> ########## ########## Simulation of a possible application of NMA in biotechnology ########## ##########

> ########## use of the rjags package

> ########## NMA in the wastewater treatment

>

>

>

>

> # Part 1: Praparation ------------------------------------------------------------------

>

>

>

>

> ##### Clear data

> rm(list=ls())

>

>

>

> ##### load libraries

> library(rjags)

> library(runjags)

> library(coda)

> library(random)

> library(readxl)

> library(matrixStats) # additional package, calculates Median

> load.module("glm")

> load.module("lecuyer")

> load.module("dic")

>

>

>

> ##### Setting Working directory

> setwd("C:/Users/IvanB/Desktop/Masterarbeit/Ergebnisse/zusätzliche Experimente/Anwendung an die Biotec/Ansatz 1")

>

>

>

> ##### Generation of Data

> ## optimal conditions

> set.seed(1)

> Data\_opt <- rnorm(12, mean = 6000, sd = 100)

> #Data\_opt

> Data\_opt[4:6] <- Data\_opt[4:6]/10

> Data\_opt[7:9] <- Data\_opt[7:9]/100

> Data\_opt[10:12] <- Data\_opt[10:12]/1000

> #Data\_opt

> Data\_opt <- as.integer( Data\_opt)

> #Data\_opt

>

> ## Toxin A-free conditions

> set.seed(2)

> Data\_tox\_free <- rnorm(12, mean = 4000, sd = 100)

> #Data\_tox\_free

> Data\_tox\_free[4:6] <- Data\_tox\_free[4:6]/10

> Data\_tox\_free[7:9] <- Data\_tox\_free[7:9]/100

> Data\_tox\_free[10:12] <- Data\_tox\_free[10:12]/1000

> #Data\_tox\_free

> Data\_tox\_free <- as.integer( Data\_tox\_free)

> #Data\_tox\_free

>

> ## Toxin A containing conditions

> set.seed(3)

> Data\_toxin <- rnorm(12, mean = 3800, sd = 100)

> #Data\_toxin

> Data\_toxin[4:6] <- Data\_toxin[4:6]/(10\*2)

> Data\_toxin[7:9] <- Data\_toxin[7:9]/(100\*3)

> Data\_toxin[10:12] <- Data\_toxin[10:12]/(1000\*4)

> #Data\_toxin

> Data\_toxin <- as.integer( Data\_toxin)

> #Data\_toxin

>

> ## all Data

> Data\_opt

[1] 5937 6018 5916 615 603 591 60 60 60 5 6 6

> Data\_tox\_free

[1] 3910 4018 4158 388 399 401 40 39 41 3 4 4

> Data\_toxin

[1] 3703 3770 3825 184 190 190 12 13 12 0 0 0

> # -> Data seems plausible and are saved in the Daten.xlsx-file

>

>

>

> ##### Read the data into R.

> #data = read\_xlsx("Daten.xlsx")

> data = as.matrix(read.table("Daten.txt", sep = "", header=F))

> head(data) # Shows the first six entries

V1 V2 V3 V4 V5 V6 V7

[1,] 4312 2952 7482 7482 1 2 2

[2,] 5717 2298 6235 6235 1 2 2

[3,] 4336 3766 5759 5759 1 2 2

[4,] 348 258 816 816 1 2 2

[5,] 501 319 535 535 1 2 2

[6,] 361 378 649 649 1 2 2

>

>

>

> ##### Values for simulation, prepare dat for JAGS (allocation values from data)

> ns <- nrow(data)

> # ns # check

> nt <- ncol(data[,5:6])

> #nt # check

>

> na <- data[,7]

> # na # check

> r <- data[,1:2]

> # r # Check

> n <- data[,3:4]

> # n # Check

> t <- data[,5:6]

> # t # Check

>

>

>

> ##### read in inits with chains

> inits1 <- list(d=c( NA, 0),

+ sd=1,

+ mu=c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),

+ .RNG.name="lecuyer::RngStream", .RNG.seed=1)

>

> inits2 <- list(d=c( NA, -1),

+ sd=4,

+ mu=c(-3, -3, -3, -3, -3, -3, -3, -3, -3, -3, -3, -3),

+ .RNG.name="lecuyer::RngStream", .RNG.seed=2)

>

> inits3 <- list(d=c( NA, 2),

+ sd=2,

+ mu=c(-3, 5, -1, -3, 7, -3, -4, -3, -3, 0, -3, -3),

+ .RNG.name="lecuyer::RngStream", .RNG.seed=3 )

>

> all.inits <- list(inits1, inits2, inits2)

>

>

>

> ##### define JAGS model within R

> cat("model{ # \*\*\* PROGRAM STARTS

+ for(i in 1:ns){ # LOOP THROUGH STUDIES

+ w[i,1] <- 0 # adjustment for multi-arm trials is zero for control arm

+ delta[i,1] <- 0 # treatment effect is zero for control arm

+ mu[i] ~ dnorm(0,.0001) # vague priors for all trial baselines

+ for (k in 1:na[i]) { # LOOP THROUGH ARMS

+ r[i,k] ~ dbin(p[i,k],n[i,k]) # binomial likelihood

+ logit(p[i,k]) <- mu[i] + delta[i,k] # model for linear predictor

+ rhat[i,k] <- p[i,k] \* n[i,k] # expected value of the numerators

+ dev[i,k] <- 2 \* (r[i,k] \* (log(r[i,k])-log(rhat[i,k]))+ (n[i,k]-r[i,k]) \* (log(n[i,k]-r[i,k]) - log(n[i,k]-rhat[i,k]))) #Deviance contribution

+ }

+ resdev[i] <- sum(dev[i,1:na[i]]) # summed residual deviance contribution for this trial

+ for (k in 2:na[i]) { # LOOP THROUGH ARMS

+ delta[i,k] ~ dnorm(md[i,k],taud[i,k]) # trial-specific LOR distributions

+ md[i,k] <- d[t[i,k]] - d[t[i,1]] + sw[i,k] # mean of LOR distributions (with multi-arm trial correction)

+ taud[i,k] <- tau \*2\*(k-1)/k # precision of LOR distributions (with multi-arm trial correction)

+ w[i,k] <- (delta[i,k] - d[t[i,k]] + d[t[i,1]]) # adjustment for multi-arm RCTs

+ sw[i,k] <- sum(w[i,1:(k-1)])/(k-1) # cumulative adjustment for multi-arm trials

+ }

+ }

+ totresdev <- sum(resdev[]) #Total Residual Deviance

+ d[1] <- 0 # treatment effect is zero for reference treatment

+ sd ~ dunif(0,5) # vague prior for between-trial SD.

+ tau <- pow(sd,-2) # between-trial precision = (1/between-trial variance)

+

+ # Provide estimates of treatment effects T[k] on the natural (probability) scale

+ # Given a Mean Effect, meanA, for 'standard' treatment 1, with precision (1/variance) precA

+

+ for (k in 2:nt){ d[k] ~ dnorm(0,0.0001) }

+ for (k in 1:nt) { logit(T[k]) <- A + d[k] }

+ A ~ dnorm(-2.2, 3.3)

+ } ",

+ file="Wastewater\_Random.txt")

>

>

>

>

>

> # Part 2: Simulation with rujags ----------------------------------------------------------------

>

>

>

> ##### Set up the JAGS model and settings

> jags.m <- run.jags(model="Wastewater\_Random.txt", monitor=c("d[2]", "T[1]", "T[2]", "sd" , "totresdev", "deviance", "pd", "dic" , "full.pd" ),

+ data=list("ns"=ns, "nt"=nt, "na"=na, "r"=r, "n"=n, "t"=t) , n.chains=3, inits=all.inits, burnin = 10000, sample = 20000, adapt = 1500)

Compiling rjags model...

Calling the simulation using the rjags method...

Adapting the model for 1500 iterations...

|++++++++++++++++++++++++++++++++++++++++++++++++++| 100%

Burning in the model for 10000 iterations...

|\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*| 100%

Running the model for 20000 iterations...

|\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*| 100%

Extending 20000 iterations for pD/DIC estimates...

|\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*| 100%

Simulation complete

Calculating summary statistics...

Calculating the Gelman-Rubin statistic for 6 variables....

Finished running the simulation

>

>

>

> #### optional, if not convergated:

> #jags.m <- autorun.jags(model="Wastewater\_Random.txt", monitor=c("d[2]", "T[1]", "T[2]", "sd" , "totresdev", "deviance", "pd", "dic" , "full.pd" ),

> # data=list("ns"=ns, "nt"=nt, "na"=na, "r"=r, "n"=n, "t"=t) , n.chains=3,

> # inits=all.inits, burnin = 10000, sample = 30000, adapt = 1500, , max.time="1hr")

>

>

>

>

>

>

> # Part 3: Output posterior values and calculation median --------------------------

>

>

>

>

> ##### summarize posterior samples

> print(jags.m)

JAGS model summary statistics from 60000 samples (chains = 3; adapt+burnin = 11500):

Lower95 Median Upper95 Mean SD Mode MCerr MC%ofSD SSeff AC.10 psrf

d[2] -3.2913 -1.8077 -0.68932 -1.879 0.66768 -- 0.0061629 0.9 11737 0.06941 1.0001

T[1] 0.026854 0.10036 0.21871 0.1108 0.054553 -- 0.00032533 0.6 28118 0.0059314 1.0002

T[2] 0.0002893 0.017321 0.060675 0.023061 0.021058 -- 0.0001334 0.6 24919 0.011478 1

sd 0.79841 1.7157 3.3115 1.8638 0.70192 -- 0.01043 1.5 4529 0.22357 1.0001

totresdev 12.471 24.55 39.725 25.188 7.0914 -- 0.061877 0.9 13134 0.036688 0.99995

deviance 136.68 148.76 163.93 149.39 7.0914 -- 0.061877 0.9 13134 0.036688 0.99996

Model fit assessment:

DIC = 164.3437 (range between chains: 164.408 - 164.4668)

[PED not available from the stored object]

Estimated effective number of parameters: pD = 15.01929

Total time taken: 33.6 seconds

> summary(jags.m$mcmc) # für 2.5 - 97.5 CrI

Iterations = 11501:31500

Thinning interval = 1

Number of chains = 3

Sample size per chain = 20000

1. Empirical mean and standard deviation for each variable,

plus standard error of the mean:

Mean SD Naive SE Time-series SE

d[2] -1.88636 0.66614 2.720e-03 0.0058620

T[1] 0.11089 0.05502 2.246e-04 0.0002246

T[2] 0.02292 0.02083 8.505e-05 0.0001006

sd 1.86260 0.69983 2.857e-03 0.0100609

totresdev 25.20353 7.09054 2.895e-02 0.0582679

deviance 149.40834 7.09054 2.895e-02 0.0582679

2. Quantiles for each variable:

2.5% 25% 50% 75% 97.5%

d[2] -3.409353 -2.241728 -1.8183 -1.45257 -0.7601

T[1] 0.036494 0.070830 0.1005 0.13923 0.2470

T[2] 0.002676 0.009845 0.0172 0.02921 0.0774

sd 0.945472 1.363336 1.7141 2.19029 3.7192

totresdev 13.251287 20.156307 24.5390 29.54292 40.9141

deviance 137.456094 144.361115 148.7438 153.74773 165.1189

>

>

>

> ##### Generate plots and save to separate file

> png(filename="Plot\_Mo\_B.png")

> plot(jags.m)

Generating plots...

> dev.off()

RStudioGD

2

>

>

>

>

>

> ##### ##### ##### simulation finished ##### ##### #####