> ########## ########## Simulation Blocker Beispiel mit Random 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)

Attache Paket: ‘jagsUI’

The following object is masked from ‘package:utils’:

View

> #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/Ex1 Blocker")

>

>

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

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

> data = as.matrix(read.table("Blocker\_Data\_neu sortiert.txt", sep = "", header=F))

> head(data) # Shows the first six entries

V1 V2 V3 V4 V5 V6 V7

[1,] 3 3 39 38 1 2 2

[2,] 14 7 116 114 1 2 2

[3,] 11 5 93 69 1 2 2

[4,] 127 102 1520 1533 1 2 2

[5,] 27 28 365 355 1 2 2

[6,] 6 4 52 59 1 2 2

> data2 = read.table("Data\_Blocker\_Rest.txt")

> head(data2) # Shows the first six entries

V1 V2 V3

1 nt <- 2

2 ns <- 22

>

>

>

> ##### 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

>

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

>

>

>

> ##### Parameter to monitor/save

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

>

>

> ##### 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, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),

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

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

>

>

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

+ sd=2,

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

+ .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

+ 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="Blocker\_Random.txt")

>

>

>

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

>

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

+ inits=all.inits,

+ parameters.to.save=params,

+ model.file="Blocker\_Random.txt",

+ n.chains=3,

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

+ store.data=TRUE)

Processing function input.......

Done.

Compiling model graph

Resolving undeclared variables

Allocating nodes

Graph information:

Observed stochastic nodes: 44

Unobserved stochastic nodes: 47

Total graph size: 964

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%

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 'Blocker\_Random.txt', generated by jagsUI.

Estimates based on 3 chains of 20000 iterations,

adaptation = 100 iterations (sufficient),

burn-in = 10000 iterations and thin rate = 1,

yielding 30000 total samples from the joint posterior.

MCMC ran for 0.506 minutes at time 2019-11-04 19:05:02.

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

d[2] -0.246 0.065 -0.371 -0.249 -0.116 FALSE 1 1.001 30000

T[1] 0.110 0.054 0.037 0.099 0.245 FALSE 1 1.000 21802

T[2] 0.089 0.046 0.029 0.079 0.204 FALSE 1 1.000 22715

sd 0.133 0.082 0.009 0.126 0.315 FALSE 1 1.009 233

totresdev 41.926 8.128 27.147 41.563 59.414 FALSE 1 1.001 2719

deviance 259.357 8.128 244.578 258.994 276.845 FALSE 1 1.001 2719

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 = 33 and DIC = 292.368

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="Blocker\_Random.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: 44

Unobserved stochastic nodes: 47

Total graph size: 964

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

>

>

> #### Berechnung Median

> # Median von T1

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

[1] 0.09942919

>

> # Median von T2

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

[1] 0.07938813

>

> # Median von d

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

[1] -0.248589

>

> # Median von sd

> median(jagsbasic.m\_II$`sd`)

[1] 0.1259102

>

>

>

>

> # Leverage Plot -----------------------------------------------------------

>

> ## Erzeugen Daten für dev und rhat

> params\_levplot <- c("dev", "rhat")

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

+ inits=all.inits,

+ parameters.to.save=params\_levplot,

+ model.file="Blocker\_Random.txt",

+ n.chains=3,

+ n.iter=10000, n.burnin=5000)

Processing function input.......

Done.

Compiling model graph

Resolving undeclared variables

Allocating nodes

Graph information:

Observed stochastic nodes: 44

Unobserved stochastic nodes: 47

Total graph size: 964

Initializing model

Adaptive phase.....

Adaptive phase complete

Burn-in phase, 5000 iterations x 3 chains

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

Sampling from joint posterior, 5000 iterations x 3 chains

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

MCMC took 0.303 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

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

>

>

> ## Berechnung w\_ik

> dev\_1.1 <- mean(jagsbasic.m\_levplot$`dev[1,1]`)

> dev\_2.1 <- mean(jagsbasic.m\_levplot$`dev[2,1]`)

> dev\_3.1 <- mean(jagsbasic.m\_levplot$`dev[3,1]`)

> dev\_4.1 <- mean(jagsbasic.m\_levplot$`dev[4,1]`)

> dev\_5.1 <- mean(jagsbasic.m\_levplot$`dev[5,1]`)

> dev\_6.1 <- mean(jagsbasic.m\_levplot$`dev[6,1]`)

> dev\_7.1 <- mean(jagsbasic.m\_levplot$`dev[7,1]`)

> dev\_8.1 <- mean(jagsbasic.m\_levplot$`dev[8,1]`)

> dev\_9.1 <- mean(jagsbasic.m\_levplot$`dev[9,1]`)

> dev\_10.1 <- mean(jagsbasic.m\_levplot$`dev[10,1]`)

> dev\_11.1 <- mean(jagsbasic.m\_levplot$`dev[11,1]`)

> dev\_12.1 <- mean(jagsbasic.m\_levplot$`dev[12,1]`)

> dev\_13.1 <- mean(jagsbasic.m\_levplot$`dev[13,1]`)

> dev\_14.1 <- mean(jagsbasic.m\_levplot$`dev[14,1]`)

> dev\_15.1 <- mean(jagsbasic.m\_levplot$`dev[15,1]`)

> dev\_16.1 <- mean(jagsbasic.m\_levplot$`dev[16,1]`)

> dev\_17.1 <- mean(jagsbasic.m\_levplot$`dev[17,1]`)

> dev\_18.1 <- mean(jagsbasic.m\_levplot$`dev[18,1]`)

> dev\_19.1 <- mean(jagsbasic.m\_levplot$`dev[19,1]`)

> dev\_20.1 <- mean(jagsbasic.m\_levplot$`dev[20,1]`)

> dev\_21.1 <- mean(jagsbasic.m\_levplot$`dev[21,1]`)

> dev\_22.1 <- mean(jagsbasic.m\_levplot$`dev[22,1]`)

> dev\_1.2 <- mean(jagsbasic.m\_levplot$`dev[1,2]`)

> dev\_2.2 <- mean(jagsbasic.m\_levplot$`dev[2,2]`)

> dev\_3.2 <- mean(jagsbasic.m\_levplot$`dev[3,2]`)

> dev\_4.2 <- mean(jagsbasic.m\_levplot$`dev[4,2]`)

> dev\_5.2 <- mean(jagsbasic.m\_levplot$`dev[5,2]`)

> dev\_6.2 <- mean(jagsbasic.m\_levplot$`dev[6,2]`)

> dev\_7.2 <- mean(jagsbasic.m\_levplot$`dev[7,2]`)

> dev\_8.2 <- mean(jagsbasic.m\_levplot$`dev[8,2]`)

> dev\_9.2 <- mean(jagsbasic.m\_levplot$`dev[9,2]`)

> dev\_10.2 <- mean(jagsbasic.m\_levplot$`dev[10,2]`)

> dev\_11.2 <- mean(jagsbasic.m\_levplot$`dev[11,2]`)

> dev\_12.2 <- mean(jagsbasic.m\_levplot$`dev[12,2]`)

> dev\_13.2 <- mean(jagsbasic.m\_levplot$`dev[13,2]`)

> dev\_14.2 <- mean(jagsbasic.m\_levplot$`dev[14,2]`)

> dev\_15.2 <- mean(jagsbasic.m\_levplot$`dev[15,2]`)

> dev\_16.2 <- mean(jagsbasic.m\_levplot$`dev[16,2]`)

> dev\_17.2 <- mean(jagsbasic.m\_levplot$`dev[17,2]`)

> dev\_18.2 <- mean(jagsbasic.m\_levplot$`dev[18,2]`)

> dev\_19.2 <- mean(jagsbasic.m\_levplot$`dev[19,2]`)

> dev\_20.2 <- mean(jagsbasic.m\_levplot$`dev[20,2]`)

> dev\_21.2 <- mean(jagsbasic.m\_levplot$`dev[21,2]`)

> dev\_22.2 <- mean(jagsbasic.m\_levplot$`dev[22,2]`)

> dev\_1 <- mean(c(dev\_1.1,dev\_1.2 ))

> dev\_2 <- mean(c(dev\_2.1,dev\_2.2 ))

> dev\_3 <- mean(c(dev\_3.1,dev\_3.2 ))

> dev\_4 <- mean(c(dev\_4.1,dev\_4.2 ))

> dev\_5 <- mean(c(dev\_5.1,dev\_5.2 ))

> dev\_6 <- mean(c(dev\_6.1,dev\_6.2 ))

> dev\_7 <- mean(c(dev\_7.1,dev\_7.2 ))

> dev\_8 <- mean(c(dev\_8.1,dev\_8.2 ))

> dev\_9 <- mean(c(dev\_9.1,dev\_9.2 ))

> dev\_10 <- mean(c(dev\_10.1,dev\_10.2 ))

> dev\_11 <- mean(c(dev\_11.1,dev\_11.2 ))

> dev\_12 <- mean(c(dev\_12.1,dev\_12.2 ))

> dev\_13 <- mean(c(dev\_13.1,dev\_13.2 ))

> dev\_14 <- mean(c(dev\_14.1,dev\_14.2 ))

> dev\_15 <- mean(c(dev\_15.1,dev\_15.2 ))

> dev\_16 <- mean(c(dev\_16.1,dev\_16.2 ))

> dev\_17 <- mean(c(dev\_17.1,dev\_17.2 ))

> dev\_18 <- mean(c(dev\_18.1,dev\_18.2 ))

> dev\_19 <- mean(c(dev\_19.1,dev\_19.2 ))

> dev\_20 <- mean(c(dev\_20.1,dev\_20.2 ))

> dev\_21 <- mean(c(dev\_21.1,dev\_21.2 ))

> dev\_22 <- mean(c(dev\_22.1,dev\_22.2 ))

>

> dev\_hilf <- rbind(dev\_1, dev\_2, dev\_3, dev\_4, dev\_5, dev\_6, dev\_7, dev\_8, dev\_9, dev\_10, dev\_11, dev\_12, dev\_13, dev\_14, dev\_15, dev\_16, dev\_17, dev\_18, dev\_19, dev\_20, dev\_21, dev\_22 )

>

> w\_ik <- sqrt(dev\_hilf[,1])

> w\_ik\_neg <- -w\_ik

>

> fertige\_Daten\_w\_ik <- cbind( w\_ik\_neg, w\_ik)

> #fertige\_Daten\_w\_ik

>

>

>

> #### Berechnung leverage\_ik

> dev\_bar\_k1 <- rbind(dev\_1.1, dev\_2.1 , dev\_3.1, dev\_4.1, dev\_5.1 , dev\_6.1 , dev\_7.1 , dev\_8.1, dev\_9.1 , dev\_10.1 , dev\_11.1, dev\_12.1 , dev\_13.1 , dev\_14.1 , dev\_15.1 , dev\_16.1 , dev\_17.1 , dev\_18.1, dev\_19.1, dev\_20.1, dev\_21.1, dev\_22.1)

> #dev\_bar\_k1

> dev\_bar\_k2 <- rbind(dev\_1.2, dev\_2.2 , dev\_3.2, dev\_4.2, dev\_5.2 , dev\_6.2 , dev\_7.2 , dev\_8.2, dev\_9.2 , dev\_10.2 , dev\_11.2, dev\_12.2 , dev\_13.2 , dev\_14.2 , dev\_15.2 , dev\_16.2 , dev\_17.2 , dev\_18.2, dev\_19.2, dev\_20.2, dev\_21.2, dev\_22.2)

> #dev\_bar\_k2

>

> rhat\_1.1 <- mean(jagsbasic.m\_levplot$`rhat[1,1]`)

> rhat\_2.1 <- mean(jagsbasic.m\_levplot$`rhat[2,1]`)

> rhat\_3.1 <- mean(jagsbasic.m\_levplot$`rhat[3,1]`)

> rhat\_4.1 <- mean(jagsbasic.m\_levplot$`rhat[4,1]`)

> rhat\_5.1 <- mean(jagsbasic.m\_levplot$`rhat[5,1]`)

> rhat\_6.1 <- mean(jagsbasic.m\_levplot$`rhat[6,1]`)

> rhat\_7.1 <- mean(jagsbasic.m\_levplot$`rhat[7,1]`)

> rhat\_8.1 <- mean(jagsbasic.m\_levplot$`rhat[8,1]`)

> rhat\_9.1 <- mean(jagsbasic.m\_levplot$`rhat[9,1]`)

> rhat\_10.1 <- mean(jagsbasic.m\_levplot$`rhat[10,1]`)

> rhat\_11.1 <- mean(jagsbasic.m\_levplot$`rhat[11,1]`)

> rhat\_12.1 <- mean(jagsbasic.m\_levplot$`rhat[12,1]`)

> rhat\_13.1 <- mean(jagsbasic.m\_levplot$`rhat[13,1]`)

> rhat\_14.1 <- mean(jagsbasic.m\_levplot$`rhat[14,1]`)

> rhat\_15.1 <- mean(jagsbasic.m\_levplot$`rhat[15,1]`)

> rhat\_16.1 <- mean(jagsbasic.m\_levplot$`rhat[16,1]`)

> rhat\_17.1 <- mean(jagsbasic.m\_levplot$`rhat[17,1]`)

> rhat\_18.1 <- mean(jagsbasic.m\_levplot$`rhat[18,1]`)

> rhat\_19.1 <- mean(jagsbasic.m\_levplot$`rhat[19,1]`)

> rhat\_20.1 <- mean(jagsbasic.m\_levplot$`rhat[20,1]`)

> rhat\_21.1 <- mean(jagsbasic.m\_levplot$`rhat[21,1]`)

> rhat\_22.1 <- mean(jagsbasic.m\_levplot$`rhat[22,1]`)

> rhat\_1.2 <- mean(jagsbasic.m\_levplot$`rhat[1,2]`)

> rhat\_2.2 <- mean(jagsbasic.m\_levplot$`rhat[2,2]`)

> rhat\_3.2 <- mean(jagsbasic.m\_levplot$`rhat[3,2]`)

> rhat\_4.2 <- mean(jagsbasic.m\_levplot$`rhat[4,2]`)

> rhat\_5.2 <- mean(jagsbasic.m\_levplot$`rhat[5,2]`)

> rhat\_6.2 <- mean(jagsbasic.m\_levplot$`rhat[6,2]`)

> rhat\_7.2 <- mean(jagsbasic.m\_levplot$`rhat[7,2]`)

> rhat\_8.2 <- mean(jagsbasic.m\_levplot$`rhat[8,2]`)

> rhat\_9.2 <- mean(jagsbasic.m\_levplot$`rhat[9,2]`)

> rhat\_10.2 <- mean(jagsbasic.m\_levplot$`rhat[10,2]`)

> rhat\_11.2 <- mean(jagsbasic.m\_levplot$`rhat[11,2]`)

> rhat\_12.2 <- mean(jagsbasic.m\_levplot$`rhat[12,2]`)

> rhat\_13.2 <- mean(jagsbasic.m\_levplot$`rhat[13,2]`)

> rhat\_14.2 <- mean(jagsbasic.m\_levplot$`rhat[14,2]`)

> rhat\_15.2 <- mean(jagsbasic.m\_levplot$`rhat[15,2]`)

> rhat\_16.2 <- mean(jagsbasic.m\_levplot$`rhat[16,2]`)

> rhat\_17.2 <- mean(jagsbasic.m\_levplot$`rhat[17,2]`)

> rhat\_18.2 <- mean(jagsbasic.m\_levplot$`rhat[18,2]`)

> rhat\_19.2 <- mean(jagsbasic.m\_levplot$`rhat[19,2]`)

> rhat\_20.2 <- mean(jagsbasic.m\_levplot$`rhat[20,2]`)

> rhat\_21.2 <- mean(jagsbasic.m\_levplot$`rhat[21,2]`)

> rhat\_22.2 <- mean(jagsbasic.m\_levplot$`rhat[22,2]`)

>

> rhat\_k1 <- rbind(rhat\_1.1, rhat\_2.1, rhat\_3.1, rhat\_4.1, rhat\_5.1, rhat\_6.1, rhat\_7.1, rhat\_8.1, rhat\_9.1, rhat\_10.1, rhat\_11.1, rhat\_12.1, rhat\_13.1, rhat\_14.1 , rhat\_15.1 , rhat\_16.1, rhat\_17.1, rhat\_18.1, rhat\_19.1, rhat\_20.1 , rhat\_21.1 , rhat\_22.1)

> #rhat\_k1

>

> dev\_tilde\_erst\_k1 <- data[,1]\*log(data[,1]/rhat\_k1[,1])

> #dev\_tilde\_erst\_k1

> dev\_tilde\_zweit\_k1 <- (data[,3]-data[,1])\*log((data[,3]-data[,1])/(data[,3]-rhat\_k1))

> #dev\_tilde\_zweit\_k1

> dev\_tilde\_gesamt\_k1 <- 2\*(dev\_tilde\_erst\_k1+dev\_tilde\_zweit\_k1)

>

> leverage\_k1 <- dev\_bar\_k1[,1] - dev\_tilde\_gesamt\_k1

>

> #leverage\_k1

>

>

> rhat\_k2 <- rbind(rhat\_1.2, rhat\_2.2, rhat\_3.2, rhat\_4.2, rhat\_5.2, rhat\_6.2, rhat\_7.2, rhat\_8.2, rhat\_9.2, rhat\_10.2, rhat\_11.2, rhat\_12.2, rhat\_13.2, rhat\_14.2 , rhat\_15.2 , rhat\_16.2, rhat\_17.2, rhat\_18.2, rhat\_19.2, rhat\_20.2 , rhat\_21.2 , rhat\_22.2)

> #rhat\_k2

>

> dev\_tilde\_erst\_k2 <- data[,2]\*log(data[,2]/rhat\_k2[,1])

> #dev\_tilde\_erst\_k2

> dev\_tilde\_zweit\_k2 <- (data[,4]-data[,2])\*log((data[,4]-data[,2])/(data[,4]-rhat\_k2[,1]))

> #dev\_tilde\_zweit\_k2

> dev\_tilde\_gesamt\_k2 <- 2\*(dev\_tilde\_erst\_k2+dev\_tilde\_zweit\_k2)

>

> leverage\_k2 <- dev\_bar\_k2[,1] - dev\_tilde\_gesamt\_k2

>

> #leverage\_k2

>

>

>

>

> #### Erzeugen leverage plot

> library(car)

Lade nötiges Paket: carData

> scatterplot(c(fertige\_Daten\_w\_ik[,"w\_ik\_neg"], fertige\_Daten\_w\_ik[,"w\_ik"]), c(leverage\_k1, leverage\_k2), main="Leverage plot for the random effects model", xlim=c(-3,3), ylim=c(0,4.5), xlab=expression('w'[ik]), ylab=expression('leverage'[ik]), regLine =F, smooth=F, boxplots=F )

> curve(-x^2+1, from=-3, to=3, col="red", lty="solid", add=T)

> curve(-x^2+2, from=-3, to=3, col="green", lty="dashed", add=T)

> curve(-x^2+3, from=-3, to=3, col="blueviolet", lty="dotted", add=T)

> curve(-x^2+4, from=-3, to=3, col="blue", lty="dotdash", add=T)

>

>

>

>

>

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