Hands-on Workshop: Exploring TMS Data

Welcome

This short tutorial will guide you through exploring Transcranial Magnetic Stimulation (TMS) study data using R. You don't need any prior coding or statistics experience, we'll go step by step.

Requirements

Before getting started, make sure you have R installed on your computer. You can download it for free from the official CRAN website: https://cran.r-project.org/

To make things easier, we also recommend using an IDE (Integrated Development Environment), which provides an interface that helps you write and run R code more smoothly.

If you're new to R, the best option is RStudio, since it's built specifically for R and is very beginner-friendly. You can download RStudio here: https://posit.co/download/rstudio-desktop/

Both R and RStudio are free to download and use.

Setup

Once you have R installed, you will also need a few extra tools (called packages). These are small add-ons that make R more powerful.

Step 1 — Install the required packages

Run this code once on your computer to install what we'll use today.

```
# Install the required packages if not already installed
install.packages("dplyr")
install.packages("ggplot2")
install.packages("tableone")
install.packages("lme4")
install.packages("Rmarkdown")
install.packages("lmerTest")
```

Step 2 — Load the packages

You'll need to load these every time you open a new R session.

```
# Load the required packages
library(dplyr)
library(ggplot2)
library(tableone)
library(lme4)
library(lmerTest)
```

Load and peek at the data

Let's start by reading our dataset into R.

What does this mean?

- `read.csv()` tells R to open a spreadsheet file that's in CSV format.
- The result is stored in something called a data frame, a bit like an Excel sheet.

```
# Path to your CSV file
path <- "data.csv"

# Read the CSV
data <- read.csv(path)
```

Take a quick look at the first few rows using `head()`, so you can confirm the data looks right.

```
# Peek at the first few rows
head(data)
```

```
##   subject_id age gender medication coil_intensity treatment site session_count
## 1          1  33 female          0       74.30402    active    B            28
## 2          1  33 female          0       74.30402    active    B            28
## 3          1  33 female          0       74.30402    active    B            28
## 4          2  37 female          0       70.46629    active    C            20
## 5          2  37 female          0       70.46629    active    C            20
## 6          2  37 female          0       70.46629    active    C            20
##   MEP_amplitude  EEG_theta  EEG_alpha  EEG_beta     time symptom_score
## 1      2.167524 -0.5158105  0.8567855  0.628965      pre      8.009644
## 2      2.167524 -0.5158105  0.8567855  0.628965     post      4.492344
## 3      2.167524 -0.5158105  0.8567855  0.628965 followup      8.681195
## 4      1.681755  0.4457241 -1.1037521 -1.262195      pre      7.937556
## 5      1.681755  0.4457241 -1.1037521 -1.262195     post     -0.888527
## 6      1.681755  0.4457241 -1.1037521 -1.262195 followup     -2.433257
##   symptom_baseline
## 1         8.009644
## 2         8.009644
## 3         8.009644
## 4         7.937556
## 5         7.937556
## 6         7.937556
```

Then, see the structure (column names and data types) using `str()`. This is useful to check that, for example, numbers are numeric and not text. This command shows:

- Each column name
- The type of data (numbers, text, etc.)
- A few example values from each column

Think of this like "What kind of information do I have in each column?"

```
str(data)
```

```
## 'data.frame':    450 obs. of  15 variables:
##  $ subject_id      : int  1 1 1 2 2 2 3 3 3 4 ...
##  $ age             : int  33 33 33 37 37 37 59 59 59 41 ...
##  $ gender          : chr  "female" "female" "female" "female" ...
##  $ medication      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ coil_intensity  : num  74.3 74.3 74.3 70.5 70.5 ...
##  $ treatment       : chr  "active" "active" "active" "active" ...
##  $ site            : chr  "B" "B" "B" "C" ...
##  $ session_count   : int  28 28 28 20 20 20 19 19 19 22 ...
```

```
##  $ MEP_amplitude   : num  2.17 2.17 2.17 1.68 1.68 ...
##  $ EEG_theta       : num  -0.516 -0.516 -0.516 0.446 0.446 ...
##  $ EEG_alpha       : num  0.857 0.857 0.857 -1.104 -1.104 ...
##  $ EEG_beta        : num  0.629 0.629 0.629 -1.262 -1.262 ...
##  $ time            : chr  "pre" "post" "followup" "pre" ...
##  $ symptom_score   : num  8.01 4.492 8.681 7.938 -0.889 ...
##  $ symptom_baseline: num  8.01 8.01 8.01 7.94 7.94 ...
```

Finally, `summary()` provides summary statistics for each column. This gives quick numerical summaries:

- For numbers: minimum, mean, median, maximum
- For text or categories: how many of each type

At this point, you should have a good sense of what's inside your dataset.

```
# Quick summary of each column
summary(data)
```

```
##    subject_id         age           gender            medication
##  Min.   :  1.0   Min.   :12.00   Length:450         Min.   :0.0
##  1st Qu.: 38.0   1st Qu.:32.00   Class :character   1st Qu.:0.0
##  Median : 75.5   Median :39.00   Mode  :character   Median :0.0
##  Mean   : 75.5   Mean   :39.67                      Mean   :0.4
##  3rd Qu.:113.0   3rd Qu.:47.00                      3rd Qu.:1.0
##  Max.   :150.0   Max.   :66.00                      Max.   :1.0
##
##  coil_intensity   treatment            site           session_count
##  Min.   :31.90   Length:450         Length:450         Min.   :10.00
##  1st Qu.:54.43   Class :character   Class :character   1st Qu.:18.00
##  Median :60.90   Mode  :character   Mode  :character   Median :21.00
##  Mean   :60.64                                         Mean   :20.36
##  3rd Qu.:67.41                                         3rd Qu.:23.00
##  Max.   :84.30                                         Max.   :31.00
##
##  MEP_amplitude        EEG_theta          EEG_alpha            EEG_beta
##  Min.   :-0.04048   Min.   :-2.38319   Min.   :-2.313736   Min.   :-2.69533
##  1st Qu.: 0.84058   1st Qu.:-0.76843   1st Qu.:-0.701804   1st Qu.:-0.56611
##  Median : 1.24759   Median :-0.11979   Median :-0.005609   Median : 0.24571
##  Mean   : 1.24166   Mean   :-0.07554   Mean   : 0.019734   Mean   : 0.07721
##  3rd Qu.: 1.60023   3rd Qu.: 0.55262   3rd Qu.: 0.681830   3rd Qu.: 0.72468
##  Max.   : 2.61846   Max.   : 3.28690   Max.   : 3.390371   Max.   : 2.28412
##
##      time           symptom_score    symptom_baseline
##  Length:450         Min.   :-9.751   Min.   :-3.36
##  Class :character   1st Qu.: 8.612   1st Qu.:12.00
##  Mode  :character   Median :15.300   Median :18.98
##                     Mean   :15.289   Mean   :18.67
##                     3rd Qu.:22.225   3rd Qu.:25.14
##                     Max.   :41.800   Max.   :41.80
##                     NA's   :39
```

Exploratory analysis

Now the fun part, visualizing what's happening in the data.

Before plotting, let's make sure the `time` variable appears in the correct order (pre > post > follow-up):

```r
data <- data %>%
  mutate(time = factor(time, levels = c("pre", "post", "followup")))
```

Plot 1 - Symptom change over time by treatment

In TMS research, we often want to know: Do patients receiving active TMS improve more than those in the sham (placebo) group?

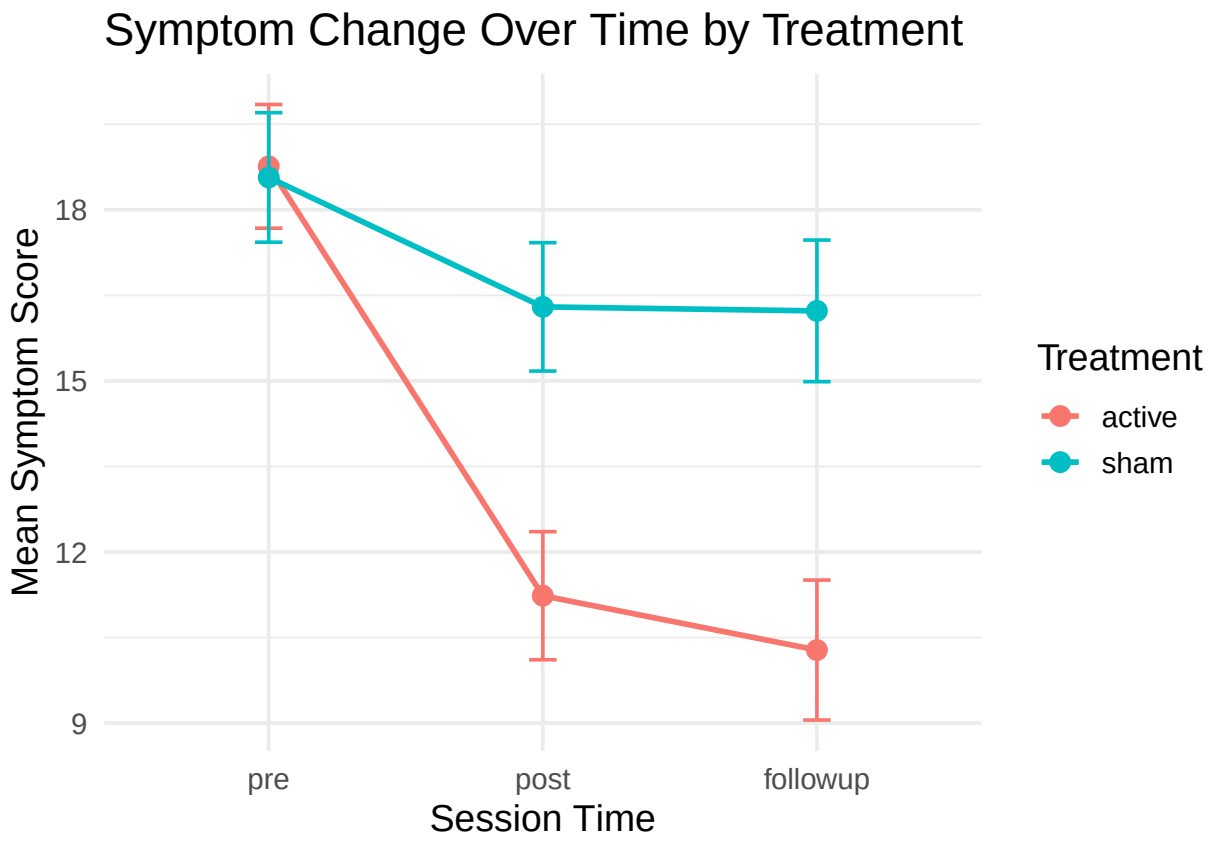
This plot shows average symptom scores at each stage of the study:

- **Pre**: before any treatment
- **Post**: immediately after treatment
- **Follow-up**: some time later

We'll add:

- **Dots** for mean symptom scores
- **Lines** connecting them over time
- **Error bars** showing uncertainty (standard error)

```r
ggplot(data, aes(x = time, y = symptom_score, color = treatment, group = treatment)) +
  stat_summary(fun = mean, geom = "line", linewidth = 1) +
  stat_summary(fun = mean, geom = "point", size = 3) +
  stat_summary(fun.data = mean_se, geom = "errorbar", width = 0.1) +
  labs(
    title = "Symptom Change Over Time by Treatment",
    x = "Session Time",
    y = "Mean Symptom Score",
    color = "Treatment"
  ) +
  theme_minimal(base_size = 14)
```



How to read this

- The y-axis shows average symptom severity.
- The x-axis shows time: pre → post → follow-up.
- Two lines: one for each treatment type (active vs sham).
- If the active line goes down more steeply, patients improved more with active TMS.
- If both lines look similar, the sham effect may be similar (placebo or no effect).

Clinically speaking

This is your first visual check: is there an apparent treatment benefit? Don't jump to conclusions yet, but you'll see whether the trend looks promising.

Plot 2 - Change from baseline

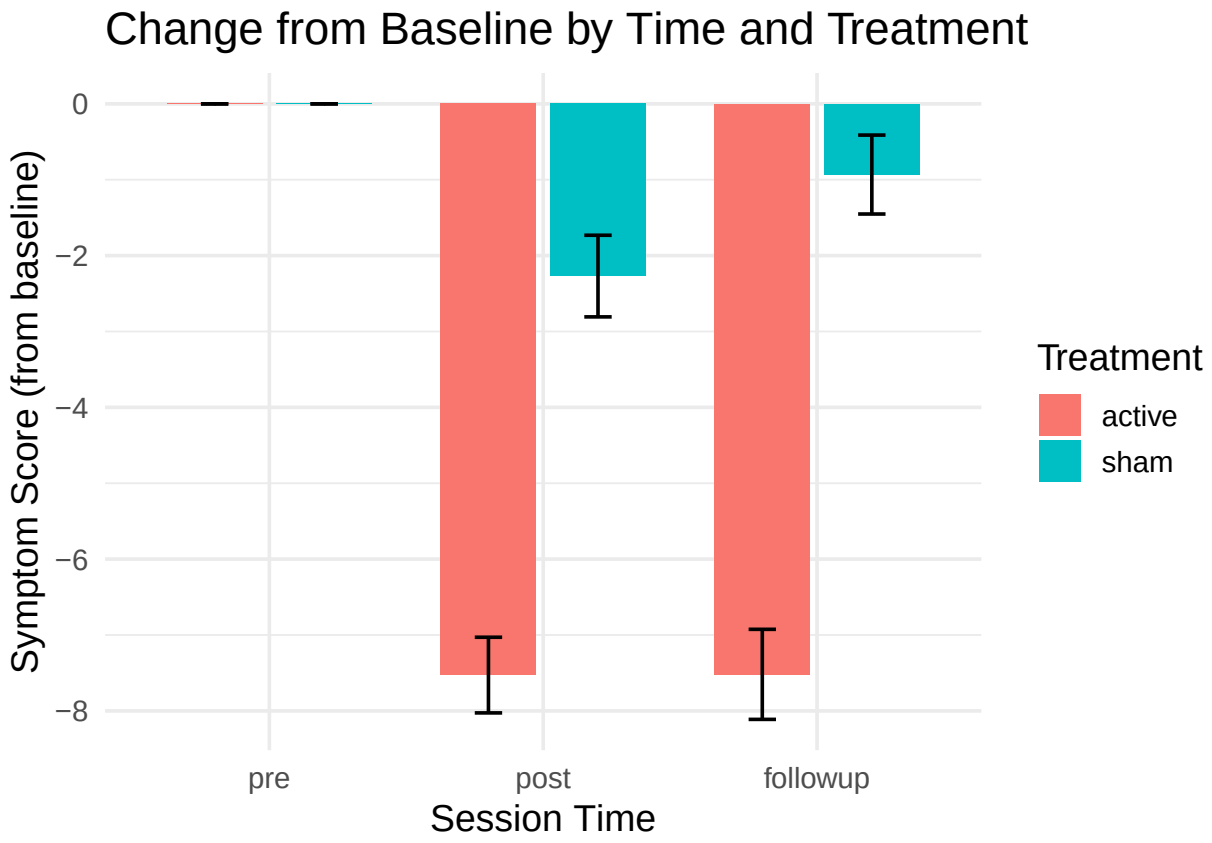
Absolute scores are useful, but clinicians often care more about change from baseline: How much did the patient improve compared to where they started?

We'll plot these changes for each group and time point.

```
# First, let's calculate the delta
data <- data %>%
  mutate(delta_from_baseline = symptom_score - symptom_baseline)

# Then plot it
ggplot(data, aes(x = time, y = delta_from_baseline, fill = treatment)) +
  stat_summary(fun = mean, geom = "bar", position = position_dodge(0.8), width = 0.7) +
  stat_summary(fun.data = mean_se, geom = "errorbar",
               position = position_dodge(0.8), width = 0.2) +
```

```r
labs(
  title = "Change from Baseline by Time and Treatment",
  x = "Session Time",
  y = "Symptom Score (from baseline)",
  fill = "Treatment"
) +
theme_minimal(base_size = 14)
```



How to read this

- Each bar shows the average change from baseline.
- Compare active vs sham at post and follow-up.
- Because the score measures symptom severity, a drop (more negative value) means symptoms improved.
- Look at the height difference: larger drops mean more improvement.
- In other words: the lower the bar, the greater the improvement.

Clinically speaking

This plot gives a quick sense of treatment effect size, how much better the active group got relative to the sham group.

Your turn

Now that you've made two fundamental plots, try adding or changing things to dig deeper. Below are a few extra plots to explore, each demonstrating a different concept.

- Check whether brain response (MEP amplitude) is related to symptom improvement.
- Compare EEG signals between groups.

Table one

A common clinical research step is to create a Table One summarizing baseline characteristics of your groups. This helps ensure groups are comparable before treatment. Let's create one using the `tableone` package.

```
# Define variables for Table One
vars <- c(
  "age", "gender", "medication", "site", "coil_intensity",
  "MEP_amplitude",  "EEG_theta",  "EEG_alpha",  "EEG_beta","symptom_score"
  )
catVars <- c("gender", "site") # Categorical variables
strataVar <- "treatment" # Grouping variable

# Adjust variable type to properly represent it in the table
data$medication <- as.logical(data$medication)

# Create Table One with baseline data
data_baseline <- data %>% filter(time == "pre")
tableOne <- CreateTableOne(vars = vars, data = data_baseline, factorVars = catVars, strata = strataVar)
print(tableOne)
```

```
##                              Stratified by treatment
##                               active        sham          p      test
##   n                              79           71
##   age (mean (SD))             40.13 (12.02) 39.15 (10.72)  0.604
##   gender = male (%)              42 (53.2)     43 (60.6)   0.454
##   medication = TRUE (%)          32 (40.5)     28 (39.4)   1.000
##   site (%)                                                 0.978
##      A                           28 (35.4)     24 (33.8)
##      B                           24 (30.4)     22 (31.0)
##      C                           27 (34.2)     25 (35.2)
##   coil_intensity (mean (SD)) 60.69 (9.60)  60.59 (9.34)   0.953
##   MEP_amplitude (mean (SD))   1.27 (0.59)   1.21 (0.55)   0.468
##   EEG_theta (mean (SD))      -0.19 (1.04)   0.06 (0.97)   0.128
##   EEG_alpha (mean (SD))      -0.02 (1.03)   0.06 (0.91)   0.612
##   EEG_beta (mean (SD))        0.10 (1.00)   0.05 (0.87)   0.757
##   symptom_score (mean (SD))  18.76 (9.64)  18.57 (9.57)   0.902
```

Now, it is your turn! Modify the variables in `vars` and `catVars` to include other baseline characteristics relevant to your study. Run the code to see how the groups compare before treatment.

Regression model

Simple linear regression

First, we'll fit a simple linear regression model to see how treatment affects symptom scores at follow-up, controlling for baseline scores.

```
data_followup <- data %>% filter(time == "followup")
model <- lm(symptom_score ~ treatment + symptom_baseline, data = data_followup)
summary(model)
```

```
##
## Call:
## lm(formula = symptom_score ~ treatment + symptom_baseline, data = data_followup)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.9312  -2.8261   0.1856   2.9218   8.0640
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -5.5539     0.9226  -6.020 2.44e-08 ***
## treatmentsham      6.5159     0.7873   8.277 3.66e-13 ***
## symptom_baseline   0.8896     0.0425  20.932  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.124 on 108 degrees of freedom
##   (39 observations deleted due to missingness)
## Multiple R-squared:  0.821,  Adjusted R-squared:  0.8177
## F-statistic: 247.6 on 2 and 108 DF,  p-value: < 2.2e-16
```

This first model helps us understand whether treatment type (active vs. sham) has an effect on symptom scores at follow-up, while controlling for baseline severity.

Key takeaways

The treatmentsham estimate (~6.52, $p < 0.001$) suggests that participants in the sham group had, on average, higher symptom scores at follow-up compared to those in the active TMS group.

The baseline severity coefficient (~0.89, $p < 0.001$) shows a positive association: participants starting with more severe symptoms tended to also present higher symptom scores at follow-up.

The model explains a substantial portion of the variability in follow-up symptoms ($R^2 = 0.82$), suggesting that treatment type and baseline severity together are strong predictors of outcome.

Clinically speaking

This model indicates a clear benefit of active TMS compared to sham stimulation when adjusting for initial severity. While individual responses may vary, the overall trend strongly supports a meaningful treatment effect.

Your turn

Now, also add more predictors like age, gender and more to see if they influence outcomes.

Mixed effects model

In this section, we'll fit a mixed effects model to account for repeated measures over time within subjects, which is common in clinical trials where we have multiple observations per patient (at pre, post, and follow-up).

```
model_mixed <- lmer(symptom_score ~ treatment * time + (1 | subject_id), data = data)
summary(model_mixed)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: symptom_score ~ treatment * time + (1 | subject_id)
##    Data: data
##
## REML criterion at convergence: 2567.6
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -1.96817 -0.58003 -0.03979  0.59277  2.36810
```

```
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  subject_id (Intercept) 82.726   9.095
##  Residual                9.786   3.128
## Number of obs: 411, groups:  subject_id, 150
##
## Fixed effects:
##                             Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)                  18.7618     1.0821 168.9301  17.338  < 2e-16 ***
## treatmentsham                -0.1942     1.5729 168.9301  -0.123    0.902
## timepost                     -7.5280     0.4977 256.7917 -15.124  < 2e-16 ***
## timefollowup                 -7.5994     0.5491 258.5120 -13.839  < 2e-16 ***
## treatmentsham:timepost        5.2583     0.7235 256.7917   7.268 4.38e-12 ***
## treatmentsham:timefollowup    6.6932     0.8139 258.8158   8.224 9.65e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##               (Intr) trtmnt timpst tmfllw trtmntshm:tmp
## treatmntshm   -0.688
## timepost      -0.230  0.158
## timefollowp   -0.208  0.143  0.453
## trtmntshm:tmp  0.158 -0.230 -0.688 -0.312
## trtmntshm:tmf  0.141 -0.204 -0.306 -0.675  0.444
```

This second model accounts for repeated measures—each participant was assessed multiple times (pre, post, follow-up)—and thus includes a random intercept for each subject to capture individual differences.

What the model includes

Fixed effects: treatment, time, and their interaction (treatment × time).

Random effects: subject-level intercepts, allowing each participant to have their own baseline level of symptoms.

Key results

Time effects: Both timepost (-7.53) and timefollowup (-7.60) show large, negative coefficients (t = -15 and -14), meaning that symptom severity decreased significantly over time compared to the pre-treatment phase.

Treatment main effect: The treatmentsham term alone (-0.19) was small and not significant, indicating no overall difference between groups at baseline.

Interaction effects: Both treatmentsham:timepost (5.26) and treatmentsham:timefollowup (6.69) were large and significant, showing that sham participants improved less over time than those receiving active TMS.

Random effects interpretation

The subject-level variance (82.7) shows there are meaningful individual differences in baseline symptom levels.

The residual variance (9.8) reflects remaining within-person variability not explained by the model.

Clinically speaking

This mixed-effects approach confirms the pattern seen earlier: active TMS produced greater improvement over time. The strong time effects indicate that participants generally improved with treatment, but the interaction shows that the active group's improvement was larger and more sustained at follow-up. This supports the idea that active stimulation drives the observed changes.

Additional resources:

If you'd like to keep exploring R on your own, here are some great resources to help you learn, practice, and grow your skills:

1. Hands-On Programming with R: A beginner-friendly introduction to R programming.
Link: https://rstudio-education.github.io/hopr/

2. RStudio Cheat Sheets: Handy reference guides for data visualization, wrangling, and more.
Link: https://posit.co/resources/cheatsheets/

3. R for Data Science (by Hadley Wickham & Garrett Grolemund): A free, in-depth book that covers tidy data workflows.
Link: https://r4ds.hadley.nz/

4. Documentation for the tableone package: A practical guide to using the tableone package for creating descriptive summary tables commonly used in medical and epidemiological research.
Link: https://cran.r-project.org/web/packages/tableone/vignettes/introduction.html

5. YaRrr! The Pirate's Guide to R: Another introduction to R programming. It also includes a great beginner-friendly overview of linear models using the lm() function.
Link: https://bookdown.org/ndphillips/YaRrr/

6. An R companion to Statistics: data analysis and modelling: A companion guide for learning statistical modeling in R, including detailed chapters on linear mixed-effects models using lmer().
Link: https://mspeekenbrink.github.io/sdam-r-companion/index.html

7. Fitting Linear Mixed-Effects Models Using lme4: The official documentation for the lme4 package, explaining the theory, syntax, and practical examples for fitting mixed-effects models with lmer().
Link: https://cran.r-project.org/web/packages/lme4/vignettes/lmer.pdf