Appendix 1: Evaluation of the three species in Figure 1

Load dependencies

```r
library(dplyr)
library(magrittr)
library(rosalia)
library(knitr)
library(corpcor)
```

```r
set.seed(1)
```

Generate the species interaction matrix from Figure 1.

```r
alpha = c(3, 2, 1)
beta = matrix(
  c(0, -3, -3, -3, 0, -1, -3, -1, 0),
  nrow = 3,
  dimnames = replicate(2, c("tree", "shrub1", "shrub2"), simplify = FALSE)
)

kable(beta)
```

	tree	shrub1	shrub2
tree	0	-3	-3
shrub1	-3	0	-1
shrub2	-3	-1	0

```r
# Simulated community has three species at some of 100 locations
n_spp = 3
n_sites = 100

# Re-organize the coefficients for rosalia
truth = c(beta[upper.tri(beta)], alpha)

# Enumerate the 8 possible species assemblages
# using an internal rosalia function
possibilities = rosalia:::generate_possibilities(n_spp)

# In each of these 8 possible assemblages, which of the 6 possible
# species pairs *both* occur?
possible_cooc = sapply(
  1:2^n_spp,
  function(i){
    tcp = tcrossprod(possibilities[i, ])
    c(tcp[upper.tri(tcp)], diag(tcp))
  }
)
```

Simulate 1000 assemblages of 3 species across 100 sites.

Once a landscape has been simulated, test whether `rosalia`, the sample correlation, and the partial correlation got the sign of the shrub-shrub interaction correct.

```
evaluation = sapply(
  1:1000,
  function(i){
    x = rosalia::simulate_data(
      par = truth,
      possibilities = possibilities,
      possible_cooc = possible_cooc,
      n_sites = n_sites
    )
    if(paste0("fakedata/three/", i, ".txt") %in% dir("fakedata/three", full.names = TRUE)){
      # Do nothing: simulated data was already written to disk
    }else{
      write.table(t(x[rowSums(x) > 0, ]), file = paste0("fakedata/three/", i, ".txt"))
    }

    rosie = rosalia(x, trace = 0, prior = make_flat_prior(), hessian = TRUE)

    c(
      cor = cor(x)[2, 3] < 0,
      partial = cor2pcor(cor(x))[2, 3] < 0,
      rosalia = rosie$beta[2,3],
      rosalia_se = sqrt(diag(solve(rosie$opt$hessian)))["beta3"]
    )
  })
```

Evaluate the frequentist coverage of the confidence intervals (CIs). If the coverage is good, then approximately 95% of the confidence intervals generated on independent data sets should contain the true value (i.e., in 95% of cases, the upper CI should be above the true value of -1 and the lower CI should be below the true value).

```
upper_ci_above_true = evaluation["rosalia", ] +
  qnorm(.975) * evaluation["rosalia_se.beta3", ] > -1
lower_ci_below_true = evaluation["rosalia", ] +
  qnorm(.025) * evaluation["rosalia_se.beta3", ] < -1

# Coverage is close to 95%
100 * mean(upper_ci_above_true & lower_ci_below_true)
```

```
## [1] 96.5
```

```
# According to this binomial test, the confidence intervals include the
# true value a bit too often to be consistent with 95% coverage (i.e. there
# is some evidence that the confidence intervals are too conservative).
binom.test(x = sum(upper_ci_above_true & lower_ci_below_true),
           n = length(upper_ci_above_true),
           p = 0.95,
           conf.level = 0.95
)
```

```
##
##  Exact binomial test
##
## data:  sum(upper_ci_above_true & lower_ci_below_true) and length(upper_ci_above_true)
## number of successes = 965, number of trials = 1000, p-value =
## 0.02915
## alternative hypothesis: true probability of success is not equal to 0.95
## 95 percent confidence interval:
##  0.9516576 0.9755025
## sample estimates:
## probability of success
##                  0.965
```

After running the “Pairs” software on transposed versions of the files simulated above, we can import its results to see how the method performed.

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

# Find the column that contains the Z-score
z_score_column = grep("Z-Score", Pairs.txt, value = TRUE)[2] %>%
  strsplit(" +") %>%
  extract2(1) %>%
  grep("Z-Score", .)

# Find the right rows of the text file, split them on whitespace, pull out
# the column identified above, and save the Z-score
pairs_z = Pairs.txt[grep("(Var2.*Var1)|(Var1.*Var2)", Pairs.txt)] %>%
    strsplit(" +") %>%
    sapply(. %>% extract(z_score_column) %>% as.numeric)


# Z>0 implies negative interactions because the Z-score relates to the
# distribution of C-scores, which describe the frequency with which species *do
# not* co-occur. It has the opposite sign of metrics based on when species *do*
# co-occur.
pairs_evaluation = pairs_z > 0
```

```
out = c(pairs = mean(pairs_evaluation) * 100, rowMeans(evaluation)[1:3] * 100)

kable(data_frame(model = names(out), percent_correct = paste(out, "%")))
```

```
# percentage of Pairs runs that found statistically significant mutualism at the
# p<.05 level (two-tailed test)
mean(pnorm(pairs_z) < 0.025) * 100

# percentage of Pairs runs that found statistically significant competition at the
# p<.05 level (two-tailed test)
pairs_significant_competition = pnorm(pairs_z) > 0.975

# percentage of rosalia runs whose 95% confidence intervals did not include 0
rowMeans(evaluation)[c("rosalia_confident_negative", "rosalia_confident_positive")] * 100
```