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

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

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

> ########## Effect of A-Toxin on MO C

>

>

>

>

> # 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 = 4000, sd = 100)

> #Data\_toxin

> Data\_toxin[1:3] <- Data\_toxin[1:3]/(1\*1.5)

> Data\_toxin[4:6] <- Data\_toxin[4:6]/(10\*1.5)

> Data\_toxin[7:9] <- Data\_toxin[7:9]/(100\*1.5)

> Data\_toxin[10:12] <- Data\_toxin[10:12]/(1000\*1.5)

> #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] 2602 2647 2683 258 267 266 26 27 25 2 2 2

> # -> 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("DatenMOC.txt", sep = "", header=F))

> head(data) # Shows the first six entries

V1 V2 V3 V4 V5 V6 V7

[1,] 3910 2602 5937 5937 1 2 2

[2,] 4018 2647 6018 6018 1 2 2

[3,] 4158 2683 5916 5916 1 2 2

[4,] 388 258 615 615 1 2 2

[5,] 399 267 603 603 1 2 2

[6,] 401 266 591 591 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....

Note: Unable to calculate the multivariate psrf

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] -1.0365 -0.95207 -0.85701 -0.95113 0.044137 -- 0.00057167 1.3 5961 0.15925 1.0001

T[1] 0.026927 0.099785 0.22086 0.11034 0.054403 -- 0.00031113 0.6 30574 -0.0029385 0.99995

T[2] 0.0096764 0.0411 0.097818 0.046671 0.025599 -- 0.00014779 0.6 30000 -0.0020721 0.99995

sd 0.00095193 0.063739 0.16891 0.073776 0.050189 -- 0.0013365 2.7 1410 0.60327 1.0005

totresdev 7.4063 17.355 29.449 17.961 5.8247 -- 0.049268 0.8 13977 0.024806 1.0001

deviance 142.64 152.59 164.68 153.2 5.8247 -- 0.049268 0.8 13977 0.024806 1

Model fit assessment:

DIC = 163.7705 (range between chains: 163.7006 - 163.7901)

[PED not available from the stored object]

Estimated effective number of parameters: pD = 10.54523

Total time taken: 26.4 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] -0.95110 0.04414 0.0001802 0.0005616

T[1] 0.11025 0.05431 0.0002217 0.0002217

T[2] 0.04663 0.02557 0.0001044 0.0001044

sd 0.07372 0.05013 0.0002046 0.0013574

totresdev 17.95085 5.78445 0.0236149 0.0424048

deviance 153.18522 5.78445 0.0236149 0.0424048

2. Quantiles for each variable:

2.5% 25% 50% 75% 97.5%

d[2] -1.03835 -0.97669 -0.95204 -0.92662 -0.8584

T[1] 0.03633 0.07094 0.09976 0.13798 0.2443

T[2] 0.01432 0.02862 0.04113 0.05824 0.1117

sd 0.00705 0.03993 0.06366 0.09553 0.2048

totresdev 8.53760 13.77051 17.34238 21.38056 30.9722

deviance 143.77198 149.00489 152.57676 156.61494 166.2065

>

>

>

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

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

> plot(jags.m)

Generating plots...

> dev.off()

RStudioGD

2

>

>

>

>

>

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