MATH5835M Statistical Computing
Exercise Sheet 2 (answers)

https://www1.maths.leeds.ac.uk/~voss/2023/MATH5835M/
Jochen Voss, J.Voss@leeds.ac.uk

2023/24, semester 1

Answer 4.

a) We can generate the samples using `rexp()` as follows:

```
n <- 10
mu <- 2
X <- rexp(n, rate = 1/mu)
```

Formula (1) then translates into R in a straightforward way: `mean(X)` computes $\bar{x}$, the call `sqrt(n)` gives $\sqrt{n}$ and `sd(X)` computes the sample standard deviation s_x:

```
T <- sqrt(n) * (mean(X) - mu) / sd(X)
```

b) We can use a loop of the form `for (i in 1:N) {...}` to execute the code from part (a) N times. For each iteration of the loop, we need to generate samples using `rexp()` and then compute T. Finally, we need to store the computed T-values in a vector (`res` in the code below). Here we write the code as an R function (see appendix B.3.2 of the book), so that it is easy to re-use:

```
gen.T.sample <- function(N, n=10, mu=2) {
    res <- numeric(N)
    for (i in 1:N) {
        X <- rexp(n, rate = 1/mu)
        res[i] <- sqrt(n) * (mean(X) - mu) / sd(X)
    }
    res
}
```

When we try out this function, we see that it indeed produces the required number of samples:

```
> gen.T.sample(5)
[1] -0.5987437  0.9067587  0.1926406 -2.3336042  0.2054727
```

We can call `gen.T.sample(100)` to get $N = 100$ samples:

```
N <- 100
T <- gen.T.sample(N)
```

c) A type I error occurs whenever $|T| > t_{n-1}(0.975)$. We can easily test this condition using R:

```
> crit <- qt(0.975, 9)
> abs(T) > crit
  [1] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
 [11] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
 [21] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [31] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [41] FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE
 [51] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [71] FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE
 [81] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [91] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
> sum(abs(T) > crit)
[1] 8
```

So we see that 8 out of the 100 samples lead to a type I error here. To estimate the probability of type I errors, we divide this number by N, to get $\frac{1}{N} \sum_{j=1}^{N} 1_{|T_j| > t_{n-1}(0.975)}$:

```
> mean(abs(T) > crit)
[1] 0.08
```

and we can estimate the RMSE as

```
> sqrt(var(abs(T) > crit) / N)
[1] 0.02726599
```

d) As a rule of thumb, the true value is likely to be within two RMSEs of the estimate, so we will likely have $p > 0.080 - 2 \cdot 0.027 = 0.026$. Thus, $N = 100$ is too small to conclude that $p > 0.05$. Since N is still relatively small, the values of the estimate and the RMSE are not very precise, and we should not try to get the required N from these values. Instead, we increase N by a factor of 100 and try again:

```
> N <- 10000
> T <- gen.T.sample(N)
> mean(abs(T) > crit)
[1] 0.0991
> sqrt(var(abs(T) > crit) / N)
[1] 0.002988112
> 0.0991 - 2 * 0.002988
[1] 0.093124
```

Clearly, this N is now large enough, and since the computation takes less than a second on my laptop, there is no need to search for smaller N which also would do the job.

e) We can use a loop to repeat the code above for different values of n to generate the required plot. The smallest possible value of n is $n = 2$, since for smaller n the sample standard deviation (in the definition of t) is not defined. We also need to be careful to re-compute the critical value for every n:

```
n <- c(2, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100)
prob <- numeric(length(n))
rmse <- numeric(length(n))
for (i in seq_along(n)) {
    ni <- n[i]
    crit <- qt(0.975, ni - 1)
    T <- gen.T.sample(N, ni)
    prob[i] <- mean(abs(T) > crit)
    rmse[i] <- sqrt(var(abs(T) > crit) / N)
}
plot(n, prob, ylim = c(0, 0.12), ylab = "P(type I error)")
arrows(n, y0 = prob - 2*rmse, y1 = prob + 2*rmse,
       code = 3, angle = 90, length = .05)
abline(h = 0.05, lty=2)
```

(see figure 1.) We (mis-)use the `arrows()` function to draw error bars, indicating the range of plus/minus two RMSEs. From the error bars we see that the peak around $n = 10$ is likely real, whereas the smaller peak around $n = 70$ may be an artefact of the Monte-Carlo error. It is known that when n goes to ∞, the probability of type I errors for this t-test converges to 5%; this is consistent with the markers in the plot getting closer to the dashed horizontal line as n increases.

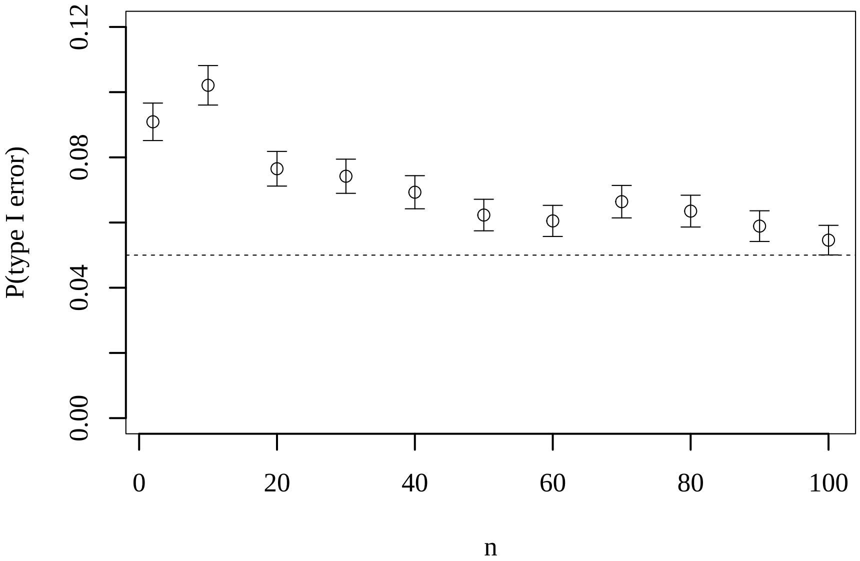

Figure 1. The estimated probability of type I errors as a function of n. The intervals indicated by the error bars have a half-width of two root mean-squared errors. See question 4(e) for details.