Bayesian item response models for citizen science ecological data

Codes to reproduce the results from Section 4.3. Item response modeling for big data 11/05/2020

Contents

Summary: In this document, we illustrate the implementation of the case study from Section 4.3 (Item response model for big data). The first part contains the codes for fitting the model in Stan on a High-Performance Computing system (HPC). We then show how to combine the subposterior estimates via consensus Monte Carlo. R codes for producing several of the figures are also included.


```
# loading the libraries we need
library('Rcpp')
library('ggplot2')
library('plyr')
library('dplyr')
library('reshape2')
library('spdep')
library('rstan')
library('ggmcmc')
library('StanHeaders')
library('dismo')
library('sp')
library('rgeos')
library('MASS')
library('ggvoronoi')
library('RColorBrewer')
library('GGally')
library('mirt')
library('loo')
library('ggrepel')
library('broom')
library('splitstackshape')
library('GGally')
```

```r
library('viridis')
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
options(scipen = 99)
```

We need to download the data from the repository: https://github.com/the-anonymous-author/Item-Response-JASA We then create the subsets (shards) of data:

```r
s5_sets <- readRDS('serengety.RDS')

s5_sets$True_Species_num <- as.numeric(as.factor(s5_sets$True_Species))
# produce the 10 shards
s5_1 <- dplyr::filter(s5_sets, set == 1)
s5_2 <- dplyr::filter(s5_sets, set == 2)
s5_3 <- dplyr::filter(s5_sets, set == 3)
s5_4 <- dplyr::filter(s5_sets, set == 4)
s5_5 <- dplyr::filter(s5_sets, set == 5)
s5_6 <- dplyr::filter(s5_sets, set == 6)
s5_7 <- dplyr::filter(s5_sets, set == 7)
s5_8 <- dplyr::filter(s5_sets, set == 8)
s5_9 <- dplyr::filter(s5_sets, set == 9)
s5_10 <- dplyr::filter(s5_sets, set == 10)
```

1 Fitting the models in Stan

Models were fit to the subsets on independent machines with no communication on an HPC. We used the following PBS .sh file to request resources on the HPC (one node with four processors for 100 hours). Note that the requested memory is 280Gb.

```
#!/bin/bash -l
#PBS -N Serengeti_bigdata_shard1
#PBS -lnodes=1:ppn=4
#PBS -l walltime=99:50:00
#PBS -l mem=280000mb
#PBS -m aeb
module load geos/3.7.1-foss-2018a
module load r/3.5.1-foss-2018a
cd $PBS_O_WORKDIR
Rscript model.R
```

The following chunk of code (saved as model.R) contains the R and Stan codes for fitting the model to the first shard of data. Please, do not fit this model on a single standard computer. The MCMC output file will have approximately 45Gb.

```r
library('Rcpp')
library('rstan')
library('StanHeaders')
library('dismo')
library('sp')
library('spdep')
library('rgeos')
```

```r
library('loo')

rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
options(scipen = 99)

source('functions.R') # these functions will produce the adjacency matrix and edges
serengeti <- readRDS('serengeti.RDS')
s1 <- <- dplyr::filter(serengeti, set == 1)

s1$True_Species_num <- as.numeric(as.factor(s1$True_Species)) # a numeric index for species
s1$id <- as.numeric(as.factor(as.character(s1$SiteID)))# a numeric index for the sites
s1$user <- as.numeric(as.factor(as.character(s1$UserID))) # a numeric index for the users

# spatial part
# adjacency matrix
adjMat <- adj_matrix(s1$LocationX, s1$LocationY)
num <- apply(adjMat,1,sum)
adj <- c(unlist(apply(adjMat,1,function(x) which(x == 1))))
nei_data <- mungeCARdata4stan(adjBUGS = adj, numBUGS = num)

node1 <- nei_data$node1
node2 <- nei_data$node2
N_edges <-  nei_data$N_edges # number of edges

# creating a data list for stan
data_lltm <- list(N = nrow(s1),                                  # total numb of anotations
             M = length(unique(s1$id)),                          # numb sites
                   Nspecies = length(unique(s1$True_Species_num)), # numb species
             Nindiv = length(unique(s1$user)),                   # numb users
             id = s1$id,                                                 # site id
                   species_id = s1$True_Species_num,                        # species id
             annot = s1$user,                                          # users id
             y = s1$correct,                                                  # outcome variable
             N_edges= N_edges,                                               # edges
             node1 = node1,
             node2 = node2
             )


model_3PLUS <- '
data {
  int<lower=1> N;                              // num elicitation points classified
  int<lower=1> M;                              // number of unique locations.
  int<lower=1> Nindiv;                         // number users
  int<lower=1,upper=M> id[N];                  // site id
  int<lower=1,upper=Nindiv> annot[N];          // user id
  int<lower=0,upper=1> y[N];                   // outcome variable
  int<lower=0> N_edges;                        // edges
  int<lower=1, upper=N> node1[N_edges];        // node1 adjacent to node2
  int<lower=1, upper=N> node2[N_edges];        // node2
  int<lower=1> Nspecies;                       // number of species
  int<lower=1,upper=Nspecies> species_id[N];   // species id
```

```
}

parameters {
  vector[Nindiv] abil;                            // ability
  real<lower=0>sigma_abil;                        // sd of the abilities
  vector<lower=0>[M] alpha;                       // slope
  real<lower=0>sigma_alpha;                       // sd of the slope
  vector<lower=0,upper=1>[Nspecies] eta;          // guessing
  real<lower=0> tau_phi;                          // precision
  vector[M] phi;                                  // spatial effect
  vector[Nspecies] diff_species;                  // difficulty based on species
  real<lower=0>sigma_diff_species;                // sd of species difficulties
}

transformed parameters {
  vector[M] difficulty;                           // site difficulty
  real<lower=0> sigma_phi = inv(sqrt(tau_phi)); // sigma for precision
  difficulty = phi * sigma_phi;
}

model {
  vector [N] pijk;                                // prob of correct classification
  abil ~ normal(0, sigma_abil);                   // informative prior for ability
  sigma_abil ~ uniform(0,10);                     // flat prior on the abilities sd
  mean(abil) ~ normal(0,0.001);                   // sum to zero for the abil
  tau_phi ~ gamma(1, 1) ;
  target += -0.5 * dot_self(phi[node1] - phi[node2]);  // prior on phi
  alpha ~ normal(1,sigma_alpha);                  // slope
  sigma_alpha ~  cauchy(0,5);                     // sd for the slope.
  eta ~ beta(1,14);                               // prior on guessing
  diff_species ~ normal(0, sigma_diff_species); // prior for the species difficulties
  sigma_diff_species ~ uniform(0,10);             // flat prior on the diff_species sd

  for (i in 1 : N) {
    pijk[i] = eta[species_id[i]] + (1 - eta[species_id[i]]) * (inv_logit(alpha[id[i]] *
    (abil[annot[i]] - (diff_species[species_id[i]] + difficulty[id[i]] )  ) ) );
    y[i] ~ bernoulli( pijk[i] );
  }
}

generated quantities {                            // to compute the log likelihood
  vector [N] lin;
  vector[N] log_lik;
  for (i in 1 : N) {
    lin[i] = eta[species_id[i]] + (1 - eta[species_id[i]]) *
    (inv_logit(alpha[id[i]] * (abil[annot[i]] -
    (diff_species[species_id[i]] + difficulty[id[i]] )) ) );
    log_lik[i] = bernoulli_log(y[i], lin[i]);
  }
}
'

nwarmup <- 3000 # warm up simulations
```

```
niter <- 8000   # number of simulations after warm up

time_start <- Sys.time()
fit_lltm_3PLUS <- stan(model_code = model_3PLUS,
                 model_name = "model_3PLUS",
                 pars = c('abil', 'difficulty',
                          'alpha', 'eta',
                                      'log_lik', 'diff_species'),
                 data = data_lltm,
                 iter = niter,
                 warmup = nwarmup,
                 thin = 2,
                 chains = 3,
                 verbose = F,
                 seed = 1,
                 refresh = max(niter/100, 1)
)

time_end <- Sys.time()
as.numeric(difftime(time_end, time_start, units = "mins")) # time

# save the chains
saveRDS(fit_lltm_3PLUS, paste0(gsub(":", "", Sys.time()),'fit_serengeti_3PL.rds'))

# summary stats
stats_lltm_3PL <- summary(fit_lltm_3PLUS)
stats_lltm_3PL <- stats_lltm_3PL$summary
saveRDS(stats_lltm_3PL, 'stats_lltm_3PL.rds') # saving the summary stats


recomp <- T
if(recomp == T){
  fit.mcmc.abil <- As.mcmc.list(fit_lltm_3PLUS,pars="abil")
  fit.mcmc.site <- As.mcmc.list(fit_lltm_3PLUS,pars="difficulty")
  fit.mcmc.alpha <- As.mcmc.list(fit_lltm_3PLUS,pars="alpha")
  fit.mcmc.guess <- As.mcmc.list(fit_lltm_3PLUS,pars="eta")
  fit.mcmc.diff.species <- As.mcmc.list(fit_lltm_3PLUS,pars="diff_species")
}

saveRDS(fit.mcmc.abil, 'fit.mcmc.abil.rds')
saveRDS(fit.mcmc.site, 'fit.mcmc.site.rds')
saveRDS(fit.mcmc.alpha, 'fit.mcmc.alpha.rds')
saveRDS(fit.mcmc.guess, 'fit.mcmc.guess.rds')
saveRDS(fit.mcmc.diff.species, 'fit.mcmc.diff.species.rds')


log_lik <- extract_log_lik(fit_lltm_3PLUS, parameter_name = "log_lik")
waic <- waic(log_lik) # computes the Watanabe-Akaike information criterion
saveRDS(waic, 'waic.rds') # saving the waic
```

The file functions.R sourced above can be found below in the Appendix section.

Once the Stan models are fit to each of the shards, we read the MCMC and summary statistics output files. Note that the codes below will not work if the stats (e.g. stats_lltm_3PLUS) and MCMC files

(e.g. fit.mcmc.abil) are not available. Please, change the path below to the location where your model outputs are.

2 Obtaining the consensus posterior distribution for each of the parameters.

Reading the summary statistics from the HPC:

```
stats_seren_1 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA//stats_lltm_3PL.rds')
stats_seren_2 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_2//stats_lltm_3PL.rds')
stats_seren_3 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_3//stats_lltm_3PL.rds')
stats_seren_4 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_4//stats_lltm_3PL.rds')
stats_seren_5 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_5//stats_lltm_3PL.rds')
stats_seren_6 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_6//stats_lltm_3PL.rds')
stats_seren_7 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_7//stats_lltm_3PL.rds')
stats_seren_8 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_8//stats_lltm_3PL.rds')
stats_seren_9 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_9//stats_lltm_3PL.rds')
stats_seren_10 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_10//stats_lltm_3PL.rds')
```

Generating a new numeric user id.

```
s5_1$user <- as.numeric(as.factor(as.character(s5_1$UserID)))
s5_2$user <- as.numeric(as.factor(as.character(s5_2$UserID)))
s5_3$user <- as.numeric(as.factor(as.character(s5_3$UserID)))
s5_4$user <- as.numeric(as.factor(as.character(s5_4$UserID)))
s5_5$user <- as.numeric(as.factor(as.character(s5_5$UserID)))
s5_6$user <- as.numeric(as.factor(as.character(s5_6$UserID)))
s5_7$user <- as.numeric(as.factor(as.character(s5_7$UserID)))
s5_8$user <- as.numeric(as.factor(as.character(s5_8$UserID)))
s5_9$user <- as.numeric(as.factor(as.character(s5_9$UserID)))
s5_10$user <- as.numeric(as.factor(as.character(s5_10$UserID)))
```

2.1 User's abilities

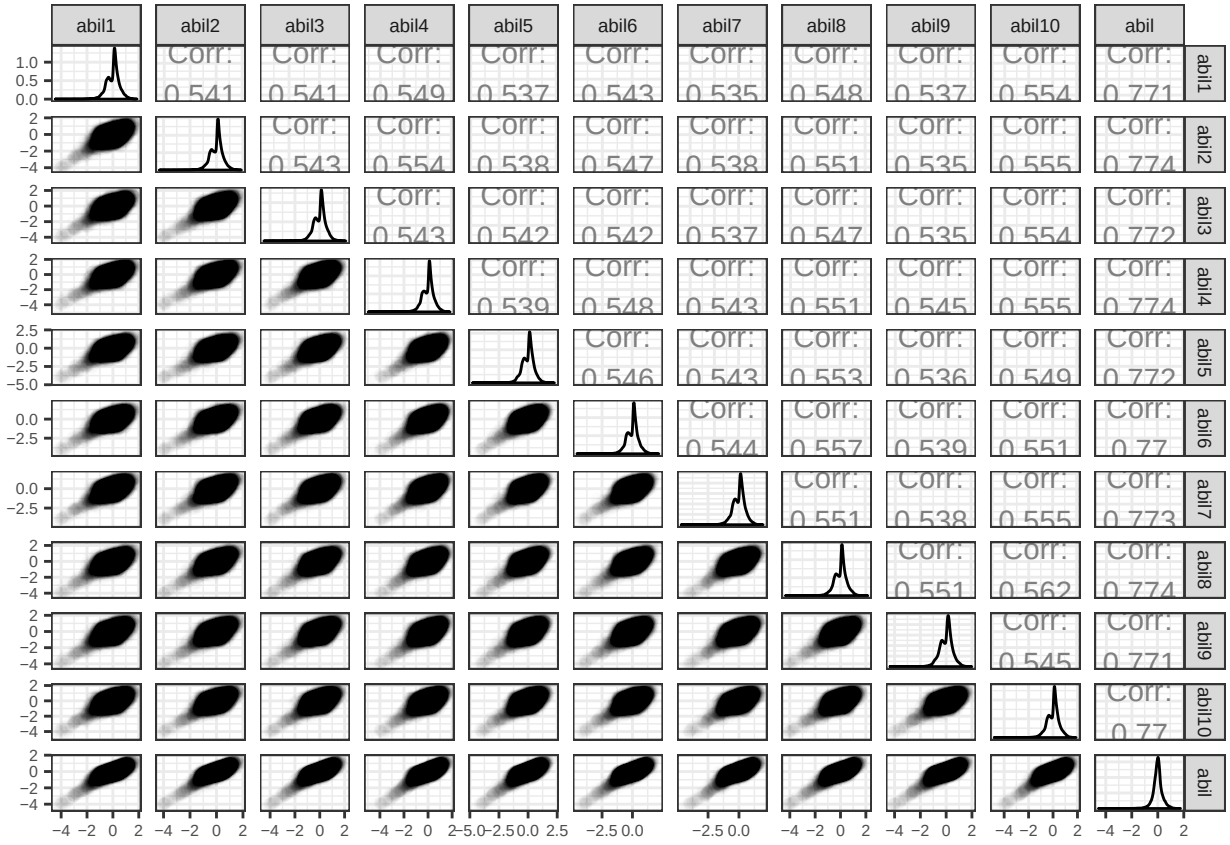
```
# abilities
abil1 = data.frame(stats_seren_1[grep("abil\\[", row.names(stats_seren_1)),c(1,3)])
names(abil1) <- c('abil1','sd_1'); abil1$user <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(a
abil2 = data.frame(stats_seren_2[grep("abil\\[", row.names(stats_seren_2)),c(1,3)])
names(abil2) <- c('abil2','sd_2'); abil2$user <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(a
abil3 = data.frame(stats_seren_3[grep("abil\\[", row.names(stats_seren_3)),c(1,3)])
names(abil3) <- c('abil3','sd_3'); abil3$user <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(a
abil4 = data.frame(stats_seren_4[grep("abil\\[", row.names(stats_seren_4)),c(1,3)])
names(abil4) <- c('abil4','sd_4'); abil4$user <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(a
abil5 = data.frame(stats_seren_5[grep("abil\\[", row.names(stats_seren_5)),c(1,3)])
names(abil5) <- c('abil5','sd_5'); abil5$user <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(a
abil6 = data.frame(stats_seren_6[grep("abil\\[", row.names(stats_seren_6)),c(1,3)])
names(abil6) <- c('abil6','sd_6'); abil6$user <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(a
abil7 = data.frame(stats_seren_7[grep("abil\\[", row.names(stats_seren_7)),c(1,3)])
names(abil7) <- c('abil7','sd_7'); abil7$user <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(a
```

```
abil8 = data.frame(stats_seren_8[grep("abil\\[", row.names(stats_seren_8)),c(1,3)])
names(abil8) <- c('abil8','sd_8'); abil8$user <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(a
abil9 = data.frame(stats_seren_9[grep("abil\\[", row.names(stats_seren_9)),c(1,3)])
names(abil9) <- c('abil9','sd_9'); abil9$user <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(a
abil10 = data.frame(stats_seren_10[grep("abil\\[", row.names(stats_seren_10)),c(1,3)])
names(abil10) <- c('abil10','sd_10'); abil10$user <- as.numeric(as.character(gsub("[^0-9.-]", "", rownar


abil1 <- abil1 %>% left_join((dplyr::select(s5_1, user, UserID) %>%
                                distinct()), by = c('user' = 'user'))
abil2 <- abil2 %>% left_join((dplyr::select(s5_2, user, UserID) %>%
                                distinct()), by = c('user' = 'user'))
abil3 <- abil3 %>% left_join((dplyr::select(s5_3, user, UserID) %>%
                                distinct()), by = c('user' = 'user'))
abil4 <- abil4 %>% left_join((dplyr::select(s5_4, user, UserID) %>%
                                distinct()), by = c('user' = 'user'))
abil5 <- abil5 %>% left_join((dplyr::select(s5_5, user, UserID) %>%
                                distinct()), by = c('user' = 'user'))
abil6 <- abil6 %>% left_join((dplyr::select(s5_6, user, UserID) %>%
                                distinct()), by = c('user' = 'user'))
abil7 <- abil7 %>% left_join((dplyr::select(s5_7, user, UserID) %>%
                                distinct()), by = c('user' = 'user'))
abil8 <- abil8 %>% left_join((dplyr::select(s5_8, user, UserID) %>%
                                distinct()), by = c('user' = 'user'))
abil9 <- abil9 %>% left_join((dplyr::select(s5_9, user, UserID) %>%
                                distinct()), by = c('user' = 'user'))
abil10 <- abil10 %>% left_join((dplyr::select(s5_10, user, UserID) %>%
                                  distinct()), by = c('user' = 'user'))


df_abil <- join_all(list(abil1, abil2,abil3, abil4,abil5,abil6,abil7,abil8,abil9, abil10), by='UserID',
df_abil <- df_abil[,c("UserID", names(df_abil)[grep('abil',names(df_abil))], names(df_abil)[grep('sd_',n
```

Assessing agreement between 10 subsets

```
W <- 1 / df_abil[,names(df_abil)[grep('sd_',names(df_abil))]] ^ 2
names(W) <- paste0('W', gsub("[^0-9.-]", "", names(W)) )
df_abil <- cbind(df_abil,W)

df_abil$abil <- rowSums(W * df_abil[,grep('abil',names(df_abil))], na.rm=T)/rowSums(W)

df_abil_melt <- df_abil
df_abil_melt$user <- as.numeric(as.factor(as.character(df_abil_melt$UserID)))

df_abil_melt <- df_abil_melt[, c( 'user', names(df_abil_melt)[grep('abil',names(df_abil_melt))]   )]

df_abil_melt <- melt(df_abil_melt, id.vars = 'user' )


ggpairs(data = df_abil, columns = grep('abil',names(df_abil)) ,
  lower = list(continuous = wrap("points", alpha = 0.01))) +
  theme_bw() +
theme(strip.text.x = element_text(size = 7),
```

```
      strip.text.y = element_text(size = 7),
      plot.title = element_text(hjust = 0.5, size=8.5),
      legend.text=element_text(size=7.5),
      legend.title = element_blank(),
      legend.position="none",
      legend.spacing.x = unit(0.05, 'cm'),
      axis.text.x = element_text(size = 6),
      axis.text.y = element_text(size = 6))
```

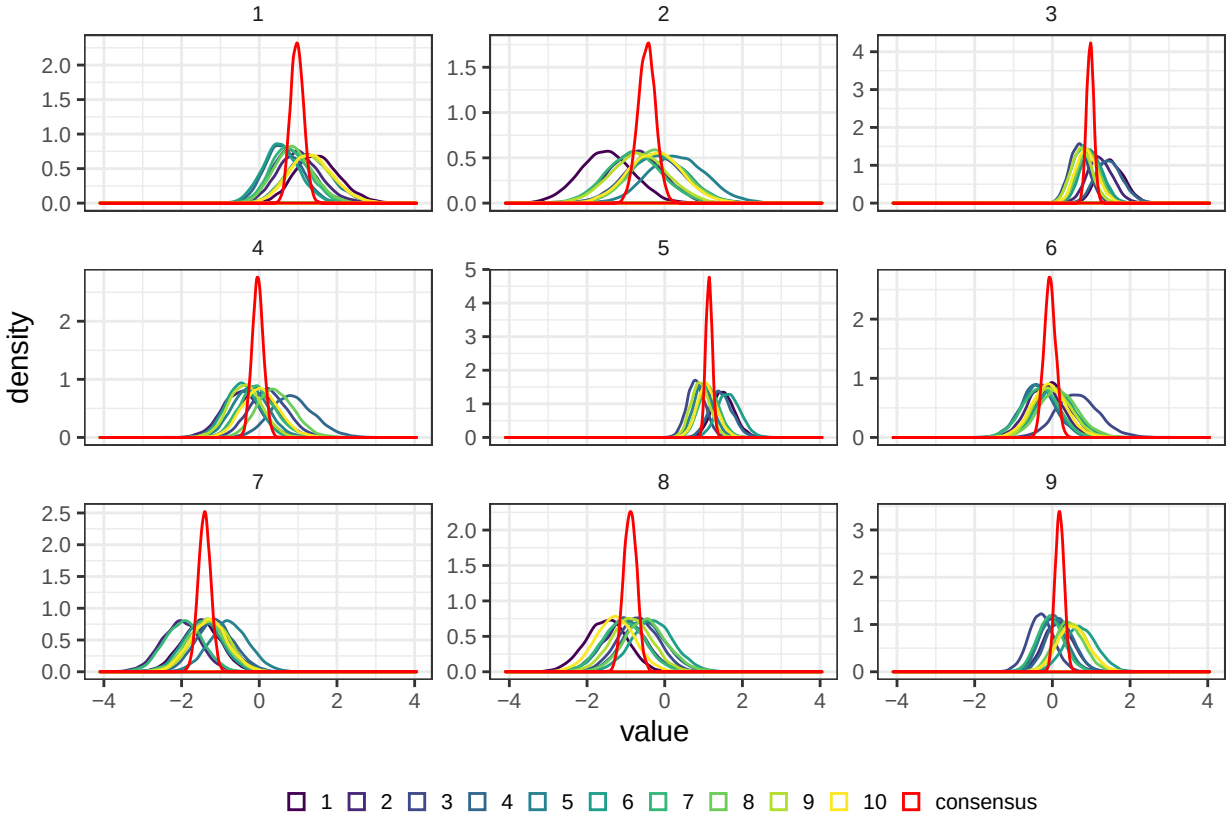



```
citi <- c('80c3e443a01284ce8be767b3b68b921d',
'1ee963bb67658ec7fb7ff5568f2e5c52',
'52a791d0b1715d3a64e3304853d78c7c',

'6ed2e808d62f729c45912856bffb1ae7',
'b1132cb9f684ac9bf8b08522c7537ea6',
'e6b9208fb9d03ce1c3885fbb866a39f8',

'43047c40867c00a83d0a3982f12f515b',
'cb93d3701ea002dd736828eb1a87cb88',
'cb7671ced14ad6f669a2601b249e0e93')

mcmc_abil <- function(fit.mcmc.abil, abil, machine ){
  names(abil)[1] <- 'abil'
  fit.mcmc.abil <- do.call(rbind.data.frame, fit.mcmc.abil)
  fit.mcmc.abil <- melt(fit.mcmc.abil)
```

```
  fit.mcmc.abil$user <- as.numeric(as.character(gsub("[^0-9.-]", "", fit.mcmc.abil$variable)))
  fit.mcmc.abil <- fit.mcmc.abil %>% left_join(abil[,c('user', 'UserID')],by= c('user' ))
  fit.mcmc.abil$machine = machine
  fit.mcmc.abil <- dplyr::filter(fit.mcmc.abil, UserID %in% citi)
  return(fit.mcmc.abil)
}

user.abil_1 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA//fit.mcmc.abil.rds')
user.abil_1 <- mcmc_abil(fit.mcmc.abil=user.abil_1, abil=abil1, machine = 1 )

user.abil_2 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_2//fit.mcmc.abil.rds')
user.abil_2 <- mcmc_abil(fit.mcmc.abil=user.abil_2, abil=abil2, machine = 2 )

user.abil_3 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_3//fit.mcmc.abil.rds')
user.abil_3 <- mcmc_abil(fit.mcmc.abil=user.abil_3, abil=abil3 , machine = 3 )

user.abil_4 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_4//fit.mcmc.abil.rds')
user.abil_4 <- mcmc_abil(fit.mcmc.abil=user.abil_4, abil=abil4, machine = 4  )

user.abil_5 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_5//fit.mcmc.abil.rds')
user.abil_5 <- mcmc_abil(fit.mcmc.abil=user.abil_5, abil=abil5, machine = 5  )

user.abil_6 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_6//fit.mcmc.abil.rds')
user.abil_6 <- mcmc_abil(fit.mcmc.abil=user.abil_6, abil=abil6, machine = 6  )

user.abil_7 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_7//fit.mcmc.abil.rds')
user.abil_7 <- mcmc_abil(fit.mcmc.abil=user.abil_7, abil=abil7, machine = 7  )

user.abil_8 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_8//fit.mcmc.abil.rds')
user.abil_8 <- mcmc_abil(fit.mcmc.abil=user.abil_8, abil=abil8, machine = 8  )

user.abil_9 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_9//fit.mcmc.abil.rds')
user.abil_9 <- mcmc_abil(fit.mcmc.abil=user.abil_9, abil=abil9, machine = 9  )

user.abil_10 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_10//fit.mcmc.abil.rds')
user.abil_10 <- mcmc_abil(fit.mcmc.abil=user.abil_10, abil=abil10, machine = 10  )


user.abil_all <- rbind(user.abil_1, user.abil_2, user.abil_3, user.abil_4, user.abil_5, user.abil_6, us



M = 10 # machines
d = length(unique(user.abil_all$UserID))# dimensions (species)
mach <- sort(unique(user.abil_all$machine)) # machines
veri <- user.abil_all %>% group_by(machine, user) %>%
  dplyr::summarise(ns=n())

ii=1
nas <- dplyr::filter(user.abil_all, machine == mach[ii]) %>% arrange(UserID) %>%
  dplyr::group_by(UserID) %>% dplyr::mutate(id = row_number()) %>%
  dplyr::select(user , UserID, id, value) %>% mutate(value = NA) %>%
  mutate(user = NA)
```

```
theta <- NULL
for(ii in 1:length(mach)){
  m1 <- dplyr::filter(user.abil_all, machine == mach[ii]) %>% arrange(UserID)
  m1 <- m1 %>% dplyr::group_by(UserID) %>% dplyr::mutate(id = row_number())
  m1 <- m1 %>% dplyr::select(user , UserID, id, value)
  nn <- dcast(m1, user + UserID  ~ id )
  nn <- nn %>% arrange(UserID)
  theta[[ii]] <- nn

}


subchain_0 <- array((unlist(theta)), dim=c(9, 7502, 10)) #9 users, 10 machines,  7502 = 3 chains x 2500
subchain <- subchain_0[,3:7502,]


W <- matrix(NA, d, M)
sampletotT <- dim(subchain)[2]
for (j in 1:d)
  for (s in 1:M)
    W[j, s] <- 1/var(subchain[j, , s], na.rm=T)
theta_ <- matrix(NA, nrow = d, ncol = sampletotT)
for (i in 1:sampletotT){
  theta_[, i] <- rowSums(W * as.numeric(as.character(subchain[,i, ])), dims = 1, na.rm=T)/rowSums(W, na
}
# adding as 11th dimension the consensus posterior
theta[[11]] <- cbind(theta[[1]][,1:2], theta_)

user_comb <- do.call(rbind.data.frame, theta)
user_comb$machine <- rep(1:11, each = 9) # 9 users
user_comb_melt <- melt(user_comb, id.vars = c('user', 'UserID', 'machine'))

colsp <- c(viridis(10), 2)

user_comb_melt$user <- as.numeric(as.factor(as.character(user_comb_melt$UserID)))

user_comb_melt[user_comb_melt$machine ==11,]$machine <- 'consensus'
user_comb_melt$machine <- factor(user_comb_melt$machine)
levels(user_comb_melt$machine) <- c("1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "consensus")

dplyr::filter(user_comb_melt) %>%
  ggplot(.) + geom_density(aes(x = value, col = machine)) +
  scale_color_manual(values = colsp) +
  facet_wrap(~user, scales = 'free_y',nrow = 3)+ theme_bw()+
  guides(col=guide_legend(nrow=1,byrow=TRUE)  )+
  theme(strip.text.x = element_text(size = 8),
        strip.text.y = element_text(size = 8),
        plot.title = element_text(hjust = 0.5, size=8.5),
        legend.text=element_text(size=7.5),
        legend.title = element_blank(),
        legend.position="bottom",
        axis.text.x = element_text( size = 8),
```

```
        axis.text.y = element_text( size = 8),
        legend.key.size = unit(0.5,"line"), # size of symbols
        strip.background = element_blank())  # removes gray background top
```



2.2 Species difficulties

```
s5_1$True_Species_num <- as.numeric(as.factor(as.character(s5_1$True_Species)))
s5_2$True_Species_num <- as.numeric(as.factor(as.character(s5_2$True_Species)))
s5_3$True_Species_num <- as.numeric(as.factor(as.character(s5_3$True_Species)))
s5_4$True_Species_num <- as.numeric(as.factor(as.character(s5_4$True_Species)))
s5_5$True_Species_num <- as.numeric(as.factor(as.character(s5_5$True_Species)))
s5_6$True_Species_num <- as.numeric(as.factor(as.character(s5_6$True_Species)))
s5_7$True_Species_num <- as.numeric(as.factor(as.character(s5_7$True_Species)))
s5_8$True_Species_num <- as.numeric(as.factor(as.character(s5_8$True_Species)))
s5_9$True_Species_num <- as.numeric(as.factor(as.character(s5_9$True_Species)))
s5_10$True_Species_num <- as.numeric(as.factor(as.character(s5_10$True_Species)))

species1 = data.frame(stats_seren_1[grep("species\\[", row.names(stats_seren_1)),c(1,3)])
names(species1) <- c('species1','sd_1'); species1$species <- as.numeric(as.character(gsub("[^0-9.-]", "
species2 = data.frame(stats_seren_2[grep("species\\[", row.names(stats_seren_2)),c(1,3)])
names(species2) <- c('species2','sd_2'); species2$species <- as.numeric(as.character(gsub("[^0-9.-]", "
species3 = data.frame(stats_seren_3[grep("species\\[", row.names(stats_seren_3)),c(1,3)])
names(species3) <- c('species3','sd_3'); species3$species <- as.numeric(as.character(gsub("[^0-9.-]", "
```

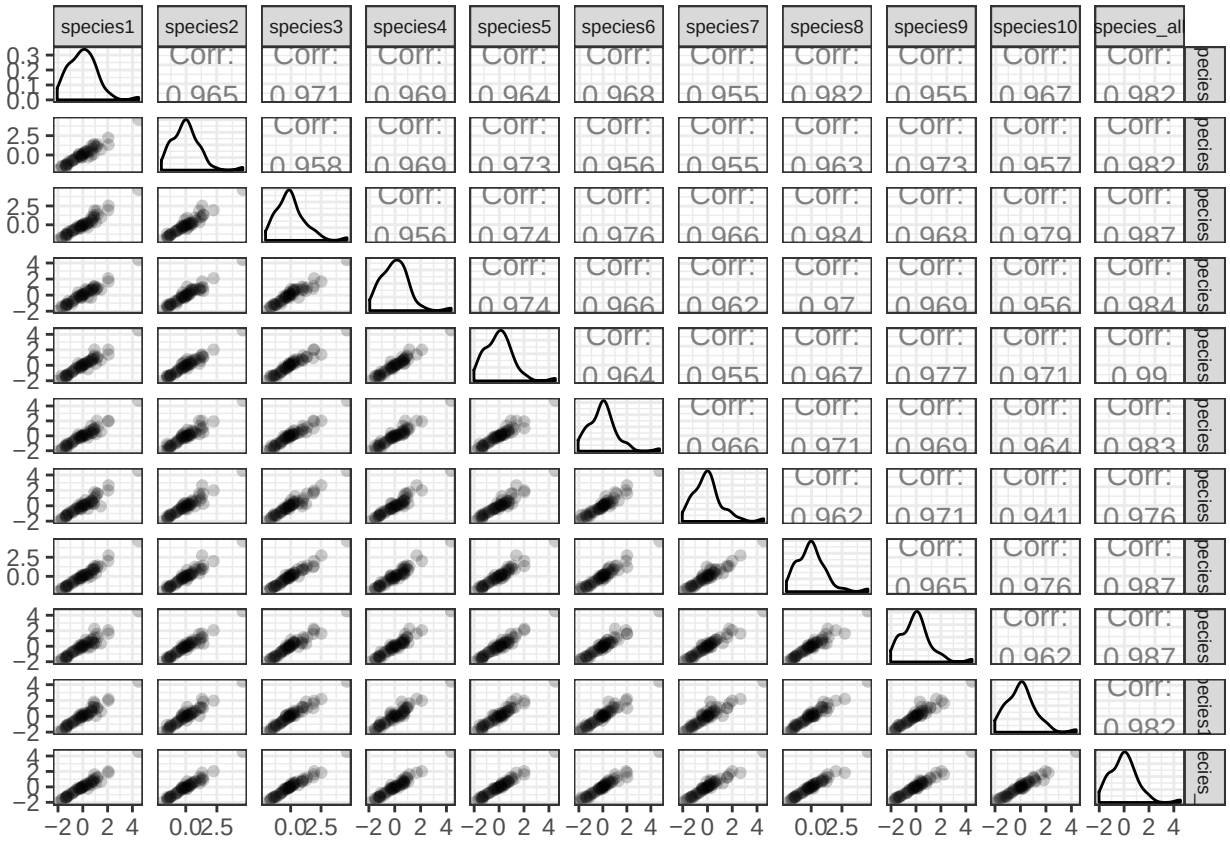
```r
species4 = data.frame(stats_seren_4[grep("species\\[", row.names(stats_seren_4)),c(1,3)])
names(species4) <- c('species4','sd_4'); species4$species <- as.numeric(as.character(gsub("[^0-9.-]", "
species5 = data.frame(stats_seren_5[grep("species\\[", row.names(stats_seren_5)),c(1,3)])
names(species5) <- c('species5','sd_5'); species5$species <- as.numeric(as.character(gsub("[^0-9.-]", "
species6 = data.frame(stats_seren_6[grep("species\\[", row.names(stats_seren_6)),c(1,3)])
names(species6) <- c('species6','sd_6'); species6$species <- as.numeric(as.character(gsub("[^0-9.-]", "
species7 = data.frame(stats_seren_7[grep("species\\[", row.names(stats_seren_7)),c(1,3)])
names(species7) <- c('species7','sd_7'); species7$species <- as.numeric(as.character(gsub("[^0-9.-]", "
species8 = data.frame(stats_seren_8[grep("species\\[", row.names(stats_seren_8)),c(1,3)])
names(species8) <- c('species8','sd_8'); species8$species <- as.numeric(as.character(gsub("[^0-9.-]", "
species9 = data.frame(stats_seren_9[grep("species\\[", row.names(stats_seren_9)),c(1,3)])
names(species9) <- c('species9','sd_9'); species9$species <- as.numeric(as.character(gsub("[^0-9.-]", "
species10 = data.frame(stats_seren_10[grep("species\\[", row.names(stats_seren_10)),c(1,3)]); names(spe
species10$species <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(species10))))
```

```r
species1 <- species1 %>% left_join( ( dplyr::select(s5_1, True_Species_num, True_Species) %>% distinct()
species2 <- species2 %>% left_join( (dplyr::select(s5_2, True_Species_num, True_Species) %>% distinct()
species3 <- species3 %>% left_join( (dplyr::select(s5_3, True_Species_num, True_Species) %>% distinct()
species4 <- species4 %>% left_join( (dplyr::select(s5_4, True_Species_num, True_Species) %>% distinct()
species5 <- species5 %>% left_join( (dplyr::select(s5_5, True_Species_num, True_Species) %>% distinct()
species6 <- species6 %>% left_join( (dplyr::select(s5_6, True_Species_num, True_Species) %>% distinct()
species7 <- species7 %>% left_join( (dplyr::select(s5_7, True_Species_num, True_Species) %>% distinct()
species8 <- species8 %>% left_join( (dplyr::select(s5_8, True_Species_num, True_Species) %>% distinct()
species9 <- species9 %>% left_join( (dplyr::select(s5_9, True_Species_num, True_Species) %>% distinct()
species10 <- species10 %>% left_join( (dplyr::select(s5_10, True_Species_num, True_Species) %>% distinc
```

```r
df_species <- join_all(list(species1, species2 ,species3, species4,species5,
                            species6,species7,species8,species9, species10),
                       by='True_Species', type='left') # had an issue joining by species numeric . shar

df_species <- df_species[,
                         c("True_Species",
                           #names(df_species)[grep('species',names(df_species))],
                           paste0('species',1:10),
                           names(df_species)[grep('sd_',names(df_species))])]
```

```r
# recombining the posteriors
# consensus monte carlo.
# Bayes and Big Data: The Consensus Monte Carlo Algorithm
W <- 1 / (df_species[,names(df_species)[grep('sd_',names(df_species))]] ^ 2)
names(W) <- paste0('W', gsub("[^0-9.-]", "", names(W)) )
df_species <- cbind(df_species, W)

df_species$species_all <- rowSums( W * df_species[,grep('species',
                                                        names(df_species))], na.rm=T)/
                                      rowSums(W, na.rm=T)

df_species_melt <- df_species
df_species_melt$species <- as.numeric(as.factor(as.character(df_species_melt$True_Species)))
df_species_melt <- df_species_melt[, c('True_Species' , names(df_species_melt)[grep('species',names(df_
df_species_melt <- melt(df_species_melt, id.vars = c('True_Species','species') )
```

```
ggpairs(data = df_species, columns = grep('species',names(df_species)) ,
  lower = list(continuous = wrap("points", alpha = 0.2))) +
  theme_bw() + theme(strip.text.x = element_text(size = 7),
                     strip.text.y = element_text(size = 7))
```

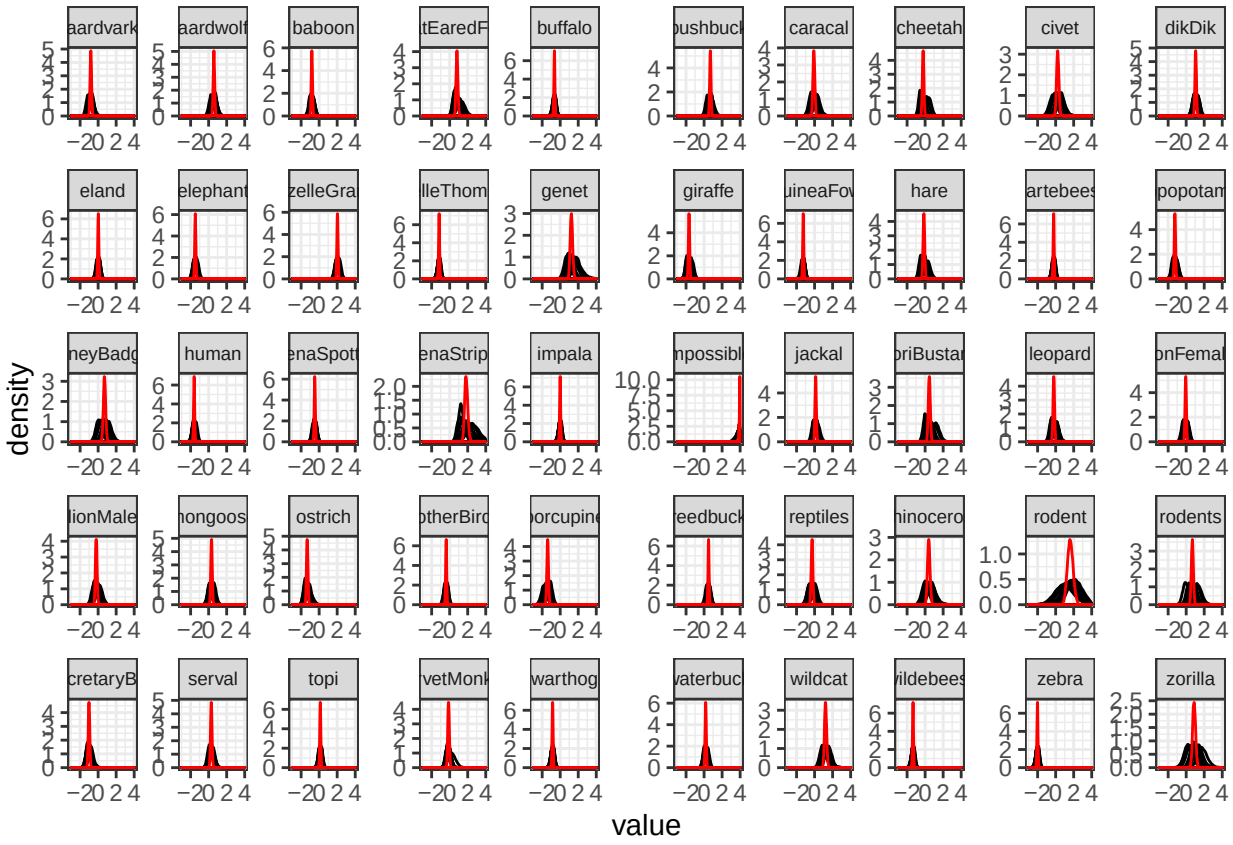



```
mcmc_species <- function(fit.mcmc.diff.species_2, species2, machine ){
  fit.mcmc.diff.species_2 <- do.call(rbind.data.frame, fit.mcmc.diff.species_2)
  fit.mcmc.diff.species_2 <- melt(fit.mcmc.diff.species_2)
  fit.mcmc.diff.species_2$species <- as.numeric(as.character(gsub("[^0-9.-]", "", fit.mcmc.diff.species_
  fit.mcmc.diff.species_2 <- fit.mcmc.diff.species_2 %>% left_join(species2[,c('species', 'True_Species
  fit.mcmc.diff.species_2$machine = machine
  #fit.mcmc.diff.species_2 <- sample_frac(fit.mcmc.diff.species_2, 0.1) # NB : sampling
  return(fit.mcmc.diff.species_2)
  }

diff.sp_1 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA//fit.mcmc.diff.species.rds')
diff.sp_1 <- mcmc_species(fit.mcmc.diff.species_2=diff.sp_1, species2=species1, machine = 1 )

diff.sp_2 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_2//fit.mcmc.diff.species.rds')
diff.sp_2 <- mcmc_species(fit.mcmc.diff.species_2=diff.sp_2, species2=species2, machine = 2 )

diff.sp_3 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_3//fit.mcmc.diff.species.rds')
diff.sp_3 <- mcmc_species(fit.mcmc.diff.species_2=diff.sp_3, species2=species3 , machine = 3 )

diff.sp_4 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_4//fit.mcmc.diff.species.rds')
```

```
diff.sp_4 <- mcmc_species(fit.mcmc.diff.species_2=diff.sp_4, species2=species4, machine = 4  )

diff.sp_5 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_5//fit.mcmc.diff.species.rds')
diff.sp_5 <- mcmc_species(fit.mcmc.diff.species_2=diff.sp_5, species2=species5, machine = 5  )

diff.sp_6 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_6//fit.mcmc.diff.species.rds')
diff.sp_6 <- mcmc_species(fit.mcmc.diff.species_2=diff.sp_6, species2=species6, machine = 6  )

diff.sp_7 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_7//fit.mcmc.diff.species.rds')
diff.sp_7 <- mcmc_species(fit.mcmc.diff.species_2=diff.sp_7, species2=species7, machine = 7  )

diff.sp_8 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_8//fit.mcmc.diff.species.rds')
diff.sp_8 <- mcmc_species(fit.mcmc.diff.species_2=diff.sp_8, species2=species8, machine = 8  )

diff.sp_9 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_9//fit.mcmc.diff.species.rds')
diff.sp_9 <- mcmc_species(fit.mcmc.diff.species_2=diff.sp_9, species2=species9, machine = 9  )

diff.sp_10 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_10//fit.mcmc.diff.species.rds')
diff.sp_10 <- mcmc_species(fit.mcmc.diff.species_2=diff.sp_10, species2=species10, machine = 10  )

diff.sp_all <- rbind(diff.sp_1, diff.sp_2, diff.sp_3, diff.sp_4,
                     diff.sp_5, diff.sp_6, diff.sp_7, diff.sp_8,
                     diff.sp_9, diff.sp_10)
```

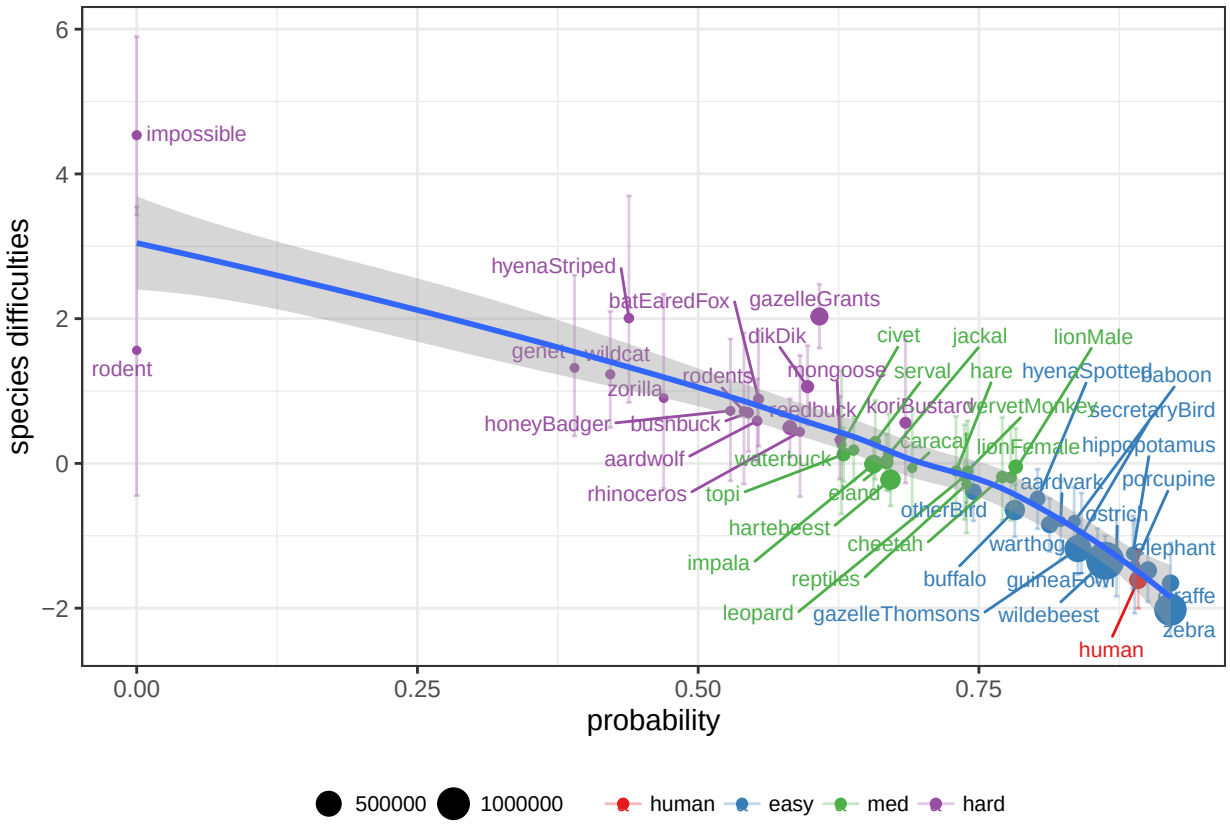
```
M = 10 # machines
d = length(unique(diff.sp_all$True_Species))# dimensions (species)
mach <- sort(unique(diff.sp_all$machine))

ii=1
nas <- dplyr::filter(diff.sp_all, machine == mach[ii]) %>% arrange(True_Species) %>%
  dplyr::group_by(True_Species) %>% dplyr::mutate(id = row_number()) %>%
  dplyr::select(species , True_Species, id, value) %>% mutate(value = NA) %>%
  mutate(species = NA)


theta <- NULL
for(ii in 1:length(mach)){
 m1 <- dplyr::filter(diff.sp_all, machine == mach[ii]) %>% arrange(True_Species)
 m1 <- m1 %>% dplyr::group_by(True_Species) %>% dplyr::mutate(id = row_number())
 #m1$num <- ave(m1$True_Species, m1$value, FUN = seq_along)
 m1 <- m1 %>% dplyr::select(species , True_Species, id, value)

 if(ii==2){
 m1 <- rbind(m1, nas)
 m1 <- m1 %>% dplyr::group_by(True_Species, id) %>%
   dplyr::summarise(species = mean(species, na.rm=T) ,
                    value = mean(value, na.rm=T))
 }

 nn <- dcast(m1, species + True_Species  ~ id )
 nn <- nn %>% arrange(True_Species)
 theta[[ii]] <- nn
```

```
}

subchain_0 <- array((unlist(theta)), dim=c(50, 7502, 10))
subchain <- subchain_0[,3:7502,]


W <- matrix(NA, d, M) # inverse of the chain variance on each species/machine combination
sampletotT <- dim(subchain)[2]
for (j in 1:d)
  for (s in 1:M)
    W[j, s] <- 1/var(subchain[j,, s], na.rm=T)
theta_ <- matrix(NA, nrow = d, ncol = sampletotT)
for (i in 1:sampletotT)
  theta_[, i] <- rowSums(W * as.numeric(as.character(subchain[,i, ])),
                         dims = 1, na.rm=T)/rowSums(W, na.rm=T)


theta[[11]] <- cbind(theta[[1]][,1:2], theta_)
species_comb <- do.call(rbind.data.frame, theta)
species_comb$machine <- rep(1:11, each = 50)
species_comb_melt <- reshape2::melt(species_comb, id.vars = c('species', 'True_Species', 'machine'))


colsp <- c(rep(1,10), 2)
  ggplot(species_comb_melt) + geom_density(aes(x = value, col = factor(machine))) +
  xlim(-3,4) +
  scale_color_manual(values = colsp) +
  facet_wrap(~True_Species, scales = 'free', nrow = 5)+ theme_bw()+
  theme(strip.text.x = element_text(size = 7),
        strip.text.y = element_text(size = 7),
        plot.title = element_text(hjust = 0.5, size=8.5),
        legend.text=element_text(size=7.5),
        legend.title = element_blank(),
        legend.position="none",
        legend.spacing.x = unit(0.05, 'cm')) +
  guides(fill=guide_legend(nrow=1,byrow=TRUE)
         )
```




```
stats_species <- species_comb_melt %>% dplyr::select(True_Species, value) %>%
  dplyr::group_by(True_Species) %>%
  dplyr::summarise(mean = mean(value, na.rm = T),
                   sd = sd(value, na.rm = T),
                   med = median(value, na.rm = T),
                   q2.5 = quantile(value, na.rm = T, .025),
                   q97.5 = quantile(value, na.rm = T, .975))

stats_species$param <- paste0('diff_species_',1:nrow(stats_species))
stats_species <- stats_species %>% dplyr::select(param, everything()); names(stats_species)[2] <- 'id'


s3 <- s5_sets
s3$True_Species_num <- as.numeric(as.factor(s3$True_Species)) # 40 species
s3$True_Species <- relevel(factor(s3$True_Species) , ref = "human")

s3_species <- s3 %>%
  dplyr::group_by(True_Species, True_Species_num) %>%
  dplyr::summarise( prob = mean(correct),
                    ns = n()) %>% arrange(True_Species_num)

s3_species <- s3_species %>% left_join(stats_species[,c('id', 'mean', 'q2.5', "q97.5")], by = c('True_Sp

names(s3_species)[names(s3_species)=='mean'] <- 'diff_species'

s3_species$True_Species_col <- NA
```

```
s3_species$True_Species_col <- cut(s3_species$diff_species,
                                   breaks = c(-1000, quantile(s3_species$diff_species, probs = seq(0, 1
                                   c('easy', 'med','hard'), include.lowest = T)
s3_species$True_Species_col <- as.character(s3_species$True_Species_col)

s3_species[s3_species$True_Species == 'human',]$True_Species_col <- 'human'
s3_species$True_Species_col <- factor(s3_species$True_Species_col, levels = c("human", "easy",  "med", 

sp_col <- brewer.pal(4, "Set1")

ggplot(s3_species, aes(x = prob, y = diff_species)) +
  geom_point(aes(col = True_Species_col, size = ns, col = True_Species_col))+
  geom_text_repel(aes(label = True_Species, col = True_Species_col), size = 2.75)+
  geom_errorbar(aes(ymin=q2.5, ymax=q97.5, col = True_Species_col),
                width=0.005, alpha = 0.3,  #, col = True_Species_col
                position=position_dodge(.005) ) +
  scale_color_manual(values = sp_col) +
  geom_smooth(aes(x = prob, y = diff_species))+
  xlab('probability')+
  ylab('species difficulties') +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5, size=8.5),
        legend.text=element_text(size=7.5),
        legend.title = element_blank(),
        legend.position="bottom",
        legend.spacing.x = unit(0.05, 'cm')) +
  guides(fill=guide_legend(nrow=1,byrow=TRUE))
```



2.3 Guessing

```
guess1 = data.frame(stats_seren_1[grep("eta\\[", row.names(stats_seren_1)),c(1,3)])
names(guess1) <- c('guess1','sd_1'); guess1$species <- as.numeric(as.character(gsub("[^0-9.-]", "", row
guess2 = data.frame(stats_seren_2[grep("eta\\[", row.names(stats_seren_2)),c(1,3)])
names(guess2) <- c('guess2','sd_2'); guess2$species <- as.numeric(as.character(gsub("[^0-9.-]", "", row
guess3 = data.frame(stats_seren_3[grep("eta\\[", row.names(stats_seren_3)),c(1,3)])
names(guess3) <- c('guess3','sd_3'); guess3$species <- as.numeric(as.character(gsub("[^0-9.-]", "", row
guess4 = data.frame(stats_seren_4[grep("eta\\[", row.names(stats_seren_4)),c(1,3)])
names(guess4) <- c('guess4','sd_4'); guess4$species <- as.numeric(as.character(gsub("[^0-9.-]", "", row
guess5 = data.frame(stats_seren_5[grep("eta\\[", row.names(stats_seren_5)),c(1,3)])
names(guess5) <- c('guess5','sd_5'); guess5$species <- as.numeric(as.character(gsub("[^0-9.-]", "", row
guess6 = data.frame(stats_seren_6[grep("eta\\[", row.names(stats_seren_6)),c(1,3)])
names(guess6) <- c('guess6','sd_6'); guess6$species <- as.numeric(as.character(gsub("[^0-9.-]", "", row
guess7 = data.frame(stats_seren_7[grep("eta\\[", row.names(stats_seren_7)),c(1,3)])
names(guess7) <- c('guess7','sd_7'); guess7$species <- as.numeric(as.character(gsub("[^0-9.-]", "", row
guess8 = data.frame(stats_seren_8[grep("eta\\[", row.names(stats_seren_8)),c(1,3)])
names(guess8) <- c('guess8','sd_8'); guess8$species <- as.numeric(as.character(gsub("[^0-9.-]", "", row
guess9 = data.frame(stats_seren_9[grep("eta\\[", row.names(stats_seren_9)),c(1,3)])
names(guess9) <- c('guess9','sd_9'); guess9$species <- as.numeric(as.character(gsub("[^0-9.-]", "", row
guess10 = data.frame(stats_seren_10[grep("eta\\[", row.names(stats_seren_10)),c(1,3)])
names(guess10) <- c('guess10','sd_10'); guess10$species <- as.numeric(as.character(gsub("[^0-9.-]", "",
```

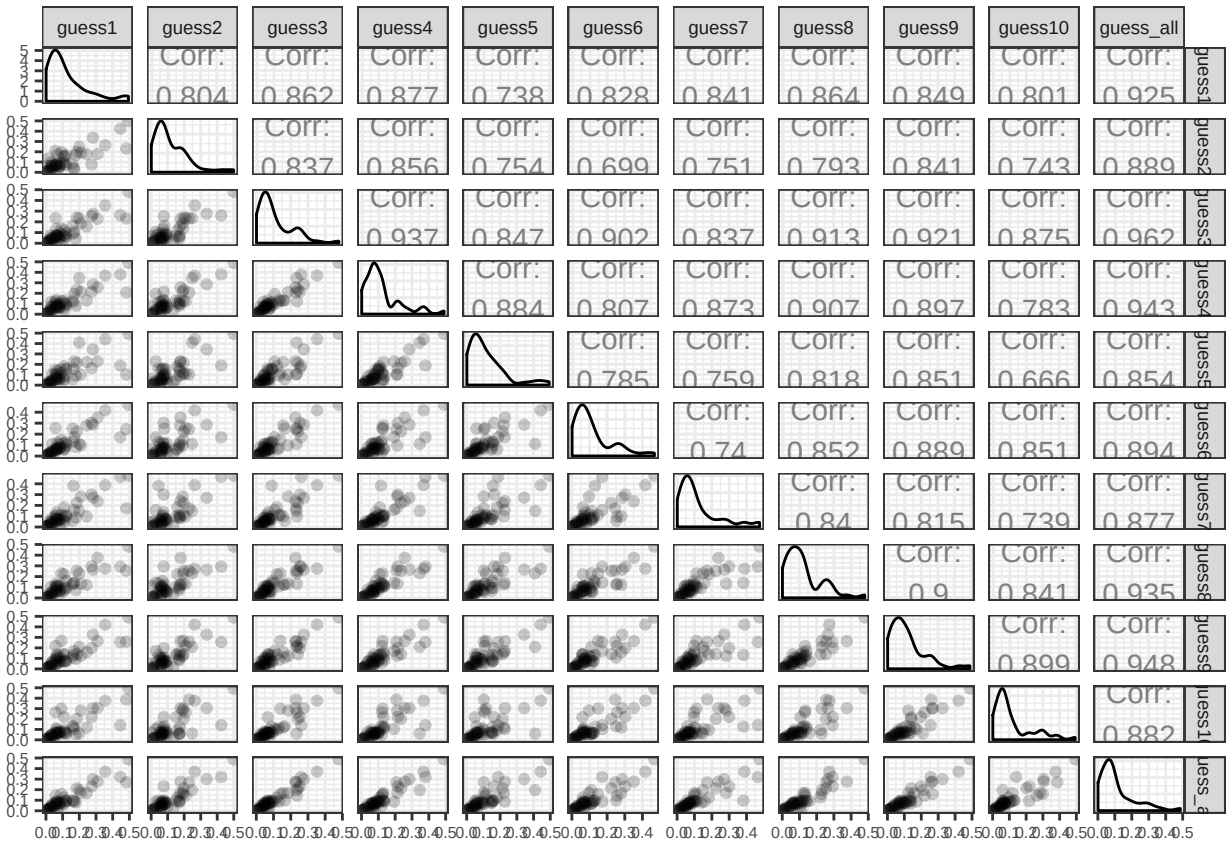
```
guess1 <- guess1 %>% left_join( ( dplyr::select(s5_1, True_Species_num, True_Species) %>% distinct() ),
guess2 <- guess2 %>% left_join( (dplyr::select(s5_2, True_Species_num, True_Species) %>% distinct() ), b
guess3 <- guess3 %>% left_join( (dplyr::select(s5_3, True_Species_num, True_Species) %>% distinct() ), b
guess4 <- guess4 %>% left_join( (dplyr::select(s5_4, True_Species_num, True_Species) %>% distinct() ), b
guess5 <- guess5 %>% left_join( (dplyr::select(s5_5, True_Species_num, True_Species) %>% distinct() ), b
guess6 <- guess6 %>% left_join( (dplyr::select(s5_6, True_Species_num, True_Species) %>% distinct() ), b
guess7 <- guess7 %>% left_join( (dplyr::select(s5_7, True_Species_num, True_Species) %>% distinct() ), b
guess8 <- guess8 %>% left_join( (dplyr::select(s5_8, True_Species_num, True_Species) %>% distinct() ), b
guess9 <- guess9 %>% left_join( (dplyr::select(s5_9, True_Species_num, True_Species) %>% distinct() ), b
guess10 <- guess10 %>% left_join( (dplyr::select(s5_10, True_Species_num, True_Species) %>% distinct()
```

```
df_guess <- join_all(list(guess1, guess2 ,guess3, guess4,guess5,guess6,guess7,guess8,guess9, guess10), b

df_guess <- df_guess[,c("True_Species", names(df_guess)[grep('guess',names(df_guess))], names(df_guess)
df_guess$species <- NULL
```

Recombining the guessing subposteriors

```
W <- 1 / (df_guess[,names(df_guess)[grep('sd_',names(df_guess))]] ^ 2)
names(W) <- paste0('W', gsub("[^0-9.-]", "", names(W)) )
df_guess <- cbind(df_guess, W)

df_guess$guess_all <- rowSums( W * df_guess[,grep('guess',
                      names(df_guess))], na.rm=T)/
                       rowSums(W, na.rm=T)

df_guess_melt <- df_guess
df_guess_melt$species <- as.numeric(as.factor(as.character(df_guess_melt$True_Species)))

df_guess_melt <- df_guess_melt[, c('True_Species' ,'species', names(df_guess_melt)[grep('guess',names(di

df_guess_melt <- melt(df_guess_melt, id.vars = c('True_Species','species') )
```

```
ggpairs(data = df_guess,
         columns = grep('guess',names(df_guess)) ,
         lower = list(continuous = wrap("points",
          alpha = 0.2))) +
  theme_bw() + theme(strip.text.x = element_text(size = 7),
                     strip.text.y = element_text(size = 7),
                     axis.text.x = element_text( size = 6),
                     axis.text.y = element_text( size = 6))
```



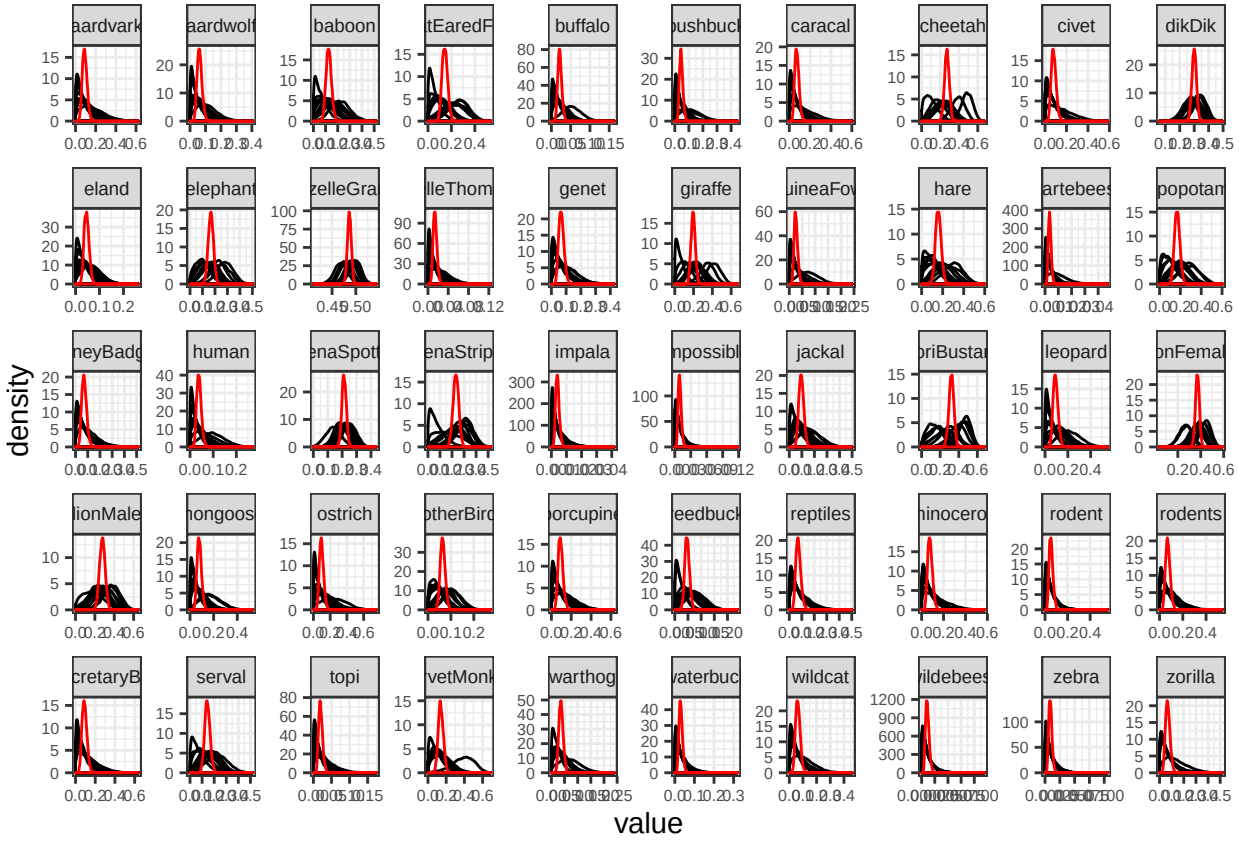

```
guess.sp_1 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA//fit.mcmc.guess.rds')
guess.sp_1 <- mcmc_species(fit.mcmc.diff.species_2=guess.sp_1, species2=guess1, machine = 1 )

guess.sp_2 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_2//fit.mcmc.guess.rds')
guess.sp_2 <- mcmc_species(fit.mcmc.diff.species_2=guess.sp_2, species2=guess2, machine = 2 )

guess.sp_3 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_3//fit.mcmc.guess.rds')
guess.sp_3 <- mcmc_species(fit.mcmc.diff.species_2=guess.sp_3, species2=guess3 , machine = 3 )

guess.sp_4 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_4//fit.mcmc.guess.rds')
guess.sp_4 <- mcmc_species(fit.mcmc.diff.species_2=guess.sp_4, species2=guess4, machine = 4  )

guess.sp_5 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_5//fit.mcmc.guess.rds')
guess.sp_5 <- mcmc_species(fit.mcmc.diff.species_2=guess.sp_5, species2=guess5, machine = 5  )

guess.sp_6 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_6//fit.mcmc.guess.rds')
guess.sp_6 <- mcmc_species(fit.mcmc.diff.species_2=guess.sp_6, species2=guess6, machine = 6  )

guess.sp_7 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_7//fit.mcmc.guess.rds')
guess.sp_7 <- mcmc_species(fit.mcmc.diff.species_2=guess.sp_7, species2=guess7, machine = 7  )

guess.sp_8 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_8//fit.mcmc.guess.rds')
guess.sp_8 <- mcmc_species(fit.mcmc.diff.species_2=guess.sp_8, species2=guess8, machine = 8  )

guess.sp_9 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_9//fit.mcmc.guess.rds')
guess.sp_9 <- mcmc_species(fit.mcmc.diff.species_2=guess.sp_9, species2=guess9, machine = 9  )
```

```
guess.sp_10 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_10//fit.mcmc.guess.rds')
guess.sp_10 <- mcmc_species(fit.mcmc.diff.species_2=guess.sp_10, species2=guess10, machine = 10  )

guess.sp_all <- rbind(guess.sp_1, guess.sp_2, guess.sp_3, guess.sp_4, guess.sp_5, guess.sp_6, guess.sp_
```

```
M = 10 # machines
d = length(unique(guess.sp_all$True_Species))# dimensions (species)
mach <- sort(unique(guess.sp_all$machine))

ii=1
nas <- dplyr::filter(guess.sp_all, machine == mach[ii]) %>% arrange(True_Species) %>%
  dplyr::group_by(True_Species) %>% dplyr::mutate(id = row_number()) %>%
  dplyr::select(species , True_Species, id, value) %>% mutate(value = NA) %>%
  mutate(species = NA)
theta <- NULL
for(ii in 1:length(mach)){
  m1 <- dplyr::filter(guess.sp_all, machine == mach[ii]) %>% arrange(True_Species)
  m1 <- m1 %>% dplyr::group_by(True_Species) %>% dplyr::mutate(id = row_number())
  m1 <- m1 %>% dplyr::select(species , True_Species, id, value)

  if(ii==2){
    m1 <- rbind(m1, nas)
    m1 <- m1 %>% dplyr::group_by(True_Species, id) %>%
      dplyr::summarise(species = mean(species, na.rm=T) ,
                       value = mean(value, na.rm=T))
  }

  nn <- dcast(m1, species + True_Species  ~ id )
  nn <- nn %>% arrange(True_Species)
  theta[[ii]] <- nn
}

subchain_0 <- array((unlist(theta)), dim=c(50, 7502, 10))
subchain <- subchain_0[,3:7502,]


W <- matrix(NA, d, M)
sampletotT <- dim(subchain)[2]
for (j in 1:d)
  for (s in 1:M)
    W[j, s] <- 1/var(subchain[j,, s], na.rm=T)
theta_ <- matrix(NA, nrow = d, ncol = sampletotT)
for (i in 1:sampletotT)
  theta_[, i] <- rowSums(W * as.numeric(as.character(subchain[,i, ])), dims = 1, na.rm=T)/rowSums(W, na

theta[[11]] <- cbind(theta[[1]][,1:2], theta_)

guess_comb <- do.call(rbind.data.frame, theta)
dim(guess_comb)
```

```
## [1]  550 7502
```

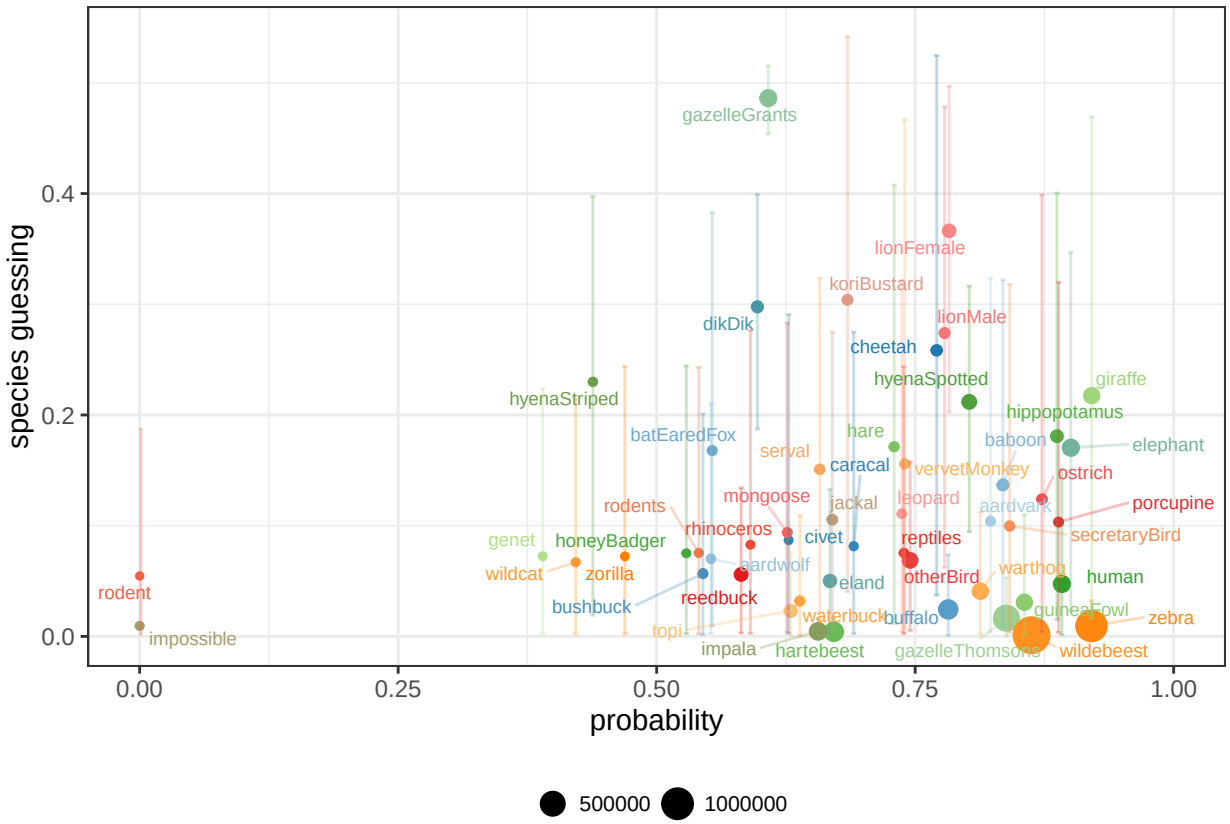
```
guess_comb$machine <- rep(1:11, each = 50)
guess_comb_melt <- melt(guess_comb, id.vars = c('species', 'True_Species', 'machine'))

colsp <- c(rep(1,10), 2)
ggplot(guess_comb_melt) + geom_density(aes(x = value, col = factor(machine))) +
  scale_color_manual(values = colsp) +
  facet_wrap(~True_Species, scales = 'free', nrow = 5)+ theme_bw()+
  theme(strip.text.x = element_text(size = 7),
        strip.text.y = element_text(size = 7),
        plot.title = element_text(hjust = 0.5, size=8.5),
        legend.text=element_text(size=7.5),
        legend.title = element_blank(),
        legend.position="none",
        legend.spacing.x = unit(0.05, 'cm'),
        axis.text.x = element_text( size = 6),
        axis.text.y = element_text( size = 6)) +
  guides(fill=guide_legend(nrow=1,byrow=TRUE)
  )
```




```
stats_guess <- guess_comb_melt %>% dplyr::select(True_Species, value) %>%
  dplyr::group_by(True_Species) %>%
  dplyr::summarise(mean = mean(value, na.rm = T),
                   sd = sd(value, na.rm = T),
                   med = median(value, na.rm = T),
                   q2.5 = quantile(value, na.rm = T, .025),
```

```
                  q97.5 = quantile(value, na.rm = T, .975))
stats_guess$param <- paste0('guessing_',1:nrow(stats_guess));  stats_guess <- stats_guess %>% dplyr::sel
```

```
s3 <- s5_sets
s3$True_Species_num <- as.numeric(as.factor(s3$True_Species)) # 40 species
s3$True_Species <- relevel(factor(s3$True_Species) , ref = "human")

s3_species <- s3 %>%
  dplyr::group_by(True_Species, True_Species_num) %>%
  dplyr::summarise( prob = mean(correct),
                    ns = n()) %>% arrange(True_Species_num)

s3_species <- s3_species %>% left_join(stats_guess[,c('id', 'mean', 'q2.5', "q97.5")], by = c('True_Spec

set.seed(12121)

nb.cols <- 50
sp_guess_col <- colorRampPalette(brewer.pal(8, "Paired"))(nb.cols) #Set2

ggplot(s3_species, aes(x = prob, y = mean, col = True_Species )) +
  geom_point(aes(size = ns))+
  geom_errorbar(aes(ymin=q2.5, ymax=q97.5),
                width=0.005, alpha = 0.3,  #
                position=position_dodge(.005) ) +
  geom_text_repel(aes(label = True_Species), size = 2.6, segment.alpha = 0.3)+
  scale_color_manual(values = sp_guess_col, guide = 'none') +
  xlab('probability')+
  ylab('species guessing') +
  theme_bw() +
  xlim(0,1) +
  theme(plot.title = element_text(hjust = 0.5, size=8.5),
        legend.text=element_text(size=7.5),
        legend.title = element_blank(),
        legend.position="bottom",
        legend.spacing.x = unit(0.05, 'cm'))
```



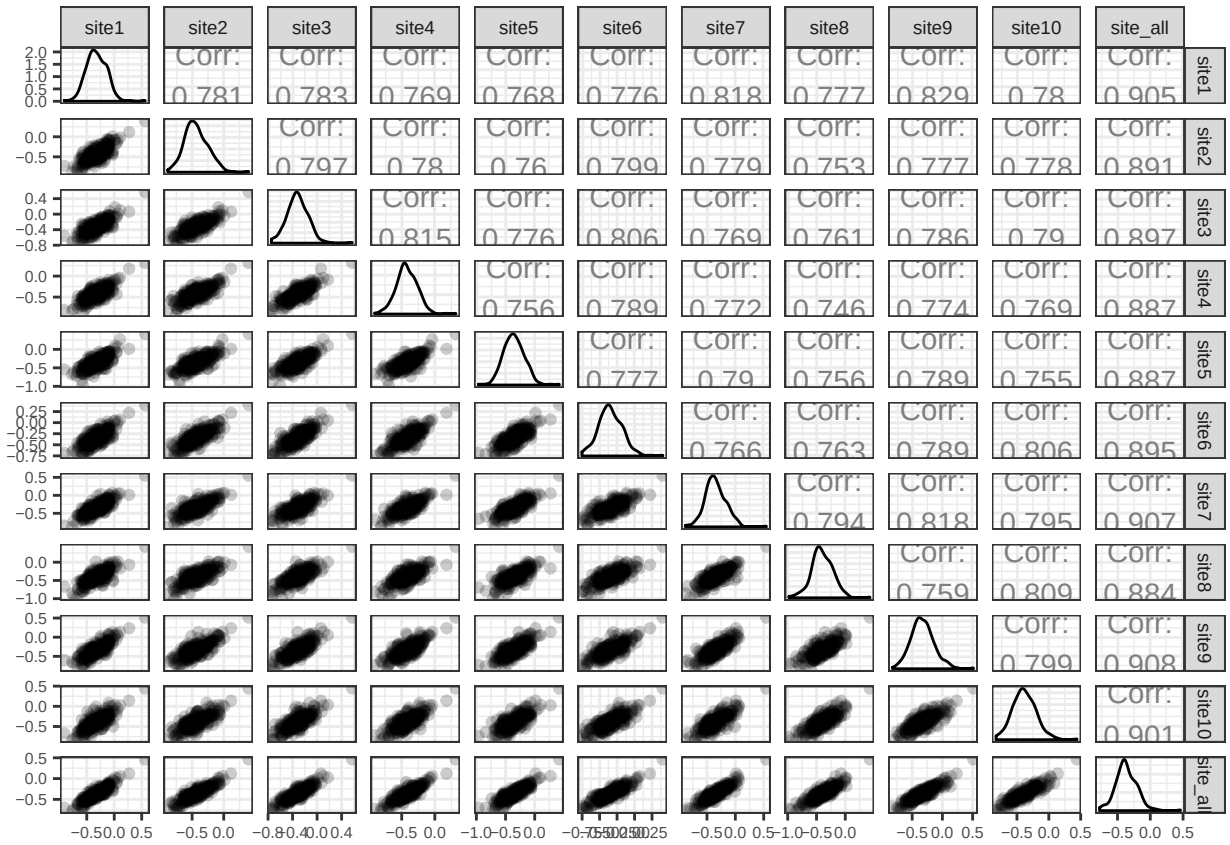
2.4 Site difficulties

```
s5_1$id <- as.numeric(as.factor(as.character(s5_1$SiteID)))
s5_2$id <- as.numeric(as.factor(as.character(s5_2$SiteID)))
s5_3$id <- as.numeric(as.factor(as.character(s5_3$SiteID)))
s5_4$id <- as.numeric(as.factor(as.character(s5_4$SiteID)))
s5_5$id <- as.numeric(as.factor(as.character(s5_5$SiteID)))
s5_6$id <- as.numeric(as.factor(as.character(s5_6$SiteID)))
s5_7$id <- as.numeric(as.factor(as.character(s5_7$SiteID)))
s5_8$id <- as.numeric(as.factor(as.character(s5_8$SiteID)))
s5_9$id <- as.numeric(as.factor(as.character(s5_9$SiteID)))
s5_10$id <- as.numeric(as.factor(as.character(s5_10$SiteID)))


site1 = data.frame(stats_seren_1[grep("difficulty\\[", row.names(stats_seren_1)),c(1,3)])
names(site1) <- c('site1','sd_1'); site1$site <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(s
site2 = data.frame(stats_seren_2[grep("difficulty\\[", row.names(stats_seren_2)),c(1,3)])
names(site2) <- c('site2','sd_2'); site2$site <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(s
site3 = data.frame(stats_seren_3[grep("difficulty\\[", row.names(stats_seren_3)),c(1,3)])
names(site3) <- c('site3','sd_3'); site3$site <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(s
site4 = data.frame(stats_seren_4[grep("difficulty\\[", row.names(stats_seren_4)),c(1,3)])
names(site4) <- c('site4','sd_4'); site4$site <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(s
site5 = data.frame(stats_seren_5[grep("difficulty\\[", row.names(stats_seren_5)),c(1,3)])
names(site5) <- c('site5','sd_5'); site5$site <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(s
```

```
site6 = data.frame(stats_seren_6[grep("difficulty\\[", row.names(stats_seren_6)),c(1,3)])
names(site6) <- c('site6','sd_6'); site6$site <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(s
site7 = data.frame(stats_seren_7[grep("difficulty\\[", row.names(stats_seren_7)),c(1,3)])
names(site7) <- c('site7','sd_7'); site7$site <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(s
site8 = data.frame(stats_seren_8[grep("difficulty\\[", row.names(stats_seren_8)),c(1,3)])
names(site8) <- c('site8','sd_8'); site8$site <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(s
site9 = data.frame(stats_seren_9[grep("difficulty\\[", row.names(stats_seren_9)),c(1,3)])
names(site9) <- c('site9','sd_9'); site9$site <- as.numeric(as.character(gsub("[^0-9.-]", "", rownames(s
site10 = data.frame(stats_seren_10[grep("difficulty\\[", row.names(stats_seren_10)),c(1,3)])
names(site10) <- c('site10','sd_10'); site10$site <- as.numeric(as.character(gsub("[^0-9.-]", "", rownar

site1 <- site1 %>% left_join( ( dplyr::select(s5_1, id, SiteID) %>% distinct() ), by = c('site' = 'id'))
site2 <- site2 %>% left_join( (dplyr::select(s5_2, id, SiteID) %>% distinct() ), by = c('site' = 'id'))
site3 <- site3 %>% left_join( (dplyr::select(s5_3, id, SiteID) %>% distinct() ), by = c('site' = 'id'))
site4 <- site4 %>% left_join( (dplyr::select(s5_4, id, SiteID) %>% distinct() ), by = c('site' = 'id'))
site5 <- site5 %>% left_join( (dplyr::select(s5_5, id, SiteID) %>% distinct() ), by = c('site' = 'id'))
site6 <- site6 %>% left_join( (dplyr::select(s5_6, id, SiteID) %>% distinct() ), by = c('site' = 'id'))
site7 <- site7 %>% left_join( (dplyr::select(s5_7, id, SiteID) %>% distinct() ), by = c('site' = 'id'))
site8 <- site8 %>% left_join( (dplyr::select(s5_8, id, SiteID) %>% distinct() ), by = c('site' = 'id'))
site9 <- site9 %>% left_join( (dplyr::select(s5_9, id, SiteID) %>% distinct() ), by = c('site' = 'id'))
site10 <- site10 %>% left_join( (dplyr::select(s5_10, id, SiteID) %>% distinct() ), by = c('site' = 'id

df_site <- join_all(list(site1, site2 ,site3, site4,site5,site6,site7,site8,site9, site10), by='SiteID'
remove <- which(names(df_site) == 'site')[2:length(which(names(df_site) == 'site'))]
df_site <- subset(df_site, select = -remove )

df_site <- df_site[,c("SiteID", names(df_site)[grep('site',names(df_site))], names(df_site)[grep('sd_',n
df_site$site <- NULL
```

```
W <- 1 / (df_site[,names(df_site)[grep('sd_',names(df_site))]] ^ 2)
names(W) <- paste0('W', gsub("[^0-9.-]", "", names(W)) )
df_site <- cbind(df_site, W)

df_site$site_all <- rowSums( W * df_site[,grep('site',
                                               names(df_site))], na.rm=T)/
  rowSums(W, na.rm=T)

df_site_melt <- df_site
df_site_melt$site_diff <- as.numeric(as.factor(as.character(df_site_melt$SiteID)))

df_site_melt <- df_site_melt[, c('SiteID' , names(df_site_melt)[grep('site',names(df_site_melt))]   )]

df_site_melt <- melt(df_site_melt, id.vars = c('SiteID','site_diff') )
```

```
ggpairs(data = df_site,
                                    columns = grep('site',names(df_site)) ,
                                    #mapping = aes(x = variable, y = value, alpha = 0.01)
                                    lower = list(continuous = wrap("points",
                                                                   alpha = 0.2))) +
  theme_bw() + theme(strip.text.x = element_text(size = 7),
                     strip.text.y = element_text(size = 7),
                     axis.text.x = element_text( size = 6),
                     axis.text.y = element_text( size = 6))
```



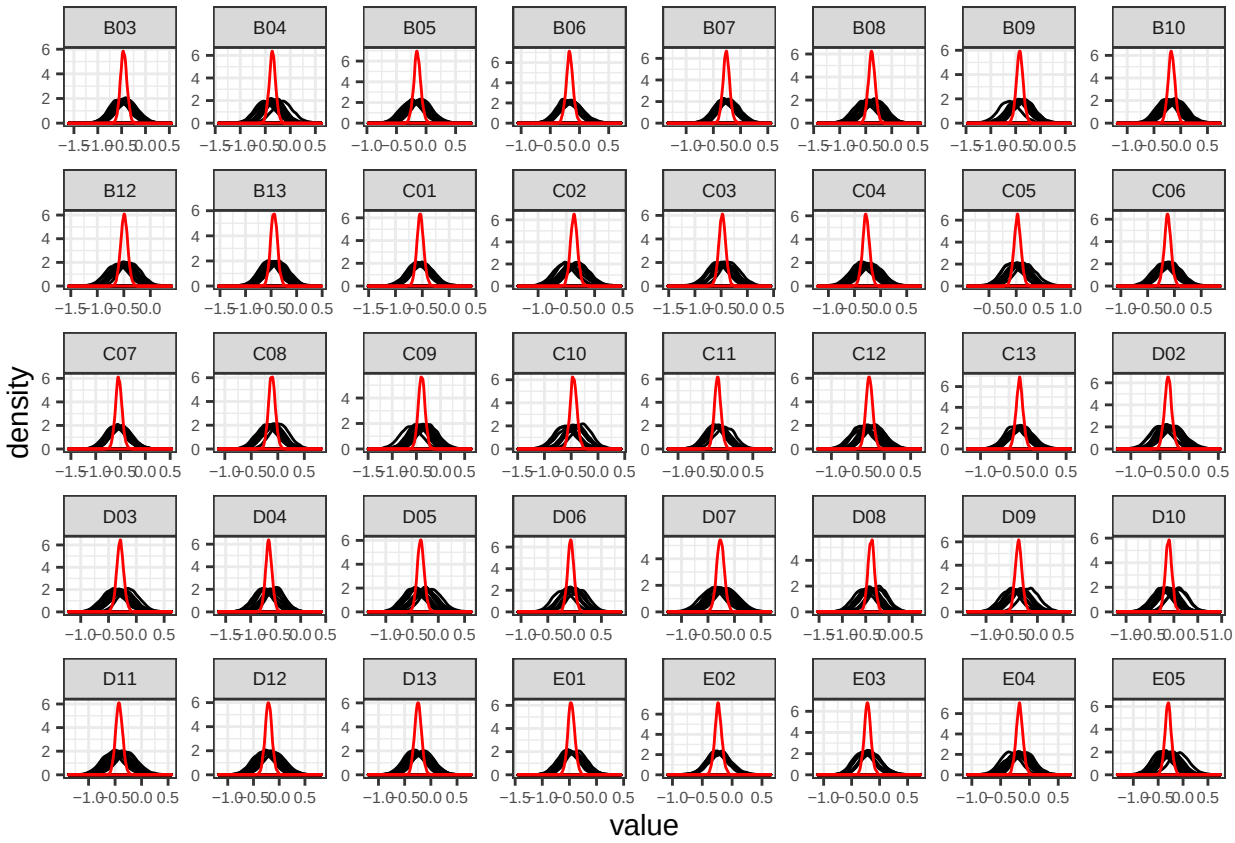

```
mcmc_site <- function(fit.mcmc.site, site, machine ){
  fit.mcmc.site <- do.call(rbind.data.frame, fit.mcmc.site)
  fit.mcmc.site <- melt(fit.mcmc.site)
  fit.mcmc.site$site <- as.numeric(as.character(gsub("[^0-9.-]", "", fit.mcmc.site$variable)))
  fit.mcmc.site <- fit.mcmc.site %>% left_join(site[,c('site', 'SiteID')],by= c('site'))
  fit.mcmc.site$machine = machine
  return(fit.mcmc.site)
}

site.sp_1 <- readRDS('Z://20191004serengllltm3PLUS_2_BIGDATA//fit.mcmc.site.rds')
site.sp_1 <- mcmc_site(fit.mcmc.site=site.sp_1, site=site1, machine = 1 )

site.sp_2 <- readRDS('Z://20191004serengllltm3PLUS_2_BIGDATA_2//fit.mcmc.site.rds')
site.sp_2 <- mcmc_site(fit.mcmc.site=site.sp_2, site=site2, machine = 2 )

site.sp_3 <- readRDS('Z://20191004serengllltm3PLUS_2_BIGDATA_3//fit.mcmc.site.rds')
site.sp_3 <- mcmc_site(fit.mcmc.site=site.sp_3, site=site3 , machine = 3 )

site.sp_4 <- readRDS('Z://20191004serengllltm3PLUS_2_BIGDATA_4//fit.mcmc.site.rds')
site.sp_4 <- mcmc_site(fit.mcmc.site=site.sp_4, site=site4, machine = 4  )

site.sp_5 <- readRDS('Z://20191004serengllltm3PLUS_2_BIGDATA_5//fit.mcmc.site.rds')
site.sp_5 <- mcmc_site(fit.mcmc.site=site.sp_5, site=site5, machine = 5  )

site.sp_6 <- readRDS('Z://20191004serengllltm3PLUS_2_BIGDATA_6//fit.mcmc.site.rds')
site.sp_6 <- mcmc_site(fit.mcmc.site=site.sp_6, site=site6, machine = 6  )
```

```
site.sp_7 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_7//fit.mcmc.site.rds')
site.sp_7 <- mcmc_site(fit.mcmc.site=site.sp_7, site=site7, machine = 7  )

site.sp_8 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_8//fit.mcmc.site.rds')
site.sp_8 <- mcmc_site(fit.mcmc.site=site.sp_8, site=site8, machine = 8  )

site.sp_9 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_9//fit.mcmc.site.rds')
site.sp_9 <- mcmc_site(fit.mcmc.site=site.sp_9, site=site9, machine = 9  )

site.sp_10 <- readRDS('Z://20191004serenglltm3PLUS_2_BIGDATA_10//fit.mcmc.site.rds')
site.sp_10 <- mcmc_site(fit.mcmc.site=site.sp_10, site=site10, machine = 10  )

site.sp_all <- rbind(site.sp_1, site.sp_2, site.sp_3, site.sp_4, site.sp_5, site.sp_6, site.sp_7, site.
```

```
M = 10 # machines
d = length(unique(site.sp_all$SiteID))# dimensions (species)
mach <- sort(unique(site.sp_all$machine)) # machines
ii=1
nas <- dplyr::filter(site.sp_all, machine == mach[ii]) %>% arrange(SiteID) %>%
  dplyr::group_by(SiteID) %>% dplyr::mutate(id = row_number()) %>%
  dplyr::select(site , SiteID, id, value) %>% mutate(value = NA) %>%
  mutate(site = NA)

theta <- NULL
for(ii in 1:length(mach)){
  m1 <- dplyr::filter(site.sp_all, machine == mach[ii]) %>% arrange(SiteID)
  m1 <- m1 %>% dplyr::group_by(SiteID) %>% dplyr::mutate(id = row_number())
  m1 <- m1 %>% dplyr::select(site , SiteID, id, value)

  if(ii==8){
    m1 <- rbind(m1, nas)
    m1 <- m1 %>% dplyr::group_by(SiteID, id) %>%
      dplyr::summarise(site = mean(site, na.rm=T) ,
                       value = mean(value, na.rm=T))
  }

  if(ii==10){
    m1 <- rbind(m1, nas)
    m1 <- m1 %>% dplyr::group_by(SiteID, id) %>%
      dplyr::summarise(site = mean(site, na.rm=T) ,
                       value = mean(value, na.rm=T))
  }

  nn <- dcast(m1, site + SiteID  ~ id )
  nn <- nn %>% arrange(SiteID)
  theta[[ii]] <- nn

}

subchain_0 <- array((unlist(theta)), dim=c(225, 7502, 10)) # 7502 = 3 chains x 2500 iter + 2 clumns sit
subchain <- subchain_0[,3:7502,]

W <- matrix(NA, d, M)
```

```
sampletotT <- dim(subchain)[2]
for (j in 1:d)
  for (s in 1:M)
    W[j, s] <- 1/var(subchain[j, , s], na.rm=T)
theta_ <- matrix(NA, nrow = d, ncol = sampletotT)
for (i in 1:sampletotT){
  theta_[, i] <- rowSums(W * as.numeric(as.character(subchain[,i, ])), dims = 1, na.rm=T)/rowSums(W, na
}
# adding as 11th dimension the consensus posterior
theta[[11]] <- cbind(theta[[1]][,1:2], theta_)

site_comb <- do.call(rbind.data.frame, theta)
site_comb$machine <- rep(1:11, each = 225)
site_comb_melt <- melt(site_comb, id.vars = c('site', 'SiteID', 'machine'))
```

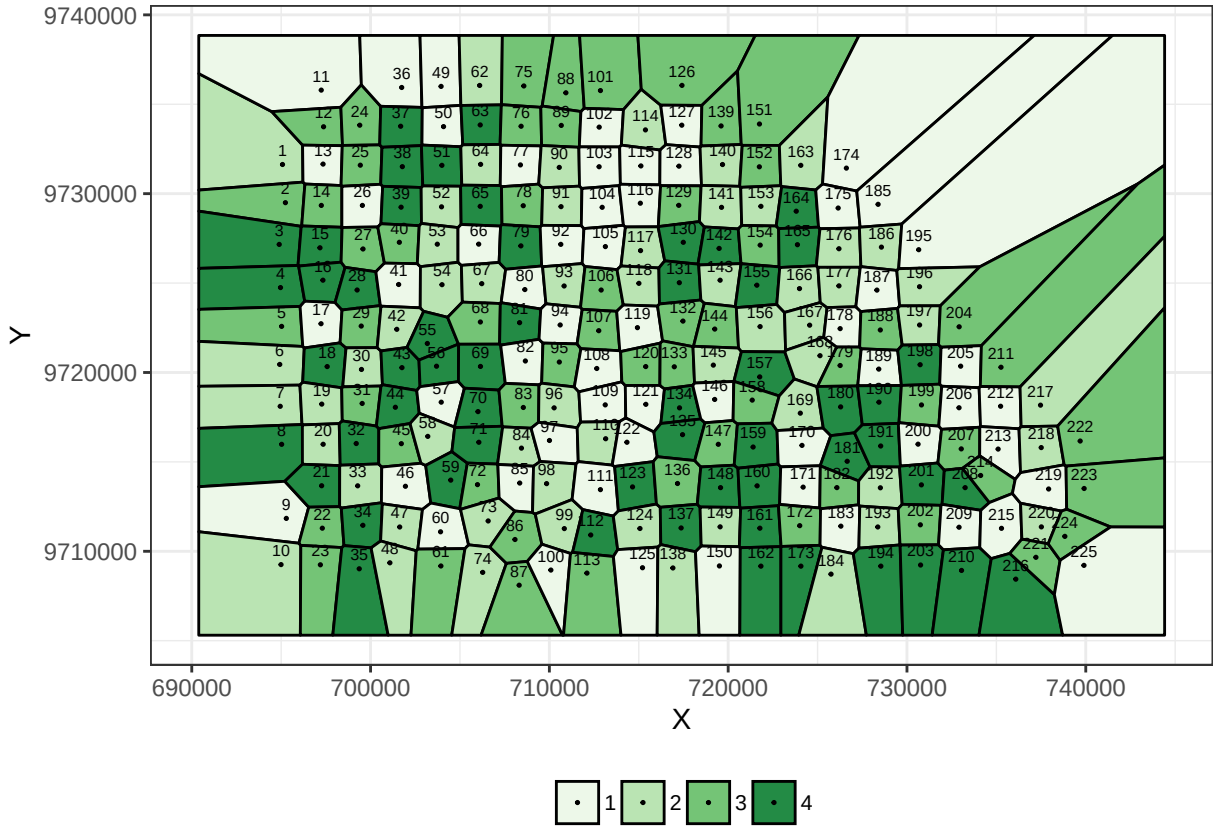
```
colsp <- c(rep(1,10), 2)
# will filter 40 sites
dplyr::filter(site_comb_melt, site %in% c(1:40)) %>%
ggplot(.) + geom_density(aes(x = value, col = factor(machine))) +
  scale_color_manual(values = colsp) +
  facet_wrap(~SiteID, scales = 'free', nrow = 5)+ theme_bw()+
  theme(strip.text.x = element_text(size = 7),
        strip.text.y = element_text(size = 7),
        plot.title = element_text(hjust = 0.5, size=8.5),
        legend.text=element_text(size=7.5),
        legend.title = element_blank(),
        legend.position="none",
        legend.spacing.x = unit(0.05, 'cm'),
        axis.text.x = element_text( size = 6),
        axis.text.y = element_text( size = 6)) +
  guides(fill=guide_legend(nrow=1,byrow=TRUE)
  )
```




```
stats_site <- site_comb_melt %>% dplyr::select( SiteID, value) %>%
  dplyr::group_by(SiteID) %>%
  dplyr::summarise(mean = mean(value, na.rm = T),
                   sd = sd(value, na.rm = T),
                   med = median(value, na.rm = T),
                    q2.5 = quantile(value, na.rm = T, .025),
                    q97.5 = quantile(value, na.rm = T, .975))
stats_site$param <- paste0('difficulty_',1:nrow(stats_site));  stats_site <- stats_site %>% dplyr::selec
```

```
s3_unique <- s5_sets %>%
    dplyr::group_by(id, SiteID, LocationX, LocationY) %>%
  dplyr::summarise(correct = mean(correct, na.rm=T)) #, site

s3_unique <-  s3_unique %>%
  left_join(stats_site[,c('id', 'mean', "sd")], by = c('SiteID' = 'id'))
names(s3_unique)[names(s3_unique)=='mean'] <- 'diff_pred'

s3_unique$diff_pred_cat <- cut(s3_unique$diff_pred, breaks = quantile(s3_unique$diff_pred), 1:4, include

cols = brewer.pal(4,'Greens')
ggplot(s3_unique , aes(LocationX, LocationY, fill = diff_pred_cat)) +
  stat_voronoi(color="black") +
  scale_fill_manual(values=cols) +
  geom_point(size = 0.25) +
  geom_text(aes(LocationX, LocationY+800, label = id), size = 2)+
```

```
  xlab('X')+
  ylab('Y')+
  coord_fixed(ratio=1)+ theme_bw() +
theme(plot.title = element_text(hjust = 0.5, size=8.5),
      legend.text=element_text(size=7.5),
      legend.title = element_blank(),
      legend.position="bottom",
      legend.spacing.x = unit(0.05, 'cm')) +
guides(fill=guide_legend(nrow=1,byrow=TRUE))
```



3 Appendix

3.1 functions.R file content

Functions to compute the adjacency matrix and edges.

```
adj_matrix <- function(LocationX, LocationY){
    nb2 <- dismo::voronoi(cbind(LocationX,LocationY))
    #from https://gis.stackexchange.com/questions/237810/adjacency-matrix-not-including-vertices
    polyids = seq_along(nb2)
    adjMat = matrix(FALSE, ncol = length(nb2), nrow = length(nb2))
    for (ii in polyids) {
      for (jj in setdiff(polyids, seq_len(ii))) {
```

```
        adjMat[ii, jj] = ifelse(class(gIntersection(nb2[ii, ], nb2[jj, ])) == 'SpatialLines',1,0)
      }
    }
    adjMat[lower.tri(adjMat)] = t(adjMat)[lower.tri(adjMat)]
return(adjMat)
}

# Codes written by Mitzi Morris
mungeCARdata4stan <- function(adjBUGS,numBUGS) {
#source: https://github.com/stan-dev/example-models/blob/master/knitr/car-iar-poisson/mungeCARdata4stan
 N = length(numBUGS);
  nn = numBUGS;
  N_edges = length(adjBUGS) / 2;
  node1 = vector(mode="numeric", length=N_edges);
  node2 = vector(mode="numeric", length=N_edges);
  iAdj = 0;
  iEdge = 0;
  for (i in 1:N) {
    for (j in 1:nn[i]) {
      iAdj = iAdj + 1;
      if (i < adjBUGS[iAdj]) {
        iEdge = iEdge + 1;
        node1[iEdge] = i;
        node2[iEdge] = adjBUGS[iAdj];
      }
    }
  }
  return (list("N"=N,"N_edges"=N_edges,"node1"=node1,"node2"=node2));
}
```