Appendix 3: Estimating species interactions

Inferring species interactions from co-occurrence data with Markov networks

David J. Harris

This document describes how the different models were fit to the simulated data from Appendix 2 and how each model's performance was evaluated.[1]

Note that the `pairs` program was run separately (outside of R) with the following options:

- Batch mode
- Sequential swap ("s")
- Printing all pairs ("y")
- C-score co-occurrence measure ("c")
- Default confidence limits (0.05)
- Default iterations (100)
- Maximum of 20 species

Initialization:

```r
mc.cores = 8

library(dplyr)        # For manipulating data structures
library(corpcor)      # For regularized partial covariances
library(rosalia)      # For Markov networks
library(arm)          # For regularized logistic regression
library(BayesComm)    # For joint species distribution modeling
library(parallel)     # for mclapply
set.seed(1)
```

A function to import the data file and run each method on it:

```r
fit_all = function(identifier){
  ######## Import ########

  data_filename = paste0("fakedata/matrices/", identifier, ".csv")
  truth_filename = paste0("fakedata/truths/", identifier, ".txt")

  # first column is row numbers; drop it
  raw_obs = as.matrix(read.csv(data_filename)[ , -1])

  # Identify species that are never present (or never absent) so they
```

[1] The PDF version of this document has been manually altered to omit 150 lines of output from the `corpcor` package of the form "`## Estimating optimal shrinkage intensity lambda (correlation matrix): 0.0326`"

```r
# can be dropped
species_is_variable = diag(var(raw_obs)) > 0
pair_is_variable = tcrossprod(species_is_variable) > 0

x = raw_obs[ , species_is_variable]
truth = unlist(read.table(truth_filename))[pair_is_variable[upper.tri(pair_is_variable)]]

splitname = strsplit(identifier, "/|-|\\.")[[1]]
n_sites = as.integer(splitname[[1]])
rep_name = splitname[[2]]

# Species IDs
sp1 = combn(colnames(x), 2)[1, ]
sp2 = combn(colnames(x), 2)[2, ]

######## Partial correlations ########
p_corr = pcor.shrink(x)

######## Correlations ########
corr = cor(x)
######## GLM ########
coef_matrix = matrix(0, ncol(x), ncol(x))
for(i in 1:ncol(x)){
  if(var(x[,i]) > 0){
    coefs = coef(bayesglm(x[,i] ~ x[ , -i], family = binomial))[-1]
    coef_matrix[i, -i] = coefs
  }
}
coef_matrix = (coef_matrix + t(coef_matrix)) / 2


######## Markov network ########
rosie = rosalia(x, maxit = 200, trace = 0, prior = make_logistic_prior(scale = 2), hessian =

rosie_point = rosie$beta[upper.tri(rosie$beta)]
rosie_se = sqrt(diag(solve(rosie$opt$hessian)))[-(1:sum(species_is_variable))]



######## BayesComm and partial BayesComm ########
bc = BC(Y = x, model = "community", its = 1000)

bc_pcors = sapply(
  1:nrow(bc$trace$R),
  function(i){
    Sigma = matrix(0, nrow = ncol(x), ncol = ncol(x))
    Sigma[upper.tri(Sigma)] <- bc$trace$R[i, ]  # Fill in upper triangle
    Sigma <- Sigma + t(Sigma)                   # Fill in lower triangle
    diag(Sigma) <- 1  # Diagonal equals 1 in multivariate probit model
    pcor = cor2pcor(Sigma)
    pcor[upper.tri(pcor)]
  }
)
```

```r
bind_rows(
  data_frame(
    rep_name = rep_name,
    n_sites = n_sites,
    sp1 = sp1,
    sp2 = sp2,
    method = "Markov network",
    truth = truth,
    estimate = rosie_point,
    lower = qnorm(.025, rosie_point, rosie_se),
    upper = qnorm(.975, rosie_point, rosie_se)
  ),
  data_frame(
    rep_name = rep_name,
    n_sites = n_sites,
    sp1 = sp1,
    sp2 = sp2,
    method = "GLM",
    truth = truth,
    estimate = coef_matrix[upper.tri(coef_matrix)],
    lower = NA,
    upper = NA
  ),
  data_frame(
    rep_name = rep_name,
    n_sites = n_sites,
    sp1 = sp1,
    sp2 = sp2,
    method = "correlation",
    truth = truth,
    estimate = corr[upper.tri(corr)],
    lower = NA,
    upper = NA
  ),
  data_frame(
    rep_name = rep_name,
      n_sites = n_sites,
      sp1 = sp1,
      sp2 = sp2,
      method = "partial correlation",
      truth = truth,
      estimate = p_corr[upper.tri(p_corr)],
      lower = NA,
      upper = NA
    ),
    data_frame(
      rep_name = rep_name,
      n_sites = n_sites,
      sp1 = sp1,
      sp2 = sp2,
      method = "partial BayesComm",
      truth = truth,
      estimate = rowMeans(bc_pcors),
      lower = apply(bc_pcors, 1, quantile, .025),
      upper = apply(bc_pcors, 1, quantile, .975)
    ),
    data_frame(
      rep_name = rep_name,
      n_sites = n_sites,
      sp1 = sp1,
      sp2 = sp2,
      method = "BayesComm",
      truth = truth,
      estimate = colMeans(bc$trace$R),
      lower = apply(bc$trace$R, 2, quantile, .025),
      upper = apply(bc$trace$R, 2, quantile, .975)
    )
  )
}
```

Run the above function on all the files:

```r
# Find all the csv files in the fakedata/matrices folder,
# then drop .csv
identifiers = dir("fakedata/matrices", pattern = "\\.csv$") %>%
  gsub("\\.csv$", "", .)

mclapply(identifiers, fit_all, mc.cores = mc.cores, mc.preschedule = FALSE, mc.silent = TRUE) 
  bind_rows() %>%
  write.csv(file = "estimates.csv")
```