THE TWO-PERIOD CROSS-OVER CLINICAL TRIAL

Eamonn O'Brien

27 July, 2019

Contents

Contents

List of Figures

List of Tables

Some functions and parameters

Hardcode in the data from Hills and Armitage Br. J. clin. Pharmac. (1979), 8, 7-20

```
patient1 <- c(1, 3, 4, 6, 7, 9, 11, 13, 16, 18, 19, 21, 22, 24,
    25, 27, 28)
y1 <- c(8, 14, 8, 9, 11, 3, 6, 0, 13, 10, 7, 13, 8, 7, 9, 10,
    2)
treatment1 <- rep("Treatment1", length(patient1))
order <- rep("First", length(treatment1))


treatment2 <- rep("Treatment2", length(treatment1))
order2 <- rep("Second", length(treatment1))
y2 <- c(5, 10, 0, 7, 6, 5, 0, 0, 12, 2, 5, 13, 10, 7, 0, 6, 2)


patient2 <- c(2, 5, 8, 10, 12, 14, 15, 17, 20, 23, 26, 29)
y1a <- c(12, 6, 13, 8, 8, 4, 8, 2, 8, 9, 7, 7)
treatment2a <- rep("Treatment2", length(patient2))
order3 <- rep("First", length(patient2))


y1b <- c(11, 8, 9, 8, 9, 8, 14, 4, 13, 7, 10, 6)
order4 <- rep("Second", length(patient2))
treatment3a <- rep("Treatment1", length(patient2))

grp1 <- cbind(as.numeric(patient1), (as.character(treatment1)),
    (as.character(order)), as.numeric(y1))

grp2 <- cbind(as.numeric(patient1), (as.character(treatment2)),
    (as.character(order2)), as.numeric(y2))

grp3 <- cbind(as.numeric(patient2), (as.character(treatment2a)),
    (as.character(order3)), as.numeric(y1a))

grp4 <- cbind(as.numeric(patient2), (as.character(treatment3a)),
    (as.character(order4)), as.numeric(y1b))


all <- as.data.frame(rbind(grp1, grp2, grp3, grp4))

all <- plyr::arrange(all, V1, V2)

names(all) <- c("Patient", "Treatment", "Order", "Response")

all$Patient <- as.numeric(as.character(all$Patient))
all$Response <- as.numeric(as.character(all$Response))

all <- plyr::arrange(all, Patient, Treatment)


knitr::kable(all)
```

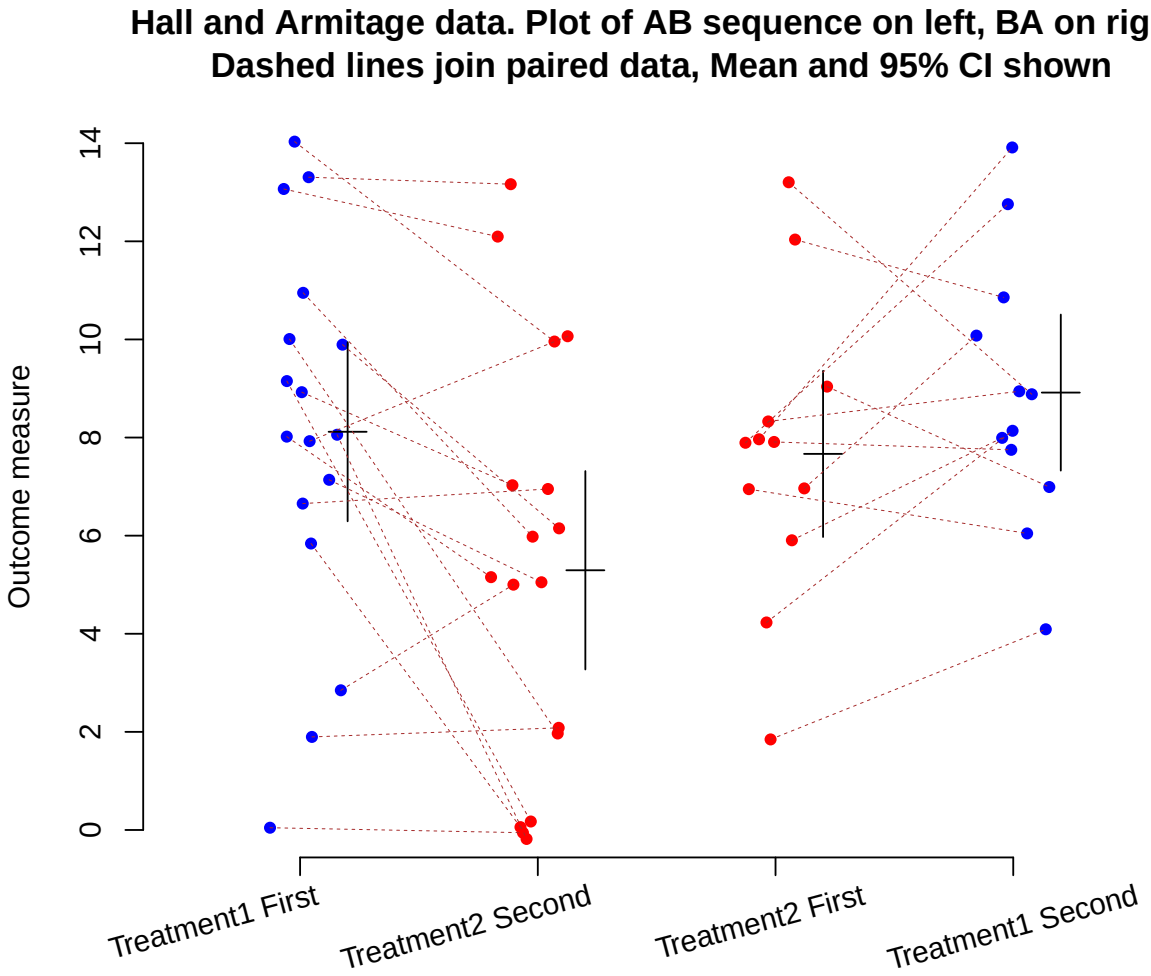
Patient	Treatment	Order	Response
1	Treatment1	First	8
1	Treatment2	Second	5
2	Treatment1	Second	11
2	Treatment2	First	12
3	Treatment1	First	14
3	Treatment2	Second	10
4	Treatment1	First	8
4	Treatment2	Second	0
5	Treatment1	Second	8
5	Treatment2	First	6
6	Treatment1	First	9
6	Treatment2	Second	7
7	Treatment1	First	11
7	Treatment2	Second	6
8	Treatment1	Second	9
8	Treatment2	First	13
9	Treatment1	First	3
9	Treatment2	Second	5
10	Treatment1	Second	8
10	Treatment2	First	8
11	Treatment1	First	6
11	Treatment2	Second	0
12	Treatment1	Second	9
12	Treatment2	First	8
13	Treatment1	First	0
13	Treatment2	Second	0
14	Treatment1	Second	8
14	Treatment2	First	4
15	Treatment1	Second	14
15	Treatment2	First	8
16	Treatment1	First	13
16	Treatment2	Second	12
17	Treatment1	Second	4
17	Treatment2	First	2
18	Treatment1	First	10
18	Treatment2	Second	2
19	Treatment1	First	7
19	Treatment2	Second	5
20	Treatment1	Second	13
20	Treatment2	First	8
21	Treatment1	First	13
21	Treatment2	Second	13
22	Treatment1	First	8
22	Treatment2	Second	10
23	Treatment1	Second	7
23	Treatment2	First	9
24	Treatment1	First	7
24	Treatment2	Second	7
25	Treatment1	First	9
25	Treatment2	Second	0
26	Treatment1	Second	10
26	Treatment2	First	7

Patient	Treatment	Order	Response
27	Treatment1	First	10
27	Treatment2	Second	6
28	Treatment1	First	2
28	Treatment2	Second	2
29	Treatment1	Second	6
29	Treatment2	First	7

Plot the data

```
#  order ABBA

d1 <- all
require(reshape)
d1$grp <- paste(d1$Treatment, d1$Order, sep=" ")
d1 <- rename(d1, c(Response="count"))
d1 <- rename(d1, c(grp="spray"))

# https://stackoverflow.com/questions/17031039/how-to-sort-a-character-vector-according-to-a-specif
ord <-  c("Treatment1 First", "Treatment2 Second", "Treatment2 First", "Treatment1 Second")
d1$spray <- factor(d1$spray,levels=ord)
d1 <- d1[order(d1$spray),]

# create jitter vars
d1$xj <- cumsum(c(TRUE, d1$spray[-1]!=d1$spray[-length(d1$spray)]))
d1$xj <- d1$xj+rnorm(nrow(d1), 0, .10)
d1$yj <- d1$count +  rnorm(nrow(d1), 0, .15)



attach(d1)

sprayTypes <- unique(spray)

y <- as.numeric(as.character(d1[,4]))

plot(y, as.factor(d1[,5]) ,  ylim=c(min(y)-0,max(y)+0) , xlim=c(0.5, 4.5) , xaxt="n",
     main="Hall and Armitage data. Plot of AB sequence on left, BA on right\n Dashed lines join pai
     xlab="", ylab="Outcome measure", frame.plot  =F , col="white")
##if not white I get a nasty boxplot

axis(1,  at=1:4, labels=F )

text(x=1:length(sprayTypes)*1.15, par("usr")[3]-.8, labels = sprayTypes,
     srt = 15, pos = 2, xpd = TRUE,
     cex.axis=.5 )

##make colours
chk <- as.character(d1$spray)
x <- as.data.frame(table(chk))

clr <- c("blue","red")



for (i in 1 : length(sprayTypes)) {

  n <- sum(spray == sprayTypes[i])

  y <- count[spray == sprayTypes[i]]

  xi <- xj[spray == sprayTypes[i]]
```

```
    yi <- yj[spray == sprayTypes[i]]

    points(x=xi, y=yi, pch = 16, cex = .9,
           col =ifelse(i %in% c(1,4),clr[1],clr[2])
      )

    # y is used here
    lines(i +  c(.12, .28), rep(mean(y), 2), lwd = 1, col="black") #

    lines(rep(i + .2, 2),  # start and end
          mean(y) + c(-1.96, 1.96) * sd(y) / sqrt(n),  lwd = 1, col="black"
    ) # vertical

  }


  # join the pairs
  # manage the data , create a new wide data set of 4 coordinates

   w1 <- d1[,c(1,3,6,7)]

   data1 <- melt(w1 , id.vars = c("Patient", "Order"))

   data1$temp <- paste(data1$Order, data1$variable)

   wx <- reshape(data1[,c(1,4,5)], idvar="Patient",
                 v.names="value", timevar=c("temp"),
                 direction="wide")

   wx <- wx[,c(1,2,4,3,5)]

   names(wx) <- c("Patient","x1","y1","x2","y2")


  for(s in 1:nrow(wx)) {

      segments(wx$x1[s], wx$y1[s], wx$x2[s], wx$y2[s], col="brown", lwd=0.5, lty=2)

  }
```

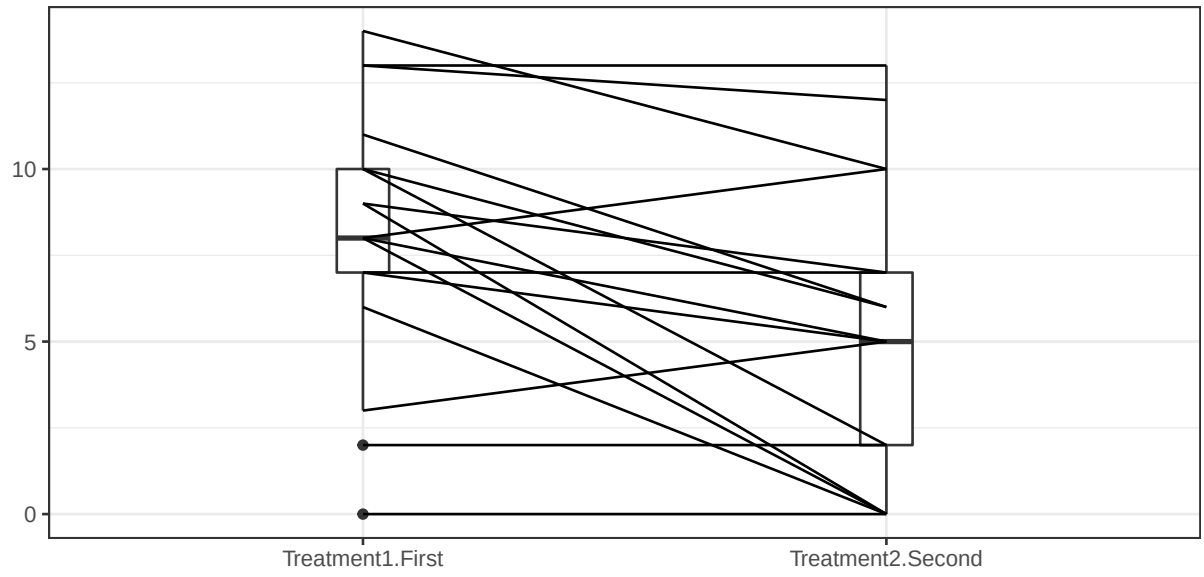


Comment

Nice plot showing all the data, the blocking in the experiment (patient), the between patient distribution and the sequences of treatment. We would ideally expect AB and BA to be a relection of each other. We show the mean and 95% CI. In fact the standard errors or confidence intervals for individual means are based on an assumption of simple random sampling that is not the case in the crossover trial. So maybe a boxplot would be better. Random noise (jitter) added to datapoints in both x and y axes to make visualisation better.

Plot the data once again

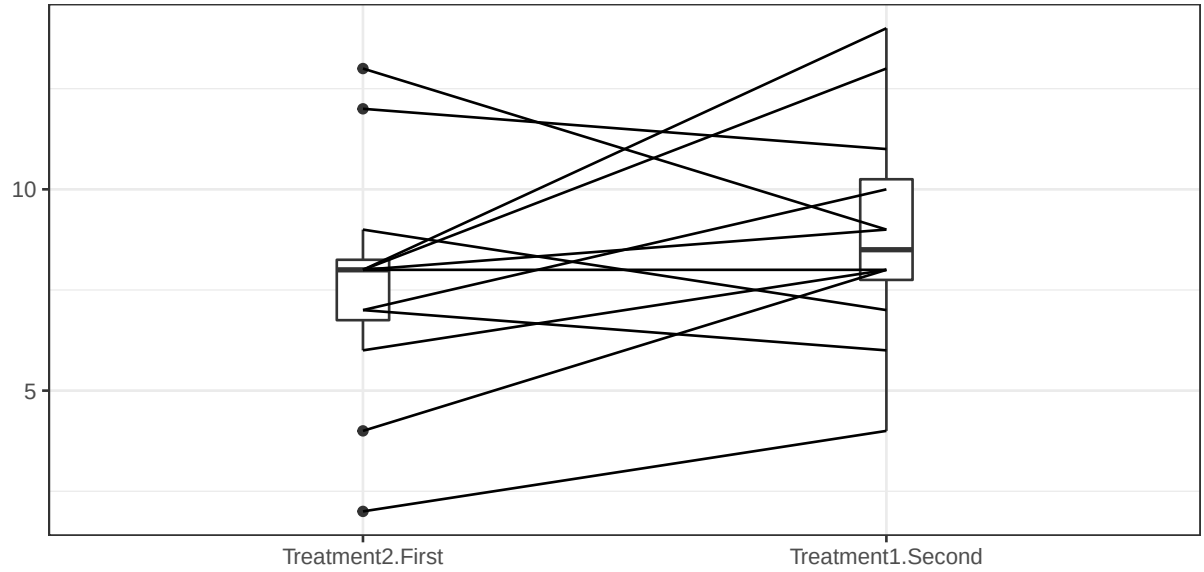
```
mydata <- all
require("PairedData")
# useful concatenation for plotting
mydata$temp <- paste(mydata$Treatment, mydata$Order, sep = ".")

Treatment1.First <- subset(mydata, temp == "Treatment1.First",
    Response, drop = TRUE)
Treatment2.Second <- subset(mydata, temp == "Treatment2.Second",
    Response, drop = TRUE)

pd <- paired(Treatment1.First, Treatment2.Second)
plot(pd, type = "profile") + theme_bw()
```




```
cat("\n\n\\pagebreak\n")
```

\pagebreak

```
Treatment2.First <- subset(mydata, temp == "Treatment2.First",
    Response, drop = TRUE)
Treatment1.Second <- subset(mydata, temp == "Treatment1.Second",
    Response, drop = TRUE)

pd <- paired(Treatment2.First, Treatment1.Second)
# plot(pd, type = 'BA') + theme_bw()
plot(pd, type = "profile") + theme_bw()
```


10
5
Treatment2.First
Treatment1.Second

Period and treatment summary stats, agree with paper

```
# with(all, tapply(Response, list(Treatment, Order), mean))

require(tidyverse)
all %>% group_by(Treatment, Order) %>% summarise_each(funs(n = length(!is.na(.)),
    mean, sd, se = sd(.)/sqrt(n())), Response)
```

```
# A tibble: 4 x 6
# Groups:   Treatment [2]
  Treatment  Order      n  mean    sd    se
  <fct>      <fct>  <int> <dbl> <dbl> <dbl>
1 Treatment1 First     17  8.12  3.84 0.931
2 Treatment1 Second    12  8.92  2.81 0.811
3 Treatment2 First     12  7.67  2.99 0.865
4 Treatment2 Second    17  5.29  4.25 1.03
```

```
# calc difference in treatements and summarise
w <- spread(select(all, -c(Order)), Treatment, Response)

w <- w %>% select(Treatment1, Treatment2) %>% mutate(Response = Treatment1 -
    Treatment2)  #%>% head()

w %>% summarise(mean = mean(Response), sd = sd(Response), n = length(!is.na(Response)),
    se = sd/sqrt(n))
```

```
      mean       sd  n        se
1 2.172414 3.317367 29 0.6160197
```

Fit a random effects model (no order nor interaction term)

```
require(nlme)
f <- lme(Response ~ Treatment, random = ~1 | Patient, data = all,
    na.action = "na.omit")
anova(f)
```

```
            numDF denDF   F-value p-value
(Intercept)     1    28 146.48262  <.0001
Treatment       1    28  12.43644  0.0015
```

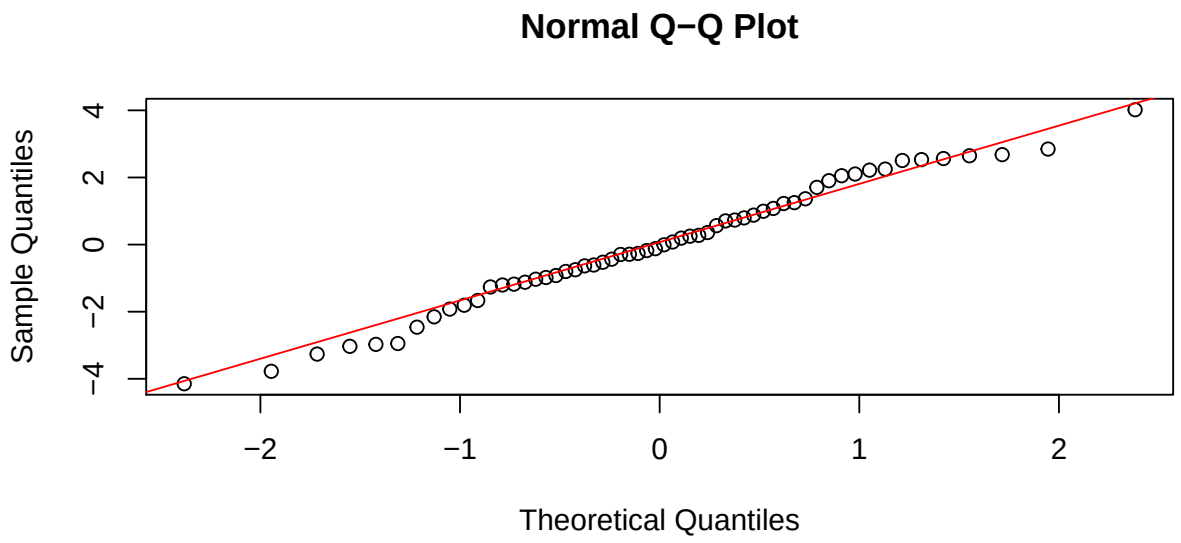
```
summary(f)$tTable
```

```
                        Value Std.Error DF   t-value
(Intercept)          8.448276 0.6818213 28 12.390749
TreatmentTreatment2 -2.172414 0.6160197 28 -3.526533
                                 p-value
(Intercept)         0.0000000000006969003
TreatmentTreatment2 0.0014712573664130472
```

```
intervals(f)
```

```
Approximate 95% confidence intervals

 Fixed effects:
                        lower      est.      upper
(Intercept)          7.051628  8.448276  9.8449234
TreatmentTreatment2 -3.434273 -2.172414 -0.9105547
attr(,"label")
[1] "Fixed effects:"

 Random Effects:
  Level: Patient
                   lower     est.   upper
sd((Intercept)) 1.963652 2.824724 4.06338

 Within-group standard error:
   lower     est.    upper
1.805238 2.345733 3.048054
```

```
qqnorm(resid(f), main = "Normal Q-Q Plot")
qqline(resid(f), col = "red")
```



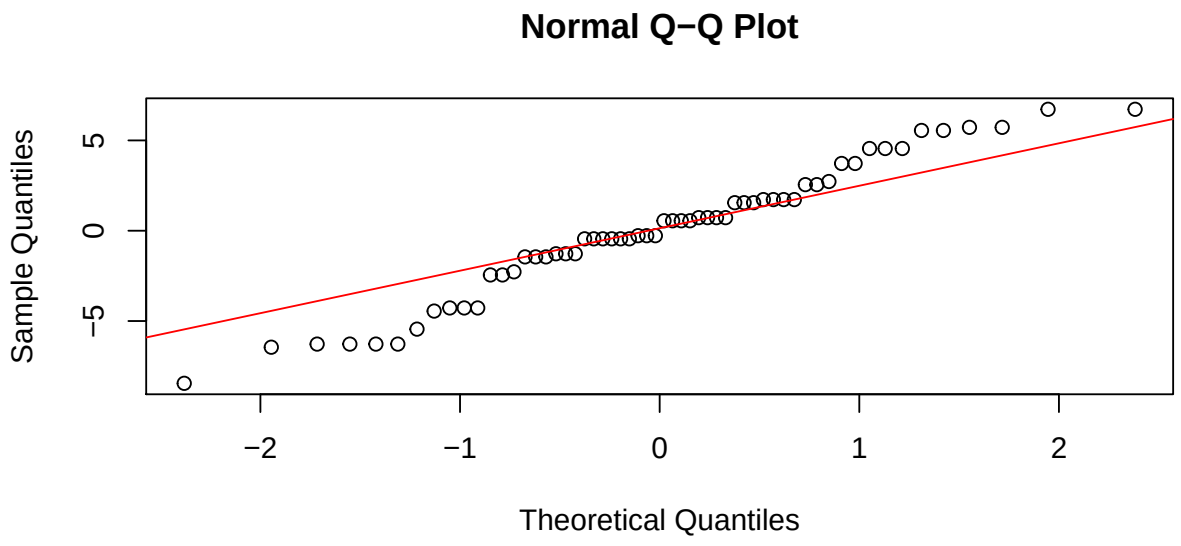

```
# collect the treatment effect estimate to make inferences
# later
z <- as.matrix(summary(f)$tTable)
Treatment <- z[2, 1][[1]]
```

Fit a linear regression model

```
f <- lm(Response ~ Treatment, data = all, na.action = "na.omit")
summary(f)
```

```

Call:
lm(formula = Response ~ Treatment, data = all, na.action = "na.omit")

Residuals:
    Min      1Q  Median      3Q     Max
-8.4483 -1.4483  0.1379  1.7241  6.7241

Coefficients:
                    Estimate Std. Error t value
(Intercept)           8.4483     0.6818  12.391
TreatmentTreatment2  -2.1724     0.9642  -2.253
                             Pr(>|t|)
(Intercept)         <0.0000000000000002 ***
TreatmentTreatment2              0.0282 *
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.672 on 56 degrees of freedom
Multiple R-squared:  0.08311,   Adjusted R-squared:  0.06674
F-statistic: 5.076 on 1 and 56 DF,  p-value: 0.0282
```

```
qqnorm(resid(f), main = "Normal Q-Q Plot")
qqline(resid(f), col = "red")
```



Function to analyse using permutation approach to duplicate Stephen Senn talk

```
Dq1 <- all
library(data.table)
Dq1 <- as.data.table(all)

# function to get permuted distribution of treatment effect

perm.dist <- function(block = "yes", n.sims = 10000) {

    # set up an array to store parameter estimates
    estArray <- array(NA, dim = c(n.sims, 4))

    for (s in 1:n.sims) {

        # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        # permute

        if (block == "yes") {

            # permute within person
            permz <- Dq1[, `:=`(y, sample(Response)), by = Patient]

        } else {

            # no blocking
            permz <- Dq1[, `:=`(y, sample(Response))]

        }

        # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        # analysis

        # respecting blocking
        possibleError <- tryCatch(f1 <- lme(y ~ Treatment, random = ~1 |
            Patient, data = permz, method = "REML"), error = function(e) e)

        # http://stackoverflow.com/questions/8093914/skip-to-next-value-of-loop-upon-error-in-r-trycatc

        if (!inherits(possibleError, "error")) {

            modelint <- possibleError
            z <- as.matrix(summary(modelint)$tTable)

        }


        # ignoring blocking
        possibleError2 <- tryCatch(f0 <- lm(y ~ Treatment, data = permz),
            error = function(e) e)

        if (!inherits(possibleError, "error")) {

            modelint1 <- possibleError2
```

```
            zz <- as.matrix(summary(modelint1)$coefficients)


        }


        estArray[s, 1] <- z[2, 1][[1]]  # collect trt effect estimate
        estArray[s, 2] <- vcov(modelint)[2, 2]  # collect variance of trt effect estimate


        estArray[s, 3] <- zz[2, 1][[1]]  # collect trt effect estimate
        estArray[s, 4] <- vcov(modelint1)[2, 2]  # collect variance of trt effect estimate

    }

    list(estArray = estArray)
}
```

Comment

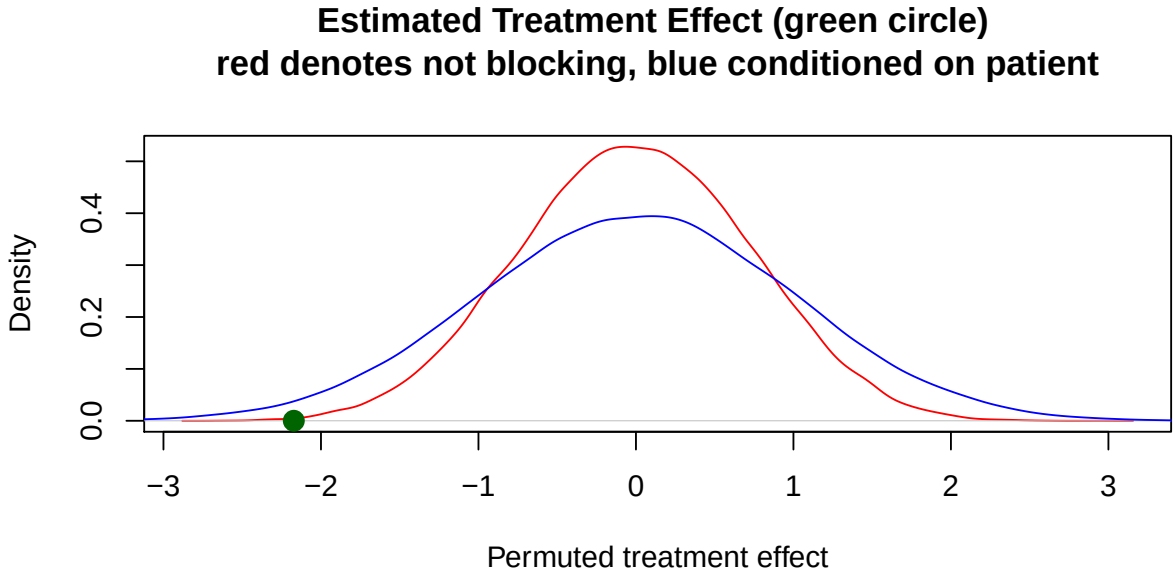
Function that allows one to permutate data either conditioning on patient (block) correctly by permuting the two values within patient or ignoring the blocking and permuting across all the data. Secondly to run the correct analysis conditioning on patient (LMM model) or not conditioning on patient OLS model.

Execute the simulations

```
block <- perm.dist(block = "yes", n.sims = sim)

no.block <- perm.dist(block = "no", n.sims = sim)
```

Plot the treatment effect permutation distributions and present the treatment effect estimate

```
A <- block$estArray[, 1]
B <- no.block$estArray[, 1]

plot(density(A), col = "red", main = "Estimated Treatment Effect (green circle) \nred denotes not block
    xlab = "Permuted treatment effect")
lines(density(B), col = "blue")
points(x = Treatment, y = 0, col = "darkgreen", cex = 1.5, pch = 19)
```



Permutation p values

```
# see Senn 34.09mins right panel youtube, good match!
sum(abs(block$estArray[, 1]) >= abs(Treatment))/sim  # Senn 0.0014
```

[1] 0.00148

```
sum(abs(no.block$estArray[, 1]) >= abs(Treatment))/sim  # Senn 0.034
```

[1] 0.02694

Summary statisitcs from the permutation approaches

```
# Manage the estimates Blocked
A <- as.data.frame(apply(block$estArray, 2, summary))
names(A) <- c("LMM Mean trt effect", "LMM Mean Var of trt effect",
    "Ols Mean trt effect", "Ols Mean Var of trt effect")


B <- t(as.data.frame(unlist(apply(block$estArray, 2, var))))
rownames(B) <- NULL
B <- as.data.frame(B)
names(B) <- c("Var of LMM trt effect", "Var of LMM Var of trt effect",
    "Var of Ols trt effect", "Var of Ols Var of trt effect")

# not blocked
C <- as.data.frame(apply(no.block$estArray, 2, summary))
names(C) <- names(A)


D <- t(as.data.frame(unlist(apply(no.block$estArray, 2, var))))
rownames(D) <- NULL
D <- as.data.frame(D)
names(D) <- names(B)
```

Blocked estimates

```
knitr::kable(A)
```

	LMM Mean trt effect	LMM Mean Var of trt effect	Ols Mean trt effect	Ols Mean Var of trt effect
Min.	-2.6551724	0.2412086	-2.6551724	0.8606251
1st Qu.	-0.5172414	0.5214880	-0.5172414	1.0007644
Median	-0.0344828	0.5384745	-0.0344828	1.0092577
Mean	-0.0055448	0.5291129	-0.0055448	1.0045769
3rd Qu.	0.5172414	0.5459486	0.5172414	1.0129947
Max.	2.9310345	0.5479870	2.9310345	1.0140139

```
knitr::kable(B)
```

Var of LMM trt effect	Var of LMM Var of trt effect	Var of Ols trt effect	Var of Ols Var of trt effect
0.529643	0.0006473	0.529643	0.0001618

Not conditioned on blocking, estimates

```
knitr::kable(C)
```

	LMM Mean trt effect	LMM Mean Var of trt effect	Ols Mean trt effect	Ols Mean Var of trt effect
Min.	-4.3793103	0.3267368	-4.3793103	0.6715645
1st Qu.	-0.6551724	0.8657211	-0.6551724	0.9909122
Median	0.0344828	0.9710379	0.0344828	1.0063700
Mean	0.0032345	0.9211857	0.0032345	0.9962786
3rd Qu.	0.6551724	1.0063700	0.6551724	1.0123153
Max.	3.8965517	1.0140139	3.8965517	1.0140139

```
knitr::kable(D)
```

Var of LMM trt effect	Var of LMM Var of trt effect	Var of Ols trt effect	Var of Ols Var of trt effect
0.9943758	0.0123249	0.9943758	0.0006075

Summary

As Stephen Senn explains in the video. Analysis must relect the block structure. Looking at the distributions of the permutated treatment effects, the effect is the same average difference. The distribution accounting for the fact the same patient is treated on two different occasions is narrower than the distribution that does not condition on patient, in consequence much more unusal event compared to the permutation distribution. (If covariates differ greatly from one patient to another we will see it in the residual error term and we make a judgement of efficacy of something compared to residual error term. Notice the different permutation p-values and compare to parametric p-values. Moral, more important than deciding to use a linear model or permutation test is to condition on block structure of experiment.

What happens when you balance but don't condition: That is to say, permute values respecting the fact that they come from a crossover but analysing them as if they came from parallel group trial:

Approach, completely randomised and analysed as such 0.9943758 . Variance of the treatment effect is equal to the mean of the variance (of treatment effect over all randomisations) 0.9962786

Approach, randomised within patient and analysed as such 0.529643 . Variance of the treatment effect is equal to the mean of the variance (of treatment effect over all randomisations) 0.5291129

Approach, randomised within patient and analysed as completely randomised 0.529643 . Variance of the treatment effect is not equal to the mean of the variance (of treatment effect over all randomisations) 1.004577

Computing Environment

```
sessionInfo()
```

```
R version 3.6.1 (2019-07-05)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 17134)

Matrix products: default

locale:
[1] LC_COLLATE=English_United Kingdom.1252
[2] LC_CTYPE=English_United Kingdom.1252
[3] LC_MONETARY=English_United Kingdom.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United Kingdom.1252

attached base packages:
[1] stats     graphics  grDevices utils     datasets
[6] methods   base

other attached packages:
 [1] data.table_1.12.2 nlme_3.1-140      forcats_0.4.0
 [4] stringr_1.4.0     dplyr_0.8.3       purrr_0.3.2
 [7] readr_1.3.1       tidyr_0.8.3       tibble_2.1.3
[10] tidyverse_1.2.1   PairedData_1.1.1  ggplot2_3.2.0
[13] lattice_0.20-38   mvtnorm_1.0-11    gld_2.5
[16] MASS_7.3-51.4     reshape_0.8.8     knitr_1.23

loaded via a namespace (and not attached):
 [1] tidyselect_0.2.5 xfun_0.8         haven_2.1.1
 [4] colorspace_1.4-1 generics_0.0.2   vctrs_0.2.0
 [7] htmltools_0.3.6  yaml_2.2.0       utf8_1.1.4
[10] rlang_0.4.0      e1071_1.7-2      pillar_1.4.2
[13] glue_1.3.1       withr_2.1.2      readxl_1.3.1
[16] modelr_0.1.4     plyr_1.8.4       cellranger_1.1.0
[19] munsell_0.5.0    gtable_0.3.0     rvest_0.3.4
[22] evaluate_0.14    labeling_0.3     lmom_2.8
[25] class_7.3-15     fansi_0.4.0      highr_0.8
[28] broom_0.5.2      Rcpp_1.0.1       scales_1.0.0
[31] backports_1.1.4  formatR_1.7      jsonlite_1.6
[34] hms_0.5.0        digest_0.6.20    stringi_1.4.3
[37] grid_3.6.1       cli_1.1.0        tools_3.6.1
[40] magrittr_1.5     lazyeval_0.2.2   crayon_1.3.4
[43] pkgconfig_2.0.2  zeallot_0.1.0    xml2_1.2.0
[46] lubridate_1.7.4  rstudioapi_0.10  assertthat_0.2.1
[49] rmarkdown_1.14   httr_1.4.0       R6_2.4.0
[52] compiler_3.6.1
```

This took 3347.18 seconds to execute.