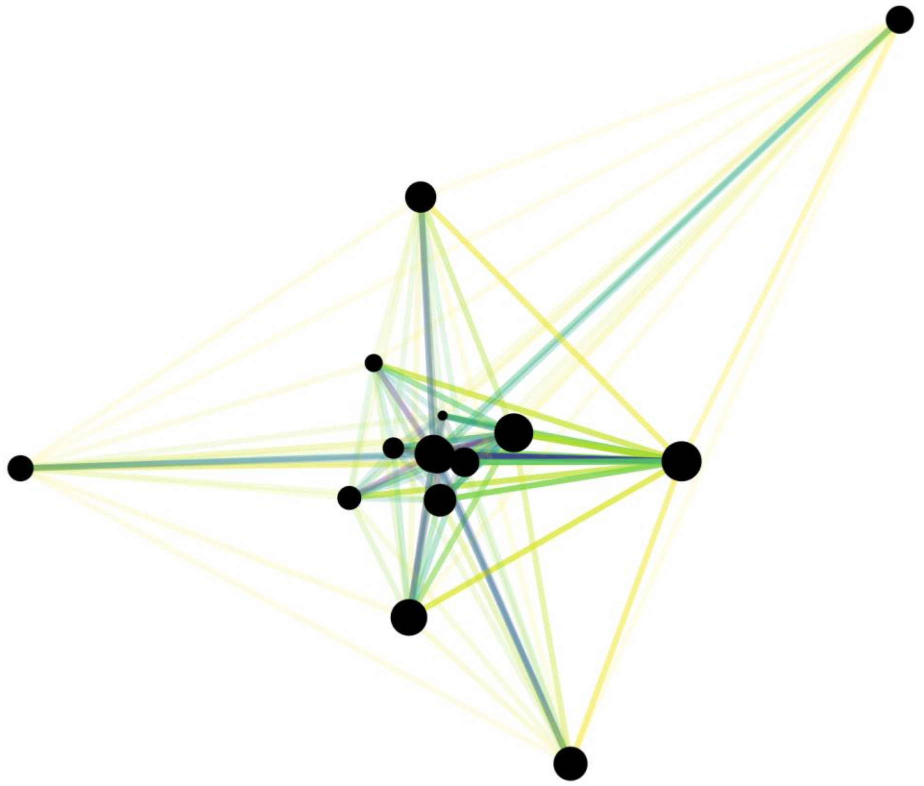
Introduction to STAN for dyadic edge regression models

Winter Workshop on complex systems 2025



Marco Fele - 2252927@swansea.ac.uk - Swansea University

We do

1. Highlight the potential of STAN
2. Introducing the basics of STAN
3. Mixture models and varying effects

Networks are often assumed as given, but should be considered as random realizations of underlying data generating processes (domain specific knowledge)

Propagate uncertainty from edges to network metrics

We do

1. Highlight the potential of STAN
2. Introducing the basics of STAN
3. Mixture models and varying effects

Networks are often assumed as given, but should be considered as random realizations of underlying data generating processes (domain specific knowledge)

Propagate uncertainty from edges to network metrics

We don't

Revise Bayesian statistics

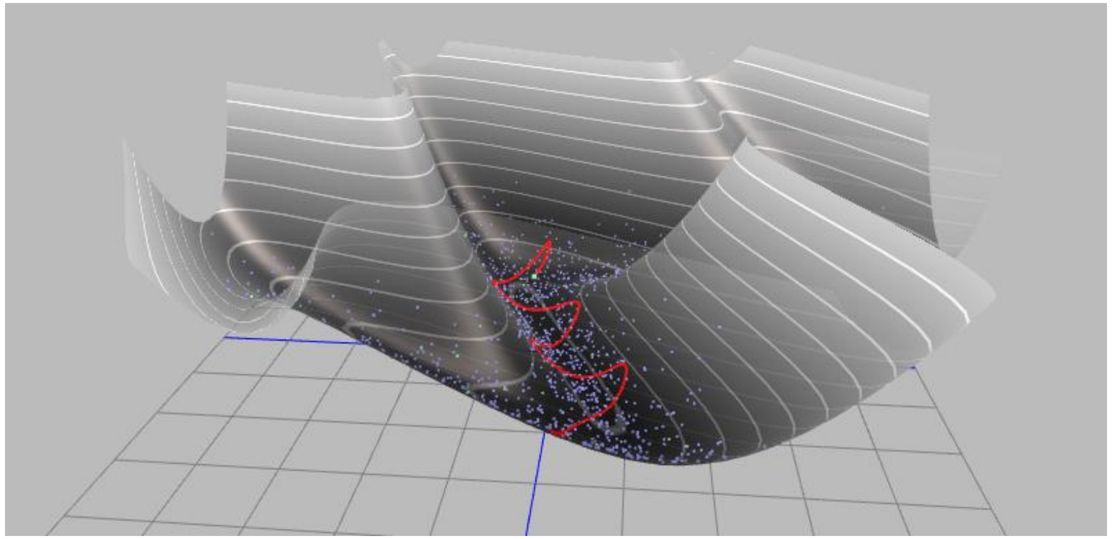
Bayesian workflow:

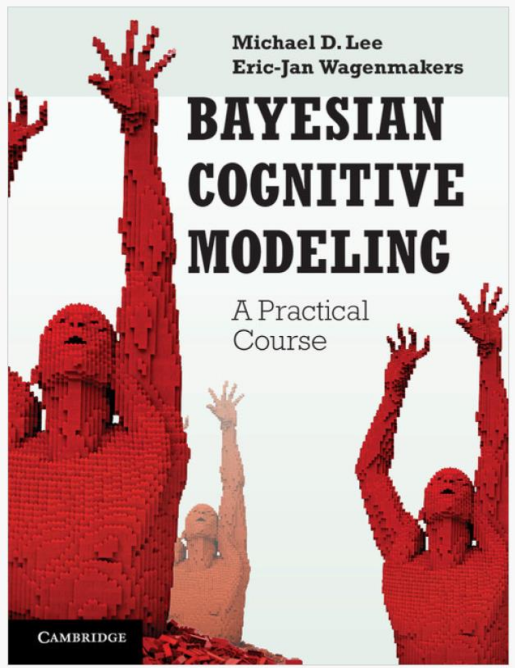
1. Prior and posterior predictive simulations
2. MCMC convergence diagnostics
3. Hypothesis testing

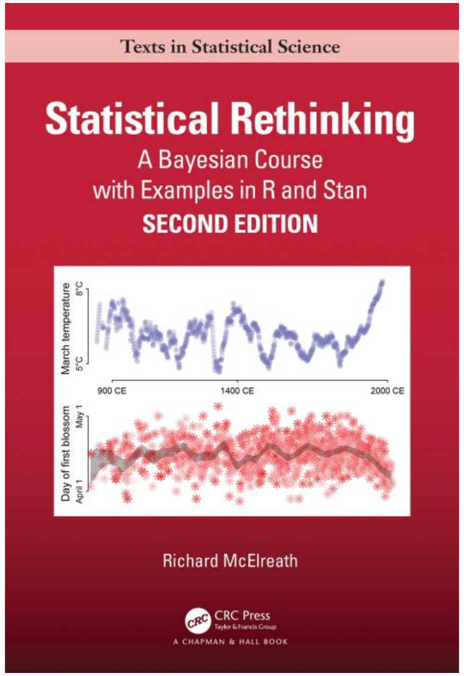
What is STAN?

Probabilistic programming language

Allows to fit a *wide* variety of statistical models through MCMC (Hamiltonian Monte Carlo NUTS)





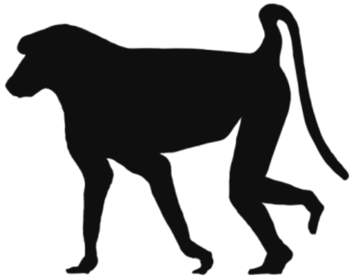


Download STAN:
https://mc-stan.org/install/

Learn STAN:
https://mc-stan.org/learn-stan/tutorials.html
https://mc-stan.org/learn-stan/case-studies.html
https://www.youtube.com/@rmcelreath

Research problem

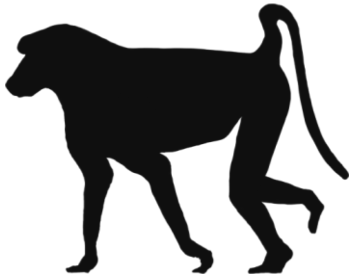
Data:
Troop of 15 baboons with dominance rank.
For 20 days, we know the number of hours two individuals are doing the same behaviour



Research problem

Data:
Troop of 15 baboons with dominance rank.
For 20 days, we know the number of hours two individuals are doing the same behaviour

Research questions:
1. What is the effect of dominance on pairwise synchrony, i.e., the probability a pair is performing the same behaviour?
2. What is the individual centrality in synchrony network?

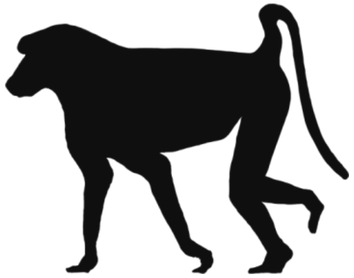


Research problem

Data:
Troop of 15 baboons with dominance rank.
For 20 days, we know the number of hours two individuals are doing the same behaviour.

Research questions:
1. What is the effect of dominance on pairwise synchrony, i.e., the probability a pair is performing the same behaviour?
2. What is the individual centrality in synchrony network?

Complications:
Different number of observations for every pair. Sometimes troop splits, but we do not know the number of subgroups, when the troop splits, and who goes where



Model

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

y_i: number of hours the pair was synchronized on observation i

λ_i: pair is together (1) or not (0)

α_1: synchrony when pair is separated

α_2: individual baseline synchrony

β: effect of dominance

$ID_{k,j}$: effect of individual k and j involved in observation i (ID is a vector)

θ: probability the pair is together (*mixture probability*)

Model

Mixture model: more than one data generating process

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

https://mc-stan.org/docs/stan-users-guide/finite-mixtures.html

y_i: number of hours the pair was synchronized on observation i

λ_i: pair is together (1) or not (0)

α_1: synchrony when pair is separated

α_2: individual baseline synchrony

β: effect of dominance

$ID_{k,j}$: effect of individual k and j involved in observation i (ID is a vector)

θ: probability the pair is together (*mixture probability*)

Model

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + \boxed{ID_{k[i]} + ID_{j[i]}} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

Dyadic regression: control for non-independency of edges belonging to a node by including a node effect (in *causal inference* terminology: close backdoor path)

Hart et al., 2023

y_i: number of hours the pair was synchronized on observation i

λ_i: pair is together (1) or not (0)

α_1: synchrony when pair is separated

α_2: individual baseline synchrony

β: effect of dominance

$ID_{k,j}$: effect of individual k and j involved in observation i (ID is a vector)

θ: probability the pair is together (*mixture probability*)

STAN implementation

STAN works by defining the model through "blocks" and finding the "posterior surface" by defining log likelihood of each observation

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

```
data {
  ?
}

parameters {
  ?
}

model {
  ?
}
```

Data block

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

Which variables are data?

Data block

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i)binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

```
data {
  int<lower=0> N; // number of data points
  int<lower=0> N_ind; // number of individuals
  int<lower=0> n_hours_same[N]; // number of hours in which both individuals are performing the same behaviour
  int<lower=0> id_1[N]; // first individual of edge
  int<lower=0> id_2[N]; // second individual of edge
  vector<lower=0,upper=1>[N] dominance_1; // dominance frst individual
  vector<lower=0,upper=1>[N] dominance_2; // dominance second individual
}
```

STAN is explicitly typed compiled language which requires variable declaration and definition
For more on data types: https://mc-stan.org/docs/reference-manual/types.html

Data block

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

```
data {
  int<lower=0> N; // number of data points
  int<lower=0> N_ind; // number of individuals
  int<lower=0> n_hours_same[N]; // number of hours in which both individuals are performing the same behaviour
  int<lower=0> id_1[N]; // first individual of edge
  int<lower=0> id_2[N]; // second individual of edge
  vector<lower=0,upper=1>[N] dominance_1; // dominance frst individual
  vector<lower=0,upper=1>[N] dominance_2; // dominance second individual
}
```

Integer arrays

Real vectors supports linear algebra operations

STAN is explicitly typed compiled language which requires variable declaration and definition
For more on data types: https://mc-stan.org/docs/reference-manual/types.html

Data block

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i)binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

```
data {
  int<lower=0> N; // number of data points
  int<lower=0> N_ind; // number of individuals
  int<lower=0> n_hours_same[N]; // number of hours in which both individuals are performing the same behaviour
  int<lower=0> id_1[N]; // first individual of edge
  int<lower=0> id_2[N]; // second individual of edge
  vector<lower=0,upper=1>[N] dominance_1; // dominance frst individual
  vector<lower=0,upper=1>[N] dominance_2; // dominance second individual
}
```

```
transformed data {
  // Centralize predictors
  vector[N] dominance_1_c = dominance_1 - mean(dominance_1);
  vector[N] dominance_2_c = dominance_2 - mean(dominance_2);
}
```

- Centralizing predictors increases sampling performance
- Response is a random variable and as such should not be centralized
- Data blocks evaluated once

Parameters block

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

Which variables are the parameters?

Parameters block

Parameters → "unknown quantities *not derived by other quantities*"

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

Every variable has a posterior distribution
Propagation of uncertainty from parameters

```
parameters {
  // intercepts
  ordered[2] alpha;
  // fixed effect
  real beta;
  // hyperparameters
  real<lower=0> sigma_id;
  // mixture mixing parameter
  real<lower=0,upper=1> theta;
  // random effects
  vector[N_ind] id_z;
}
```

Parameters block

Parameters → "unknown quantities *not derived by other quantities*"

$$y_i \sim binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim bernulli(\theta)$$

```
parameters {
  // intercepts
  ordered[2] alpha;
  // fixed effect
  real beta;
  // hyperparameters
  real<lower=0> sigma_id;
  // mixture mixing parameter
  real<lower=0,upper=1> theta;
  // random effects
  vector[N_ind] id_z;
}
```

Missing values can be imputed by including them in the parameters block
https://www.youtube.com/watch?v=Oeq6GChHOzc

ordered is a data type to solve non-identifiability of mixture models (and HMMs)

It would be possible to swap values of p_i and α_1, and change $\theta \rightarrow 1 - \theta$ https://betanalpha.github.io/assets/case_studies/identifying_mixture_models.html

Model block

$$y_i \sim Binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i)Binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim Bernulli(\theta)$$

Model block

$$y_i \sim Binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i)Binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim Bernulli(\theta)$$

Priors

$$\theta \sim Beta(4,4)$$

$$\beta \sim Normal(0,5)$$

$$\alpha_1 \sim Normal(0,5)$$

- Priors' distribution depend on parameters' domain
- Priors' parameters are chosen through prior predictive simulations and domain-specific knowledge
- Wise choice of priors' parameters allow *regularization*

Model block

$$y_i \sim Binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i)Binomial(n = 24, \alpha_1)$$

$$logit(p_i) = 2\alpha_2 + ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim Bernulli(\theta)$$

Priors

$$\theta \sim Beta(4,4)$$

$$\beta \sim Normal(0,5)$$

$$\alpha_1 \sim Normal(0,5)$$

Varying effects

$$ID \sim Normal(\alpha_2, \sigma_{ID})$$

Hyperparameters

$$\alpha_2 \sim Normal(0,5)$$

$$\sigma_{ID} \sim Exponential(0.1)$$

- Priors' distribution depend on parameters' domain
- Priors' parameters are chosen through prior predictive simulations and domain-specific knowledge
- Wise choice of priors' parameters allow *regularization*
- Varying effects consist in letting the data inform the parameters of prior → automatic regularization

Model block

Non-centered parametrization

$$ID \sim Normal(\alpha_2, \sigma_{ID})$$

$$\alpha_2 \sim Normal(0,5)$$

$$\sigma_{ID} \sim Exponential(0.1)$$

$\longrightarrow$

$$ID.z \sim Normal(0,1)$$

$$ID = ID.z + \alpha_2 * \sigma_{ID}$$

$$\alpha_2 \sim Normal(0,5)$$

$$\sigma_{ID} \sim Exponential(0.1)$$

Non-centered parametrization increases sampling performance

Non-centered parametrization for multidimensional normal distributions more complicated (involves Cholesky decomposition) but *very* necessary
https://www.youtube.com/watch?v=DPnLb5EaCkA&t=3977s

Find the log-likelihood of each observation

Model block executed each time gradient is computed

```
transformed parameters { // Back-transform centered parametrization.
  vector[N_ind] transformed_id = sigma_id * id_z + alpha[2];
}

model {
  // Priors
  // fixed effect
  alpha[1] ~ normal(-1, 5);
  beta ~ normal(0, 5);
  // Hyperparameters
  alpha[2] ~ normal(0, 5);
  sigma_id ~ exponential(0.1);
  // mixture mixing probability
  theta ~ beta(4, 4);
  // random effects
  target += normal_lpdf(id_z | 0, 1);

  // Model
  for(i in 1:N) { // for every observation
    real p = transformed_id[id_1[i]] + transformed_id[id_2[i]] + beta * (dominance_1_c[i] + dominance_2_c[i]);
    target += log_sum_exp( // marginalize likelihood.
    log(theta) + binomial_logit_lpmf(n_hours_same[i] | 24, p), // "process" 1 : the pair is in the same location
    log1m(theta) + binomial_logit_lpmf(n_hours_same[i] | 24, alpha[1]) // "process" 2 : the pair is in a different location
    );
  }
}
```

$$y_i \sim Binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) Binomial(n = 24, \alpha_1)$$

$$logit(p_i) = ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim Bernulli(\theta)$$

$$\theta \sim Beta(4,4)$$

$$\beta \sim Normal(0,5)$$

$$\alpha_1 \sim Normal(-1,5)$$

$$\alpha_2 \sim Normal(0,5)$$

$$ID.z \sim Normal(0,1)$$

$$ID = ID.z + \alpha_2 * \sigma_{ID}$$

$$\alpha_2 \sim Normal(0,5)$$

$$\sigma_{ID} \sim Exponential(0.1)$$

Find the log-likelihood of each observation

```
transformed parameters { // Back-transform centered parametrization.
  vector[N_ind] transformed_id = sigma_id * id_z + alpha[2];
}

model {
  // Priors
  // fixed effect
  alpha[1] ~ normal(-1, 5);
  beta ~ normal(0, 5);
  // Hyperparameters
  alpha[2] ~ normal(0, 5);
  sigma_id ~ exponential(0.1);
  // mixture mixing probability
  theta ~ beta(4, 4);
  // random effects
  target += normal_lpdf(id_z | 0, 1);

  // Model
  for(i in 1:N) { // for every observation
    real p = transformed_id[id_1[i]] + transformed_id[id_2[i]] + beta * (dominance_1_c[i] + dominance_2_c[i]);
    target += log_sum_exp( // marginalize likelihood.
    log(theta) + binomial_logit_lpmf(n_hours_same[i] | 24, p), // "process" 1 : the pair is in the same location
    log1m(theta) + binomial_logit_lpmf(n_hours_same[i] | 24, alpha[1]) // "process" 2 : the pair is in a different location
    );
  }
}
```

To let the data inform priors of varying effects likelihood must be specified

$$y_i \sim Binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i) Binomial(n = 24, \alpha_1)$$

$$logit(p_i) = ID_{k[i]} + ID_{j[i]} + \beta\left(d_{k[i]} + d_{j[i]}\right)$$

$$\lambda_i \sim Bernulli(\theta)$$

$$\theta \sim Beta(4,4)$$

$$\beta \sim Normal(0,5)$$

$$\alpha_1 \sim Normal(-1,5)$$

$$\alpha_2 \sim Normal(0,5)$$

$$ID.z \sim Normal(0,1)$$

$$ID = ID.z + \alpha_2 * \sigma_{ID}$$

$$\alpha_2 \sim Normal(0,5)$$

$$\sigma_{ID} \sim Exponential(0.1)$$

Find the log-likelihood of each observation

```
transformed parameters { // Back-transform centered parametrization.
  vector[N_ind] transformed_id = sigma_id * id_z + alpha[2];
}

model {
  // Priors
  // fixed effect
  alpha[1] ~ normal(-1, 5);
  beta ~ normal(0, 5);
  // Hyperparameters
  alpha[2] ~ normal(0, 5);
  sigma_id ~ exponential(0.1);
  // mixture mixing probability
  theta ~ beta(4, 4);
  // random effects
  target += normal_lpdf(id_z | 0, 1);

  // Model
  for(i in 1:N) { // for every observation
    real p = transformed_id[id_1[i]] + transformed_id[id_2[i]] + beta * (dominance_1_c[i] + dominance_2_c[i]);
    target += log_sum_exp( // marginalize likelihood.
    log(theta) + binomial_logit_lpmf(n_hours_same[i] | 24, p), // "process" 1 : the pair is in the same location
    log1m(theta) + binomial_logit_lpmf(n_hours_same[i] | 24, alpha[1]) // "process" 2 : the pair is in a different location
    );
  }
}
```

Priors consistent with
the ordered type

$$y_i \sim Binomial(n=24, p_i)(\lambda_i) + (1-\lambda_i) Binomial(n=24, \alpha_1)$$
$$logit(p_i) = ID_{k[i]} + ID_{j[i]} + \beta(d_{k[i]} + d_{j[i]})$$
$$\lambda_i \sim Bernulli(\theta)$$
$$\theta \sim Beta(4,4)$$
$$\beta \sim Normal(0,5)$$
$$\alpha_1 \sim Normal(-1,5)$$
$$\alpha_2 \sim Normal(0,5)$$
$$ID.z \sim Normal(0,1)$$
$$ID = ID.z + \alpha_2 * \sigma_{ID}$$
$$\alpha_2 \sim Normal(0,5)$$
$$\sigma_{ID} \sim Exponential(0.1)$$

Find the log-likelihood of each observation

```
transformed parameters { // Back-transform centered parametrization.
  vector[N_ind] transformed_id = sigma_id * id_z + alpha[2];
}

model {
  // Priors
  // fixed effect
  alpha[1] ~ normal(-1, 5);
  beta ~ normal(0, 5);
  // Hyperparameters
  alpha[2] ~ normal(0, 5);
  sigma_id ~ exponential(0.1);
  // mixture mixing probability
  theta ~ beta(4, 4);
  // random effects
  target += normal_lpdf(id_z | 0, 1);

  // Model
  for(i in 1:N) { // for every observation
    real p = transformed_id[id_1[i]] + transformed_id[id_2[i]] + beta * (dominance_1_c[i] + dominance_2_c[i]);
    target += log_sum_exp( // marginalize likelihood.
    log(theta) + binomial_logit_lpmf(n_hours_same[i] | 24, p), // "process" 1 : the pair is in the same location
    log1m(theta) + binomial_logit_lpmf(n_hours_same[i] | 24, alpha[1]) // "process" 2 : the pair is in a different location
    );
  }
}
```

To find likelihood of mixture model marginalize over data generating processes

$$y_i \sim Binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i)Binomial(n = 24, \alpha_1)$$

$$logit(p_i) = ID_{k[i]} + ID_{j[i]} + \beta(d_{k[i]} + d_{j[i]})$$

$$\lambda_i \sim Bernulli(\theta)$$

$$\theta \sim Beta(4,4)$$

$$\beta \sim Normal(0,5)$$

$$\alpha_1 \sim Normal(-1,5)$$

$$\alpha_2 \sim Normal(0,5)$$

$$ID.z \sim Normal(0,1)$$

$$ID = ID.z + \alpha_2 * \sigma_{ID}$$

$$\alpha_2 \sim Normal(0,5)$$

$$\sigma_{ID} \sim Exponential(0.1)$$

Find the log-likelihood of each observation

```
transformed parameters { // Back-transform centered parametrization.
  vector[N_ind] transformed_id = sigma_id * id_z + alpha[2];
}

model {
  // Priors
  // fixed effect
  alpha[1] ~ normal(-1, 5);
  beta ~ normal(0, 5);
  // Hyperparameters
  alpha[2] ~ normal(0, 5);
  sigma_id ~ exponential(0.1);
  // mixture mixing probability
  theta ~ beta(4, 4);
  // random effects
  target += normal_lpdf(id_z | 0, 1);

  // Model
  for(i in 1:N) { // for every observation
    real p = transformed_id[id_1[i]] + transformed_id[id_2[i]] + beta * (dominance_1_c[i] + dominance_2_c[i]);
    target += log_sum_exp( // marginalize likelihood.
    log(theta) + binomial_logit_lpmf(n_hours_same[i] | 24, p), // "process" 1 : the pair is in the same location
    log1m(theta) + binomial_logit_lpmf(n_hours_same[i] | 24, alpha[1]) // "process" 2 : the pair is in a different location
    );
  }
}
```

Computationally stable:
log_sum_exp() → sum probabilities
log1m() → 1 - probability

$$y_i \sim Binomial(n = 24, p_i)(\lambda_i) + (1 - \lambda_i)Binomial(n = 24, \alpha_1)$$
$$logit(p_i) = ID_{k[i]} + ID_{j[i]} + \beta(d_{k[i]} + d_{j[i]})$$
$$\lambda_i \sim Bernulli(\theta)$$
$$\theta \sim Beta(4,4)$$
$$\beta \sim Normal(0,5)$$
$$\alpha_1 \sim Normal(-1,5)$$
$$\alpha_2 \sim Normal(0,5)$$
$$ID.z \sim Normal(0,1)$$
$$ID = ID.z + \alpha_2 * \sigma_{ID}$$
$$\alpha_2 \sim Normal(0,5)$$
$$\sigma_{ID} \sim Exponential(0.1)$$

Generated quantities block

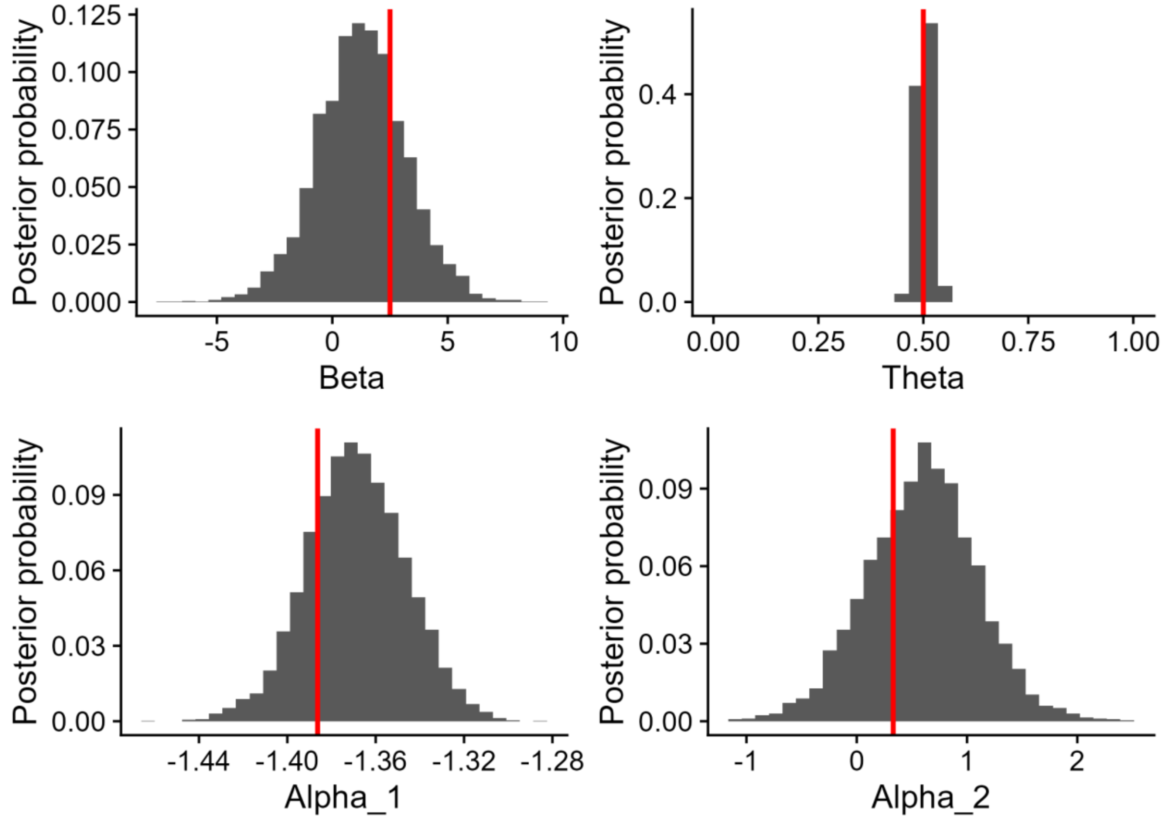
Block executed for each MCMC sample

```
generated quantities {
  // Classify observation as same or different location
  vector[N] prob_same;
  for(i in 1:N) { // the mixture probability of a data point is the likelihood mulitplied by the mixture probability normalized over the mixtures
    real p = transformed_id[id_1[i]] + transformed_id[id_2[i]] + beta * (dominance_1_c[i] + dominance_2_c[i]);
    real mixture_1 = binomial_logit_lpmf(n_hours_same[i] | 24, p) + log(theta);
    real mixture_2 = binomial_logit_lpmf(n_hours_same[i] | 24, alpha[1]) + log1m(theta);
    prob_same[i] = exp(mixture_1 - log_sum_exp(mixture_1, mixture_2));
  }
  // Generate edge posterior
  matrix[N_ind, N_ind] edges;
  for(ind_1 in 1:N_ind) {
    for(ind_2 in 1:N_ind) {
      edges[ind_1, ind_2] = inv_logit(transformed_id[ind_1] + transformed_id[ind_2] + beta * (dominance_1_c[ind_1] + dominance_2_c[ind_2]));
    }
  }
  // Generate centrality posterior
  vector[N_ind] centrality;
  for(ind_1 in 1:N_ind) {
    centrality[ind_1] = 0;
    for(ind_2 in 1:N_ind) {
      centrality[ind_1] += edges[ind_1, ind_2];
    }
  }
  centrality /= N_ind;
}
```

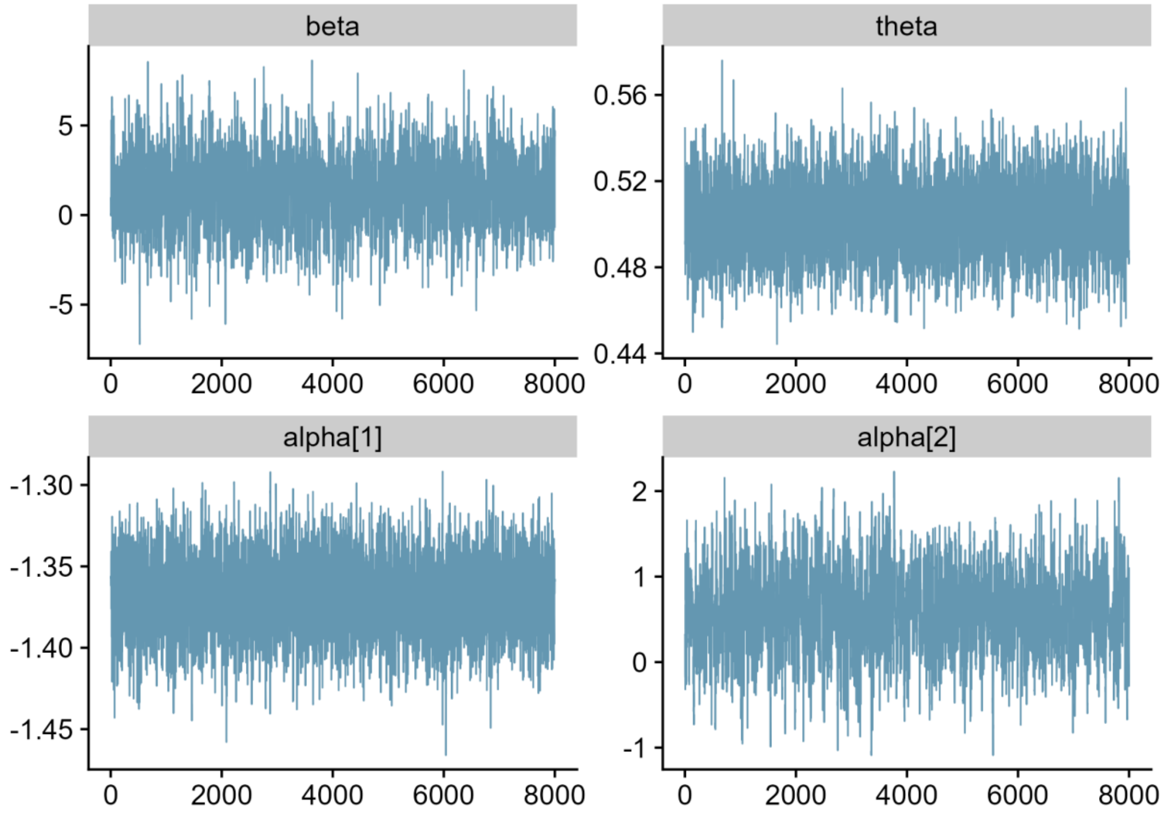
Mixture models perform unsupervised classification

To find the posterior for a data point belonging to a specific data generating process, multiply the likelihood times the mixture probability and normalize over the data generating processes

Results

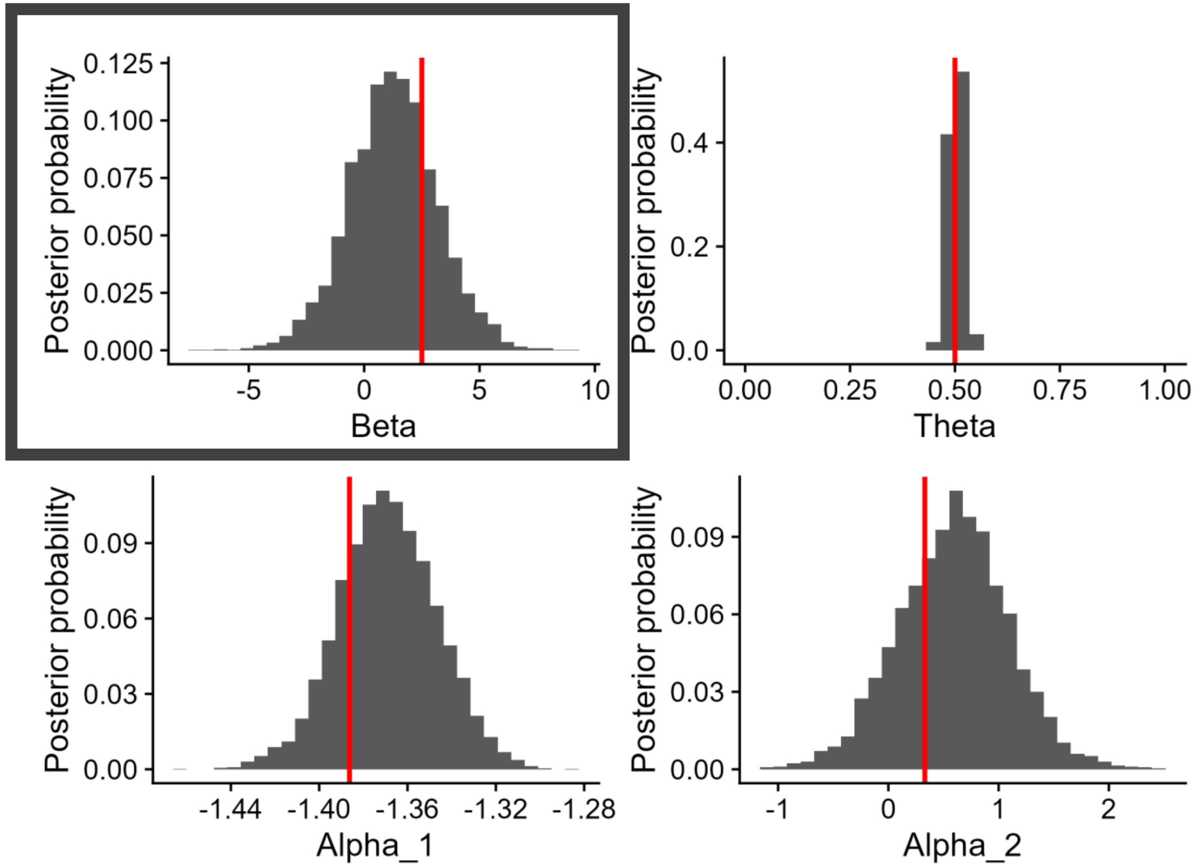


Parameters used to generate data

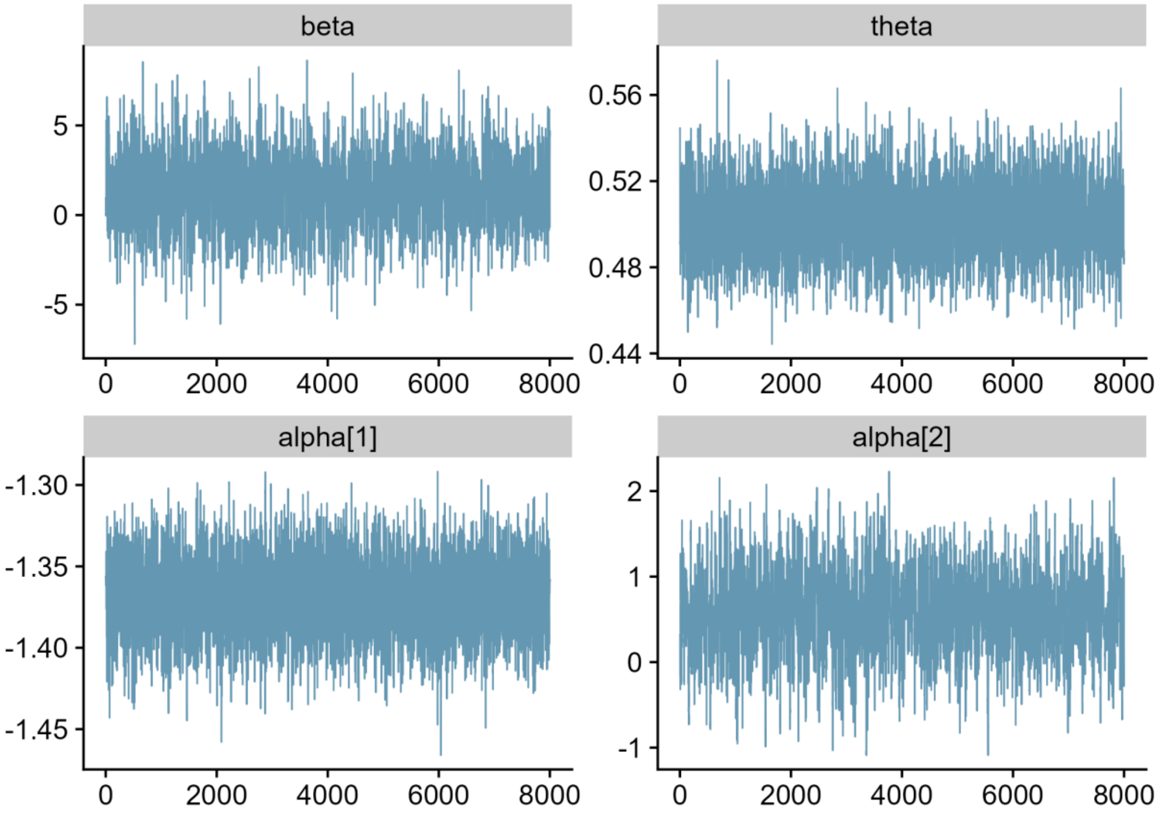


MCMC diagnostics

What is the effect of dominance on pairwise synchrony?

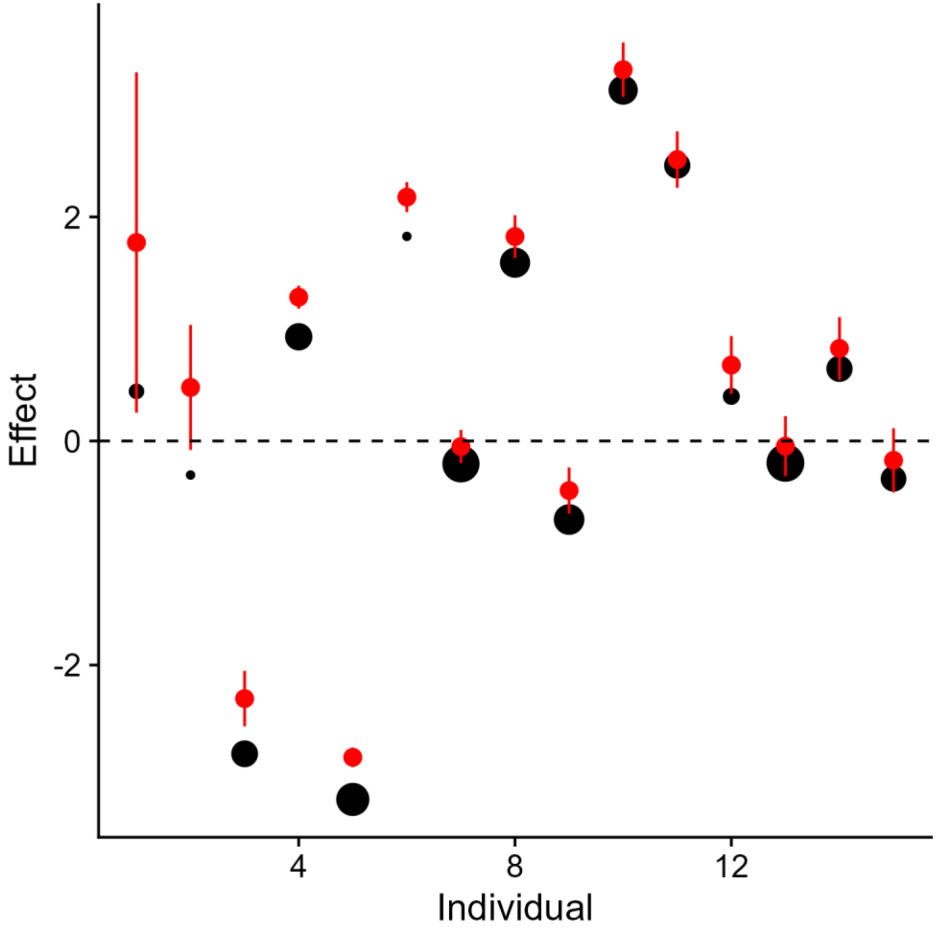


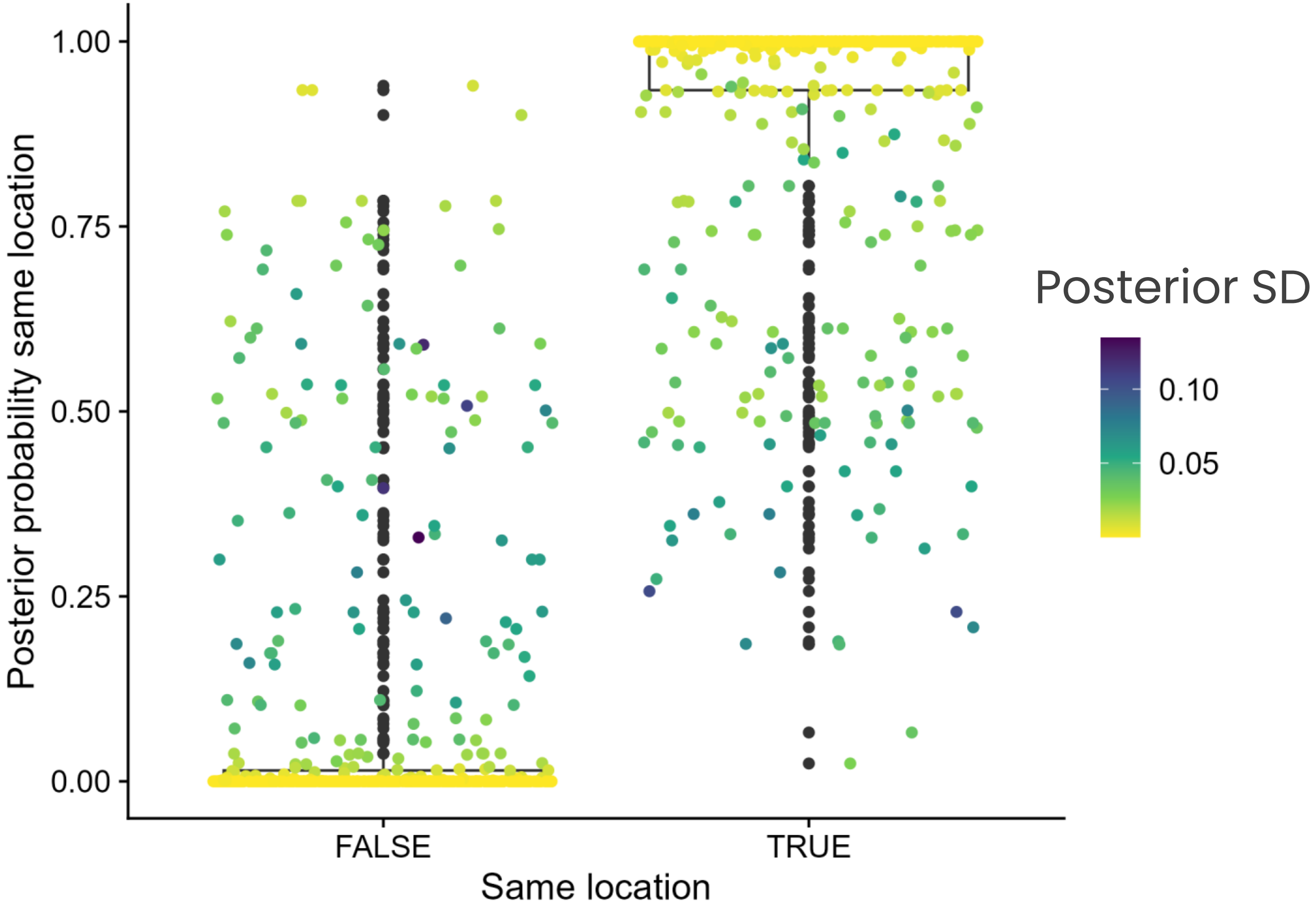
Parameters used to generate data



MCMC diagnostics

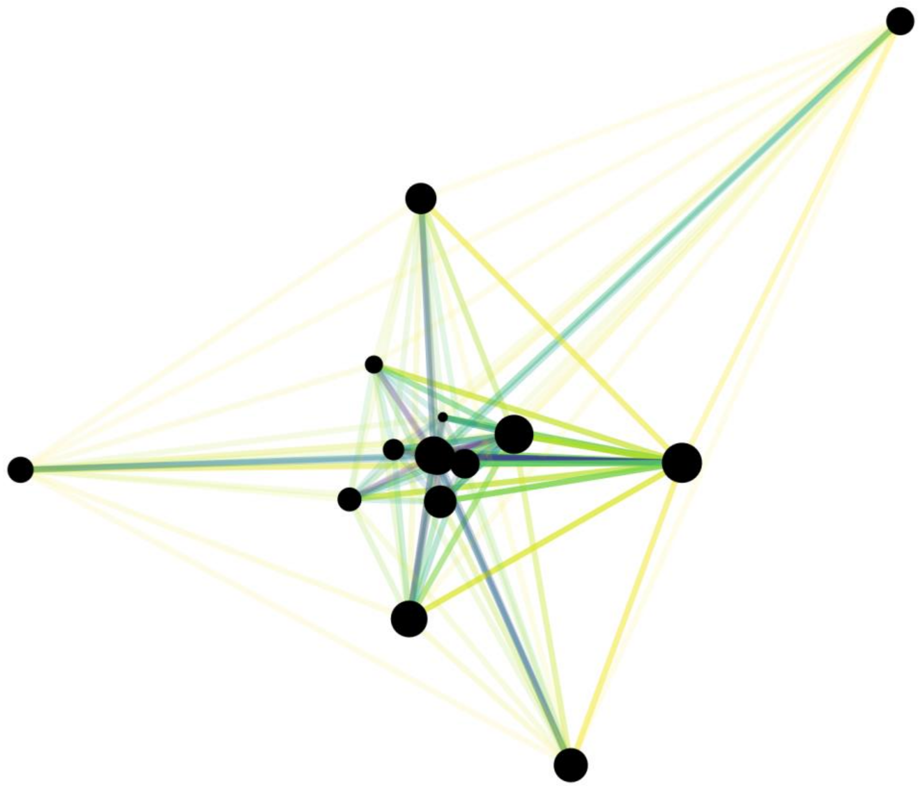
Results



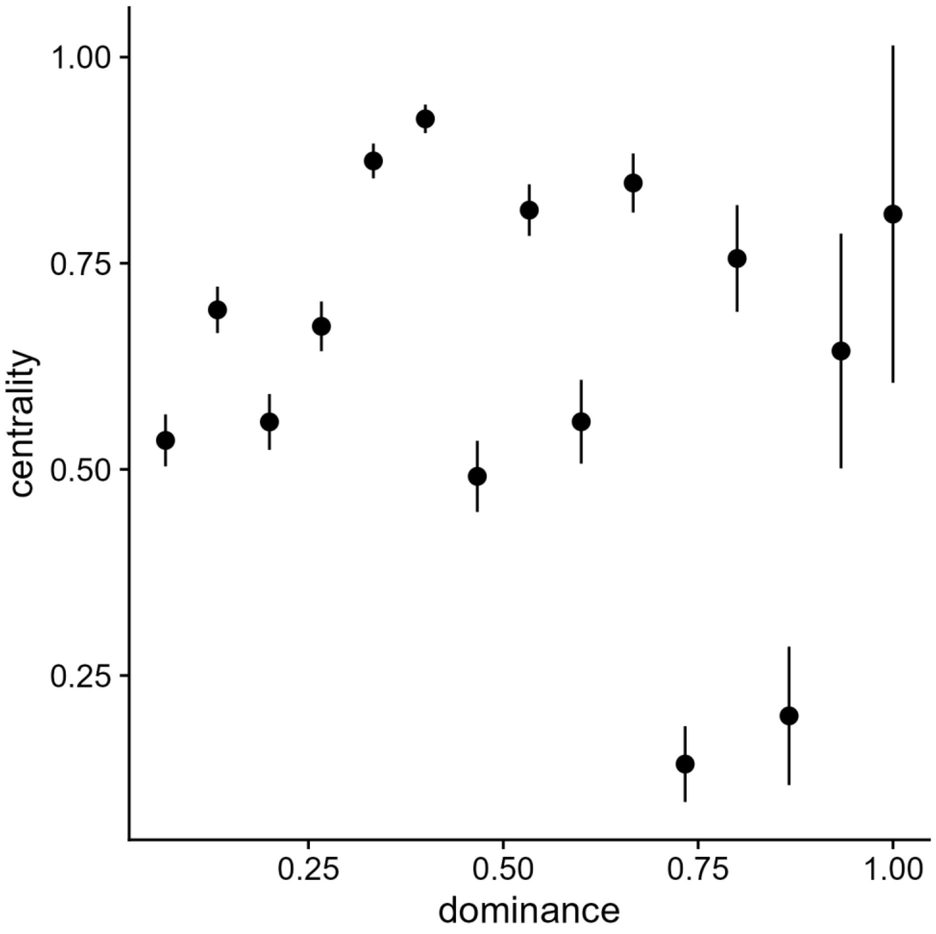


Successful unsupervised classification
Prediction is derived by parameters, uncertainty propagated

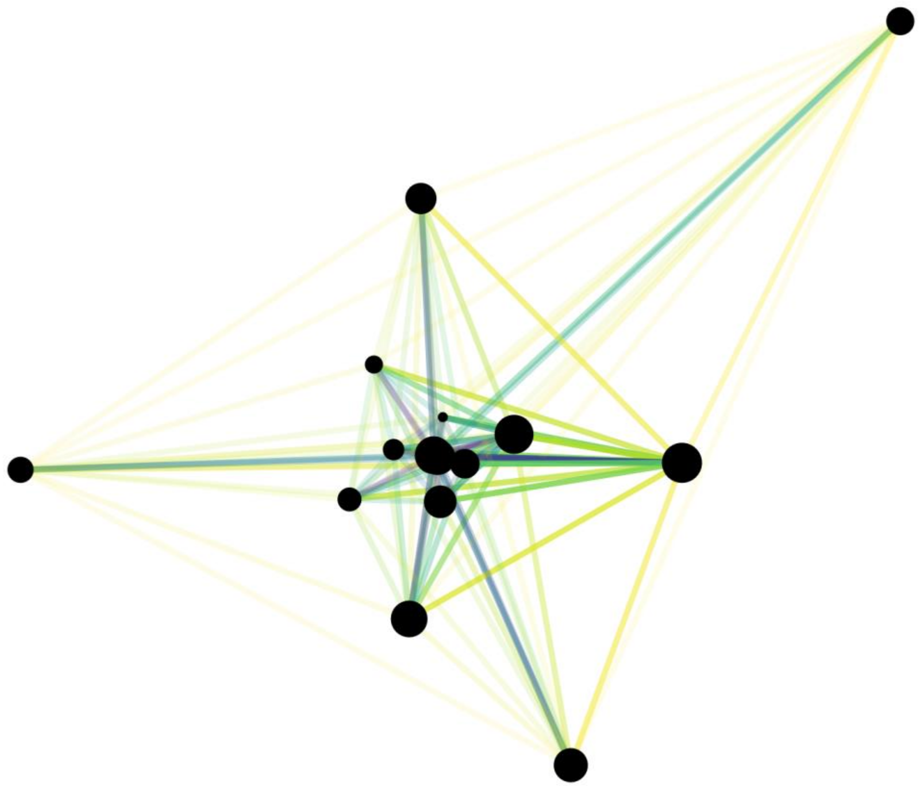
Results



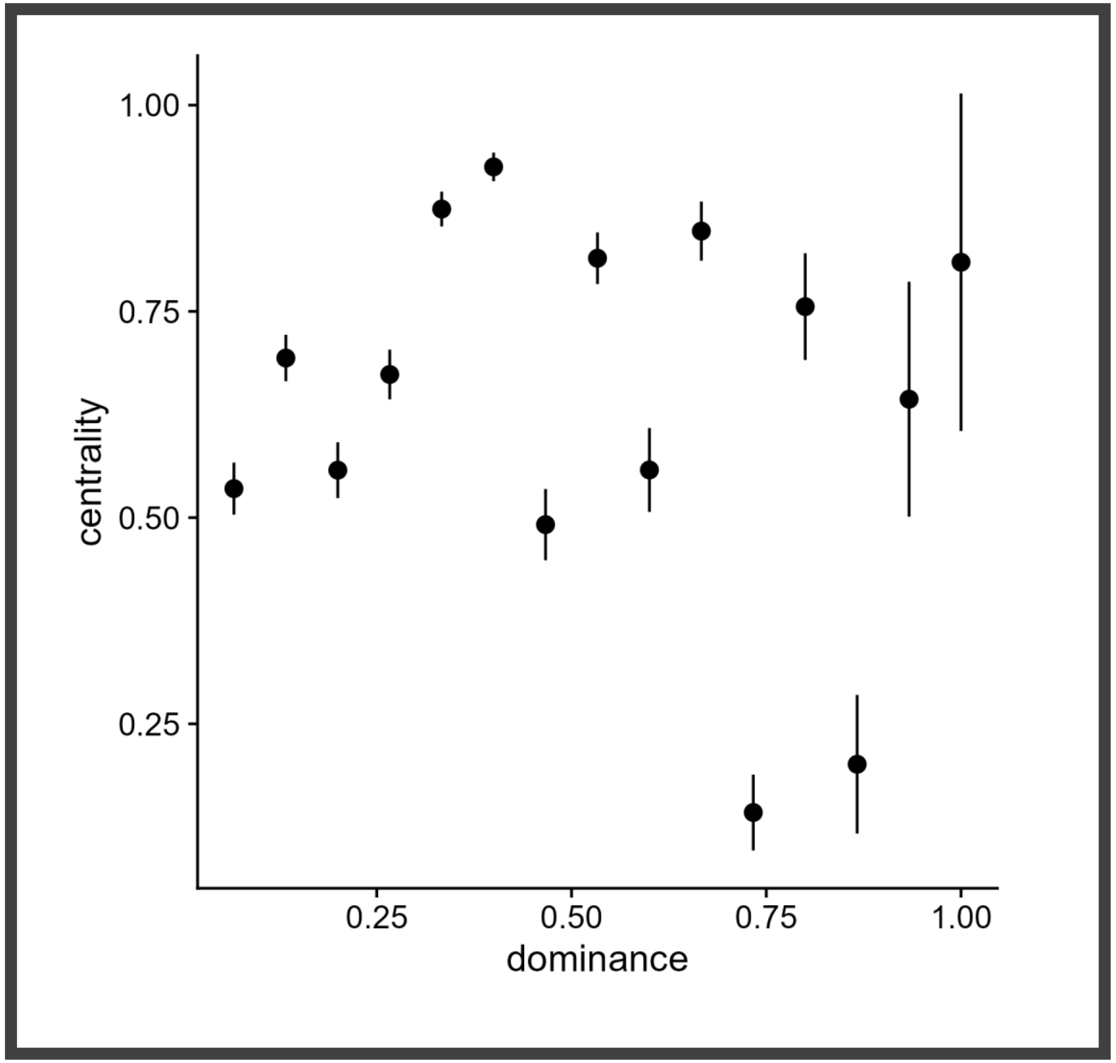
Color is posterior median
Transparency is posterior SD
Node size is dominance



What is the individual centrality in synchrony network?



Color is posterior median
Transparency is posterior sd
Node size is dominance



Uncertainty propagated from edges to node metrics

Examples of other STAN usage

Reinforcement learning

eLife RESEARCH ARTICLE

Risk-sensitive learning is a winning strategy for leading an urban invasion

Alexis J Breen[1*†], Dominik Deffner[2,3*]

Hidden Markov Models

nature communications

Article https://doi.org/10.1038/s41467-024-47010-3

Collective incentives reduce over-exploitation of social information in unconstrained human groups

Received: 23 August 2023

Accepted: 18 March 2024

Dominik Deffner[1,2], David Mezey[2,3], Benjamin Kahl[1], Alexander Schakowski[1], Pawel Romanczuk[2,3], Charley M. Wu[1,4,5] & Ralf H. J. M. Kurvers[1,2]

Gaussian processes

Inference in social networks

Methods in Ecology and Evolution Open Access BRITISH ECOLOGICAL SOCIETY

RESEARCH ARTICLE

BISoN: A Bayesian framework for inference of social networks

Jordan Hart[1] | Michael Nash Weiss[2,3] | Daniel Franks[4,5] | Lauren Brent[6]