Quantitative Methods II – Case Study 2026

Group B3 – Course 4902
Tobias Schützeneder, Magnus Schachner, Vsevolod Stepanov, Eva Vedernyikova

20 May 2026

1 Case I: Monte Carlo Valuation of a Leveraged Barrier Certificate

In this first case study we use a Monte Carlo simulation to find the fair present value of a financial product called a Leveraged Barrier Certificate. The idea behind the certificate is the following: after one year the holder receives a payoff that depends on the price of an underlying stock. If the stock price ends at or above a fixed barrier B, the holder receives λ times the difference between the final price S_F and the strike price K; if the price ends below the barrier, the holder only receives a fixed rebate R. Because the future stock price is random, the payoff is random as well, and the value of the certificate is defined as the expected payoff $V = E[f(S_F)]$.

The difficulty is that this expected value has no simple closed-form formula. The Monte Carlo method offers a way around this: instead of computing the expectation directly, we simulate a large number of possible future stock prices, calculate the payoff in each case, and take the average. By the law of large numbers, this average gets closer to the true expected value as the number of simulations grows. Throughout the case we set a fixed random seed immediately before each simulation, as required by the instructions, so that anyone re-running the code obtains exactly the same numbers and plots.

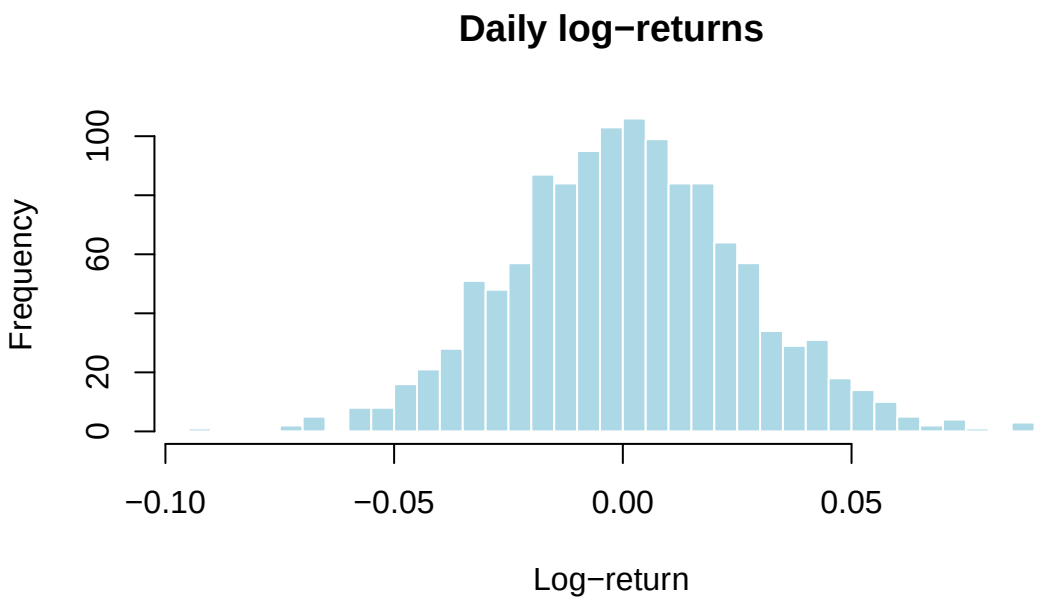
1.1 Data preparation

Before we can simulate anything, we need estimates for the average return and the volatility of the stock, and we obtain these from a series of historical prices. We load the file `asset_data.csv` into R with the `read.csv()` function; the prices themselves are stored in the column `Price`. The quantity we actually need is not the price but the daily log-return, defined as $R_k = \ln(S_{k+1}) - \ln(S_k)$, i.e. the difference between the logarithms of two consecutive prices. A convenient way to compute the whole series at once is the `diff()` function: applied to a vector, it returns the differences between successive elements, so applying it to the logarithm of the prices gives exactly the log-return series. We then use `hist()` to draw a histogram, which shows how the returns are distributed.

```
asset  <- read.csv("asset_data.csv")
prices <- asset$Price
logret <- diff(log(prices))        # R_k = ln(S_{k+1}) - ln(S_k)
length(logret)
```

```
## [1] 1259
```

```r
hist(logret, breaks = 40, col = "lightblue", border = "white",
     main = "Daily log-returns", xlab = "Log-return")
```



Interpretation. The price series contains 1260 observations, so we obtain 1259 log-returns: one fewer than the number of prices, because the first return already needs two prices. The histogram is roughly symmetric and centred slightly above zero, with most returns lying between -0.05 and 0.05. This bell-shaped pattern matters, because the price model we use later assumes that the log-returns are normally distributed.

1.2 Parameter estimation

The price formula given in the assignment depends on two unknown population parameters: the mean μ and the standard deviation σ of the daily log-returns. Since we do not know their true values, we estimate them from the sample, using the sample mean and the sample standard deviation. In R these are computed with `mean()` and `sd()`. The function `sd()` divides by $n - 1$ rather than n, which gives the unbiased sample standard deviation we used in class.

```r
mu    <- mean(logret)
sigma <- sd(logret)
c(mu = mu, sigma = sigma)
```

```
##          mu       sigma
## 0.001128283 0.025336534
```

Interpretation. The estimates are $\hat{\mu} \approx 0.00113$ and $\hat{\sigma} \approx 0.02534$. In words, the stock earned an average log-return of about 0.11% per day, with a daily standard deviation of roughly 2.5%. These two values now take the place of μ and σ in all of the simulations that follow.

1.3 Simulation setup

We now fix the contract values given in the assignment and prepare the simulation. We draw $n = 10{,}000$ values of a standard normal random variable Z with the function `rnorm()`, which generates random numbers from a normal distribution (by default with mean 0 and standard deviation 1). To make this draw reproducible, we call `set.seed()` with a fixed number directly before `rnorm()`. Each simulated stock price is then obtained by plugging Z into the price formula. Because R works with whole vectors at once, a single line computes all 10,000 prices: the expression is applied element by element to `Z` and returns a vector `SF` of the same length.

```
S0 <- 150; K <- 140; B <- 160; lambda <- 1.5; rebate <- 20
n  <- 10000

set.seed(2026)
Z  <- rnorm(n)
SF <- S0 * exp((mu - sigma^2 / 2) * 252 + sigma * sqrt(252) * Z)
summary(SF)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   41.93  140.50  183.68  199.81  240.73  864.38
```

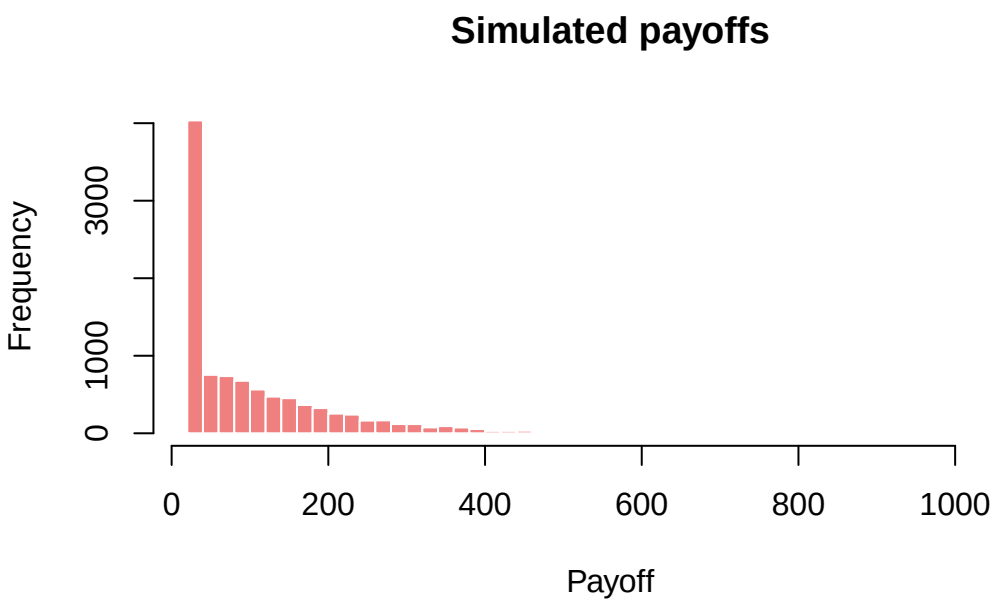
Interpretation. The `summary()` output gives the five-number summary of the 10,000 simulated year-end prices. They range from well below to well above the starting price of 150, which reflects the uncertainty about where the stock will stand in one year. (In the code we call the rebate `rebate` instead of `R`, so that it cannot be confused with the name of the programming language.)

1.4 Payoff generation

Next we turn each simulated price into the payoff the investor would receive. The payoff is defined in two pieces – the leveraged amount $\lambda(S_F - K)$ if the barrier is reached, and the fixed rebate otherwise – so the natural tool is the `ifelse()` function. It takes three arguments: a condition, the value to return where the condition is TRUE, and the value to return where it is FALSE. Applied to the whole vector `SF`, it checks the barrier condition for every simulated price at once and returns the correct payoff for each.

```
payoff <- ifelse(SF >= B, lambda * (SF - K), rebate)

hist(payoff, breaks = 50, col = "lightcoral", border = "white",
     main = "Simulated payoffs", xlab = "Payoff")
```



Interpretation. The histogram of the payoffs has a characteristic shape made of two separate parts. There is a tall, narrow bar exactly at 20, which collects all the scenarios in which the stock missed the barrier and the investor received only the rebate. The remaining payoffs all lie at $\lambda(B - K) = 30$ or higher, because the smallest possible leveraged payoff occurs right at the barrier $S_F = B = 160$. No payoff falls between 20 and 30, since the payoff function jumps at the barrier. This gap, and the resulting asymmetric two-part distribution, are exactly why the expected value is awkward to compute by hand and why a simulation is so convenient here.

1.5 Approximating the value V

We can now estimate the value of the certificate. As explained at the start, the Monte Carlo estimate of $V = E[f(S_F)]$ is simply the average of all the simulated payoffs, which we compute with `mean()`. The law of large numbers guarantees that this average approaches the true expected value as the number of simulations increases.

```
V_hat <- mean(payoff)
V_hat
```

```
## [1] 105.4748
```

Interpretation. With 10,000 simulated payoffs we obtain an estimated value of $\hat{V}_{10000} \approx 105.47$. We should keep in mind that this number is itself the result of random draws: with a different seed the estimate would change slightly. The next section looks at how large this random variation is.

1.6 Variance and convergence

1.6.1 Estimated variance of the estimator

Because $\hat{V}_n$ is an average of n independent payoffs, it is itself a random variable, and we describe its precision by its variance. As we saw in class, the variance of a sample mean equals the variance of a single observation divided by n. We therefore estimate the variance of one payoff with the sample variance `var()` and divide it by n. The square root of this quantity is the standard error, i.e. the typical size of the deviation between our estimate and the true value.

```
var_V <- var(payoff) / n
c(V_hat = V_hat, variance = var_V, std_error = sqrt(var_V))
```

```
##      V_hat   variance  std_error
## 105.474814   1.268546   1.126298
```

Interpretation. The estimated variance of $\hat{V}_n$ is about 1.27, which corresponds to a standard error of roughly 1.13. In other words, our estimate of the certificate's value is accurate to within about one unit, which is quite precise compared to a value of around 105.

1.6.2 Variance for different sample sizes

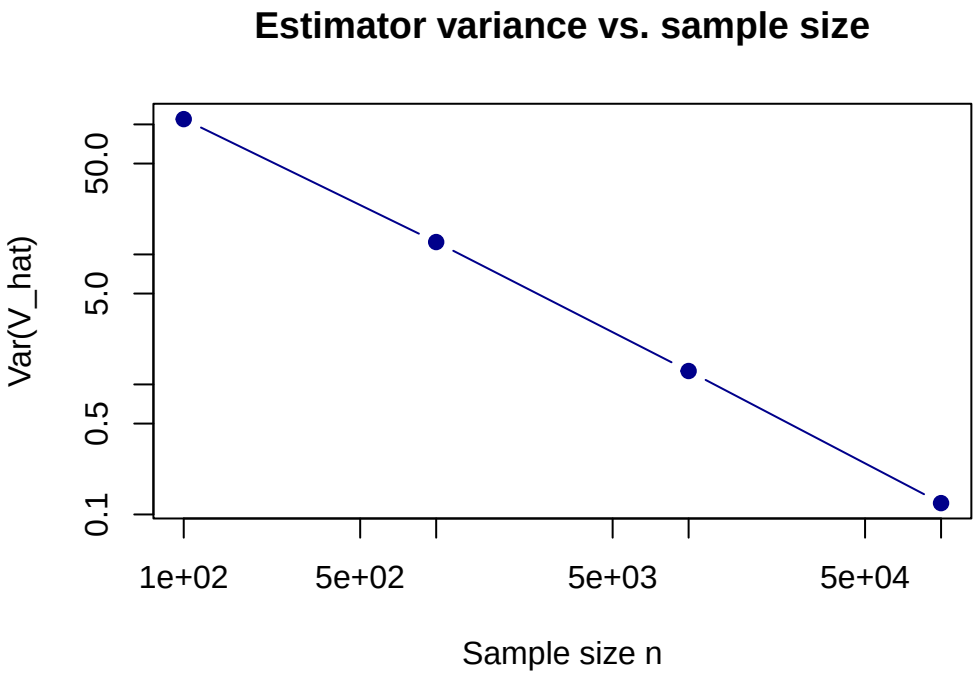
To see how the precision depends on the number of simulations, we repeat the whole experiment for four sample sizes, $n = 100,\ 1000,\ 10000$ and 100000, and record the estimated variance of $\hat{V}_n$ in each case. We use a `for`-loop that runs through the sample sizes, and we reset the same seed inside the loop before every draw so the experiment stays reproducible. Because both the sample sizes and the variances cover several orders of magnitude, we plot them on logarithmic axes (the argument `log = "xy"`), which makes the relationship easier to see.

```
ns   <- c(100, 1000, 10000, 100000)
varV <- numeric(length(ns))

for (i in seq_along(ns)) {
  set.seed(2026)
  Zi  <- rnorm(ns[i])
  SFi <- S0 * exp((mu - sigma^2 / 2) * 252 + sigma * sqrt(252) * Zi)
  fi  <- ifelse(SFi >= B, lambda * (SFi - K), rebate)
  varV[i] <- var(fi) / ns[i]
}
cbind(n = ns, variance = varV)
```

```
##          n    variance
## [1,] 1e+02 109.8077252
## [2,] 1e+03  12.4502118
## [3,] 1e+04   1.2685461
## [4,] 1e+05   0.1222312
```

```r
plot(ns, varV, log = "xy", type = "b", pch = 19, col = "darkblue",
     xlab = "Sample size n", ylab = "Var(V_hat)",
     main = "Estimator variance vs. sample size")
```



Interpretation. The table and the plot show the same thing: every time we multiply the sample size by ten, the variance shrinks by roughly a factor of ten as well. On the logarithmic plot this appears as a straight line with a slope of about -1.

1.6.3 Connection with the Central Limit Theorem

This behaviour is a direct consequence of the Central Limit Theorem, which we studied in class through simulation. There we repeatedly drew samples from a population, computed the sample mean of each sample, and saw that the distribution of these sample means becomes more and more bell-shaped and, importantly, narrower as the sample size grows. Our estimator $\hat{V}_n$ is exactly such a sample mean – the average of n independent payoffs – so the same reasoning applies here.

The Central Limit Theorem tells us that $\hat{V}_n$ is approximately normally distributed around the true value V, with a standard error equal to $\sigma_f/\sqrt{n}$, where σ_f is the standard deviation of a single payoff. This is the reason the variance falls like $1/n$: the standard error has $\sqrt{n}$ in the denominator, so the variance has n. The practical consequence is the well-known "square-root rule" of the Monte Carlo method: because the error decreases only with the square root of the sample size, reducing it by half requires four times as many simulations. The same normal approximation would also let us build a confidence interval for V from $\hat{V}_n$ and its standard error, just as we did for ordinary sample means in class.

1.7 Risk metrics

Finally, we look at the certificate from the point of view of risk. Suppose the investor pays 30 USD to buy it. The realised profit or loss (P/L) in each simulated scenario is then the payoff minus this cost.

1.7.1 Expected profit and loss

```
cost <- 30
pl   <- payoff - cost
mean(pl)
```

```
## [1] 75.47481
```

Interpretation. The expected profit or loss is the average of all simulated P/L values, which equals our estimated value minus the cost, about 75.47 USD. Since this is clearly positive, the certificate is profitable on average at a price of 30 USD under our estimated parameters. As always with an expected value, though, this is an average over many scenarios and says nothing about what happens in a single case.

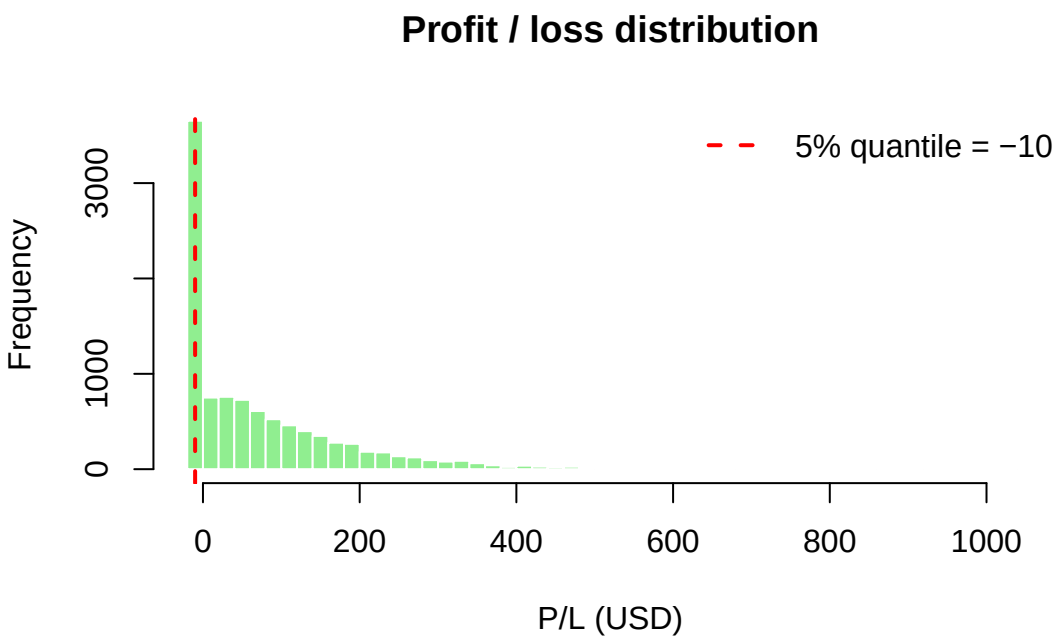
1.7.2 Value at Risk at level 5%

To describe the downside risk we use the Value at Risk (VaR). The VaR at level $\alpha = 5\%$ is based on the 5% quantile of the P/L distribution, that is the value below which only 5% of the simulated outcomes fall. We obtain it with the `quantile()` function. To visualise it, we draw the histogram of the P/L values and add a vertical line at the 5% quantile with `abline()`, together with a short `legend()`.

```
VaR <- quantile(pl, 0.05)
VaR
```

```
##  5%
## -10
```

```
hist(pl, breaks = 50, col = "lightgreen", border = "white",
     main = "Profit / loss distribution", xlab = "P/L (USD)")
abline(v = VaR, col = "red", lwd = 2, lty = 2)
legend("topright", paste("5% quantile =", round(VaR, 2)),
       col = "red", lwd = 2, lty = 2, bty = "n")
```



Interpretation. The 5% quantile turns out to be exactly -10. The reason is structural: whenever the stock misses the barrier, the investor receives the rebate of 20 and therefore loses $20-30=-10$, and the simulation misses the barrier in about 37% of all scenarios, far more than 5%. The worst 5% of outcomes are therefore all rebate cases, each with a loss of 10 USD. Interpreted as a risk measure, a Value at Risk of 10 USD at the 5% level means that, under our model, we can be at least 95% sure that the loss will not exceed 10 USD. For this particular certificate this also happens to be the largest loss that can occur at all, because the rebate is the smallest payoff the product can ever pay.

2 Case II: Lifestyle and Sleeping Quality

In the second case study we analyse a dataset on lifestyle and sleeping quality. The aim is to describe the data, to investigate how sleep quality and stress differ across occupations and genders, to test two claims made by a sleep doctor, and finally to compare sleep duration and age across the different sleep-disorder categories. We work through the tasks in the order given in the assignment.

2.1 Descriptive analysis

2.1.1 Loading the data and first overview

In order to provide a descriptive analysis of the dataset, we start by loading it into R and inspecting its overall structure. Because we want to produce frequency tables and use `summary()` later on, we set the argument `stringsAsFactors = TRUE` when reading the file. This argument instructs R to treat the text columns (gender, occupation and sleep disorder) as factors, i.e. categorical variables with a fixed set of levels. To get a sense of the size of the dataset we use the `dim()` function, which returns the number of rows and columns of a data frame.

```
set.seed(2026)
sleep <- read.csv("sleep.csv", stringsAsFactors = TRUE)
dim(sleep)
```

```
## [1] 374   8
```

In our case, the dataset contains 374 observations and 8 variables. As a next step, we dig deeper into the type of each variable using the `str()` function and check whether any values are missing.

```
str(sleep)
```

```
## 'data.frame':    374 obs. of  8 variables:
##  $ gender        : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2 ...
##  $ age           : int  27 28 28 28 28 28 29 29 29 29 ...
##  $ occupation    : Factor w/ 11 levels "Accountant","Doctor",..: 10 2 2 7 7 10 11 2 2
↪  2 ...
##  $ sleep_duration: num  6.1 6.2 6.2 5.9 5.9 5.9 6.3 7.8 7.8 7.8 ...
##  $ sleep_quality : int  6 6 6 4 4 4 6 7 7 7 ...
##  $ stress_level  : int  6 8 8 8 8 8 7 6 6 6 ...
##  $ daily_steps   : int  4200 10000 10000 3000 3000 3000 3500 8000 8000 8000 ...
##  $ sleep_disorder: Factor w/ 3 levels "Insomnia","None",..: 2 2 2 3 3 1 1 2 2 2 ...
```

```
colSums(is.na(sleep))
```

```
##         gender            age     occupation sleep_duration  sleep_quality
##              0              0              0              0              0
##   stress_level    daily_steps sleep_disorder
##              0              0              0
```

With the expression `is.na(sleep)` we produce a logical table that has the same shape as the dataset: it marks missing entries as TRUE and valid ones as FALSE. Since R treats TRUE as 1 and FALSE as 0, we can then count the missing values per column with the integrated function `colSums()`. All counts are zero, which means the dataset is complete. We do need to be careful, however, because the label "None" appearing in `sleep_disorder` is a genuine category (meaning "no diagnosed disorder") rather than a missing value.

2.1.2 Variable types

The eight variables fall into two groups. The categorical variables (factors) are gender (2 levels), occupation (11 levels) and sleep disorder (3 levels: None, Insomnia and Sleep Apnea). The numeric variables are age, sleep duration, sleep quality, stress level and daily steps. Although sleep quality and stress level are technically ordinal variables, since they are restricted to integer values between 1 and 10, we nevertheless treat them as numeric when computing means, correlations and visualisations, which is in line with standard practice for Likert-type data.

2.1.3 Numeric variables

The following call summarises the five numeric variables, reporting the minimum, the three quartiles, the mean and the maximum of each.

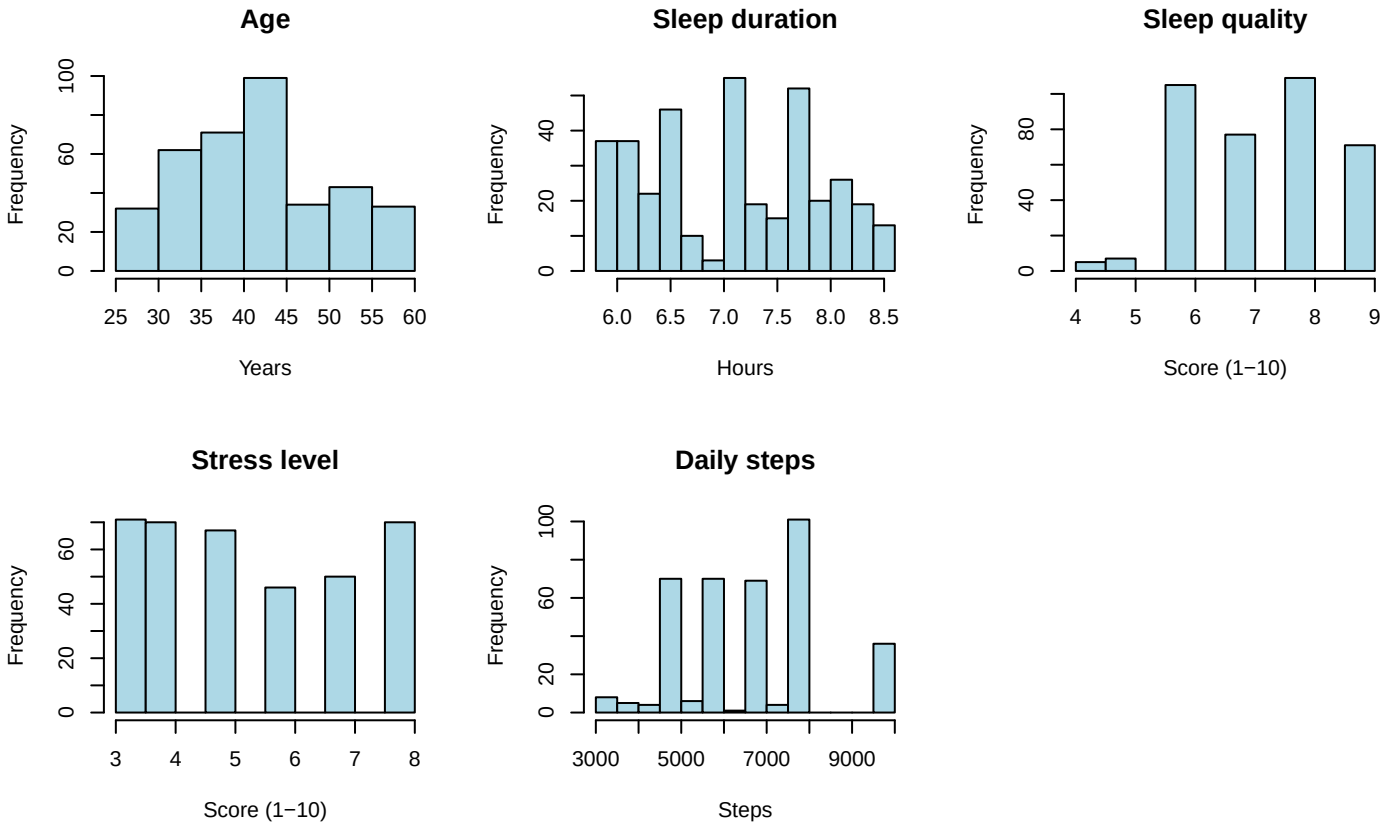
```
summary(sleep[, c("age", "sleep_duration", "sleep_quality",
                  "stress_level", "daily_steps")])
```

```
##       age        sleep_duration  sleep_quality    stress_level
##  Min.   :27.00   Min.   :5.800   Min.   :4.000   Min.   :3.000
##  1st Qu.:35.25   1st Qu.:6.400   1st Qu.:6.000   1st Qu.:4.000
##  Median :43.00   Median :7.200   Median :7.000   Median :5.000
##  Mean   :42.18   Mean   :7.132   Mean   :7.313   Mean   :5.385
##  3rd Qu.:50.00   3rd Qu.:7.800   3rd Qu.:8.000   3rd Qu.:7.000
##  Max.   :59.00   Max.   :8.500   Max.   :9.000   Max.   :8.000
##   daily_steps
##  Min.   : 3000
##  1st Qu.: 5600
##  Median : 7000
##  Mean   : 6817
##  3rd Qu.: 8000
##  Max.   :10000
```

We then visualise the five distributions using histograms. The command `par(mfrow = c(2, 3))` arranges the panels in a grid of two rows and three columns, so that all of them appear together.

```
par(mfrow = c(2, 3))
hist(sleep$age, main = "Age", xlab = "Years", col = "lightblue")
hist(sleep$sleep_duration, main = "Sleep duration",
     xlab = "Hours", col = "lightblue")
hist(sleep$sleep_quality, main = "Sleep quality",
     xlab = "Score (1-10)", col = "lightblue")
```

```r
hist(sleep$stress_level, main = "Stress level",
     xlab = "Score (1-10)", col = "lightblue")
hist(sleep$daily_steps, main = "Daily steps", xlab = "Steps", col = "lightblue")
par(mfrow = c(1, 1))
```



Interpretation. Age is roughly evenly spread across the working-age range, from 27 to 59, with a mean of approximately 42. Although an irregular bar pattern can be observed for sleep duration, the data is concentrated around the 7-hour mark, which is in line with common medical recommendations. Sleep quality is clearly skewed towards the upper end of the scale, with a median of 7, indicating that most individuals report reasonably good sleep. The stress level is also evenly distributed across the range from 3 to 8, so no clear extremes or outliers can be observed. Similar to sleep duration, daily steps show several gaps, likely because the values are reported in rounded numbers. They range from 3,000 to 10,000 steps, peaking around 8,000.

2.1.4 Categorical variables

We next examine the three categorical variables through frequency tables and bar plots. The `table()` function counts how often each category occurs, and `barplot()` turns these counts into a bar chart. For the occupation variable we sort the categories from the most to the least frequent with `sort()`, which makes the chart easier to read, and we rotate the labels with `las = 2` so that the long occupation names fit.

```r
table(sleep$gender)
```

```
##
## Female   Male
##    185    189
```

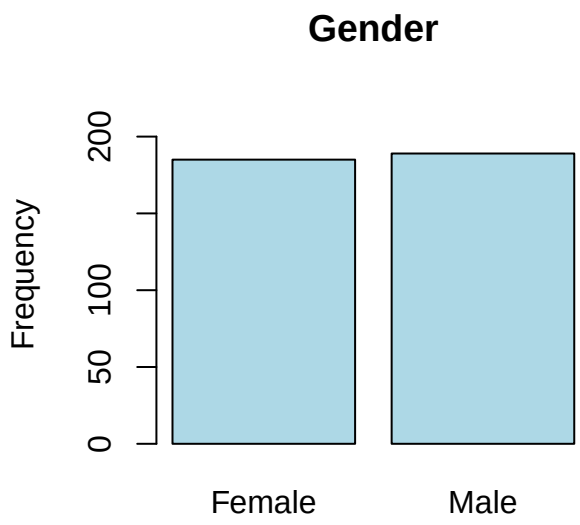
```r
table(sleep$sleep_disorder)
```

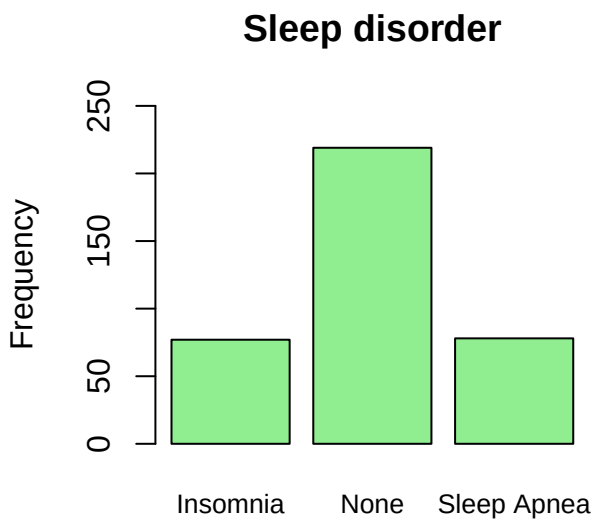
```
##
##    Insomnia        None Sleep Apnea
##          77         219          78
```

```r
table(sleep$occupation)
```

```
##
##           Accountant               Doctor             Engineer
##                   37                   71                   63
##               Lawyer              Manager                Nurse
##                   47                    1                   73
## Sales Representative          Salesperson            Scientist
##                    2                   32                    4
##    Software Engineer              Teacher
##                    4                   40
```

```r
par(mfrow = c(1, 2))
barplot(table(sleep$gender), main = "Gender", ylab = "Frequency",
        col = "lightblue", ylim = c(0, 220))
barplot(table(sleep$sleep_disorder), main = "Sleep disorder",
        ylab = "Frequency", col = "lightgreen", ylim = c(0, 250),
        cex.names = 0.8)
```






```r
par(mfrow = c(1, 1))
```

```r
op <- par(mar = c(9, 4, 3, 1))            # extra bottom space for the long labels
barplot(sort(table(sleep$occupation), decreasing = TRUE),
        main = "Occupation", ylab = "Frequency",
        las = 2, cex.names = 0.8, col = "lightcoral")
```


```
par(op)
```

Interpretation. The sample is almost perfectly balanced by gender (185 females versus 189 males). The same applies to Insomnia (77 cases) and Sleep Apnea (78 cases), which together represent approximately 41% of the observations. The remaining share is taken by the "None" category, comprising 219 cases and making up 59% of the sample. Lastly, the occupation variable is highly imbalanced, as Nurse (73), Doctor (71) and Engineer (63) together account for more than half of the dataset. Meanwhile, Manager, Sales Representative, Scientist and Software Engineer each occur fewer than five times. This information is important to keep in mind for the occupation × gender analysis later, where some combinations rely on very small subsamples and therefore need to be interpreted with caution.

2.2 Analysis by occupation and gender

2.2.1 Best sleep quality and lowest stress by occupation

Occupational differences can be examined by computing the mean of sleep quality and stress level for each occupation group. In R this is implemented using the `aggregate()` function. In simple terms, a numeric variable is split by a categorical variable, and a chosen function – such as the mean in this case – is then applied to each subgroup. The expression `sleep_quality ~ occupation` should be read as "sleep quality grouped by occupation". We sort the output so that the highest and lowest values are easy to spot.

```r
quality_by_occ <- aggregate(sleep_quality ~ occupation, sleep, mean)
quality_by_occ[order(-quality_by_occ$sleep_quality), ]
```

```
##              occupation sleep_quality
## 3              Engineer      8.412698
## 4                Lawyer      7.893617
## 1            Accountant      7.891892
## 6                 Nurse      7.369863
## 5               Manager      7.000000
## 11              Teacher      6.975000
## 2                Doctor      6.647887
## 10    Software Engineer      6.500000
## 8           Salesperson      6.000000
## 9             Scientist      5.000000
## 7  Sales Representative      4.000000
```

```r
stress_by_occ <- aggregate(stress_level ~ occupation, sleep, mean)
stress_by_occ[order(stress_by_occ$stress_level), ]
```

```
##              occupation stress_level
## 3              Engineer     3.888889
## 11              Teacher     4.525000
## 1            Accountant     4.594595
## 5               Manager     5.000000
## 4                Lawyer     5.063830
## 6                 Nurse     5.547945
## 10    Software Engineer     6.000000
## 2                Doctor     6.732394
## 8           Salesperson     7.000000
## 9             Scientist     7.000000
## 7  Sales Representative     8.000000
```

Interpretation. Looking at the output, among the occupations represented in the sample Engineers report the highest sleep quality (8.41 out of 10), together with the lowest average stress level (3.89 out of 10). At the other end of the spectrum the same pattern appears in reverse: Sales Representatives and Scientists report both the lowest sleep quality and the highest stress levels. These results need to be treated with caution, however, since the sample contains only 2 Sales Representatives and 4 Scientists, which means their group means are based on very few observations and are therefore unreliable.

2.2.2 Sleep disorders by gender

Good tools to identify the most common sleep disorder overall and by gender are, on the one hand, frequency tables and, on the other hand, contingency tables. The first is used to examine the marginal distribution, whereas the second is used to analyse the joint distribution of gender and sleep disorder.

```r
table(sleep$sleep_disorder)
```

```
##
##    Insomnia        None Sleep Apnea
##          77         219          78
```

```r
table(sleep$gender, sleep$sleep_disorder)
```

```
##
##          Insomnia None Sleep Apnea
##   Female       36   82          67
##   Male         41  137          11
```

Interpretation. As already seen in the descriptive analysis, the most common category in the full sample is by far "None", accounting for about 59%. Among the actual sleep disorders, Sleep Apnea and Insomnia appear with almost identical frequency. However, when we look at men and women separately, the picture changes considerably. While Insomnia (41 cases versus only 11 for Sleep Apnea) is the most common disorder among men, women are more often affected by Sleep Apnea (67 cases versus 36 for Insomnia). This reveals a striking asymmetry: Sleep Apnea is much more prevalent among females in this dataset, while Insomnia is more common among males.

2.2.3 Lowest sleep quality and highest stress by occupation and gender

As a next step, we group by two categorical variables at the same time, expanding the `aggregate()` formula with a plus sign to obtain every occupation × gender combination present in the data. We also print a table of group sizes, so that we can judge how reliable each average is.

```r
combo_quality <- aggregate(sleep_quality ~ occupation + gender, sleep, mean)
head(combo_quality[order(combo_quality$sleep_quality), ])
```

```
##              occupation gender sleep_quality
## 13 Sales Representative   Male       4.00000
## 7             Scientist Female       5.00000
## 14          Salesperson   Male       6.00000
## 16              Teacher   Male       6.00000
## 15    Software Engineer   Male       6.50000
## 10               Doctor   Male       6.57971
```

```r
combo_stress <- aggregate(stress_level ~ occupation + gender, sleep, mean)
head(combo_stress[order(-combo_stress$stress_level), ])
```

```
##              occupation gender stress_level
## 13 Sales Representative   Male      8.00000
## 7             Scientist Female      7.00000
## 14          Salesperson   Male      7.00000
## 10               Doctor   Male      6.84058
## 16              Teacher   Male      6.20000
## 9             Accountant   Male      6.00000
```

```r
table(sleep$occupation, sleep$gender)
```

```
##
##                         Female Male
##   Accountant                36    1
##   Doctor                     2   69
##   Engineer                  32   31
##   Lawyer                     2   45
##   Manager                    1    0
##   Nurse                     73    0
##   Sales Representative       0    2
##   Salesperson                0   32
##   Scientist                  4    0
##   Software Engineer          0    4
##   Teacher                   35    5
```

Interpretation. Similar to before, the combination with the lowest mean sleep quality is again male Sales Representatives, with a mean of 4.00, together with the highest stress level, with a mean of 8.00. However, this group contains only two individuals in the dataset, which makes its estimates to some extent meaningless. If we restrict the analysis to subsamples of at least 30 individuals, we obtain a more reliable basis. Under this restriction, the lowest sleep quality is observed for male Salespersons (mean = 6.00, n = 32) and male Doctors (mean = 6.58, n = 69), while the highest stress level is observed for the same two groups (male Salespersons mean = 7.00, male Doctors mean = 6.84). This shows that one should always consider aggregated means together with the underlying group size to reach meaningful interpretations; otherwise, very small subgroups can produce misleading "extremes".

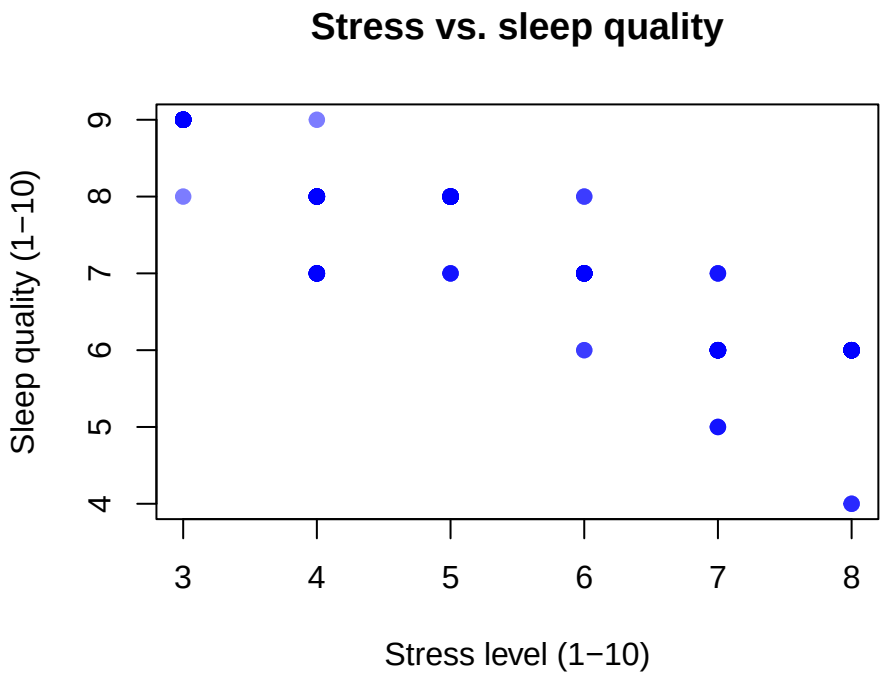
2.2.4 Correlation between stress and sleep quality

In order to find out the relationship between stress level and sleep quality, we finally compute the correlation coefficient and visualise it in a scatter plot. We use the integrated R function `cor()`, which by default returns the Pearson correlation. The outcome can take values between -1 and 1, where values close to ± 1 indicate a strong linear relationship and a value near 0 indicates no linear relationship. In the `plot()`, the arguments are chosen so that, with `pch = 19`, filled circles are drawn in a semi-transparent blue, achieved by `col = rgb(0, 0, 1, 0.3)`. The fourth argument, 0.3, controls the opacity, meaning that overlapping points appear darker and therefore reveal concentrations of observations.

```r
cor(sleep$stress_level, sleep$sleep_quality)
```

```
## [1] -0.898752
```

```r
plot(sleep$stress_level, sleep$sleep_quality,
     xlab = "Stress level (1-10)", ylab = "Sleep quality (1-10)",
     main = "Stress vs. sleep quality",
     pch = 19, col = rgb(0, 0, 1, 0.3))
```



Interpretation. In our case, the Pearson correlation between stress level and sleep quality is $r \approx -0.9$, which is a very strong negative linear association. In other words, individuals who report higher stress levels also tend to report substantially lower sleep quality. This goes hand in hand with the occupation-level finding above, where Engineers had simultaneously the lowest stress and the highest sleep quality, while Salespersons and Sales Representatives had the highest stress together with the lowest sleep quality. However, it is important to mention that even though the data show that stress and sleep quality move together, we do not gain any information about whether high stress causes poor sleep, whether poor sleep amplifies stress, or whether a third factor, such as workload, working hours or occupation type, drives both. Although correlation does not imply causation, the evidence in this sample does support the hypothesis that lower stress is associated with better sleep quality.

2.3 Testing the doctor's claims

2.3.1 Claim (a): 41% of males suffer from insomnia

This claim concerns a single proportion, so the appropriate procedure is a one-sample test of a proportion. We test $H_0 : p = 0.41$ versus $H_1 : p \neq 0.41$, where p is the true proportion of males with insomnia. In R, the function `prop.test()` implements this test; we pass it the number of male insomniacs, the total number of males and the hypothesised proportion 0.41.

```
males      <- subset(sleep, gender == "Male")
insomniacs <- sum(males$sleep_disorder == "Insomnia")
prop.test(insomniacs, nrow(males), p = 0.41)
```

```
##
##  1-sample proportions test with continuity correction
##
## data:  insomniacs out of nrow(males), null probability 0.41
## X-squared = 28.331, df = 1, p-value = 1.022e-07
## alternative hypothesis: true p is not equal to 0.41
## 95 percent confidence interval:
##  0.1617851 0.2838553
## sample estimates:
##         p
## 0.2169312
```

Interpretation. In the sample, 41 males out of 189 in total suffer from insomnia, giving an observed proportion of approximately 21.7%. The test yields a chi-squared statistic of 28.33 with one degree of freedom and a p-value of about 10^{-7}, which is far below the 5% significance level. Therefore, we reject H_0. Equivalently, the 95% confidence interval $[16.2\%, 28.4\%]$ does not contain the doctor's value of 41%. This shows that the data provide strong evidence against the first claim, as the true insomnia rate in this population appears to be roughly half of what the doctor stated.

2.3.2 Claim (b): male insomniacs walk on average 7065 steps

This claim concerns a single mean, so the appropriate procedure is a one-sample t-test. We test $H_0 : \mu = 7065$ versus $H_1 : \mu \neq 7065$, where μ is the true mean daily step count of male insomniacs. In R, the function `t.test()` implements this test.

```r
mi <- subset(sleep, gender == "Male" & sleep_disorder == "Insomnia")
t.test(mi$daily_steps, mu = 7065)
```

```
##
##  One Sample t-test
##
## data:  mi$daily_steps
## t = -7.1934, df = 40, p-value = 1.01e-08
## alternative hypothesis: true mean is not equal to 7065
## 95 percent confidence interval:
##  5460.205 6164.185
## sample estimates:
## mean of x
##  5812.195
```

Interpretation. Among the 41 male insomniacs in the sample, daily steps average 5,812 per day, with a standard deviation of about 1,115 steps. The t-statistic is -7.19 with 40 degrees of freedom, and the p-value is about 10^{-8}. This is again far below the 5% significance level, and we therefore reject H_0. As with the first claim, the 95% confidence interval [5,460, 6,164] does not contain the doctor's claimed value of 7,065. Taking this into account, there is strong evidence against the second claim as well, as male insomniacs walk substantially less than the doctor stated.

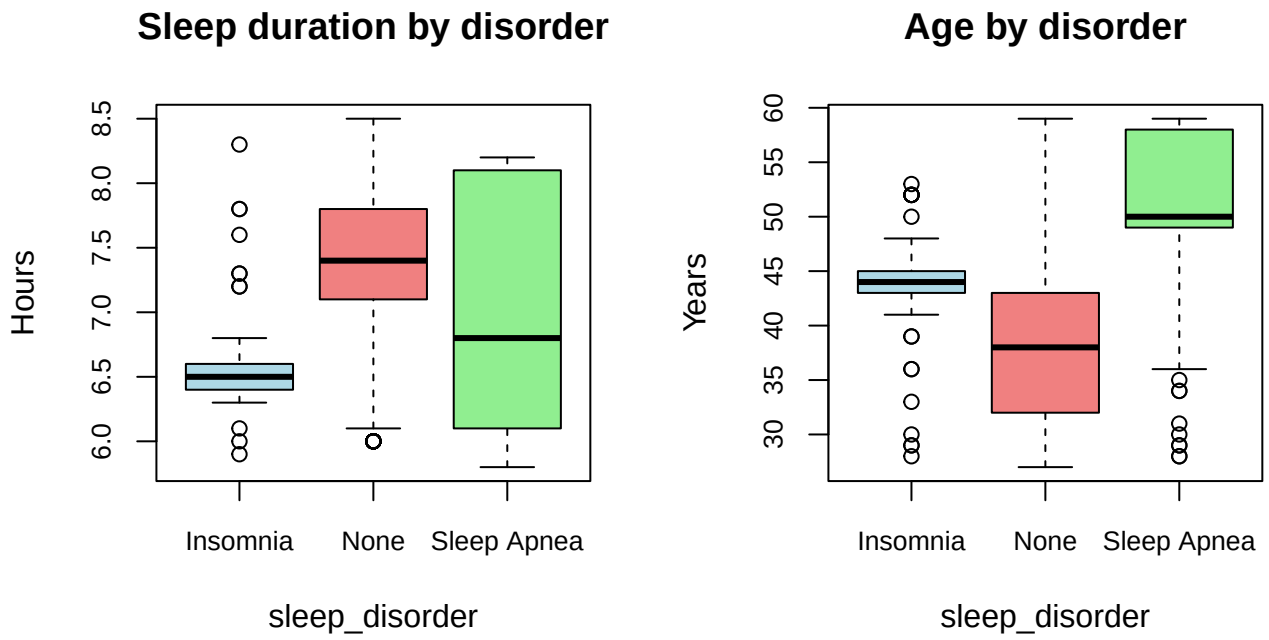
2.4 Sleep duration and age across disorder categories

We now compare sleep duration and age across the three sleep disorder categories (None, Insomnia and Sleep Apnea). Before using confidence intervals to formally compare the group means, we visualise the distributions with boxplots.

2.4.1 Boxplots

A boxplot displays the median (the thick line), the interquartile range (the box) and the range of typical observations (the whiskers) for each group, plus any outliers as individual points. We plot sleep duration and age side by side.

```
par(mfrow = c(1, 2))
boxplot(sleep_duration ~ sleep_disorder, data = sleep,
        main = "Sleep duration by disorder", ylab = "Hours",
        col = c("lightblue", "lightcoral", "lightgreen"), cex.axis = 0.8)
boxplot(age ~ sleep_disorder, data = sleep,
        main = "Age by disorder", ylab = "Years",
        col = c("lightblue", "lightcoral", "lightgreen"), cex.axis = 0.8)
```




```
par(mfrow = c(1, 1))
```

Interpretation. The boxplots clearly reveal differences across the categories. For sleep duration (left panel), individuals without a diagnosed disorder sleep on average 7.4 hours, while those with insomnia sleep noticeably less (median around 6.6 hours). This confirms the definition of insomnia,

which is characterised by difficulties falling or staying asleep, so a reduced sleep duration for this group is not surprising. Individuals with sleep apnea fall in between, but they show much greater variability, as their box is wider than for the other two groups. Concerning age (right panel), there is a clear ordering: individuals without a disorder are the youngest, with a median around 39 years, followed by those with insomnia (around 43 years) and lastly those suffering from sleep apnea, who are noticeably older, with a median close to 50 years. This is consistent with clinical experience, which documents that sleep apnea is more prevalent in older adults.

2.4.2 Comparing mean sleep duration with confidence intervals

The brief asks us to compare the mean sleep duration across the three categories using 99% confidence intervals ($\alpha = 1\%$): if two intervals do not intersect, this provides evidence that the underlying population means differ at the corresponding significance level. We read each interval off the output of the `t.test()` function, which reports a confidence interval for the mean.

```
none  <- sleep$sleep_duration[sleep$sleep_disorder == "None"]
insom <- sleep$sleep_duration[sleep$sleep_disorder == "Insomnia"]
apnea <- sleep$sleep_duration[sleep$sleep_disorder == "Sleep Apnea"]

t.test(none,  conf.level = 0.99)$conf.int
```

```
## [1] 7.229856 7.487039
## attr(,"conf.level")
## [1] 0.99
```

```
t.test(insom, conf.level = 0.99)$conf.int
```

```
## [1] 6.473040 6.706181
## attr(,"conf.level")
## [1] 0.99
```

```
t.test(apnea, conf.level = 0.99)$conf.int
```

```
## [1] 6.740527 7.323575
## attr(,"conf.level")
## [1] 0.99
```

Interpretation. The three 99% confidence intervals for the mean sleep duration are roughly $[7.23, 7.49]$ for None, $[6.47, 6.71]$ for Insomnia and $[6.74, 7.32]$ for Sleep Apnea. Comparing them pairwise, the None and Insomnia intervals do not intersect, so we have strong evidence that these two means differ at a significance level smaller than 1%. The same applies to the Insomnia and Sleep Apnea intervals, which also do not overlap. Only the None and Sleep Apnea intervals overlap, so we do not have sufficient evidence at $\alpha = 1\%$ to claim that their mean sleep durations differ. In the end, it is not surprising that insomnia is associated with shorter sleep; the lack of a clear difference between None and Sleep Apnea is more interesting, as it reflects the fact that sleep apnea reduces sleep quality more than total sleep duration.

2.4.3 Comparing mean age with confidence intervals

We repeat the same procedure for the variable age.

```
none_age  <- sleep$age[sleep$sleep_disorder == "None"]
insom_age <- sleep$age[sleep$sleep_disorder == "Insomnia"]
apnea_age <- sleep$age[sleep$sleep_disorder == "Sleep Apnea"]

t.test(none_age,  conf.level = 0.99)$conf.int
```

```
## [1] 37.66201 40.41105
## attr(,"conf.level")
## [1] 0.99
```

```
t.test(insom_age, conf.level = 0.99)$conf.int
```

```
## [1] 42.07169 44.96727
## attr(,"conf.level")
## [1] 0.99
```

```
t.test(apnea_age, conf.level = 0.99)$conf.int
```

```
## [1] 47.01638 52.39388
## attr(,"conf.level")
## [1] 0.99
```

Interpretation. The three 99% confidence intervals for the mean age are roughly $[37.7, 40.4]$ for None, $[42.1, 45.0]$ for Insomnia and $[47.0, 52.4]$ for Sleep Apnea. Unlike for sleep duration, the three intervals here are all pairwise disjoint: the interval for None lies entirely below that for Insomnia, which in turn lies entirely below that for Sleep Apnea. The CI-based method therefore gives evidence at $\alpha = 1\%$ that the mean age differs between all three pairs of groups. This is consistent with our observations from the boxplots in the previous section, as it confirms that sleep apnea sufferers in this sample are considerably older than the other two groups, while individuals with no diagnosed disorder are the youngest on average.