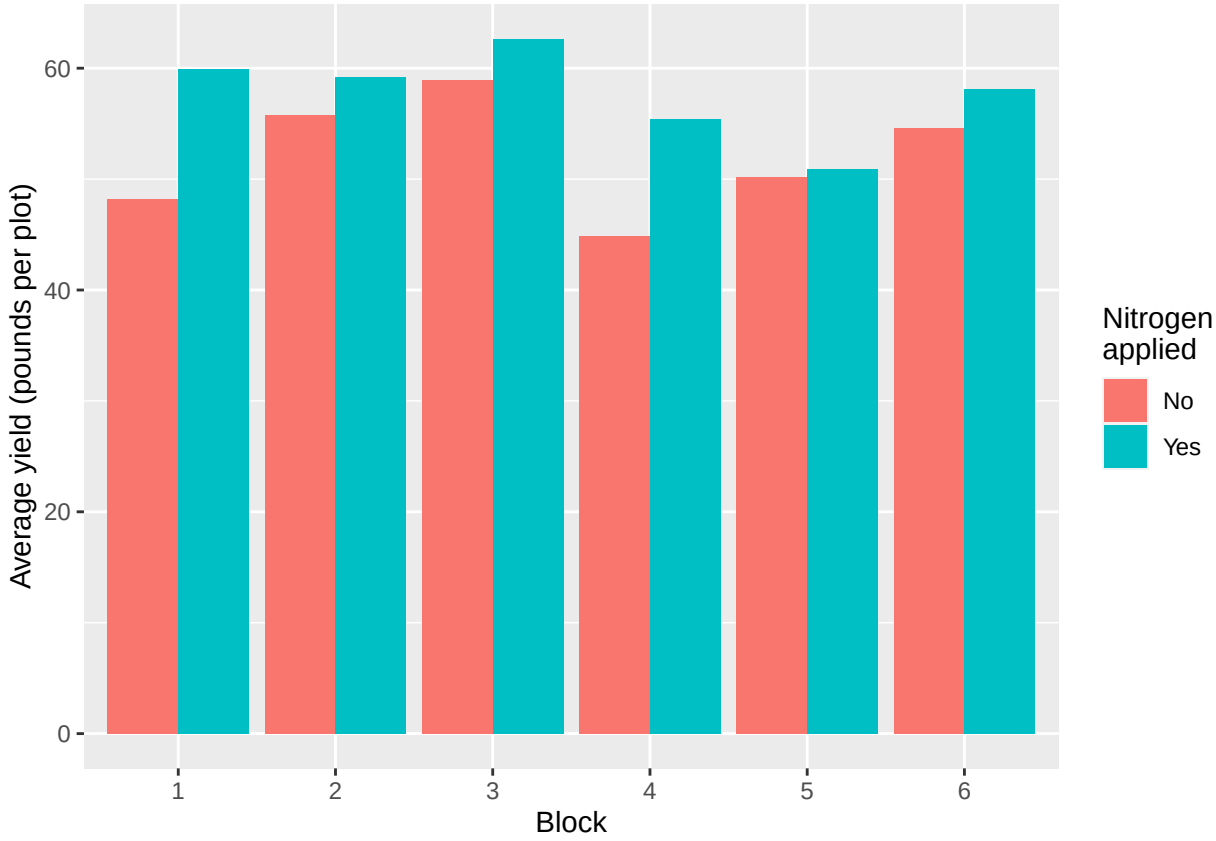
```
library(dplyr)
npk_avg <- npk %>%
  group_by(block, N) %>%
```

```
  summarize(avg_yield = mean(yield))
```

```
## `summarise()` has grouped output by 'block'. You can override using the
## `.groups` argument.
```

Make a plot of the average yield per block for each treatment

```
library(ggplot2)
ggplot(npk_avg, aes(x = factor(block), y = avg_yield, fill = factor(N))) +
  geom_col(position = "dodge") +
  labs(x = "Block", y = "Average yield (pounds per plot)", fill = "Nitrogen") +
  scale_fill_discrete(name = "Nitrogen\napplied", labels = c("No", "Yes"))
```



This code creates a data frame npk_avg that contains the average yield per block for each treatment combination (nitrogen applied or not applied) using the dplyr package. The code then uses ggplot2 to create a bar plot of the average yield per block, with separate bars for the two nitrogen treatments, and with blocks on the x-axis. The position = "dodge" argument in geom_col() ensures that the bars for the two nitrogen treatments are displayed side-by-side for each block.

By including the factor block in the analysis, we account for any variability in yield that may be due to differences between blocks (e.g., soil type, sun exposure, etc.). This allows us to focus on the effect of nitrogen specifically, rather than confounding it with block effects. Additionally, the randomized treatment assignments ensure that any differences in yield between the two nitrogen treatments are not due to systematic differences in the placement of nitrogen-treated plots within blocks.

4C

```r
anova = aov(yield ~ block + N, data = npk)
summary(anova)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## block        5  343.3   68.66   3.395 0.0262 *
## N            1  189.3  189.28   9.360 0.0071 **
## Residuals   17  343.8   20.22
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Df Sum Sq Mean Sq F value Pr(>F)

#block 5 343.3 68.66 3.395 0.0262 * #N 1 189.3 189.28 9.360 0.0071 ** #Residuals 17 343.8 20.22 #— #Signif. codes: 0 ‘’ ***0.001*** ‘’ *0.01* ‘’ 0.05 ‘.’ 0.1 ‘ ’ 1

We see significantly low p-values (0.0262, 0.0071), indicating a significant difference between variable yield and the N and block factors

It wasn't sensible to include the block since there are the same amount of additives in each block which got randomly distributed in the plots

The Friedman test couldn't be applied here since every treatment must be able to be applied to every block, it was not a complete block design.

4D

```r
lm = lm(yield ~., data = npk)
step_model <- stepAIC(lm, direction = "both")
```

```
## Start:  AIC=73.28
## yield ~ block + N + P + K
##
##         Df Sum of Sq    RSS    AIC
## - P      1      8.40 248.59 72.106
## <none>                240.19 73.281
## - K      1     95.20 335.39 79.294
## - block  5    343.29 583.48 84.583
## - N      1    189.28 429.47 85.228
##
## Step:  AIC=72.11
## yield ~ block + N + K
##
##         Df Sum of Sq    RSS    AIC
## <none>                248.59 72.106
## + P      1      8.40 240.19 73.281
## - K      1     95.20 343.79 77.887
## - block  5    343.29 591.88 82.926
## - N      1    189.28 437.87 83.693
```

#Start: AIC=73.28 #yield ~ block + N + P + K

Df Sum of Sq RSS AIC

#- P 1 8.40 248.59 72.106 # 240.18 73.281 #- K 1 95.20 335.39 79.294 #- block 5 343.30 583.48 84.583 #- N 1 189.28 429.47 85.228

#Step: AIC=72.11 #yield ~ block + N + K

Df Sum of Sq RSS AIC

248.59 72.106 #+ P 1 8.40 240.18 73.281 #- K 1 95.20 343.79 77.887 #- block 5 343.30 591.88 82.926 #- N 1 189.28 437.87 83.693

```
lm2 = lm(formula = yield ~ block + N + K + block*N, data = npk)
lm3 = lm(formula = yield ~ block + N + block*K, data = npk)
```

Chosen best model is: lm(formula = yield ~ block + N + K, data = npk)