> ########## ########## Simulation Dietary Fat Beispiel mit Random Effects mit STAN ########## ##########

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

>

>

>

>

> ##### Clear data

> rm(list=ls())

>

>

>

> ##### Setting working directory

> setwd("C:/Users/IvanB/Desktop/Masterarbeit/Statistische Programme und Gibbs Sampler/STAN/Nachrechnen TSD2/Ex3 Diabetes")

>

>

>

> ##### Requiering stan

> library("rstan")

> library("rstantools")

> rstan\_options(auto\_write = TRUE)

> #options(mc.cores = parallel::detectCores())

> Sys.setenv(LOCAL\_CPPFLAGS = '-march=native')

>

>

>

> ##### Read in data

> data = read.csv2("Diabetes\_Data\_neu sortiert\_II\_ohneNA.csv", header=TRUE, sep = ";", quote = "\"", dec = ",", fill = TRUE, comment.char = "")

> head(data) # Shows the first six entries

time Treatment\_t Erfolge\_r Gesamtanzahl\_n Studienarm Studie Studie\_test X.Study

1 5.8 1 43 1081 3 1 1 #MRC-E 38

2 4.7 1 29 416 2 2 2 #EWPH 32

3 3.0 1 140 1631 2 3 3 #SHEP 42

4 3.8 1 75 3272 2 4 4 #HAPPHY 33

5 4.0 1 302 6766 3 5 5 #ALLHAT 26

6 3.0 1 176 2511 2 6 6 #INSIGHT 35

> #data2 = read.table("Diabetes\_Data\_Rest.txt")

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

>

>

>

> ##### Assignment data to stan

> NO =nrow(data)

> #NO =48

> NT=max(data$Treatment\_t )

> NS=max(data$Studie)

> time=data$time

> r=data$Erfolge\_r

> t=data$Treatment\_t

> s=data$Studie

> base=data$Treatment\_t

> n=data$Gesamtanzahl\_n

>

> data\_list <- list(NO=NO, NT=NT, NS=NS, time=time, r=r, t=t, s=s,base=base, n=n)

>

>

>

> ##### Read in inits

> # Anmerkung: da cloglog als Link: wurden die Inits von STAN generiert.

> # Die manuellen Inits sind zwar drin, können aber rausgelassen werden

>

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

+ 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),

+

+

+ delta= c(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ))

> # die anderen Inits müssen nach der Art von Inits1 angepasst werden

>

> # delta= c(NA, 0, 0, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, 0, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA,

> # 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0,0, NA, 0, 0, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA,

> # NA, 0, NA)) # structure(.Data= , .Dim=c(22, 3))

> # mu=c(0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0, 0,0,0,0,0, 0,0),

> # A=c(0, 0, 0, 0, 0, 0) ,

>

> inits2 <- list(d=c(NA,-1,4,-1,2,3),

+ sd=3,

+ mu=c(1,1,0,1,0, 0,1,0,0,0, 1,1,0,0,0, 0,1,0,0,0, 1,1),

+ A=1 ,

+ delta= c(NA, 0, 0, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, 0, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA,

+ 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0,0, NA, 0, 0, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA,

+ NA, 0, NA)) # structure(.Data= , .Dim=c(22, 3))

>

> inits3 <- list(d=c(NA,1,4,-3,-2,3),

+

+ sd=4.5,

+ mu=c(1,1,0,1,0, 0,1,0,0,0, 1,1,0,-2,0, 0,1,0,-2,0, 1,1),

+ A=2 ,

+

+ delta= c(NA, 0, 0, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, 0, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA,

+ 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0,0, NA, 0, 0, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA, NA, 0, NA,

+ NA, 0, NA)) # structure(.Data= , .Dim=c(22, 3))

>

> all.inits <- list(inits1, inits2, inits3)

> #function(chain\_id = 1) {

>

>

>

> ##### Compiling

> m <- stan\_model('Model\_Random\_final.stan')

hash mismatch so recompiling; make sure Stan code ends with a blank line

> m <- stan\_model('Model\_Random\_final.stan')

>

>

>

> stan\_samples <- sampling(m, data = data\_list, iter=20000, verbose=T, chain=3, init\_r=0.1)

CHECKING DATA AND PREPROCESSING FOR MODEL 'Model\_Random\_final' NOW.

COMPILING MODEL 'Model\_Random\_final' NOW.

STARTING SAMPLER FOR MODEL 'Model\_Random\_final' NOW.

SAMPLING FOR MODEL 'Model\_Random\_final' NOW (CHAIN 1).

Chain 1:

Chain 1: Gradient evaluation took 0 seconds

Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.

Chain 1: Adjust your expectations accordingly!

Chain 1:

Chain 1:

Chain 1: Iteration: 1 / 20000 [ 0%] (Warmup)

Chain 1: Iteration: 2000 / 20000 [ 10%] (Warmup)

Chain 1: Iteration: 4000 / 20000 [ 20%] (Warmup)

Chain 1: Iteration: 6000 / 20000 [ 30%] (Warmup)

Chain 1: Iteration: 8000 / 20000 [ 40%] (Warmup)

Chain 1: Iteration: 10000 / 20000 [ 50%] (Warmup)

Chain 1: Iteration: 10001 / 20000 [ 50%] (Sampling)

Chain 1: Iteration: 12000 / 20000 [ 60%] (Sampling)

Chain 1: Iteration: 14000 / 20000 [ 70%] (Sampling)

Chain 1: Iteration: 16000 / 20000 [ 80%] (Sampling)

Chain 1: Iteration: 18000 / 20000 [ 90%] (Sampling)

Chain 1: Iteration: 20000 / 20000 [100%] (Sampling)

Chain 1:

Chain 1: Elapsed Time: 130.755 seconds (Warm-up)

Chain 1: 140.885 seconds (Sampling)

Chain 1: 271.64 seconds (Total)

Chain 1:

SAMPLING FOR MODEL 'Model\_Random\_final' NOW (CHAIN 2).

Chain 2:

Chain 2: Gradient evaluation took 0 seconds

Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.

Chain 2: Adjust your expectations accordingly!

Chain 2:

Chain 2:

Chain 2: Iteration: 1 / 20000 [ 0%] (Warmup)

Chain 2: Iteration: 2000 / 20000 [ 10%] (Warmup)

Chain 2: Iteration: 4000 / 20000 [ 20%] (Warmup)

Chain 2: Iteration: 6000 / 20000 [ 30%] (Warmup)

Chain 2: Iteration: 8000 / 20000 [ 40%] (Warmup)

Chain 2: Iteration: 10000 / 20000 [ 50%] (Warmup)

Chain 2: Iteration: 10001 / 20000 [ 50%] (Sampling)

Chain 2: Iteration: 12000 / 20000 [ 60%] (Sampling)

Chain 2: Iteration: 14000 / 20000 [ 70%] (Sampling)

Chain 2: Iteration: 16000 / 20000 [ 80%] (Sampling)

Chain 2: Iteration: 18000 / 20000 [ 90%] (Sampling)

Chain 2: Iteration: 20000 / 20000 [100%] (Sampling)

Chain 2:

Chain 2: Elapsed Time: 113.2 seconds (Warm-up)

Chain 2: 150.181 seconds (Sampling)

Chain 2: 263.381 seconds (Total)

Chain 2:

SAMPLING FOR MODEL 'Model\_Random\_final' NOW (CHAIN 3).

Chain 3: Rejecting initial value:

Chain 3: Log probability evaluates to log(0), i.e. negative infinity.

Chain 3: Stan can't start sampling from this initial value.

Chain 3:

Chain 3: Gradient evaluation took 0 seconds

Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.

Chain 3: Adjust your expectations accordingly!

Chain 3:

Chain 3:

Chain 3: Iteration: 1 / 20000 [ 0%] (Warmup)

Chain 3: Iteration: 2000 / 20000 [ 10%] (Warmup)

Chain 3: Iteration: 4000 / 20000 [ 20%] (Warmup)

Chain 3: Iteration: 6000 / 20000 [ 30%] (Warmup)

Chain 3: Iteration: 8000 / 20000 [ 40%] (Warmup)

Chain 3: Iteration: 10000 / 20000 [ 50%] (Warmup)

Chain 3: Iteration: 10001 / 20000 [ 50%] (Sampling)

Chain 3: Iteration: 12000 / 20000 [ 60%] (Sampling)

Chain 3: Iteration: 14000 / 20000 [ 70%] (Sampling)

Chain 3: Iteration: 16000 / 20000 [ 80%] (Sampling)

Chain 3: Iteration: 18000 / 20000 [ 90%] (Sampling)

Chain 3: Iteration: 20000 / 20000 [100%] (Sampling)

Chain 3:

Chain 3: Elapsed Time: 121.107 seconds (Warm-up)

Chain 3: 157.008 seconds (Sampling)

Chain 3: 278.115 seconds (Total)

Chain 3:

Warning messages:

1: There were 29726 divergent transitions after warmup. Increasing adapt\_delta above 0.8 may help. See

http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

2: Examine the pairs() plot to diagnose sampling problems

3: The largest R-hat is 2.09, indicating chains have not mixed.

Running the chains for more iterations may help. See

http://mc-stan.org/misc/warnings.html#r-hat

4: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be unreliable.

Running the chains for more iterations may help. See

http://mc-stan.org/misc/warnings.html#bulk-ess

5: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quantiles may be unreliable.

Running the chains for more iterations may help. See

http://mc-stan.org/misc/warnings.html#tail-ess

> #stan\_samples <- sampling(m, data = data\_list, iter=20000, verbose=T, chain=4, control = list(adapt\_delta = 0.99)) # dauert zu lange

> #stan\_samples <- sampling(m, data = data\_list, i

>

>

>

>

>

> # Ausgabe posteriore Werte, Berechnung Median --------------------------

>

>

>

>

> ##### Plotting and summarizing the posterior distribution

> stan\_samples # = print(stan\_samples)

Inference for Stan model: Model\_Random\_final.

3 chains, each with iter=20000; warmup=10000; thin=1;

post-warmup draws per chain=10000, total post-warmup draws=30000.

mean se\_mean sd 2.5% 25% 50% 75% 97.5% n\_eff Rhat

d[1] -540151.13 822183.23 1364428.76 -4471723.62 -902659.48 -37398.82 58523.03 1441270.51 3 1.99

d[2] -0.02 0.02 0.98 -1.95 -0.68 -0.02 0.65 1.91 3285 1.00

d[3] 0.00 0.02 1.00 -1.95 -0.68 0.00 0.68 1.95 3222 1.00

d[4] 0.02 0.02 0.98 -1.90 -0.65 0.02 0.69 1.91 3409 1.00

d[5] 0.01 0.02 0.99 -1.89 -0.68 0.00 0.68 1.97 3238 1.00

d[6] -0.05 0.02 1.01 -2.04 -0.74 -0.05 0.64 1.95 2950 1.00

mu[1] -4.56 0.00 0.16 -4.87 -4.66 -4.56 -4.45 -4.25 3017 1.00

mu[2] -3.87 0.00 0.18 -4.24 -3.99 -3.87 -3.75 -3.52 2962 1.00

mu[3] -3.40 0.00 0.13 -3.65 -3.48 -3.40 -3.31 -3.16 2150 1.00

mu[4] -4.88 0.00 0.14 -5.17 -4.98 -4.88 -4.79 -4.62 2343 1.00

mu[5] -4.30 0.00 0.11 -4.52 -4.38 -4.30 -4.23 -4.08 1967 1.00

mu[6] -3.61 0.00 0.12 -3.85 -3.69 -3.61 -3.52 -3.37 2097 1.00

mu[7] -3.91 0.00 0.12 -4.15 -3.99 -3.91 -3.83 -3.67 2296 1.00

mu[8] -2.64 0.01 0.28 -3.20 -2.83 -2.64 -2.45 -2.12 2663 1.00

mu[9] -1.57 0.01 0.57 -2.69 -1.94 -1.57 -1.19 -0.44 1536 1.00

mu[10] -0.90 0.01 0.57 -2.03 -1.29 -0.90 -0.51 0.21 1721 1.00

mu[11] -1.48 0.01 0.57 -2.59 -1.86 -1.48 -1.10 -0.35 1950 1.00

mu[12] -1.23 0.01 0.56 -2.31 -1.61 -1.24 -0.86 -0.13 1779 1.00

mu[13] -1.24 0.01 0.58 -2.36 -1.63 -1.25 -0.85 -0.10 1696 1.00

mu[14] -1.39 0.01 0.57 -2.50 -1.77 -1.39 -1.01 -0.24 1937 1.00

mu[15] -0.92 0.01 0.48 -1.87 -1.25 -0.91 -0.59 0.01 1405 1.01

mu[16] -1.37 0.01 0.47 -2.30 -1.69 -1.37 -1.05 -0.44 1778 1.00

mu[17] -1.56 0.01 0.54 -2.61 -1.92 -1.55 -1.19 -0.52 1836 1.00

mu[18] -1.82 0.01 0.55 -2.87 -2.19 -1.81 -1.45 -0.77 1893 1.00

mu[19] -1.55 0.01 0.55 -2.63 -1.93 -1.54 -1.18 -0.47 1704 1.00

mu[20] -1.80 0.01 0.55 -2.89 -2.17 -1.80 -1.44 -0.71 2055 1.00

mu[21] -1.64 0.01 0.55 -2.69 -2.01 -1.64 -1.27 -0.55 1703 1.00

mu[22] -1.35 0.01 0.53 -2.42 -1.71 -1.34 -1.00 -0.32 1617 1.00

mu[23] -5.97 0.02 0.99 -7.92 -6.62 -5.96 -5.31 -4.02 1611 1.00

mu[24] -4.59 0.03 1.01 -6.54 -5.27 -4.60 -3.90 -2.63 1567 1.00

mu[25] -3.66 0.02 1.00 -5.59 -4.34 -3.67 -2.98 -1.71 2009 1.00

mu[26] -4.93 0.02 0.86 -6.65 -5.51 -4.92 -4.35 -3.28 1754 1.00

mu[27] -4.64 0.02 0.82 -6.24 -5.20 -4.63 -4.08 -3.06 1458 1.00

mu[28] -3.98 0.02 0.79 -5.54 -4.50 -3.98 -3.45 -2.41 1941 1.01

mu[29] -4.41 0.02 0.77 -5.93 -4.91 -4.40 -3.89 -2.92 2076 1.00

mu[30] -5.88 0.03 1.51 -9.31 -6.75 -5.73 -4.86 -3.36 2300 1.00

mu[31] -3.49 0.02 0.84 -5.10 -4.06 -3.49 -2.94 -1.82 1769 1.00

mu[32] -2.32 0.02 0.81 -3.89 -2.87 -2.32 -1.77 -0.76 1796 1.00

mu[33] -4.08 0.02 0.79 -5.64 -4.60 -4.07 -3.54 -2.53 1908 1.01

mu[34] -3.23 0.02 0.79 -4.77 -3.74 -3.22 -2.70 -1.69 1765 1.00

mu[35] -3.41 0.02 0.77 -4.92 -3.94 -3.41 -2.88 -1.92 1839 1.00

mu[36] -3.86 0.02 0.78 -5.40 -4.39 -3.86 -3.33 -2.34 1656 1.00

mu[37] -2.44 0.02 0.84 -4.09 -3.00 -2.45 -1.89 -0.76 1666 1.00

mu[38] -3.38 0.02 0.80 -4.98 -3.91 -3.39 -2.85 -1.80 1661 1.00

mu[39] -3.42 0.02 0.85 -5.04 -4.00 -3.42 -2.84 -1.74 1510 1.00

mu[40] -3.77 0.02 0.81 -5.33 -4.32 -3.77 -3.22 -2.17 1730 1.00

mu[41] -3.24 0.02 0.82 -4.81 -3.79 -3.26 -2.70 -1.61 1929 1.00

mu[42] -3.85 0.02 0.79 -5.42 -4.37 -3.86 -3.32 -2.29 1823 1.00

mu[43] -3.84 0.02 0.79 -5.38 -4.37 -3.85 -3.32 -2.29 1549 1.00

mu[44] -3.00 0.02 0.77 -4.48 -3.55 -3.01 -2.45 -1.49 1491 1.00

mu[45] -5.18 0.03 1.01 -7.15 -5.86 -5.18 -4.49 -3.23 1619 1.00

mu[46] -4.93 0.02 0.81 -6.52 -5.47 -4.92 -4.37 -3.37 1651 1.00

mu[47] -2.80 0.02 0.84 -4.44 -3.35 -2.80 -2.24 -1.13 1606 1.00

mu[48] -3.43 0.02 0.83 -5.07 -3.99 -3.41 -2.88 -1.84 1514 1.00

delta[1] -0.14 0.00 0.10 -0.34 -0.20 -0.14 -0.07 0.05 2416 1.00

delta[2] -0.08 0.00 0.10 -0.27 -0.15 -0.08 -0.01 0.12 2686 1.00

delta[3] -0.07 0.00 0.10 -0.25 -0.13 -0.07 0.00 0.13 2082 1.00

delta[4] -0.10 0.00 0.10 -0.28 -0.16 -0.10 -0.03 0.10 2400 1.00

delta[5] -0.13 0.00 0.10 -0.32 -0.20 -0.13 -0.06 0.06 2097 1.00

delta[6] -0.08 0.00 0.10 -0.27 -0.14 -0.08 -0.01 0.12 2212 1.00

delta[7] -0.08 0.00 0.10 -0.27 -0.15 -0.08 -0.01 0.12 2287 1.00

delta[8] -0.05 0.00 0.10 -0.24 -0.12 -0.05 0.01 0.14 3102 1.00

delta[9] -3.05 0.01 0.57 -4.17 -3.43 -3.05 -2.67 -1.92 1533 1.00

delta[10] -1.78 0.01 0.57 -2.89 -2.18 -1.78 -1.40 -0.66 1719 1.00

delta[11] -2.91 0.01 0.57 -4.04 -3.29 -2.91 -2.52 -1.79 1958 1.00

delta[12] -2.44 0.01 0.56 -3.54 -2.81 -2.43 -2.06 -1.35 1774 1.00

delta[13] -2.44 0.01 0.58 -3.58 -2.84 -2.43 -2.05 -1.32 1709 1.00

delta[14] -2.81 0.01 0.57 -3.95 -3.19 -2.81 -2.43 -1.69 1936 1.00

delta[15] -2.05 0.01 0.49 -2.98 -2.38 -2.06 -1.71 -1.09 1408 1.01

delta[16] -2.96 0.01 0.48 -3.90 -3.28 -2.96 -2.64 -2.02 1774 1.00

delta[17] -2.26 0.01 0.54 -3.29 -2.63 -2.27 -1.89 -1.21 1832 1.00

delta[18] -2.65 0.01 0.55 -3.71 -3.03 -2.66 -2.28 -1.59 1888 1.00

delta[19] -2.29 0.01 0.55 -3.36 -2.66 -2.30 -1.91 -1.21 1704 1.00

delta[20] -2.58 0.01 0.55 -3.67 -2.94 -2.58 -2.22 -1.50 2056 1.00

delta[21] -2.40 0.01 0.55 -3.48 -2.76 -2.40 -2.02 -1.35 1704 1.00

delta[22] -1.78 0.01 0.53 -2.82 -2.14 -1.79 -1.43 -0.71 1623 1.00

delta[23] 0.03 0.02 0.97 -1.88 -0.62 0.02 0.67 1.97 1600 1.00

delta[24] -0.01 0.02 0.98 -1.93 -0.68 -0.01 0.65 1.91 1559 1.00

delta[25] 0.00 0.02 0.99 -1.93 -0.67 0.01 0.68 1.93 2012 1.00

delta[26] -0.04 0.02 0.86 -1.68 -0.63 -0.05 0.53 1.65 1759 1.00

delta[27] 0.02 0.02 0.81 -1.54 -0.53 0.01 0.58 1.64 1466 1.00

delta[28] -0.01 0.02 0.79 -1.56 -0.54 -0.01 0.51 1.54 1943 1.01

delta[29] 0.01 0.02 0.76 -1.47 -0.51 0.00 0.51 1.51 2071 1.00

delta[30] 0.01 0.01 0.75 -1.51 -0.50 0.01 0.52 1.46 2556 1.00

delta[31] -0.99 0.02 0.83 -2.66 -1.54 -0.99 -0.42 0.60 1768 1.00

delta[32] -0.46 0.02 0.80 -2.00 -1.00 -0.46 0.10 1.12 1799 1.00

delta[33] -0.74 0.02 0.79 -2.26 -1.27 -0.74 -0.22 0.81 1905 1.01

delta[34] -0.62 0.02 0.78 -2.15 -1.15 -0.63 -0.10 0.92 1773 1.00

delta[35] -0.51 0.02 0.77 -1.99 -1.03 -0.51 0.03 0.99 1843 1.00

delta[36] -0.58 0.02 0.78 -2.08 -1.10 -0.58 -0.05 0.95 1670 1.00

delta[37] -0.67 0.02 0.82 -2.31 -1.21 -0.66 -0.12 0.95 1667 1.00

delta[38] -1.01 0.02 0.80 -2.58 -1.54 -1.00 -0.49 0.57 1674 1.00

delta[39] -0.77 0.02 0.84 -2.45 -1.35 -0.77 -0.19 0.86 1507 1.00

delta[40] -0.88 0.02 0.81 -2.48 -1.42 -0.88 -0.33 0.69 1723 1.00

delta[41] -0.76 0.02 0.81 -2.39 -1.31 -0.75 -0.21 0.80 1927 1.00

delta[42] -0.66 0.02 0.79 -2.22 -1.19 -0.65 -0.15 0.91 1831 1.00

delta[43] -0.51 0.02 0.78 -2.06 -1.03 -0.50 0.02 1.03 1549 1.00

delta[44] -0.36 0.02 0.77 -1.87 -0.91 -0.36 0.19 1.12 1489 1.00

delta[45] 0.03 0.02 1.00 -1.88 -0.65 0.03 0.71 1.98 1624 1.00

delta[46] 0.01 0.02 0.80 -1.54 -0.54 0.01 0.55 1.59 1664 1.00

[ reached getOption("max.print") -- omitted 594 rows ]

Samples were drawn using NUTS(diag\_e) at Thu Aug 29 11:23:42 2019.

For each parameter, n\_eff is a crude measure of effective sample size,

and Rhat is the potential scale reduction factor on split chains (at

convergence, Rhat=1).

> plot(stan\_samples)

'pars' not specified. Showing first 10 parameters by default.

ci\_level: 0.8 (80% intervals)

outer\_level: 0.95 (95% intervals)

> Stan\_summary <- summary(stan\_samples, pars = c("d[2]", "d\_II[2]", "d[3]", "d[4]", "d[5]", "d[6]", "T[1]", "T[2]", "T[3]", "T[4]", "T[5]", "T[6]", "s\_d" , "totalresdev"), probs = c(0.025, 0.975))$summary

> Stan\_summary

mean se\_mean sd 2.5% 97.5% n\_eff Rhat

d[2] -1.720561e-02 0.0170932196 0.97972378 -1.9503474 1.9056558 3285.183 1.001057

d\_II[2] -1.720561e-02 0.0170932196 0.97972378 -1.9503474 1.9056558 3285.183 1.001057

d[3] -9.149453e-05 0.0175475023 0.99609069 -1.9468599 1.9456905 3222.309 1.000843

d[4] 1.775095e-02 0.0167594025 0.97857402 -1.9046449 1.9122431 3409.340 1.000496

d[5] 8.424739e-03 0.0173677236 0.98834636 -1.8884696 1.9719003 3238.415 1.000200

d[6] -5.103455e-02 0.0186528000 1.01313353 -2.0360465 1.9511400 2950.158 1.000168

T[1] -3.111299e+00 0.0179359903 0.95470913 -5.0144700 -1.2422413 2833.293 1.000394

T[2] -3.111585e+00 0.0263786694 1.38181687 -5.7998428 -0.3133396 2744.070 1.001830

T[3] -3.113472e+00 0.0264498627 1.36510292 -5.8253011 -0.4703535 2663.692 1.000701

T[4] -3.100464e+00 0.0235219982 1.36163629 -5.7465859 -0.4347906 3350.995 1.000444

T[5] -3.114720e+00 0.0235754468 1.37138748 -5.8271882 -0.4371272 3383.767 1.000381

T[6] -3.123233e+00 0.0254942241 1.39586198 -5.8002854 -0.3694655 2997.791 1.000356

s\_d 9.790990e-01 0.0004419889 0.01994873 0.9265102 0.9993795 2037.079 1.004480

totalresdev 5.783579e+01 0.1326236335 11.41709245 37.7061580 82.3947547 7410.870 1.000301

>

>

>

> ##### Additional Lines for Median

> Median\_d2 <- median(as.matrix(stan\_samples, pars = c("d\_II[2]")))

> Median\_d2

[1] -0.02050058

> Median\_d3 <- median(as.matrix(stan\_samples, pars = c("d\_II[3]")))

> Median\_d3

[1] 0.002488227

> Median\_d4 <- median(as.matrix(stan\_samples, pars = c("d\_II[4]")))

> Median\_d4

[1] 0.02057897

> Median\_d5 <- median(as.matrix(stan\_samples, pars = c("d\_II[5]")))

> Median\_d5

[1] 0.00049925

> Median\_d6 <- median(as.matrix(stan\_samples, pars = c("d\_II[6]")))

> Median\_d6

[1] -0.05210117

>

> Median\_T1 <- median(as.matrix(stan\_samples, pars = c("T[1]")))

> Median\_T1

[1] -3.110962

> Median\_T2 <- median(as.matrix(stan\_samples, pars = c("T[2]")))

> Median\_T2

[1] -3.143915

> Median\_T3 <- median(as.matrix(stan\_samples, pars = c("T[3]")))

> Median\_T3

[1] -3.117181

> Median\_T4 <- median(as.matrix(stan\_samples, pars = c("T[4]")))

> Median\_T4

[1] -3.118612

> Median\_T5 <- median(as.matrix(stan\_samples, pars = c("T[5]")))

> Median\_T5

[1] -3.10843

> Median\_T6 <- median(as.matrix(stan\_samples, pars = c("T[6]")))

> Median\_T6

[1] -3.122797

>

> Median\_sd <- median(as.matrix(stan\_samples, pars = c("s\_d")))

> Median\_sd

[1] 0.9851291

>

>

>

>

>

> # Section for Convergence Diagnostic --------------------------------------------------

>

>

>

>

> # konkreter Vergleich mit BUGS nicht möglich ->kein pD in dem Sinne

> # DIC (und damit pD) ist veraltet

> # => loo()-Fkt und WAIC

> #(pD Code für Python)

> #allerdings andere Konvergenz - Diagnostika:

> # Diagnostik mir rstan Paket

> sampler\_params <- get\_sampler\_params(stan\_samples, inc\_warmup = TRUE)

> summary(do.call(rbind, sampler\_params), digits = 2)

accept\_stat\_\_ stepsize\_\_ treedepth\_\_ n\_leapfrog\_\_ divergent\_\_ energy\_\_

Min. :0.00 Min. :1.3e-04 Min. : 0.0 Min. : 1 Min. :0.00 Min. : 38213

1st Qu.:0.76 1st Qu.:2.7e-02 1st Qu.: 4.0 1st Qu.: 23 1st Qu.:1.00 1st Qu.: 38243

Median :0.87 Median :2.7e-02 Median : 5.0 Median : 48 Median :1.00 Median : 38250

Mean :0.82 Mean :3.3e-02 Mean : 4.8 Mean : 58 Mean :0.98 Mean : 38321

3rd Qu.:0.94 3rd Qu.:3.3e-02 3rd Qu.: 6.0 3rd Qu.: 83 3rd Qu.:1.00 3rd Qu.: 38257

Max. :1.00 Max. :1.4e+01 Max. :10.0 Max. :1023 Max. :1.00 Max. :529030

> # each chain separately

> lapply(sampler\_params, summary, digits = 2)

[[1]]

accept\_stat\_\_ stepsize\_\_ treedepth\_\_ n\_leapfrog\_\_ divergent\_\_ energy\_\_

Min. :0.00 Min. :1.3e-04 Min. : 0.0 Min. : 1 Min. :0.00 Min. : 38213

1st Qu.:0.76 1st Qu.:2.7e-02 1st Qu.: 4.0 1st Qu.: 22 1st Qu.:1.00 1st Qu.: 38243

Median :0.86 Median :2.7e-02 Median : 5.0 Median : 47 Median :1.00 Median : 38250

Mean :0.82 Mean :3.3e-02 Mean : 4.8 Mean : 58 Mean :0.98 Mean : 38318

3rd Qu.:0.93 3rd Qu.:3.3e-02 3rd Qu.: 6.0 3rd Qu.: 83 3rd Qu.:1.00 3rd Qu.: 38257

Max. :1.00 Max. :1.4e+01 Max. :10.0 Max. :1023 Max. :1.00 Max. :529030

[[2]]

accept\_stat\_\_ stepsize\_\_ treedepth\_\_ n\_leapfrog\_\_ divergent\_\_ energy\_\_

Min. :0.00 Min. :0.00041 Min. :0.0 Min. : 1 Min. :0.00 Min. : 38214

1st Qu.:0.76 1st Qu.:0.02712 1st Qu.:4.0 1st Qu.: 22 1st Qu.:1.00 1st Qu.: 38243

Median :0.87 Median :0.02712 Median :5.0 Median : 46 Median :1.00 Median : 38250

Mean :0.82 Mean :0.03316 Mean :4.8 Mean : 56 Mean :0.99 Mean : 38328

3rd Qu.:0.94 3rd Qu.:0.03262 3rd Qu.:6.0 3rd Qu.: 79 3rd Qu.:1.00 3rd Qu.: 38258

Max. :1.00 Max. :5.79578 Max. :9.0 Max. :626 Max. :1.00 Max. :523907

[[3]]

accept\_stat\_\_ stepsize\_\_ treedepth\_\_ n\_leapfrog\_\_ divergent\_\_ energy\_\_

Min. :0.00 Min. :2.1e-04 Min. :0.0 Min. : 1 Min. :0.00 Min. : 38213

1st Qu.:0.76 1st Qu.:2.7e-02 1st Qu.:4.0 1st Qu.: 23 1st Qu.:1.00 1st Qu.: 38243

Median :0.87 Median :2.7e-02 Median :5.0 Median : 50 Median :1.00 Median : 38250

Mean :0.82 Mean :3.3e-02 Mean :4.9 Mean : 61 Mean :0.98 Mean : 38317

3rd Qu.:0.94 3rd Qu.:3.3e-02 3rd Qu.:6.0 3rd Qu.: 87 3rd Qu.:1.00 3rd Qu.: 38257

Max. :1.00 Max. :1.4e+01 Max. :9.0 Max. :742 Max. :1.00 Max. :515321

> Stan\_summary\_lp\_\_ <- summary(stan\_samples, pars = c("lp\_\_"), probs = c(0.025, 0.975))$summary # sigmasq\_delta entspricht sd, nachher ändern

> Stan\_summary\_lp\_\_

mean se\_mean sd 2.5% 97.5% n\_eff Rhat

lp\_\_ -38195.29 0.1153233 7.347515 -38210.43 -38181.85 4059.262 1.001428

> # weitere Möglichkeit: Package 'shinystan'

>

>

>

>

>

> # manuelle Berechnung von pD ----------------------------------------------

>

>

>

>

> SingeValues\_dev <- summary(stan\_samples, pars = c("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]", "dev[23]", "dev[24]", "dev[25]", "dev[26]", "dev[27]", "dev[28]", "dev[29]", "dev[30]", "dev[31]", "dev[32]", "dev[33]", "dev[34]", "dev[35]", "dev[36]", "dev[37]", "dev[38]", "dev[39]", "dev[40]", "dev[41]", "dev[42]", "dev[43]", "dev[44]", "dev[45]", "dev[46]", "dev[47]", "dev[48]"))$summary

> out\_lePlo <- capture.output( SingeValues\_dev)

> cat("Hilf\_pD", out\_lePlo, file="Hilf\_pD.txt", sep="\n", append=TRUE)

> Hilf\_dev\_pD <- read.table("Hilf\_pD.txt", sep = "", header=F, skip=2, nrows=21)

> #dev ist Std-Abweichung jedes einzelnen Werts

> # insg 48 Werte

> Var\_manuell <- sum(Hilf\_dev\_pD[,2])^2/48

> Var\_manuell

[1] 23330094101

> pD\_manuell <- Var\_manuell/2

> pD\_manuell

[1] 11665047051

>

>

>

>

>

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