Appendix 4: Results

Inferring species interactions from co-occurrence data with Markov networks

David J. Harris

```
library(dplyr)
library(mgcv)
library(ggplot2)
library(tidyr)
library(knitr)
library(lme4)
```

Import the results from Appendix 4:

```
x = read.csv("estimates.csv", stringsAsFactors = FALSE)
x$simulation_type = gsub("[0-9]", "", x$rep_name)
```

Import the results from the *Pairs* software:

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

# 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
) - 1

partial_names = sapply(
  strsplit(grep("^>", pairs_txt, value = TRUE), " +"),
  function(x) x[[3]]
)
filename_lines = grep("^>", pairs_txt)

# Sort a vector of alphanumeric strings by the numeric component
# as if they were integers.  For example, V20 is larger than V12,
# even though V12 comes first alphabetically
alnum_sort = function(x){
  raw = as.integer(gsub("[[:alpha:]]", "", x))
```

```r
  x[order(raw)]
}

pairs_results = lapply(
  1:length(filename_lines),
  function(i){

    n_sites = as.integer(strsplit(partial_names[[i]], "-")[[1]][[1]])
    rep_name = strsplit(partial_names[[i]], "-")[[1]][[2]]


    # Find the line where the current data set is mentioned in
    # pairs.txt
    filename_line = filename_lines[i]

    # 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 corresponding chunk of the "x" data frame, based on n_sites
    # and rep_name
    is_correct_sim = x$n_sites == n_sites & x$rep_name == rep_name
    x_subset = x[is_correct_sim & x$method == "correlation", ]

    # Pull out the species numbers and their Z-scores, then join to x_subset
    pairs_results = lapply(
      splitted,
      function(x){
        # in the x data frame, species 1 is always a lower number than species 2
        spp = alnum_sort(x[3:4])
        data.frame(
          sp1 = spp[1],
          sp2 = spp[2],
          z = x[14],
          stringsAsFactors = FALSE
        )
      }
    ) %>%
      bind_rows %>%
      mutate(spp = paste(sp1, sp2, sep = "-"))

    n_spp = 20

    pairs_results$z = as.numeric(pairs_results$z)
```

```r
    # Re-order the pairs_results to match the other methods
    m = matrix(NA, n_spp, n_spp)
    new_order = match(
      paste0("V", row(m)[upper.tri(m)], "-V", col(m)[upper.tri(m)]),
      pairs_results$spp
    )
    ordered_pairs_results = pairs_results[na.omit(new_order), ]

    ordered_pairs_results = ordered_pairs_results %>%
      filter(sp1 %in% c(x_subset$sp1) & sp2 %in% x_subset$sp2)

    x_subset$estimate = ordered_pairs_results$z
    x_subset$method = "null"

    x_subset
  }
) %>% bind_rows()

# Manually adjust the Z values less than -1000 so that these outliers
# won't completely dominate the analyses below
pairs_results$estimate[pairs_results$estimate < -1000] = -50

x = rbind(x, pairs_results)
```

Calculate model performance:

```r
resids = function(data){
  resid(lm(truth ~ estimate + 0, data = data))
}

result_summary = x %>%
  group_by(method, simulation_type) %>%
  do(data.frame(., resids = resids(.))) %>%
  ungroup %>%
  group_by(method, simulation_type, n_sites) %>%
  summarise(r2 = 1 - sum(resids^2) / sum(truth^2))

result_summary$method = reorder(result_summary$method, -result_summary$r2)

result_summary$simulation_type = reorder(
  result_summary$simulation_type,
  -result_summary$r2
)
```

```
result_summary = result_summary %>%
  group_by(method) %>%
  summarise(mean_r2 = round(100 * mean(r2))) %>%
  mutate(method_r2 = paste0(method, " (0.", mean_r2, ")")) %>%
  select(method, method_r2) %>%
  inner_join(result_summary, "method")

result_summary$simulation_type_long = plyr::revalue(
  result_summary$simulation_type,
  c(no_env = "constant environment",
    env = "heterogeneous environment",
    abund = "abundance")
)

result_summary$method_r2 = reorder(result_summary$method_r2, -result_summary$r2)
```

Calculate average R-squared across methods and simulation types

```
result_summary %>%
  group_by(method, simulation_type) %>%
  summarise(mean(r2)) %>%
  spread(simulation_type, `mean(r2)`) %>%
  kable(digits = 3)
```

method	no_env	env	abund
Markov network	0.525	0.451	0.384
GLM	0.472	0.405	0.283
partial correlation	0.403	0.322	0.200
partial BayesComm	0.394	0.302	0.166
correlation	0.291	0.183	0.117
null	0.227	0.125	0.075
BayesComm	0.206	0.110	0.060

Save the R-squared results as Figure 3

```
legend_name = expression(Method~(mean~R^2))

pdf("manuscript-materials/figures/performance.pdf", width = 8, height = 2.5)
ggplot(result_summary, aes(x = n_sites, y = r2, col = method_r2, shape = method_r2)) +
  facet_grid(~simulation_type_long) +
  geom_line(size = .5) +
  geom_point(size = 2.5, fill = "white") +
  scale_shape_manual(values = c(16, 22, 17, 23, 18, 24, 15), name = legend_name) +
```

```r
  geom_hline(yintercept = 0, size = 1/2) +
  geom_vline(xintercept = 0, size = 1) +
  #scale_shape_manual(values = c(21, 25, 15, 18)) +
  coord_cartesian(ylim = c(-.01, 0.76)) +
  ylab(expression(R^2)) +
  xlab("Number of sites (log scale)") +
  scale_x_log10(breaks = unique(x$n_sites), limits = range(x$n_sites)) +
  theme_bw(base_size = 11) +
  theme(panel.margin = grid::unit(1.25, "lines")) +
  theme(panel.border = element_blank(), axis.line = element_blank()) +
  theme(
    panel.grid.minor = element_blank(),
    panel.grid.major.y = element_line(color = "lightgray", size = 1/4),
    panel.grid.major.x = element_blank()
  ) +
  theme(strip.background = element_blank(), legend.key = element_blank()) +
  theme(plot.margin = grid::unit(c(.01, .01, .75, .1), "lines")) +
  theme(
    axis.title.x = element_text(vjust = -0.2, size = 12),
    axis.title.y = element_text(angle = 0, hjust = -.1, size = 12)
  ) +
  scale_color_brewer(palette = "Dark2", name = legend_name)
dev.off()
```

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

Estimate R-squared uncertainty:

Fit a linear mixed model describing R-squared as a function of method, landscape size, and simulation type. Note the small standard errors associated with the effect of estimation method.

```r
landscape_estimates = x %>%
  group_by(method, simulation_type) %>%
  do(data.frame(., resids = resids(.))) %>%
  ungroup %>%
  group_by(method, simulation_type, rep_name, n_sites) %>%
  summarise(r2 = 1 - sum(resids^2) / sum(truth^2))


summary(
  lmer(
    r2 ~ method + n_sites + simulation_type + (1|rep_name),
    data = landscape_estimates
  )
)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: r2 ~ method + n_sites + simulation_type + (1 | rep_name)
##    Data: landscape_estimates
##
## REML criterion at convergence: -4945.2
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -7.3997 -0.6350  0.0758  0.6883  2.7104
##
## Random effects:
##  Groups   Name        Variance  Std.Dev.
##  rep_name (Intercept) 0.0008304 0.02882
##  Residual             0.0113238 0.10641
## Number of obs: 3150, groups:  rep_name, 150
##
## Fixed effects:
##                             Estimate Std. Error t value
## (Intercept)               -2.481e-02  7.186e-03   -3.45
## methodcorrelation          6.965e-02  7.094e-03    9.82
## methodGLM                  2.544e-01  7.094e-03   35.85
## methodMarkov network       3.217e-01  7.094e-03   45.35
## methodnull                 1.504e-02  7.094e-03    2.12
## methodpartial BayesComm    1.616e-01  7.094e-03   22.78
## methodpartial correlation  1.796e-01  7.094e-03   25.31
## n_sites                    1.050e-04  2.690e-06   39.04
## simulation_typeenv         8.853e-02  7.402e-03   11.96
## simulation_typeno_env      1.795e-01  7.402e-03   24.25
##
## Correlation of Fixed Effects:
##             (Intr) mthdcr mthGLM mthdMn mthdnl mtBysC mthdpc n_sits smltn_
## methdcrrltn -0.494
## methodGLM   -0.494  0.500
## mthdMrkvntw -0.494  0.500  0.500
## methodnull  -0.494  0.500  0.500  0.500
## mthdprtBysC -0.494  0.500  0.500  0.500  0.500
## mthdprtlcrr -0.494  0.500  0.500  0.500  0.500  0.500
## n_sites     -0.228  0.000  0.000  0.000  0.000  0.000  0.000
## smltn_typnv -0.515  0.000  0.000  0.000  0.000  0.000  0.000  0.000
## smltn_typn_ -0.515  0.000  0.000  0.000  0.000  0.000  0.000  0.000  0.500
```

Identify statistical significance for the Markov network and Pairs

```r
pairs_summary = x[x$method == "null" & !grepl("pop", x$rep_name), ]
markov_summary = x[x$method == "Markov network" & !grepl("pop", x$rep_name), ]
```

```r
# Pairs's Z score is based on C-scores, which are positive when species are
# disaggregated.  So significantly positive interactions == negative estimates
pairs_summary$sig_pos = pairs_summary$estimate < qnorm(.025)
pairs_summary$sig_neg = pairs_summary$estimate > qnorm(.975)

# Markov network is significant when lower bound is above zero or lower bound is
# below zero
markov_summary$sig_pos = markov_summary$lower > 0
markov_summary$sig_neg = markov_summary$upper < 0
```

Create Figure 4:

```r
# Function for smoothing the error rates with a generalized additive model
# and plotting the results
my_geom_smooth = function(data, color, ...){
  geom_smooth(
    data = data,
    aes(
      x = truth,
      y = as.integer((sig_neg & truth > 0) | (sig_pos & truth < 0)),
      color = color
    ),
    method = gam,
    family = binomial,
    formula = y ~ s(x),
    se = FALSE,
    n = 1024,
    ...
  )
}


truth_seq = seq(min(markov_summary$truth), max(markov_summary$truth), length = 1000)

plotfun = function(..., add){
  if(add){
    lines(...)
  }else{
    plot(...)
  }
}


error_smoother = function(data){
  predict(
```

```r
    gam(
      I((sig_neg & truth > 0) | (sig_pos & truth < 0)) ~ s(truth),
      data = data,
      family = binomial
    ),
    data.frame(truth = truth_seq),
    type = "response"
  )
}

y_pairs = error_smoother(pairs_summary)
y_markov = error_smoother(markov_summary)

pdf("manuscript-materials/figures/error_rates.pdf", height = 8.5, width = 8.5/3)
par(mfrow = c(3, 1))

# Compare estimates
spread_estimates = x %>%
  dplyr::select(-lower, -upper, -X) %>%
  spread(method, estimate) %>%
  na.omit()

# R-squared for null versus correlation
round(summary(lm(null ~ I(correlation*sqrt(n_sites)), data = spread_estimates))$r.squared, 2)
```

```
## [1] 0.95
```

```r
# R-squared for glm markov network versus glm
round(summary(lm(`Markov network` ~ GLM, data = spread_estimates))$r.squared, 2)
```

```
## [1] 0.94
```

```r
with(
  spread_estimates,
  plot(
    `Markov network`,
    GLM,
    pch = ".",
    col = "#00000020",
    ylab = "Markov network estimate",
    xlab = "GLM estimate",
    bty = "l"
  )
)
mtext("A. Markov network estimates\nvs. GLM estimates", adj = 0, side = 3,
      font = 2, line = 1.2, cex = .9)
```

```r
abline(lm(`Markov network` ~ GLM, data = spread_estimates))
text(0, 5, expression(R^2==0.94))

with(
  spread_estimates,
  plot(
    correlation * sqrt(n_sites),
    null,
    pch = ".",
    col = "#00000020",
    ylab = "Z-score",
    xlab = expression("correlation" %*% sqrt(number~~of~~sites)),
    bty = "l"
  )
)
mtext("B. Null model estimates vs.\nscaled correlation coefficients",
      adj = 0, side = 3, font = 2, line = 1.2, cex = .9)
abline(lm(null ~ I(correlation*sqrt(n_sites)), data = spread_estimates))
text(10, 12, expression(R^2==0.95))

plotfun(
  truth_seq,
  y_pairs,
  type = "l",
  xlab = "\"True\" interaction strength",
  ylab = "P(confidently predict wrong sign)",
  bty = "l",
  yaxs = "i",
  col = 2,
  add = FALSE,
  ylim = c(0, .4),
  lwd = 2
)
mtext("C. Error rate vs.\ninteraction strength", side = 3, adj = 0,
      font = 2, line = 1.2, cex = .9)
plotfun(truth_seq, y_markov, add = TRUE, lwd = 2)
legend("topleft", lwd = 2, legend = c("Null model", "Markov network"),
       col = c(2, 1), bty = "n")
dev.off()
```

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

Summarize inferential statistics:

```
# P(Pairs confidently wrong)
with(pairs_summary,  mean((sig_neg & truth > 0) | (sig_pos & truth < 0)))
```

```
## [1] 0.1533758
```

```
# P(Markov network confidently wrong)
with(markov_summary, mean((sig_neg & truth > 0) | (sig_pos & truth < 0)))
```

```
## [1] 0.02861541
```

```
# P(Pairs rejects null)
with(pairs_summary,  mean(sig_neg | sig_pos))
```

```
## [1] 0.4520535
```

```
# P(Markov network rejects null)
with(markov_summary,  mean(sig_neg | sig_pos))
```

```
## [1] 0.2116561
```

```
# P(reject null | small interaction) for Makrov network
# (approximate Type I error rates)
markov_summary %>%
  group_by(simulation_type) %>%
  filter(abs(truth) < .1) %>%
  summarize(Type_1_error_rate = mean(sig_neg | sig_pos)) %>%
  kable(digits = 3)
```

simulation_type	Type_1_error_rate
abund	0.220
env	0.136
no_env	0.019

```
# P(reject null | small interaction) for Pairs
# (approximate Type I error rates)
pairs_summary %>%
  group_by(simulation_type) %>%
  filter(abs(truth) < .1) %>%
  summarize(Type_1_error_rate = mean(sig_neg | sig_pos)) %>%
```

```r
  kable(digits = 3)
```

simulation_type	Type_1_error_rate
abund	0.575
env	0.501
no_env	0.297

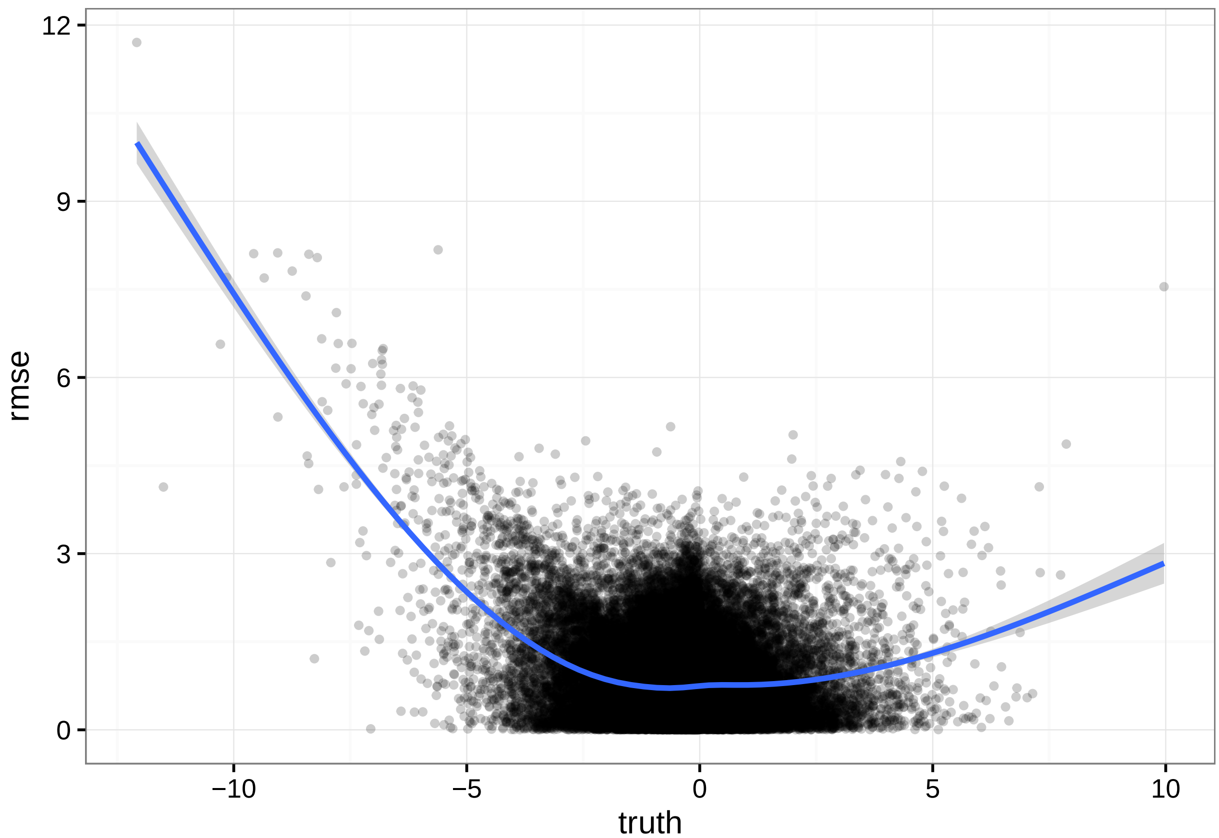
```r
# Confidence interval coverage across all values
# What's the probability that any "true" value falls inside the 95% CI?
markov_summary %>%
  group_by(simulation_type) %>%
  summarize(mean(truth > lower & truth < upper)) %>%
  kable(digits = 3)
```

simulation_type	mean(truth > lower & truth < upper)
abund	0.868
env	0.857
no_env	0.975

Plot root mean square error versus "true" beta

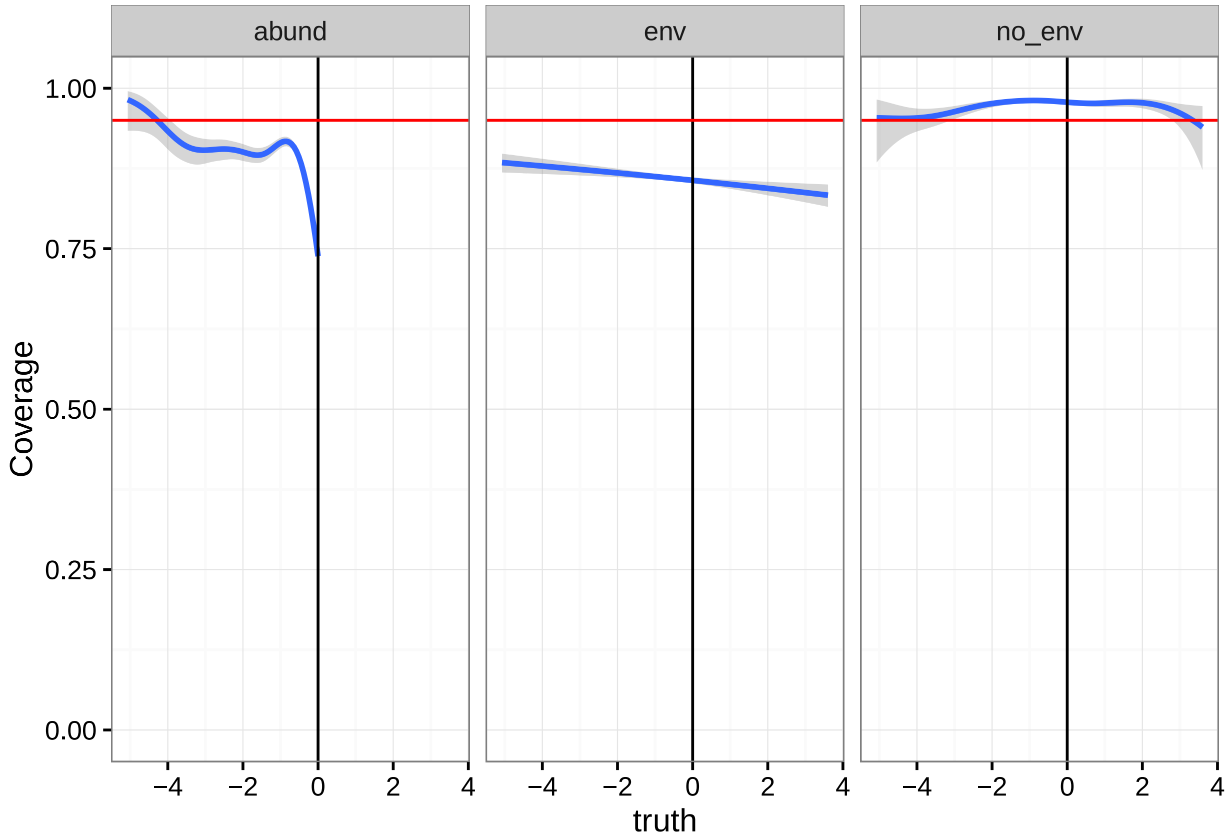
```r
# Exclude the "abundance" simulations because the interpretation of $\beta$
# is different
rmses = markov_summary %>%
  filter(simulation_type != "abund") %>%
  mutate(rmse = sqrt((truth - estimate)^2))

ggplot(rmses, aes(x = truth, y = rmse)) +
  geom_point(alpha = .2, size = 1) +
  geom_smooth() +
  theme_bw() +
  coord_cartesian(ylim = c(0, max(rmses$rmse)))
```



Plot confidence interval coverage versus true β value

```
# Coverage versus true beta value.  The top and bottom 0.5%
# of the distribution have been omitted to prevent bad behavior
# by the smoother in the tails.
markov_summary %>%
  filter(percent_rank(truth) > .005 & percent_rank(truth) < .995) %>%
  mutate(covered = truth > lower & truth < upper) %>%
  ggplot(aes(x = truth, y = as.integer(covered))) +
  facet_grid(~simulation_type) +
  geom_smooth(method = gam, formula = y ~ s(x), method.args = list(family = "binomial")) +
  theme_bw() +
  coord_cartesian(ylim = c(0, 1)) +
  geom_vline(xintercept = 0) +
  geom_hline(yintercept = 0.95, color = "red") +
  ylab("Coverage")
```


abund
env
no_env
Coverage
1.00
0.75
0.50
0.25
0.00
−4
−2
0
2
4
truth