

Foreword

Statistical analysis are done using `sessionInfo()``R.version``version.string`, which is freely available from CRAN. You may find convenient to run R through RStudio. RStudio offers really great support for editing and running R scripts. You can even organize your work into a project, with version control and automatic reporting built on the fly. I personally choose to work like in the 80s– although I was just a kid at that time–with a simple text editor and an interactive shell available within few key presses. This is possible thanks to Emacs and the brilliant ESS mode.

I replicated all SAS code using SAS University Edition. It can run locally on your computer (using, e.g., Virtual Box) or directly in the cloud. I do not hold a personal licence for SAS (although I could get from several universities where I am teaching) and so I found this solution particularly handy to compare SAS and R output.

1 Analysis of Clinical Trials using SAS

The following analyses are based on Dmitrienko et al. [2005], with data available online at Analysis of Clinical Trials Using SAS: A Practical Guide.

1.1 The HAMD17 study

Context. This is a multicenter clinical trial comparing experimental drug vs. placebo in patients with major depression disorder. The outcome is the change from baseline after 9 weeks of acute treatment, and efficacy is measured using the total score of the Hamilton depression rating scale (17 items).

This is a classical application of unbalanced design and potential heterogeneity between clinical centres, where there is an unequal number of observations per treatment (here, drug by center).

Here is one of many ways to get the data right into R:

```
raw <- textConnection("
100 P 18 100 P 14 100 D 23 100 D 18 100 P 10 100 P 17 100 D 18 100 D 22
100 P 13 100 P 12 100 D 28 100 D 21 100 P 11 100 P 6 100 D 11 100 D 25
100 P 7 100 P 10 100 D 29 100 P 12 100 P 12 100 P 10 100 D 18 100 D 14
101 P 18 101 P 15 101 D 12 101 D 17 101 P 17 101 P 13 101 D 14 101 D 7
101 P 18 101 P 19 101 D 11 101 D 9 101 P 12 101 D 11 102 P 18 102 P 15
102 P 12 102 P 18 102 D 20 102 D 18 102 P 14 102 P 12 102 D 23 102 D 19
102 P 11 102 P 10 102 D 22 102 D 22 102 P 19 102 P 13 102 D 18 102 D 24
102 P 13 102 P 6 102 D 18 102 D 26 102 P 11 102 P 16 102 D 16 102 D 17
102 D 7 102 D 19 102 D 23 102 D 12 103 P 16 103 P 11 103 D 11 103 D 25
103 P 8 103 P 15 103 D 28 103 D 22 103 P 16 103 P 17 103 D 23 103 D 18
103 P 11 103 P -2 103 D 15 103 D 28 103 P 19 103 P 21 103 D 17 104 D 13
104 P 12 104 P 6 104 D 19 104 D 23 104 P 11 104 P 20 104 D 21 104 D 25
104 P 9 104 P 4 104 D 25 104 D 19
")
d <- scan(raw, what = "character")
rm(raw)
d <- as.data.frame(matrix(d, ncol = 3, byrow = TRUE))
names(d) <- c("center", "drug", "change")
d$change <- as.numeric(as.character(d$change))
d$drug <- relevel(d$drug, ref = "P")
```

Briefly, the idea is to copy and paste the SAS DATALINES instruction as raw text and to scan the flow of characters. The next bit of code uses `matrix` to arrange the data into a tabular dataset with 3 columns corresponding to center, drug and change score. When transforming this table to a data

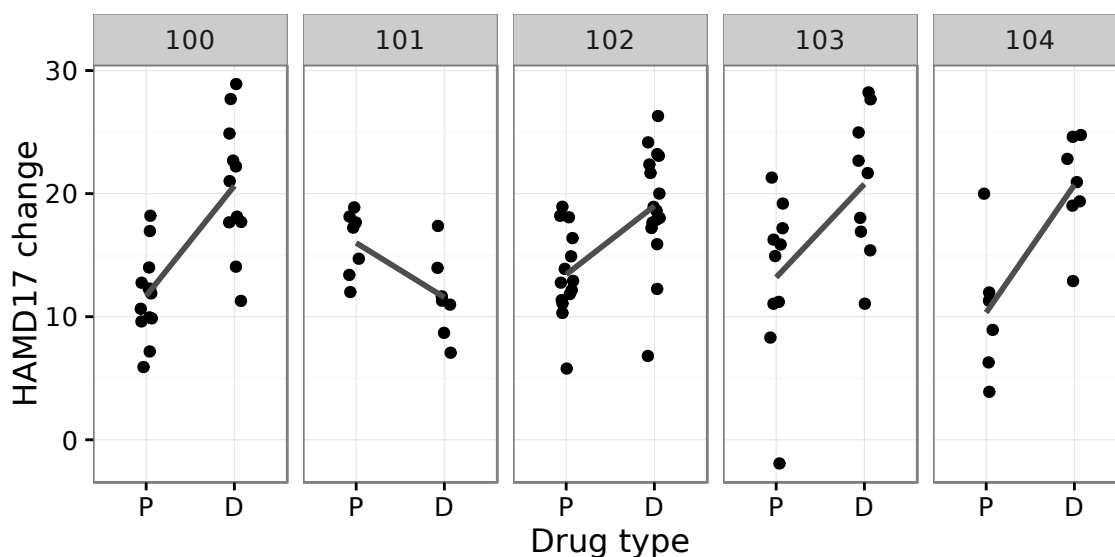


Figure 1: Distribution of change scores in each centre

frame, center and drug will be converted to factors but we need to handle the proper conversion of change to numerical values. Also, note that we set the reference category to the Placebo group to simplify things a bit.

Some basic exploratory graphical analysis follows. In the next chunk, we display the raw data for each centre and highlight the difference between drug and placebo using a trend line (Figure 1). Note the use of `aes(group = 1)` when calling `geom_smooth` as there is no real grouping variable in the data structure other than the ones that are already used (drug on the x-axis and center for facetting).

```
p <- ggplot(data = d, aes(x = drug, y = change))
p <- p + geom_jitter(width = .2)
p <- p + geom_smooth(aes(group = 1), method = "lm", se = FALSE, colour = "grey30")
p + facet_grid(~ center) + labs(x = "Drug type", y = "HAMD17 change")
```

Using Hmisc package, we can easily build a Table of summary statistics by drug and center. For simplicity, we will limit the display to the first 3 centers in Table 1.

```
fm <- change ~ drug + center
s <- summary(fm, data = subset(d, center %in% c("100", "101", "102")),
             method = "cross", fun = smean.sd)
```

Table 1: Mean HAMD17 change by drug, center

drug	100			101			102			Total		
	N	Mean	SD	N	Mean	SD	N	Mean	SD	N	Mean	SD
P	13	12	3.4	7	16	2.7	14	13	3.6	34	13	3.6
D	11	21	5.6	7	12	3.3	16	19	4.7	34	18	5.7
Total	24	16	6.3	14	14	3.7	30	16	5.0	68	16	5.3

Only 3 out of 5 centres are shown.

Now, let's consider average change scores by center, which are displayed in Figure 2. First, we need to compute the average score in each group, and then compute the difference between the two (called delta). This could be done with Hmisc `summarize` command, but we will rely on the `plyr` package and its `ddply` command. What is important is that the results are returned as a data frame to facilitate the use of `ggplot` data structure in turn.

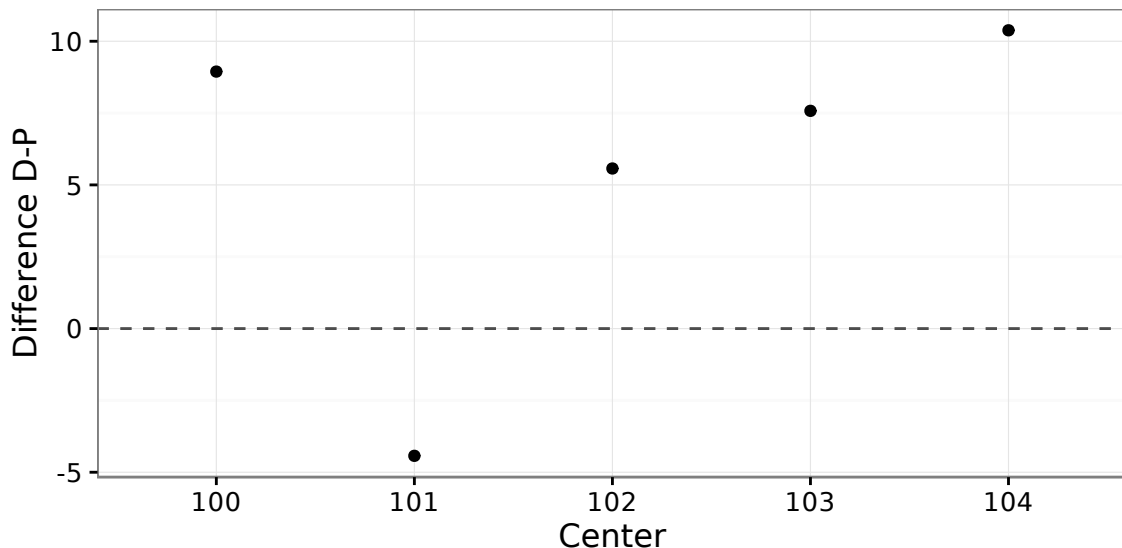


Figure 2: Average difference between drug and placebo in each centre

```
r <- ddply(d, "center", summarize,
           delta = mean(change[drug == "D"]) - mean(change[drug == "P"]))
p <- ggplot(data = r, aes(x = center, y = delta))
p <- p + geom_point() + geom_hline(yintercept = 0, linetype = 2, colour = "grey30")
p + labs(x = "Center", y = "Difference D-P")
```

Now comes the modeling stage. First, we will analyse the primary endpoint using fixed-effect models. Dmitrienko et al. [2005] provide all the maths that are necessary to understand how to derive various types of sum of squares, and this is further addressed in, e.g., REF., or on Stack Exchange.

Let us first update the formula we used for producing Table 1 to incorporate an interaction term, drug:center (in R, drug * center will expand to drug + center + drug:center):

```
fm <- change ~ drug * center

replications(change ~ drug:center, data = d)

## $`drug:center`
##      center
## drug 100 101 102 103 104
##   P   13   7  14  10   6
##   D   11   7  16   9   7
```

As can be seen, data are slightly imbalanced for all but centre 101.

By default, R computes so-called “sequential” Type I sum of squares (SS), and here is what we get when using a standard combination of `lm` (to compute parameter estimates) and `anova` (to build the ANOVA table for the regression model):

```
options(contrasts = c("contr.sum", "contr.poly"))
m <- lm(fm, data = d)
anova(m)

## Analysis of Variance Table
##
## Response: change
##      Df Sum Sq Mean Sq F value    Pr(>F)
## drug      1  888.04   888.04  40.0745 9.365e-09 ***
## center     4   87.14    21.78   0.9831 0.4209278
## drug:center 4  507.45   126.86   5.7249 0.0003761 ***
```

```
## Residuals    90 1994.38    22.16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The car package allows to work with Type II and Type III SS, the later (Yates' weighted squares of means) being the default under SPSS or SAS. So, if we are interested in Type II sum of squares

```
car::Anova(m, type = "II")
## Anova Table (Type II tests)
##
## Response: change
##           Sum Sq Df F value    Pr(>F)
## drug       889.78  1 40.1528 9.109e-09 ***
## center     87.14  4  0.9831 0.4209278
## drug:center 507.45  4  5.7249 0.0003761 ***
## Residuals  1994.38 90
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Type III analysis is readily obtained by replacing type = "II" with type = "III" as shown in the next code block. It should be noted that without altering the default contrast treatment that are used by R, as we did in the above chunk, we would not get the correct results for the Type III analysis.

```
car::Anova(m, type = "III")
## Anova Table (Type III tests)
##
## Response: change
##           Sum Sq Df F value    Pr(>F)
## (Intercept) 22344.6  1 1008.3442 < 2.2e-16 ***
## drug        709.8  1  32.0320 1.783e-07 ***
## center      91.5  4   1.0318 0.3953130
## drug:center  507.4  4   5.7249 0.0003761 ***
## Residuals   1994.4 90
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that in the case of Type III SS, we can also use the base command drop1 and we will get similar results:

```
drop1(m, scope = ~ ., test = "F")
## Single term deletions
##
## Model:
## change ~ drug * center
##           Df Sum of Sq    RSS    AIC F value    Pr(>F)
## <none>                 1994.4 319.29
## drug          1    709.82 2704.2 347.74 32.0320 1.783e-07 ***
## center        4     91.46 2085.8 315.78  1.0318 0.3953130
## drug:center   4     507.45 2501.8 333.96  5.7249 0.0003761 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

To sum up, the results from the different approaches are exposed in Table 2.

Sidenote. Here is how we could compute the parameter estimates and the SS corresponding to the drug effect in the case of a Type III analysis. The code follows that posted on Stack Exchange, with minor adaptation. How this works is quite simple: We first get the design matrix stored in our model `m` and then solve the “normal equations” $(X'X)\hat{\beta} = X'y$ in order to get $\hat{\beta} = (X'X)^{-1}X'y$.

```

D <- model.matrix(m)                                ## design matrix
bhat <- solve(t(D) %*% D) %*% t(D) %*% d$change     ## beta parameters
get.ss <- function(C) {
  require(MASS)
  teta <- C%*%bhat
  M <- C %*% ginv(t(D)%*%D) %*% t(C)
  SSH <- t(teta) %*% ginv(M) %*% teta
  return(as.numeric(SSH))
}
## SS(drug|center, drug:center)
get.ss(matrix(c(0,1,0,0,0,0,0,0,0,0), nrow = 1, ncol = 10))
## [1] 709.8196

```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
drug	1	888.04	888.04	40.07	0.0000
center	4	87.14	21.78	0.98	0.4209
drug:center	4	507.45	126.86	5.72	0.0004
Residuals	90	1994.38	22.16		

(a) Type I SS

	Sum Sq	Df	F value	Pr(>F)
drug	889.78	1	40.15	0.0000
center	87.14	4	0.98	0.4209
drug:center	507.45	4	5.72	0.0004
Residuals	1994.38	90		

(b) Type II SS

	Sum Sq	Df	F value	Pr(>F)
drug	709.82	1	32.03	0.0000
center	91.46	4	1.03	0.3953
drug:center	507.45	4	5.72	0.0004
Residuals	1994.38	90		

(c) Type III SS

Table 2: Overview of fixed-effects analysis for the HAMD17 study

1.2 The Severe sepsis trial

Context. This is a placebo-controlled RCT examining the effect of an experimental drug on 28-day all-cause mortality in patients with severe sepsis. Patients were allocated to one of four strata depending on their APACHE II score.

This is a classical application of stratified analysis of a binary outcome (dead/alive).

To enter the data in R, we will input individual values of the three-way Table of events as an array. Note that it would also be possible to create two matrix objects and then bind into to a 3-dimensional table. In what follows, we write data for the treated group first. Note that when using array, data should be entered column-wise (there is no `byrow =` option as in `matrix`).

```

varnames <- list(strata = 1:4,
                 status = c("Dead", "Alive", "Total"),
                 group = c("Experimental", "Placebo"))

d <- array(c(33,49,48,80,185,169,156,130,218,218,204,210,
            26,57,58,118,189,165,104,123,215,222,162,241),
          dim = c(4,3,2), dimnames = varnames)

```

Note also that the third column ("Total") can be safely omitted as margins can be computed automatically with R, e.g.:

```
addmargins(d[, -3, ], c(1,2))
```

```

d <- d[, -3, ]
dim(d)
## [1] 4 2 2

```

An alternative representation of this array-based Table is provided by R's flat tables (`ftable`), in long or wide format; see Table 3 for the wide format using `ftable(d, row.vars = 1, col.vars = c(3,2))`:

```

ftable(d)

##              group Experimental Placebo
## strata status

```

```
## 1    Dead      33    26
##      Alive    185   189
## 2    Dead      49    57
##      Alive    169   165
## 3    Dead      48    58
##      Alive    156   104
## 4    Dead      80   118
##      Alive    130   123
```

strata	status:	group: Experimental		Placebo	
		Dead	Alive	Dead	Alive
1		33	185	26	189
2		49	169	57	165
3		48	156	58	104
4		80	130	118	123

Table 3: 28-day mortality data from the 1690-patient sepsis study

The following code is used to depict the situation in graphical terms:

```
dd <- as.data.frame(ftable(d))
r <- ddply(dd, c("strata", "group"), mutate, prop = Freq/sum(Freq))
p <- ggplot(subset(r, status == "Dead"), aes(x = prop, y = group))
p <- p + geom_point() + facet_wrap(~ strata, nrow = 2)
p + scale_x_continuous(limits = c(0,0.5)) + labs(x = "Proportion deads", y = "")
```

```
library(vcd)
cotabplot(d, 1)
```

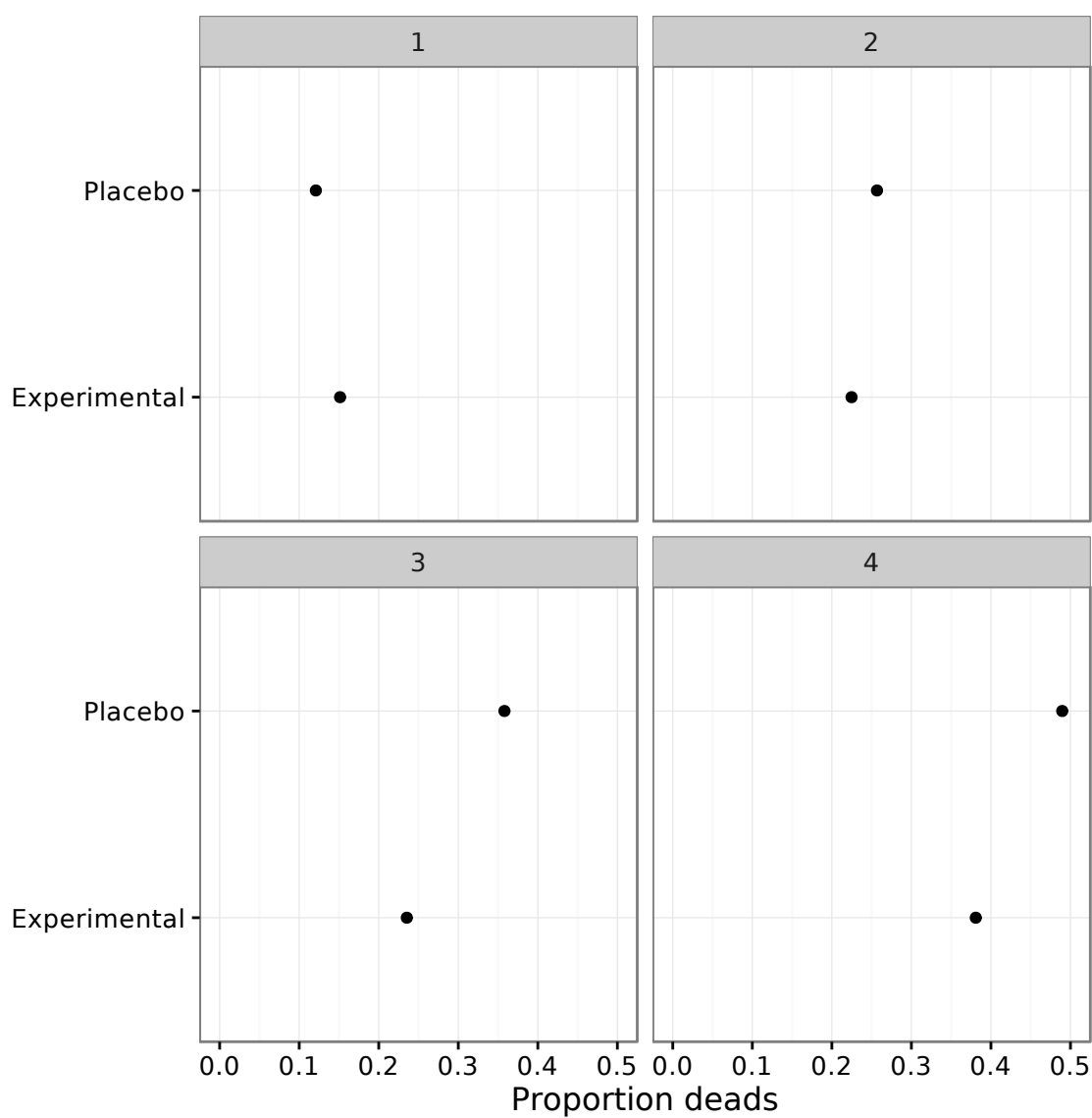


Figure 3: Proportion of patients who died by the end of the study

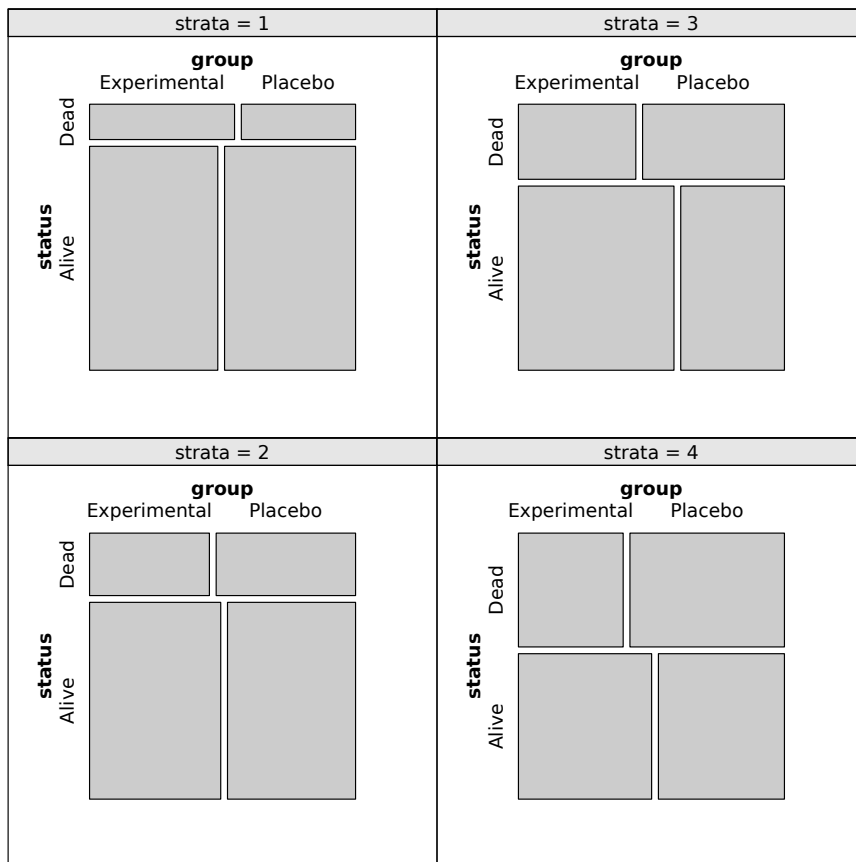


Figure 4: Conditional association plot

References

A Dmitrienko, G Molenberghs, C Chuang-Stein, and W Offen. *Analysis of Clinical Trials Using SAS: A Practical Guide*. SAS Institute Inc., Cary, NC, USA, 2005.

Contents

1 Analysis of Clinical Trials using SAS	1
1.1 The HAMD17 study	1
1.2 The Severe sepsis trial	5

List of Tables

1	Mean HAMD17 change by drug, center	2
2	Overview of fixed-effects analysis for the HAMD17 study	5
3	28-day mortality data from the 1690-patient sepsis study	6

List of Figures

1	Distribution of change scores in each centre	2
2	Average difference between drug and placebo in each centre	3
3	Proportion of patients who died by the end of the study	7
4	Conditional association plot	8