Fitting logseries and negative binomial SADs with random forests

Infering the scale of a truncated negative binomial

The zero-truncated negative binomial can be parameterized to represent most all log-normal distributions (given finite data) and in the limit of $k = 0$ is exactly Fisher's log-series. So it's a good distribution to see if we can properly parameterize with random forests.

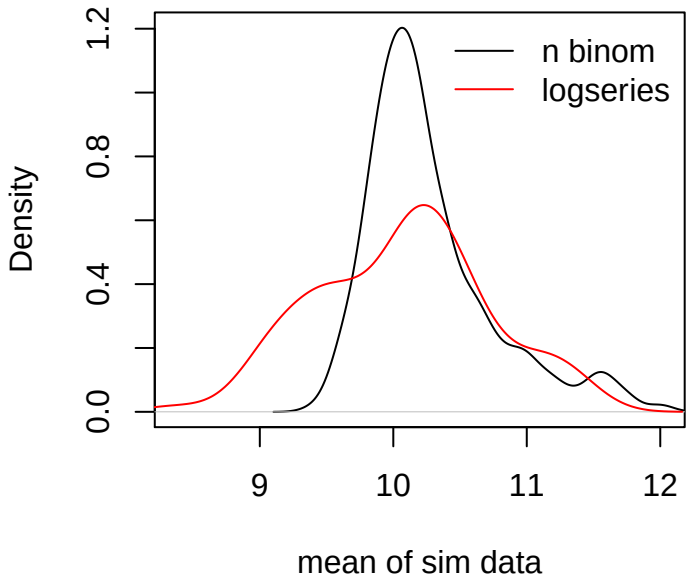
Species abundances simulated using pika package

```
# parameters for the negative binomial
mu <- 10
k <- seq(1, 6, length.out = 6)

# parameters for the log-series maintaining the mean
p <- uniroot(function(p) -1/log(1 - p) * p / (1 - p) - mu,
             c(.Machine$double.eps, 1 - .Machine$double.eps))$root
b <- -log(p)

# simulation parameters
nsim <- 100
nspp <- 500

# simulate negative binomial SADs
negbSAD <- rtnegb(nsim * nspp * length(k), mu = mu, k = rep(k, each = nsim * nspp))
negbSAD <- split(negbSAD, rep(1:(nsim * length(k)), each = nspp))
param <- rep(k, each = nsim)

# add to that fisher SADs
fishSAD <- rfish(nsim * nspp, beta = b)
fishSAD <- split(fishSAD, rep(1:nsim, each = nspp))

allSAD <- c(negbSAD, fishSAD)
allParam <- c(param, rep(0, nsim))
```

Random forest fitting

First let's make sure the RF won't just be picking up something simple like a spurious difference in means.

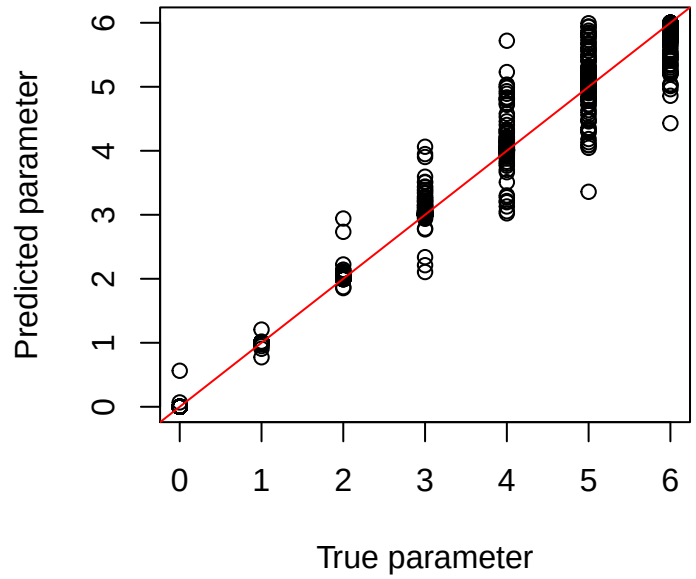
```
plot(density(sapply(negbSAD, mean)), xlim = range(sapply(allSAD, mean)),
     xlab = 'mean of sim data', main = '')
lines(density(sapply(fishSAD, mean)), col = 'red')
legend('topright', legend = c('n binom', 'logseries'),
       col = c('black', 'red'), lty = 1, bty = 'n')
```



The means of means are close for each distribution, but logseries has more variability in means across simulations. I think that's ok.

Now on to the fitting with RF based on Renyi entropy

```
# summary statistic of Renyi entropies
sumStat <- t(sapply(allSAD, renyi))

# run the random forest
rf <- randomForest(sumStat, allParam)

# see how well prediction works
plot(allParam, rf$predicted, xlab = 'True parameter', ylab = 'Predicted parameter')
abline(0, 1, col = 'red')
```

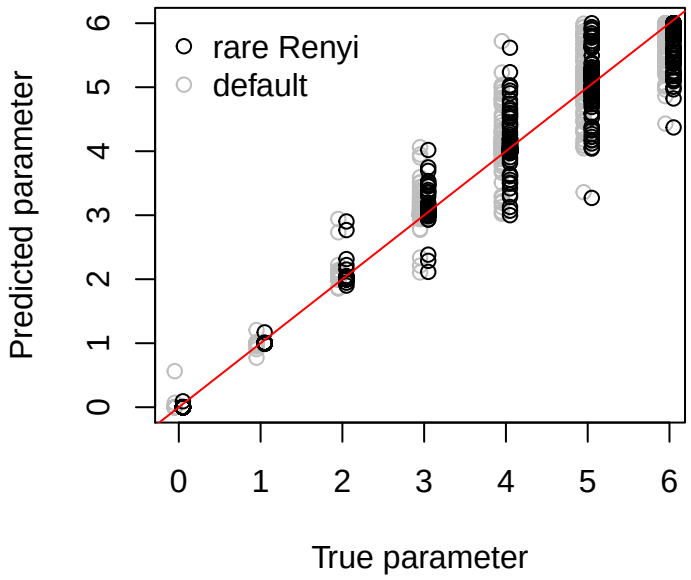


The prediction looks pretty good!

I'm curious to see if using different weights in the Renyi entropy makes a difference. For example, paying

more attention to rare species

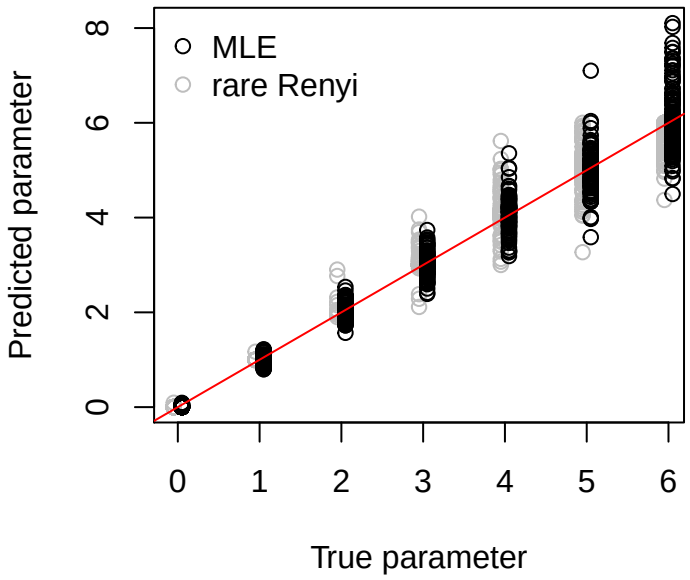
```
sumStat <- t(sapply(allSAD, renyi, scales = c(0, 2^(-6:2), Inf)))

# run the random forest
rfRare <- randomForest(sumStat, allParam)

# see how well prediction works
plot(allParam - 0.05, rf$predicted, xlab = 'True parameter', ylab = 'Predicted parameter',
     col = 'gray')
points(allParam + 0.05, rfRare$predicted)
abline(0, 1, col = 'red')
legend('topleft', legend = c('rare Renyi', 'default'), col = c('black', 'gray'),
       pch = 1, bty = 'n')
```



We can see that focusing on rare species (or rather not focusing so much on the dominants) helps pin down the difference between the logseries (true parameter = 0) and also the value of the true parameter when k is small in the negative binomial. But the results are effectively equivalent for larger values of k.

For negative binomial we can also see if the RF does any better or worse than maximum likelihood

```
# maximum likelihood estimation with *pika*
mle <- sapply(allSAD, function(x) {sad(x, 'tnegb')$MLE[2]})

plot(allParam - 0.05, rfRare$predicted, xlab = 'True parameter',
     ylab = 'Predicted parameter', col = 'gray',
     ylim = range(mle, rfRare$predicted))
points(allParam + 0.05, mle)
abline(0, 1, col = 'red')
legend('topleft', legend = c('MLE', 'rare Renyi'), col = c('black', 'gray'),
       pch = 1, bty = 'n')
```



So my take home is that RF on rare Renyi is better for negative binomial when k is small, but is a less consistent estimator (i.e. the mean of the inferred parameter can be off) than ML. That makes sense, ML is provably least biased for n approaching infinity. But maybe with more Renyi entropies RF would do just as good.