> ########## ########## Simulation Dietary Fat Beispiel mit Fixed Effects mit dem runjags package

> ########## Die Working Directory muss auf Ihre Bedürfnisse angepasst werden

>

>

>

>

>

> # Teil Simulation mit JAGS ------------------------------------------------------------------

>

>

>

> ##### Clear data

> rm(list=ls())

>

>

>

> ##### load libraries

> library(rjags)

> library(runjags)

> library(random)

> library(coda)

> load.module("glm")

> load.module("lecuyer")

> load.module("dic")

>

>

>

> ##### Sichergehen richtiger Working directory

> setwd("C:/Users/IvanB/Desktop/Masterarbeit/Statistische Programme und Gibbs Sampler/Programm JAGS/Nachrechnen TSD2-Dokument/Nachrechnen mit runjags/DietaryFat")

>

>

>

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

> #data = read.table("DietaryFat\_Data.txt", sep = "", header=F)

> data = as.matrix(read.table("DietaryFat\_Data.txt", sep = "", header=T))

> head(data) # Shows the first six entries

t..1. t..2. t..3. E..1. E..2. E..3. r..1. r..2. r..3. na..

[1,] 1 2 NA 1917.0 1925.0 NA 113 111 NA 2

[2,] 1 2 2 43.6 41.3 38 1 5 3 3

[3,] 1 2 NA 393.5 373.9 NA 24 20 NA 2

[4,] 1 2 NA 4715.0 4823.0 NA 248 269 NA 2

[5,] 1 2 NA 715.0 751.0 NA 31 28 NA 2

[6,] 1 2 NA 885.0 895.0 NA 65 48 NA 2

> #data2 = as.data.frread.table("DietaryFat\_Data\_Rest.txt")

> #head(data2) # Shows the first six entries

>

>

>

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

> ns <- nrow(data)

> #ns # check

> nt <- ncol(data[,7:9])

> #nt # check

> na <- data[,10]

> #na # check

> r <- data[,7:9]

> #r # Check

> E <- data[,4:6]

> #E # Check

> t <- data[,1:3]

> #t # Check

>

>

>

> ##### read in inits with chains

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

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

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

>

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

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

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

>

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

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

+ .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] ~ dpois(theta[i,k]) # Poisson likelihood

+ theta[i,k] <- lambda[i,k]\*E[i,k] # failure rate \* exposure

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

+ dev[i,k] <- 2\*((theta[i,k]-r[i,k]) + r[i,k]\*log(r[i,k]/theta[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

+ for (k in 2:nt){ d[k] ~ dnorm(0,.0001) } # vague priors for treatment effects

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

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

+

+ # zusätzlich eingefügt

+ A ~ dnorm(-3,1.77)

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

+ } ",

+ file="DietaryFat\_Fixed.txt")

>

>

>

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

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

+ data=list("ns"=ns, "nt"=nt, "na"=na, "r"=r, "E"=E, "t"=t) , n.chains=3, inits=all.inits, burnin = 20000, sample = 20000, adapt = 5000)

Compiling rjags model...

Calling the simulation using the rjags method...

Adapting the model for 5000 iterations...

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

Burning in the model for 20000 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 5 variables....

Finished running the simulation

>

>

>

> #### optional, falls nicht konvergiert:

> #jags.m <- autorun.jags(model="DietaryFat\_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, "E"=E, "t"=t) , n.chains=3,

> # inits=all.inits, burnin = 100000, sample = 100000, adapt = 2000, , max.time="1hr")

>

>

>

>

>

> # Ausgabe posteriore Werte und Berechnung DIC --------------------------

>

>

>

> print(jags.m)

JAGS model summary statistics from 60000 samples (chains = 3; adapt+burnin = 25000):

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

d[2] -0.18907 -0.013429 0.14955 -0.015043 0.087563 -- 0.0015516 1.8 3185 0.25931 1.0003

T[1] 0.0040837 0.049784 0.1706 0.065936 0.056649 -- 0.00032978 0.6 29509 0.004317 0.99996

T[2] 0.0048784 0.048863 0.17163 0.065257 0.056639 -- 0.00033466 0.6 28643 0.0078097 0.99999

totresdev 11.68 20.847 31.581 21.346 5.1952 -- 0.04418 0.9 13827 0.031965 0.99996

deviance 114.4 123.57 134.3 124.07 5.1952 -- 0.04418 0.9 13827 0.031965 0.99996

Model fit assessment:

DIC = 133.0408 (range between chains: 132.9847 - 133.0495)

[PED not available from the stored object]

Estimated effective number of parameters: pD = 8.93132

Total time taken: 24.1 seconds

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

Iterations = 25001:45000

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] -0.01443 0.08736 0.0003567 0.0014295

T[1] 0.06572 0.05681 0.0002319 0.0002372

T[2] 0.06505 0.05683 0.0002320 0.0002439

totresdev 21.37764 5.20477 0.0212484 0.0423396

deviance 124.09655 5.20477 0.0212484 0.0423396

2. Quantiles for each variable:

2.5% 25% 50% 75% 97.5%

d[2] -0.18994 -0.06431 -0.01295 0.03669 0.1501

T[1] 0.01142 0.02990 0.04960 0.08225 0.2142

T[2] 0.01117 0.02934 0.04884 0.08162 0.2125

totresdev 12.52908 17.71487 20.87708 24.53480 32.9781

deviance 115.24799 120.43378 123.59599 127.25371 135.6970

>

>

>

>

> ########## ########## ########## Simulation beendet ########## ########## ##########