Appendix 3: Estimating species interactions

"Estimating species interactions from observational 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]

Initialization:

```
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(RColorBrewer) # For color palette
set.seed(1)
```

Load in the results from the `pairs` program, run 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

```
pairs_txt = readLines("fakedata/Pairs.txt")

# Find areas of the data file that correspond
# to species pairs' results
beginnings = grep("Sp1", pairs_txt) + 1
ends = c(
  grep("^[^ ]", pairs_txt)[-1],
  length(pairs_txt)
) - 1

# The above code fails on the very last line of the file
ends[length(ends)] = ends[length(ends)] + 1
```

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

[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`"

```
fit_all = function(filename){
  ######## Import ########

  # Multiplying by one is necessary to prevent silly errors
  # regarding TRUE/FALSE versus 1/0
  raw_obs = readRDS(filename)[["observed"]] * 1

  # Identify species that are never present (or never absent) so they
  # 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 = readRDS(filename)[["truth"]][pair_is_variable[upper.tri(pair_is_variable)]]

  splitname = strsplit(filename, "/|-|\\.")[[1]]
  n_sites = as.integer(splitname[[3]])
  rep = as.integer(splitname[[4]])


  ######## 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))

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

  `partial BayesComm` = 0

  for(i in 1:nrow(bc$trace$R)){
    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

  `partial BayesComm` = `partial BayesComm` + cor2pcor(Sigma) / nrow(bc$trace$R)
}

######## Pairs ########
# Find the line where the current data set is mentioned in
# pairs.txt
filename_line = grep(
  paste0(
    gsub("fakedata/(.*)\\.rds", "\\1", filename),
    "-"
  ),
  pairs_txt
)

# Which chunk of the data file corresponds to this file?
chunk = min(which(beginnings > filename_line))

# Split the chunk on whitespace.
splitted = strsplit(pairs_txt[beginnings[chunk]:ends[chunk]], " +")

# Pull out the species numbers and their Z-scores
pairs_results = lapply(
  splitted,
  function(x){
    spp = sort(as.integer(x[3:4]))
    data.frame(
      sp1 = spp[1],
      sp2 = spp[2],
      z = as.numeric(x[14])
    )
  }
) %>%
  bind_rows %>%
  mutate(spp = paste(sp1, sp2, sep = "-"))

# Re-order the pairs_results to match the other methods
m = matrix(NA, ncol(x), ncol(x))
new_order = match(
  paste(row(m)[upper.tri(m)], col(m)[upper.tri(m)], sep = "-"),
  pairs_results$spp
)
ordered_pairs_results = pairs_results[new_order, ]

######## Output ########
```

```r
  data.frame(
    truth = truth,
    n_sites = n_sites,
    rep = rep,
    sp1 = ordered_pairs_results$sp1,
    sp2 = ordered_pairs_results$sp2,
    `partial correlation` = p_corr[upper.tri(p_corr)],
    correlation = corr[upper.tri(corr)],
    `Markov network` = rosie$beta[upper.tri(rosie$beta)],
    GLM = coef_matrix[upper.tri(coef_matrix)],
    `BayesComm` = colMeans(bc$trace$R),
    `partial BayesComm` = `partial BayesComm`[upper.tri(`partial BayesComm`)],
    Pairs = ordered_pairs_results$z
  )
}
```

Run the above function on all the files:

```r
# Find all the .rds files in the fakedata folder
files = dir("fakedata", pattern = "\\.rds$", full.names = TRUE)

# Run all the analyses on all the files
z = lapply(files, fit_all) %>% bind_rows %>% as.data.frame
```

```r
# Fix a formatting issue with the column names
colnames(z) = gsub("\\.", " ", colnames(z))
```

Summarize the results of all 7 methods across the simulated landscapes:

```r
# Calculate residuals for each method
resids = sapply(
  colnames(z)[-(1:5)],
  function(i){resid(lm(z$truth ~ z[,i] + 0))}
)

sizes = sort(unique(z$n_sites))

# Compute proportion of variance explained, compared with a null that
# assumes all species interactions are 0.
results = as.data.frame(
  t(
    sapply(
      sizes,
      function(n){
        total_ss = mean(resid(lm(truth ~ 0, data = z))[z$n_sites == n]^2)
        1 - colMeans(resids[z$n_sites == n , ]^2) / total_ss
```

```
      }
    )
  )
)
# Sort the results in decreasing order
results = results[order(colMeans(results), decreasing = TRUE)]
```

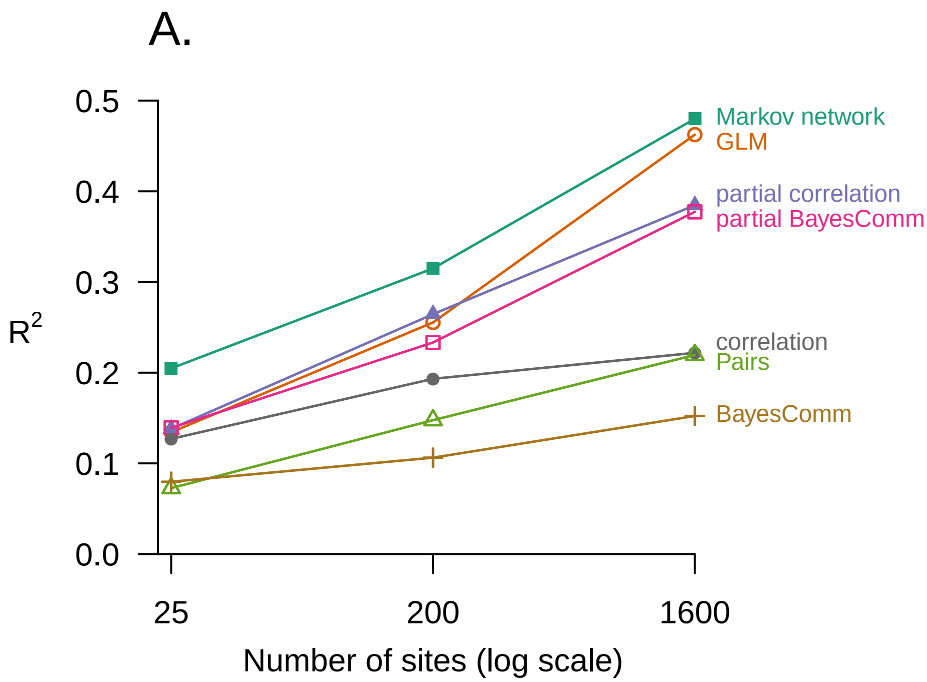
Plot the results:

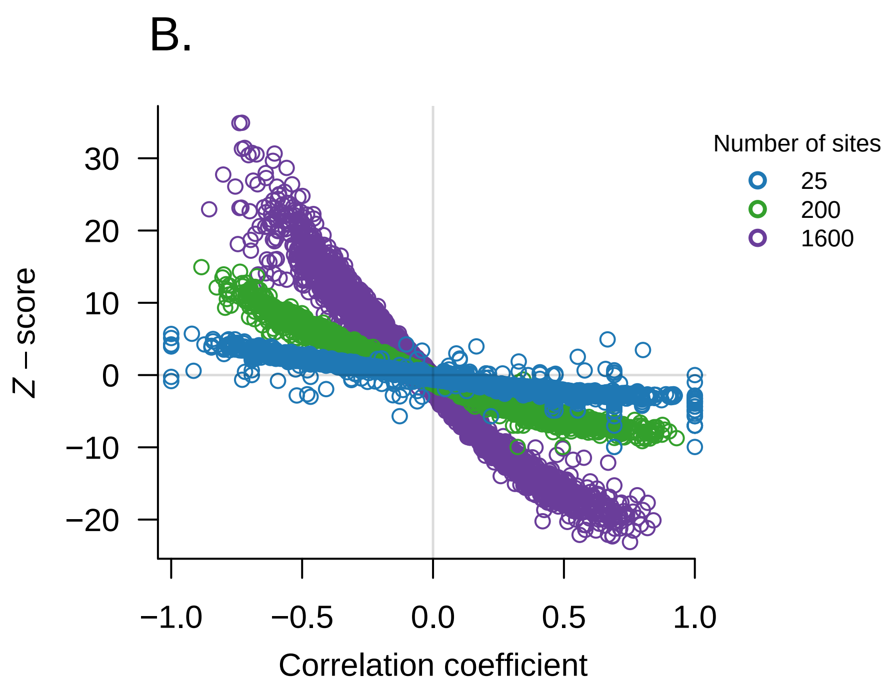
```
colors = brewer.pal(8, "Dark2")[c(1, 2, 3, 4, 8, 5, 7)]

# Set up the plotting canvas
pdf("manuscript-materials/figures/performance.pdf", height = 8, width = 5)
par(mfrow = c(2, 1))
par(mar = c(5, 4, 3, 0) + .1)

# Plot the model performance
matplot(
  sizes,
  results,
  type = "o",
  ylab = "",
  xlim = c(25 * .9, 1600 * 8),
  ylim = c(0, .5 + 1E-9),
  pch = c(15, 1, 17, 0, 16, 2, 3),
  lty = 1,
  log = "x",
  xlab = "",
  axes = FALSE,
  xaxs = "i",
  yaxs = "i",
  col = colors,
  lwd = 1.25,
  cex = 0.9,
  bty = "l"
)
axis(1, c(1, 25, 200, 1600))
axis(2, seq(0, 1, .1), las = 1)

mtext(expression(R^2), side = 2, line = 3, las = 1)
mtext("Number of sites (log scale)", side = 1, line = 2.25, at = 200)
mtext("A.", line = 1.25, at = 25, cex = 1.5)

# Determine how high on the y-axis each method label should be so they
# don't overlap.
heights = results[nrow(results), ]
```

```r
heights$Pairs = heights$Pairs - .01
heights$correlation = heights$correlation + .01

heights$`partial correlation` = heights$`partial correlation` + .01
heights$`partial BayesComm` = heights$`partial BayesComm` - .01

heights$GLM = heights$GLM - .01

text(1625, heights, colnames(results), pos = 4, cex = 0.75, col = colors)

######

# silly code to make the next graph line up prettily with the previous one
full_range = log10(c(25 * .9, 1600 * 8))
base_range = log10(c(25, 1600))

base_scaled = 2 * base_range / (base_range[2] - base_range[1])
full_scaled = 2 * full_range / (base_range[2] - base_range[1])

# New colors for plot B
colors2 = brewer.pal(10, "Paired")[c(2, 4, 10)]
plot(
  z$correlation,
  z$Pairs,
  col = colors2[factor(z$n_sites)],
  las = 1,
  xlab = "",
  ylab = expression(italic(Z)-score),
  bty = "l",
  xaxs = "i",
  axes = FALSE,
  xlim = full_scaled - base_scaled[2] + 1
)
axis(1, seq(-2, 1, .5))
axis(2, seq(-50, 50, 10), las = 1)
mtext("Correlation coefficient", side = 1, line = 2.25, at = 0)
mtext("B.", line = 1.25, at = -1, cex = 1.5)

legend(
  x = 1.05,
  y = max(z$Pairs),
  pch = 1,
  lty = 0,
  lwd = 2,
  col = colors2[1:length(unique(z$n_sites))],
  legend = levels(factor(z$n_sites)),
  title = "Number of sites",
```

```
  bty = "n",
  cex = 0.75,
  pt.cex = 1
)
segments(-1000, 0, 1.04, 0, col = "#00000025", lwd = 1.25)
segments(0, -1000, 0, 1000, col = "#00000025", lwd = 1.25)


dev.off()
```

```
## pdf
##   2
```





Summarize the results:

```
# R-squareds computed across *all* landscape types
total_ss = mean(resid(lm(truth ~ 0, data = z))^2)
round(100 * sort(1 - colMeans(resids^2) / total_ss), 1)
```

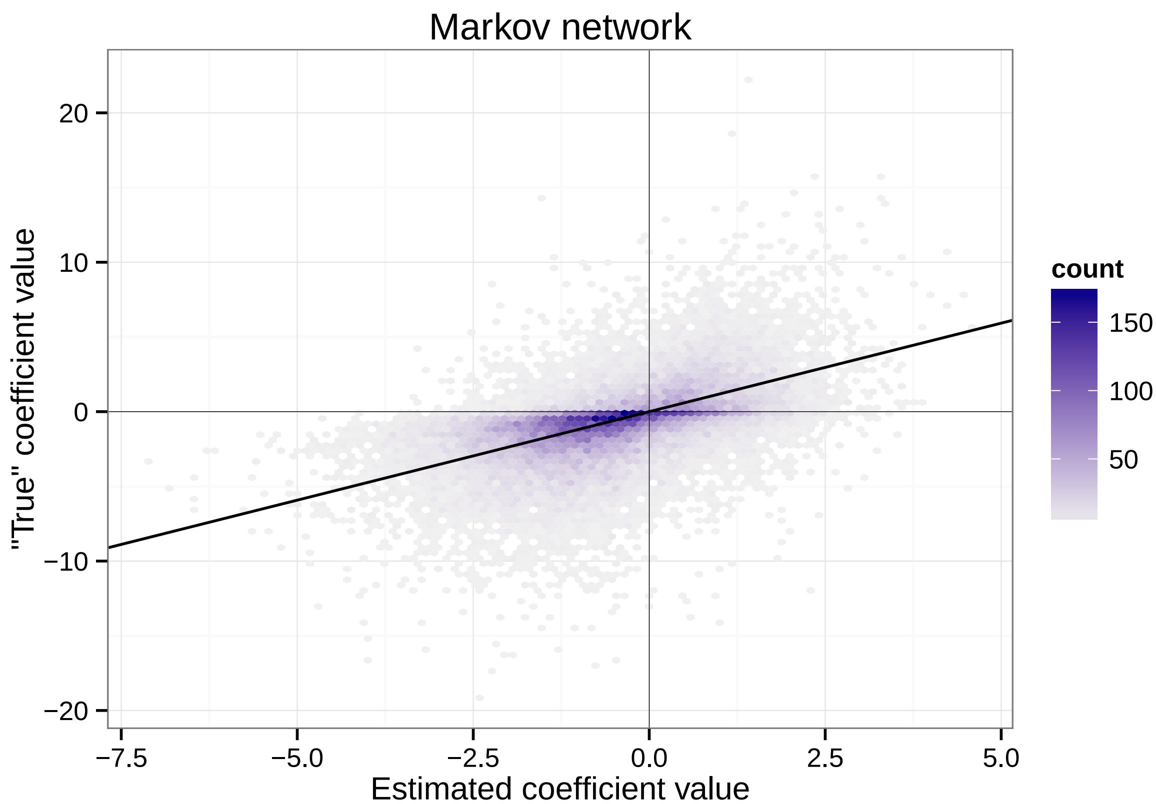
```
##           BayesComm                 Pairs           correlation
##                11.6                  15.4                  18.6
##   partial BayesComm partial correlation                   GLM
##                26.1                  27.5                  29.9
##      Markov network
##                34.6
```

```r
# Rank correlations among the methods that estimate marginal relationships
round(cor(z[ , c(7, 10, 12)], method = "spearman")[,1], 2)
```

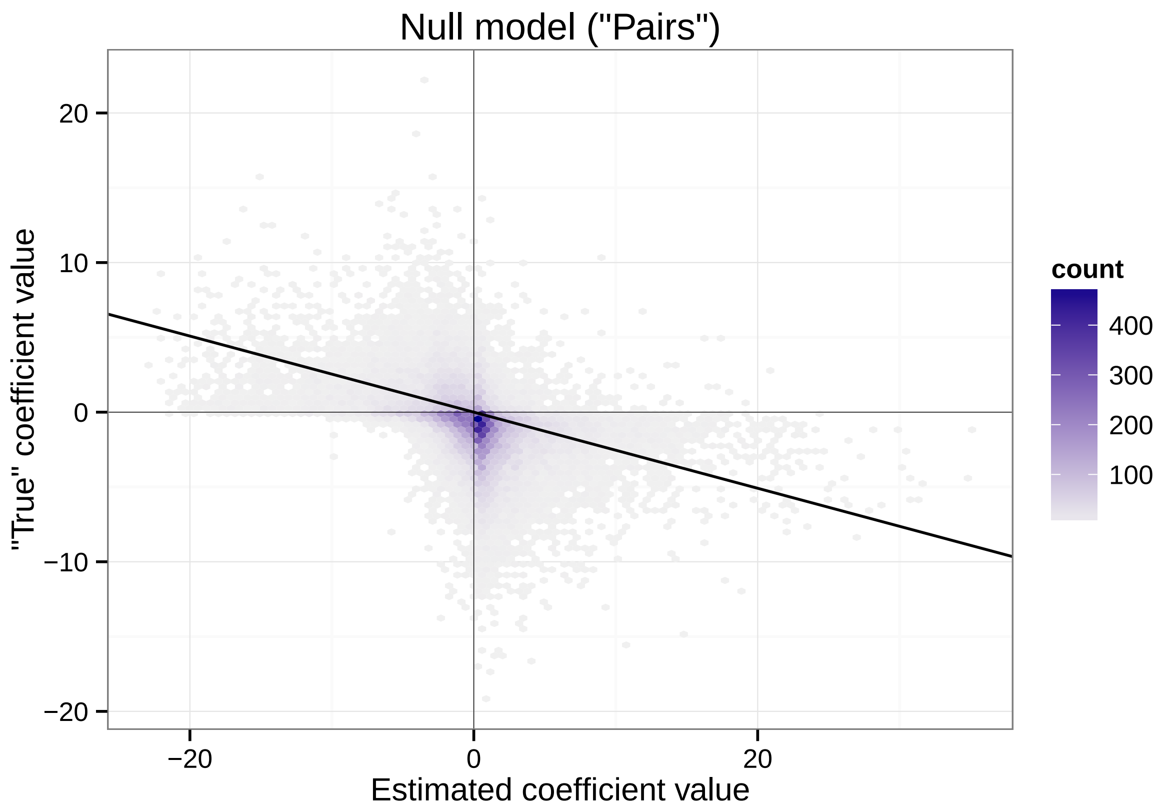
```
## correlation   BayesComm       Pairs
##        1.00        0.85       -0.92
```

Plot the true interactions versus estimated interactions for two methods:

```r
library(ggplot2)

ggplot(z[z$n_sites >0, ], aes(x = `Markov network`, y = truth)) +
  stat_binhex(bins = 100) +
  xlab("Estimated coefficient value") +
  ylab("\"True\" coefficient value") +
  stat_hline(yintercept = 0, size = 1/8) +
  stat_vline(xintercept = 0, size = 1/8) +
  scale_fill_gradient(low = "#F0F0F0", high = "darkblue", trans = "identity") +
  theme_bw() +
  ggtitle("Markov network") +
  stat_abline(intercept = 0, slope = coef(lm(z$truth ~ z$`Markov network` + 0)))
```




```
ggplot(z[z$n_sites >0, ], aes(x = `Pairs`, y = truth)) +
  stat_binhex(bins = 100) +
  xlab("Estimated coefficient value") +
  ylab("\"True\" coefficient value") +
  stat_hline(yintercept = 0, size = 1/8) +
  stat_vline(xintercept = 0, size = 1/8) +
  scale_fill_gradient(low = "#F0F0F0", high = "darkblue", trans = "identity") +
  theme_bw() +
  ggtitle("Null model (\"Pairs\")") +
  stat_abline(intercept = 0, slope = coef(lm(z$truth ~ z$Pairs + 0)))
```



Bootstrap resample from each of the three landscape size classes, generating new sets of 150 landscapes. For each one, calculate the proportion of variance explained (compared to a null baseline that assumes all species pairs' interaction strengths are zero).

```
rep = 12
n = 200
boots = replicate(500,
          {
            boot = lapply(
              sizes,
              function(n){lapply(
                sample.int(max(z$rep), replace = TRUE),
                function(rep){
                  z[z$rep == rep & z$n_sites == n, ]
                }
              ) %>% bind_rows}
            ) %>% bind_rows

            resids = sapply(
              colnames(boot)[-(1:5)],
              function(i){resid(lm(boot$truth ~ boot[[i]] + 0))}
            )
            total_ss = mean(resid(lm(truth ~ 0, data = boot))^2)
            1 - colMeans(resids^2) / total_ss
          }
```

```
)
```

Summarize the bootstrap results:

```
for(compared in c("partial correlation", "correlation", "GLM", "BayesComm", "partial BayesComm"
  CI = round(
    quantile(
      boots["Markov network", ] / boots[compared, ],
      c(.025, .975)),
    2
  )

  message("R-squared is " , CI[1], "-", CI[2], " times higher than from ", compared, " (95% boo
}
```

```
## R-squared is 1.23-1.29 times higher than from partial correlation (95% bootstrap CI)
## R-squared is 1.79-1.95 times higher than from correlation (95% bootstrap CI)
## R-squared is 1.14-1.18 times higher than from GLM (95% bootstrap CI)
## R-squared is 2.72-3.28 times higher than from BayesComm (95% bootstrap CI)
## R-squared is 1.29-1.37 times higher than from partial BayesComm (95% bootstrap CI)
## R-squared is 2.12-2.37 times higher than from Pairs (95% bootstrap CI)
```