**Student names:**

**Jurriën Boogert**
**Snr. 2085383**
**Anr. 872949**

**Giorgos Vermaak**
**Snr. 2108346**
**Anr. 482782**

**Ka Wai Tang**
**Snr. 2064845**
**Anr. 870245**

**Exercise 1.**

1. Function Lady_tasting_tea(S)

```
Input:
S: integer defining the number of experiments (counter of samples)

Output:
Proportion for each of the numbers of correctly guessed cups from 0 correct to 8 correct.

    1. Set a vector *Correct_cups* of length 9 (amount of cups)
    2. Repeat the following S times (for i in 1:S)
       Initialize a variable *Correct* with 1 (because R doesn't use zero indexing)
        a. Store a random sample with four times milk and four times tea
           (4x“milk”,4x”tea”) without replacement in the vector *Line_up*
        b. Store a random sample with four times milk and four times tea
           (4x”milk”,4x”tea”) without replacement in the vector *Guesses*
        c. Loop over the amount of cups in Line_up (for cup in 1:length of *Line_up*)
            i.  If *Guesses*[cup] == *Line_up*[cup]
                    1. *Correct* = *Correct* + 1
            ii. Else if Guesses[cup] != Line_up[cup]
                    1. *Correct* = *Correct*
        d. *Correct_cups[Correct]* = *Correct_cups[Correct]* + 1
    3. Return *Correct_cups*/S
```

2. See Appendix *Exercise 1.2* for R code of the implementation of the pseudocode from question 1.

3. Table 1 shows the results of the simulation of the lady tea tasting algorithm in R. Notice that the probability of having at least 6 cups correctly guessed, given that the null-hypothesis of by random chance having 4 out of 8 correct is true, is 0.243 (0.229 + 0.014), which is bigger than the significance level of 0.05 so we can't conclude the lady performs better than random guessing. In other words: there is a probability of 0.243 of having at least 6 cups correct while guessing which is greater than the cut-off value of 0.05, therefore we conclude the lady has no special ability and does perform equally well by guessing. Even with guessing the outcome of the experiment is likely, hence we do not reject the null-hypothesis that the lady guesses in favor of the alternative hypothesis that she can truly discriminate whether milk or the tea infusion was added first.

**Table 1**

*Results of the Lady_tasting_tea algorithm in R*

| Amount of Cups Correct | Estimated Probabilities |
|---|---|
| 0 | 0.014 |
| 1 | 0.000 |
| 2 | 0.229 |
| 3 | 0.000 |
| 4 | 0.514 |
| 5 | 0.000 |
| 6 | 0.229 |
| 7 | 0.000 |
| 8 | 0.014 |
| Total | 1.000 |

*Note.* The estimated probabilities are calculated as the sum of the number of correctly guessed cups divided by 1 million (the amount of trials). The outcome is the proportion.

**Exercise 2.**

1. The population *X* is normally distributed with a population mean of 0 and a variance equal to 5. The population parameter estimated by the three variance estimators therefore has a value of 5, the population variance.

2. The var() function in R is calculated by summing the squares of the difference between an individual data point and the mean of all data points from the sample, divided by *n-1* (the sample size minus 1). The outcome of var() is thus the unbiased estimator of the variance.

The maximum likelihood estimator can be calculated by multiplying the outcome of var() with *(n-1)/n*. The minMSE estimator can be calculated by multiplying the outcome of var() by *(n-1)/(n+1).* In both cases you first multiply by *(n-1)* to get the sum of squared differences between individual samples and the mean of these samples. Then you divide by the denominator from the corresponding estimator. In case of the maximum likelihood estimator that is *n*. In case of the minMSE estimator that is *(n+1).*

3. See appendix *Exercise 2.3* for the R code of the simulation for the three different estimators.

**Table 2**

*Comparison of the performance of the three estimators.*

| Sample Size (*n*) | Maximum Likelihood estimator | Unbiased variance estimator | MinMSE estimator |
|---|---|---|---|
| | Amount of Samples *S = 1* | | |
| 3 | 5.861 | 8.792 | 4.396 |
| | Difference with the population variance (bias) | | |
| | 0.861 | 3.792 | -0.604 |
| | Amount of Samples *S = 10000* | | |
| | Average value of the estimate | | |
| 3 | 3.309 | 4.964 | 2.482 |
| 20 | 4.751 | 5.002 | 4.525 |
| 200 | 4.972 | 4.997 | 4.947 |
| | Estimated bias | | |
| 3 | -1.691 | -0.036 | -2.518 |
| 20 | -0.249 | 0.002 | -0.475 |
| 200 | -0.028 | -0.003 | -0.053 |
| | Variance of the estimate | | |
| 3 | 11.103 | 24.981 | 6.245 |
| 20 | 2.378 | 2.635 | 2.157 |
| 200 | 0.249 | 0.252 | 0.247 |
| | Mean squared error | | |
| 3 | 13.960 | 24.980 | 12.586 |
| 20 | 2.440 | 2.634 | 2.382 |
| 200 | 0.250 | 0.252 | 0.249 |

*Note.* The samples are taken from a normal distribution with mean of 0 and variance of 5.

4. From the results of the previous simulation we can conclude that the Unbiased estimator, as the name suggests, has the lowest bias from the three estimators. The MinMSE estimator has the largest bias from the three estimators. Second largest bias has the Maximum likelihood estimator. The differences are especially apparent when the sample size is small. For all estimators, the bigger the sample size, the smaller the bias.

For every sample size the variance of the estimate is the smallest for the MinMSE estimator. After comes the Maximum likelihood estimator and the Unbiased estimator has the largest variance. Also here applies that the differences are especially apparent when the sample size is small. Again, for all estimators, the larger the sample size, the smaller the variance. All three estimator variances converge to practically the same value when sample size is large.

The average value of the estimate is the smallest for the MinMSE estimator for all sample sizes. After comes the Maximum likelihood estimator and als last comes the Unbiased estimator. Again, the bigger the sample size, the smaller the estimate.

In practice you use the point estimate of the variance estimator, which means you do not repeatedly draw samples from the population distribution. The MC simulation is just meant to show how well the different estimators behave. With this in mind it is preferable to have an estimator which does not deviate too much from the true population variance. When taking into account both the bias and the variance we can recommend using the MinMSE estimator for smaller sample size (n=3) as concluded from looking at table 2, because this estimator has the lowest MSE (mean squared error). When using a larger sample size (n=200) all values (bias, variance and MSE) of all estimators converge to each other and there is no significant difference and therefore preference for one over the other. When looking at table 3 you may think the unbiased estimator is the best, because it has the lowest estimated bias, but remember that this is the average of many estimates minus the true value. This estimator actually has a high variance for smaller sample size (n=3), which makes it hard to rely on as a (point) estimate for the true value. The MinMSE estimator has a higher bias for smaller sample size (n=3), but also a smaller variance when conducting many ‘experiments’, which makes it a better estimate to rely on for smaller sample sizes. In between to two more ‘extreme’ estimators lies the biased estimator, which more or less performs on average.

**Appendix**

R version 4.2.1

*Exercise 1.2*

```
set.seed(12)

lady_tasting_tea <- function(S){
  Correct_cups <- vector(mode = "numeric", length = 9)
  for (i in 1:S){
    Correct <- 1
    Line_up <- sample(x = c("milk","milk","milk","milk","tea","tea","tea","tea"), size = 8, replace =
FALSE)
    Guesses <- sample(x = c("milk","milk","milk","milk","tea","tea","tea","tea"), size = 8, replace =
FALSE)
    for (cup in 1:length(Line_up)){
      if (Guesses[cup] == Line_up[cup]){
        Correct = Correct + 1
        }
      else if (Guesses[cup] != Line_up[cup]){
        Correct = Correct
        }
    }
    Correct_cups[Correct] = Correct_cups[Correct] + 1
  }
  return (Correct_cups/S)
}

lady_tasting_tea(1000000)
```

*Exercise 2.3*

```
set.seed(12052022)

n <- 200
sample <- rnorm(n=n,mean=0,sd=sqrt(5))

ML <- var(sample) * ((n-1)/n)
unb <- var(sample)
minmse <- var(sample) * ((n-1)/(n+1))

ML_vec <- vector(mode = 'numeric', length = 10000)
```

```
unb_vec <- vector(mode = 'numeric', length = 10000)
minmse_vec <- vector(mode = 'numeric', length = 10000)

for (i in 1:10000){
  sample <- rnorm(n=n,mean=0,sd=sqrt(5))

  ML <- var(sample) * ((n-1)/n)
  unb <- var(sample)
  minmse <- var(sample) * ((n-1)/(n+1))

  ML_vec[i] <- ML
  unb_vec[i] <- unb
  minmse_vec[i] <- minmse
}

#average value of estimate
mean(ML_vec)
mean(unb_vec)
mean(minmse_vec)

#variance of estimate
sum((mean(ML_vec) - ML_vec)^2)/9999
sum((mean(unb_vec) - unb_vec)^2)/9999
sum((mean(minmse_vec) - minmse_vec)^2)/9999

#mse
sum((5 - ML_vec)^2)/10000
sum((5 - unb_vec)^2)/10000
sum((5 - minmse_vec)^2)/10000
```