Exercises day 8

Basic Statistics for health researchers 2020

23 November 2020

Warming up

Download the R-demo of Lecture 8 (available from the course webpage). Since it is a rather long file, we recommend that you create your own script containing parts of the R-demo file you find relevant, along with your own comments.

1. [R-users] Generate the pre-post dataset (lines 2-8).
   [non R-users] Download the file prepost.csv from the course webpage, load it.
2. Convert the dataset from the wide format to the long format (lines 33-48)
3. Make descriptive plots/compute summary statistics (lines 59-97).
4. Estimate the treatment effect using the estimated means
5. Run a t-test, linear regression, and linear regression for heteroschedastic data to test for a difference in change of vision between the two groups (lines 101-130). Compare the outputs.
6. Create a spaghetti plot (lines 150-154).
7. Fit a random intercept model and try to make sense of the R-output (lines 209-238). Display the fitted value at each time for each group.

Additional questions if times permits:

8. Fit the same random intercept model using another parametrisation (lines 269-291/slides 47-49). Can you make sense of the output?
9. In addition of testing the difference in change of vision, one could also test the difference in vision at follow-up (line 295-301/slide 50). Can you relate the output to the coefficients estimated by the random intercept model? Why does the p-value for the change differ from the one obtained when using `summary(e.lme)`?

Exercise A: Longitudinal study

The ARMD trial used as an example in the lecture followed the patients not only at week 4 but also at week 12, 24, and 52.

```
data(armd.wide, package = "nlmeU")
armdW <- armd.wide
str(armdW)
```

```
'data.frame':       240 obs. of  10 variables:
 $ subject : Factor w/ 240 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...
 $ lesion  : int  3 1 4 2 1 3 1 3 2 1 ...
 $ line0   : int  12 13 8 13 14 12 13 8 12 10 ...
 $ visual0 : int  59 65 40 67 70 59 64 39 59 49 ...
 $ visual4 : int  55 70 40 64 NA 53 68 37 58 51 ...
 $ visual12: int  45 65 37 64 NA 52 74 43 49 71 ...
 $ visual24: int  NA 65 17 64 NA 53 72 37 54 71 ...
 $ visual52: int  NA 55 NA 68 NA 42 65 37 58 NA ...
 $ treat.f : Factor w/ 2 levels "Placebo","Active": 2 2 1 1 2 2 1 1 2 1 ...
 $ miss.pat: Factor w/ 9 levels "----","---X",..: 4 1 2 1 9 1 1 1 1 2 ...
```

We will re-analyse this study considering and study the treatment effect at all follow-up points.

1. What the presence of `NA` values in the previous output indicate?

2. Make a graphical representation of the data. To convert the dataset into the long format you can use:

```
armdL <- reshape2::melt(armdW,
    id.var = c("subject","lesion","treat.f","miss.pat"),
    measure.vars = paste0("visual",c(0,4,12,24,52)),
    variable.name = "week",
    value.name = "visual")
armdL$week <- factor(armdL$week,
                     level = paste0("visual",c(0,4,12,24,52)),
                     labels = c(0,4,12,24,52))
armdL$week.num <- as.numeric(as.character(armdL$week))
armdL$visit <- as.numeric(armdL$week)
armdL.cc <- armdL[!is.na(armdL$visual),]
```

3. Compute summary statistics (mean, variance, correlation) of the dataset over time and in each treatment group. What do you notice?

4. Test whether there is a greater loss in vision in the Placebo group than in the Active group using t-tests. Should you correct for multiple comparisons before concluding? What limitation(s) do you see with the t-test approach?

5. For this question, we will ignore the correlation between repeated measurement (⚠ INCORRECT ANALYSIS). What is the difference between the following two linear models?

```
e.linear <- lm(visual ~ treat.f*week.num, data = armdL.cc)
e.np <- lm(visual ~ treat.f*week, data = armdL.cc)
```

To answer the question, it can be helpful to make a graphical representation of the fitted values (and thus of the treatment effect) with each model. This can be achieved by computing the fitted value (using the `predict` function) for each possible value of the covariates:

```
X <- unique(armdL.cc[,c("treat.f","week.num","week")])
X
```

```
    treat.f week.num week
1    Active        0    0
3   Placebo        0    0
241  Active        4    4
243 Placebo        4    4
481  Active       12   12
483 Placebo       12   12
722  Active       24   24
723 Placebo       24   24
962  Active       52   52
964 Placebo       52   52
```

- Which model is the closest to the multiple t-test approach?
- What unrealistic assumptions are we making?

6. We will now account for the correlation between the measurement using a random intercept model:

```
library(nlme)
e.lme <- lme(visual ~ treat.f*week.num,
             random =~ 1|subject, data = armdL.cc)
```

- which assumptions are we still making? (see slide 39 in the lecture) Are they satisfied? You can check using the residuals:

```
armdL.cc$residuals <- residuals(e.lme)
```

You will probably also need the residuals in the wide format:

```
armaW2.cc <- reshape2::dcast(armdL.cc,
                             formula = subject+lesion+treat.f~week,
                             value.var = "residuals")
```

- display the new fitted values and test the treatment effect.

Additional question if times permits:

7. We could could relax some of the assumptions about the variance and the correlation using the following model:

```
e.gls <- gls(visual ~ treat.f*week.num,
             correlation = corSymm(form=~visit|subject),
             weights = varIdent(form=~1|week),
             data = armdL.cc)
```

The argument **weights** models a different variance over time:

```
summary(e.gls$modelStruct$varStruct)
```

```
Variance function:
 Structure: Different standard deviations per stratum
 Formula: ~1 | week
 Parameter estimates:
       0        4       12       24       52
1.000000 1.070090 1.160112 1.247591 1.261966
```

and the argument `correlation` models a different correlation between each timepoint:

```
summary(e.gls$modelStruct$corStruct)
```

```
Correlation Structure: General
 Formula: ~visit | subject
 Parameter estimate(s):
 Correlation:
  1     2     3     4
2 0.853
3 0.736 0.842
4 0.662 0.751 0.826
5 0.516 0.592 0.699 0.84
```

- display the new fitted values and test the new treatment effect.

Exercise B: Data visualisation

In this exercise we will discuss different data visualizations using simulated data.

1. We are interested in an endpoint (say vision) measured over time (say baseline, week 4, week 8). Before generating data, we first need to decide the expected value at each timepoint:

```
mu <- 0:2
```

and the variance-covariance matrix between the repetitions:

```
sd1 <- 1 ; sd2 <- 1 ; sd3 <- 1 ; rho <- 0.7
cov <- rho * sd1 * sd2
cov2 <- rho^2 * sd1 * sd2
Sigma <- matrix(c(sd1^2, cov  , cov2 ,
                  cov  , sd2^2, cov  ,
                  cov2 , cov  , sd3^2),
                nrow = 3, ncol = 3, byrow = TRUE,
                dimnames = list(c("X1","X2","X3"),c("X1","X2","X3")))
```

- what is the interpretation of the elements contained in the matrix?

2. We now simulate data using the `rmvnorm` function (random number generator for the multivariate normal distribution) of the mvtnorm package:

```
library(mvtnorm)
n <- 200
set.seed(10)
dW <- data.frame(id = 1:n, rmvnorm(n, mean = mu, sigma = Sigma))
```

The long format version of this dataset can be obtained using:

```
dL <- reshape2::melt(dW, id.vars = "id",
                     value.name = "X", variable.name = "time")
```

- compute the empirical mean and empirical variance-covariance matrix of the simulated dataset and compare it to the true one.
- what would happen if we lower (say by a factor 10) the number of observations?
- what does the code `set.seed(10)` achieve?

3. Visualize the data using:

- a scatterplot (with three panels: (`X1`, `X2`), (`X2`, `X3`), and (`X1`, `X3`))
- a boxplot (x-axis: the type of outcome, y-axis: X value)
- a spaghetti plot (x-axis: type of outcome, y-axis: X value)

Discuss the pro- and cons- of each type of display.

Additional questions if time permits:

4. You can vary the simulation parameters, e.g.:

- correlation parameter
- variance parameters
- number of observations

and see how they affects the various plots.

5. A rather nice 2D and 3D display can be obtained using the following code (you need to install the R package `rayshader` for the 3D display):

```
library(ggplot2)
gg <- ggplot(dW, mapping = aes(x=X1, y=X2))
gg <- gg + stat_density_2d(aes(fill = ..level..), geom = "polygon")
gg <- gg + xlab("V1") + ylab("V2")
gg <- gg + scale_fill_viridis_c(option = "D")
gg ## 2D display
rayshader::plot_gg(gg) ## 3D display
```

- What parameters of the simulation (stored in `mu`, `Sigma`) can you identify on this display?
- What represent the color-scale/height?
- Do you find this representation useful?