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

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

>

>

>

>

>

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

>

>

>

>

> ##### Clear data

> rm(list=ls())

>

>

>

> ##### load libraries

> #library(rjags) # jagsUI benötigt dieses Paket

> library(lattice)

> #library(coda)

> library(jagsUI)

> #library(random)

>

>

>

> ##### Sichergehen richtiger Working directory

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

>

>

>

> ##### 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

[1] 10

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

> nt # check

[1] 3

> na <- data[,10]

> na # check

[1] 2 3 2 2 2 2 2 2 2 2

> r <- data[,7:9]

> r # Check

r..1. r..2. r..3.

[1,] 113 111 NA

[2,] 1 5 3

[3,] 24 20 NA

[4,] 248 269 NA

[5,] 31 28 NA

[6,] 65 48 NA

[7,] 3 1 NA

[8,] 28 39 NA

[9,] 177 174 NA

[10,] 2 1 NA

> E <- data[,4:6]

> E # Check

E..1. E..2. E..3.

[1,] 1917.0 1925.0 NA

[2,] 43.6 41.3 38

[3,] 393.5 373.9 NA

[4,] 4715.0 4823.0 NA

[5,] 715.0 751.0 NA

[6,] 885.0 895.0 NA

[7,] 87.8 91.0 NA

[8,] 1011.0 939.0 NA

[9,] 1544.0 1588.0 NA

[10,] 125.0 123.0 NA

> t <- data[,1:3]

> t # Check

t..1. t..2. t..3.

[1,] 1 2 NA

[2,] 1 2 2

[3,] 1 2 NA

[4,] 1 2 NA

[5,] 1 2 NA

[6,] 1 2 NA

[7,] 1 2 NA

[8,] 1 2 NA

[9,] 1 2 NA

[10,] 1 2 NA

>

> dat <- list("ns", "nt", "na", "r", "E", "t") # names list of numbers

> dat

[[1]]

[1] "ns"

[[2]]

[1] "nt"

[[3]]

[1] "na"

[[4]]

[1] "r"

[[5]]

[1] "E"

[[6]]

[1] "t"

>

>

>

> ##### Parameter to monitor/save

> params <- c("d[2]", "T[1]", "T[2]" , "dev", "totresdev", "theta" )

> params

[1] "d[2]" "T[1]" "T[2]" "dev" "totresdev" "theta"

>

>

>

> ##### 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="base::Wichmann-Hill", .RNG.seed=1)

>

>

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

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

+ .RNG.name="base::Wichmann-Hill", .RNG.seed=2)

>

>

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

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

+ .RNG.name="base::Wichmann-Hill", .RNG.seed=3 )

> # Achtung: "lecuyer::RngStream" funktioniert nicht mit jagsUI

>

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

>

>

>

> ##### define JAGS model within R

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

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

+ 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] # event rate \* exposure

+ log(lambda[i,k]) <- mu[i] + d[t[i,k]] - d[t[i,1]] # 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

+ }

+ 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

+

+ # 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 <- jags(data=dat,

+ inits=all.inits,

+ parameters.to.save=params,

+ model.file="DietaryFat\_Fixed.txt",

+ n.chains=3,

+ n.iter=21000, n.burnin=9000,

+ store.data=TRUE)

Processing function input.......

Done.

Compiling model graph

Resolving undeclared variables

Allocating nodes

Graph information:

Observed stochastic nodes: 21

Unobserved stochastic nodes: 13

Total graph size: 329

Initializing model

Adaptive phase.....

Adaptive phase complete

Burn-in phase, 9000 iterations x 3 chains

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

Sampling from joint posterior, 12000 iterations x 3 chains

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

Calculating statistics.......

Done.

Warning messages:

1: In process.input(data, parameters.to.save, inits, n.chains, n.iter, :

Suppling a list of character strings to the data argument will be deprecated in the next version

2: In process.input(data, parameters.to.save, inits, n.chains, n.iter, :

Suppling a character vector to the data argument will be deprecated in the next version

>

> #Warning messages:

> #1: In process.input(data, parameters.to.save, inits, n.chains, n.iter, :

> #Suppling a list of character strings to the data argument will be deprecated in the next version

> #2: In process.input(data, parameters.to.save, inits, n.chains, n.iter, :

> #Suppling a character vector to the data argument will be deprecated in the next version

> # => muss demnächst angepasst werden

>

>

>

> # Anzeigen der posterioren Werte und des Medians --------------------------

>

>

>

> #traceplot(jags.m) # zeigt Abbildungen einzeln nach Eingabe der Entertaste an

> jags.View(jags.m)

> jags.m

JAGS output for model 'DietaryFat\_Fixed.txt', generated by jagsUI.

Estimates based on 3 chains of 21000 iterations,

adaptation = 100 iterations (sufficient),

burn-in = 9000 iterations and thin rate = 1,

yielding 36000 total samples from the joint posterior.

MCMC ran for 0.1 minutes at time 2019-08-21 16:40:01.

mean sd 2.5% 50% 97.5% overlap0 f Rhat n.eff

d[2] -0.007 0.054 -0.111 -0.007 0.098 TRUE 0.55 1.000 36000

T[1] 0.067 0.059 0.012 0.050 0.225 FALSE 1.00 1.000 36000

T[2] 0.066 0.059 0.012 0.050 0.223 FALSE 1.00 1.000 36000

dev[1,1] 0.600 0.846 0.001 0.277 3.043 FALSE 1.00 1.000 36000

dev[2,1] 2.185 1.489 0.150 1.910 5.848 FALSE 1.00 1.000 16899

dev[3,1] 0.641 0.892 0.001 0.294 3.219 FALSE 1.00 1.000 18927

dev[4,1] 0.942 1.277 0.001 0.455 4.488 FALSE 1.00 1.001 36000

dev[5,1] 0.694 0.943 0.001 0.323 3.399 FALSE 1.00 1.000 36000

dev[6,1] 1.868 1.899 0.005 1.290 6.891 FALSE 1.00 1.000 36000

dev[7,1] 1.254 1.501 0.002 0.712 5.485 FALSE 1.00 1.000 36000

dev[8,1] 1.885 1.805 0.009 1.393 6.605 FALSE 1.00 1.000 19030

dev[9,1] 0.693 0.967 0.001 0.317 3.474 FALSE 1.00 1.000 36000

dev[10,1] 0.838 1.139 0.001 0.397 4.052 FALSE 1.00 1.000 36000

dev[1,2] 0.586 0.826 0.001 0.271 2.895 FALSE 1.00 1.000 36000

dev[2,2] 1.677 1.579 0.009 1.253 5.765 FALSE 1.00 1.000 36000

dev[3,2] 0.560 0.780 0.001 0.261 2.788 FALSE 1.00 1.000 22447

dev[4,2] 0.986 1.328 0.001 0.477 4.712 FALSE 1.00 1.000 30441

dev[5,2] 0.635 0.875 0.001 0.299 3.073 FALSE 1.00 1.002 5094

dev[6,2] 1.868 1.816 0.006 1.348 6.667 FALSE 1.00 1.000 12609

dev[7,2] 0.907 1.111 0.001 0.496 3.965 FALSE 1.00 1.000 36000

dev[8,2] 1.965 1.918 0.007 1.426 6.999 FALSE 1.00 1.000 36000

dev[9,2] 0.680 0.950 0.001 0.313 3.368 FALSE 1.00 1.001 10050

dev[10,2] 0.543 0.763 0.001 0.249 2.697 FALSE 1.00 1.000 36000

dev[2,3] 0.365 0.524 0.000 0.164 1.862 FALSE 1.00 1.000 36000

totresdev 22.371 4.739 15.114 21.721 33.298 FALSE 1.00 1.000 18042

theta[1,1] 112.172 8.130 96.856 112.002 128.825 FALSE 1.00 1.000 36000

theta[2,1] 3.196 1.077 1.437 3.080 5.657 FALSE 1.00 1.000 17955

theta[3,1] 22.655 3.462 16.361 22.500 29.933 FALSE 1.00 1.000 36000

theta[4,1] 256.437 13.234 231.071 256.402 282.715 FALSE 1.00 1.000 36000

theta[5,1] 28.868 3.804 21.862 28.740 36.804 FALSE 1.00 1.000 36000

theta[6,1] 56.445 5.566 46.068 56.284 67.802 FALSE 1.00 1.000 26711

theta[7,1] 1.972 0.999 0.527 1.797 4.365 FALSE 1.00 1.000 36000

theta[8,1] 34.876 4.315 26.840 34.709 43.884 FALSE 1.00 1.000 26961

theta[9,1] 173.636 10.343 154.072 173.401 194.713 FALSE 1.00 1.000 36000

theta[10,1] 1.533 0.887 0.316 1.369 3.714 FALSE 1.00 1.000 36000

theta[1,2] 111.848 8.083 96.768 111.612 128.255 FALSE 1.00 1.000 36000

theta[2,2] 3.006 1.011 1.355 2.897 5.291 FALSE 1.00 1.000 23903

theta[3,2] 21.376 3.269 15.476 21.217 28.239 FALSE 1.00 1.000 36000

theta[4,2] 260.467 13.374 235.091 260.172 287.180 FALSE 1.00 1.000 36000

theta[5,2] 30.107 3.956 22.902 29.914 38.272 FALSE 1.00 1.000 36000

theta[6,2] 56.681 5.569 46.161 56.495 68.178 FALSE 1.00 1.000 15126

theta[7,2] 2.029 1.026 0.543 1.849 4.483 FALSE 1.00 1.000 36000

theta[8,2] 32.168 4.006 24.724 32.018 40.466 FALSE 1.00 1.000 36000

theta[9,2] 177.331 10.555 157.089 177.107 198.507 FALSE 1.00 1.000 36000

theta[10,2] 1.498 0.868 0.309 1.336 3.609 FALSE 1.00 1.000 36000

theta[2,3] 2.766 0.930 1.247 2.665 4.868 FALSE 1.00 1.000 23903

deviance 125.090 4.739 117.833 124.440 136.017 FALSE 1.00 1.000 18042

Successful convergence based on Rhat values (all < 1.1).

Rhat is the potential scale reduction factor (at convergence, Rhat=1).

For each parameter, n.eff is a crude measure of effective sample size.

overlap0 checks if 0 falls in the parameter's 95% credible interval.

f is the proportion of the posterior with the same sign as the mean;

i.e., our confidence that the parameter is positive or negative.

DIC info: (pD = var(deviance)/2)

pD = 11.2 and DIC = 136.318

DIC is an estimate of expected predictive error (lower is better).

> # mit jags-Funktion keine weiterführende Diagnostik möglich: das Objekt jags.m zeigt nur eine Liste von 24 Elementen an.

> # ebenso wenig mit jags.basic

>

>

>

> # Versuch mit jagsbasic

> jagsbasic.m <- jags.basic(data=dat,

+ inits=all.inits,

+ parameters.to.save=params,

+ model.file="DietaryFat\_Fixed.txt",

+ n.chains=3,

+ n.iter=20000, n.burnin=10000)

Processing function input.......

Done.

Compiling model graph

Resolving undeclared variables

Allocating nodes

Graph information:

Observed stochastic nodes: 21

Unobserved stochastic nodes: 13

Total graph size: 329

Initializing model

Adaptive phase.....

Adaptive phase complete

Burn-in phase, 10000 iterations x 3 chains

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

Sampling from joint posterior, 10000 iterations x 3 chains

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

MCMC took 0.086 minutes.

Warning messages:

1: In process.input(data, parameters.to.save, inits, n.chains, n.iter, :

Suppling a list of character strings to the data argument will be deprecated in the next version

2: In process.input(data, parameters.to.save, inits, n.chains, n.iter, :

Suppling a character vector to the data argument will be deprecated in the next version

> #ja, funktioniert

> #jagsbasic.m

> jagsbasic.m\_II <- do.call(rbind.data.frame, jagsbasic.m)

>

>

>

> #### Berechnung Median

> # Median von T1

> median(jagsbasic.m\_II$`T[1]`)

[1] 0.05005423

>

> # Median von T2

> median(jagsbasic.m\_II$`T[2]`)

[1] 0.04952643

>

> # Median von d

> median(jagsbasic.m\_II$`d[2]`)

[1] -0.0067173

>

>

>

>

>

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