Solution to Exercise 3

Department of Methodology and Statistics
Utrecht University

Bayesian Statistics, Week 8

A. Recognizing the model assumption(s).

Assumptions:

- The persons within one condition constitute a homogeneous group and are interchangeable, when it comes to their chance of recovery (LD).

This assumption could be violated if:

- There are order effects (more likely with a convenience sample or self-enrollment). It is possible that those who signed up for the study immediately after it started where having more intense PTSD symptoms, whereas those who enrolled later were already 'healthier' to begin with. Or the involved therapists may (unconsciously) have improved their method within the study period, so that subjects who started out later received more effective treatment.

B. Writing the equivalent model for the raw (individual) data.

Exercise 2 - ModelC - Kladblok

Bestand Bewerken Opmaak Beeld Help

```
# Bayesian Statistics - Exercise 2 Model file
# fitting the model to the raw data, and conducting a posterior predictive check

model {
 # Likelihood of the data
 for (i in 1:n.PE) {
  LD.PE[i] ~ dbern(theta.PE)
 }

 for(i in 1:n.PC) {
  LD.PC[i] ~ dbern(theta.PC)
 }

  # Prior distributions
  theta.PE ~ dbeta(1,1)
  theta.PC ~ dbeta(1,1)

  # Relative Risk
  RR <- theta.PC / theta.PE
}
```

C. Running the analysis, checking the results against Exercise 1 - commands

```
# 1. Load the data into R
source("Exercise 2 - Data.txt")

# 2. Create a model object with the jags.model() function
# Specify the model, data and number of chains
model <- jags.model(file = "Exercise 2 - Model_specified.txt",
                                        data = dat, n.chains = 2)

# 3. Use the update function to run the Markov Chain for a
# burn-in period of 1000 iterations
update(object = model, n.iter = 1000)

# 4. Use the coda.samples function to get samples from the
# posterior distribution
parameters <- c("theta.PE", "theta.PC", "RR")
samples <- coda.samples(model = model, variable.names = parameters,
                                                n.iter = 10000)

# 5. Get an informative summary of the samples
summary(samples)
```

C. Running the analysis, checking the results against Exercise 1 - output

Small differences can arise between model runs because all the estimates are based on a sampling procedure!

```
Iterations = 1001:11000
Thinning interval = 1
Number of chains = 2
Sample size per chain = 10000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

           Mean     SD Naive SE Time-series SE
RR       0.694 0.1165 0.000824       0.000843
theta.PC 0.283 0.0374 0.000265       0.000265
theta.PE 0.412 0.0409 0.000289       0.000289

2. Quantiles for each variable:

           2.5%   25%   50%   75% 97.5%
RR       0.489 0.612 0.686 0.768 0.942
theta.PC 0.211 0.257 0.282 0.308 0.359
theta.PE 0.333 0.384 0.412 0.439 0.494
```

D. Specifying discrepancy measures for the emperical data.


```
# Discrepancy measure for the PE condition
proportion.half1.PE <- mean(dat$LD.PE[1:70])
proportion.half2.PE <- mean(dat$LD.PE[71:141])
diff.PE <- proportion.half1.PE - proportion.half2.PE

# Discrepancy measure for the PC condition
proportion.half1.PC <- mean(dat$LD.PC[1:71])
proportion.half2.PC <- mean(dat$LD.PC[72:143])
diff.PC <- proportion.half1.PC - proportion.half2.PC
```

E. Calculating the discrepancy measures for model-predicted (replicated) data sets.

One way of doing this (repeat for both chains) :

```
# Chain 1
predicted.proportion.half1.PE.chain1 <-
    apply(replicated.PE.chain1, 1, function(x) mean(x[1:70]))
predicted.proportion.half2.PE.chain1 <-
    apply(replicated.PE.chain1, 1, function(x) mean(x[71:141]))
predicted.diff.PE.chain1 <- predicted.proportion.half1.PE.chain1 -
    predicted.proportion.half2.PE.chain1

predicted.proportion.half1.PC.chain1 <-
    apply(replicated.PC.chain1, 1, function(x) mean(x[1:71]))
predicted.proportion.half2.PC.chain1 <-
    apply(replicated.PC.chain1, 1, function(x) mean(x[72:143]))
predicted.diff.PC.chain1 <- predicted.proportion.half1.PC.chain1 -
    predicted.proportion.half2.PC.chain1
```

F. Adding code to obtain posterior predictive p-values for the two groups.

One-sided vs. two-sided: depends on the model assumption tested, and which violations are possible/plausible. Here, two-sided (repeat for both chains):

```
# Chain 1
abs.diff.PE <- abs(diff.PE)
abs.pred.diff.PE.chain1 <- abs(predicted.diff.PE.chain1)
ppp.PE.chain1 <- ifelse(
    (abs.pred.diff.PE.chain1 - abs.diff.PE) >= 0, yes = 1, no = 0)

abs.diff.PC <- abs(diff.PC)
abs.pred.diff.PC.chain1 <- abs(predicted.diff.PC.chain1)
ppp.PC.chain1 <- ifelse(
    (abs.pred.diff.PC.chain1 - abs.diff.PC) >= 0, yes = 1, no = 0)
```

G. Interpreting the posterior predictive p-values.


```
# Posterior predictive p-values
mean(ppp.PC.chain1)
```

```
## [1] 0.0037
```

```
mean(ppp.PE.chain1)
```

```
## [1] 0.467
```

The binomial assumption is violated in the PC group, because the first half of the sample has a different chance of recovery than the second half, ppp $= 0.004$.