joint spp distance sampling

jmm

5/9/2020

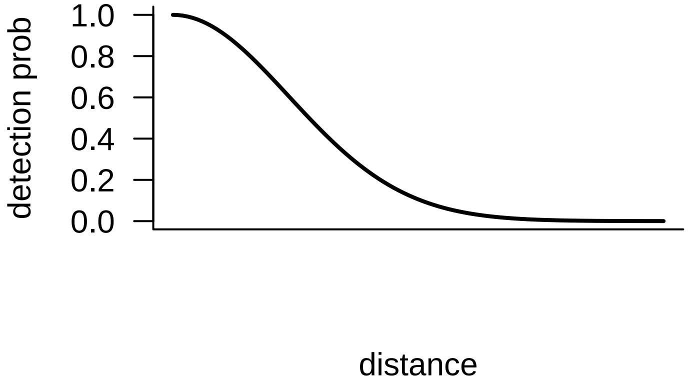
The goal here is to model changes in species abundances based on distance sampling. To model species responses, we want to use a joint species modelling approach a la Ovaskainen.

As a starting point we need a distance sampling model that can be fitted with Stan so that we can then include the joint species part.

Distance sampling is a good example of a hierarchical model where we separate the ecological quantities of interest (animal density), from the observation of such quantities.

Here I follow the idea of a data augmentation approach as presented in Royle and Dorazio (2009). But, as with Stan we cannot have discrete-valued hidden variable we have to use a zero-inflated version (see the San code below).

Let's start by simulating a data set and trying to recover known parameters. We assume that detection probability decreases with distance to the trasect as a half-normal: $\exp\left(-\frac{x^2}{2\sigma^2}\right)$

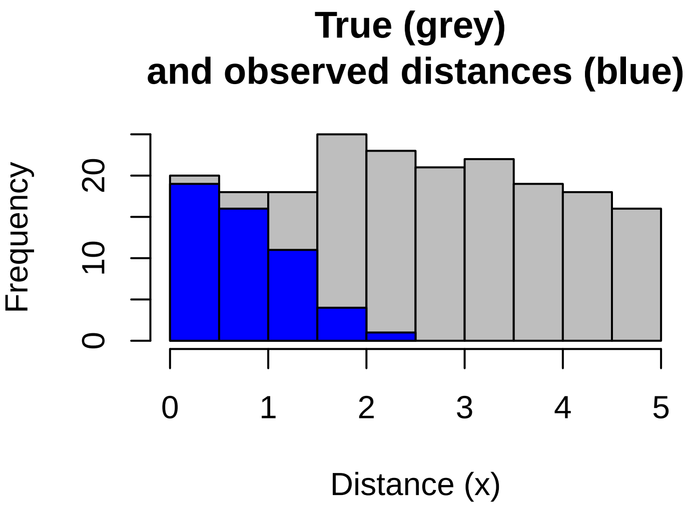


We assume further that a total of n individuals are located at random over the area being sampled. We run the transect for a length L and the boundaries of the study area are at a distance B from each side of the transect. Let's simulate some data:

```
sigma = 1
n = 200
B = 5
L = 10

x <- runif(n, -B, B)
p <- exp(-x^2/2*sigma^2) # detection probability
y_obs <- rbinom(n, size = 1, prob = p)
```

Now we plot the distribution of distances of all the animals to the transect and the distance of those that were obseved:

```
hist(abs(x), nclass = 15,
     xlab = "Distance (x)", col = "grey",
     main = "True (grey) \nand observed distances (blue)" )
 hist(abs(x[y_obs==1]), col = "blue", add = TRUE)
```



To estimate density $\frac{n}{L\times 2B}$ using data augmentation we add to the list of observed individual a number nz of zeros and define a parameter ψ for the probability that an individual (observed or not) is part of the population under study.

A Stan model for this can be found in the dist.stan file

Now we fit the model to the simulated data

```
nz = 200

datos <- list(n_obs = sum(y_obs),
              nz = nz,
              x = abs(x[y_obs==1]),
              y = c(rep(1,sum(y_obs)), rep(0,nz)),
              B = B,
              Area = 2*B*L)

pars = c("D", "psi", "sigma")

library("rstan")
```

```
## Loading required package: StanHeaders
```

```
## Loading required package: ggplot2
```

```
## rstan (Version 2.19.2, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

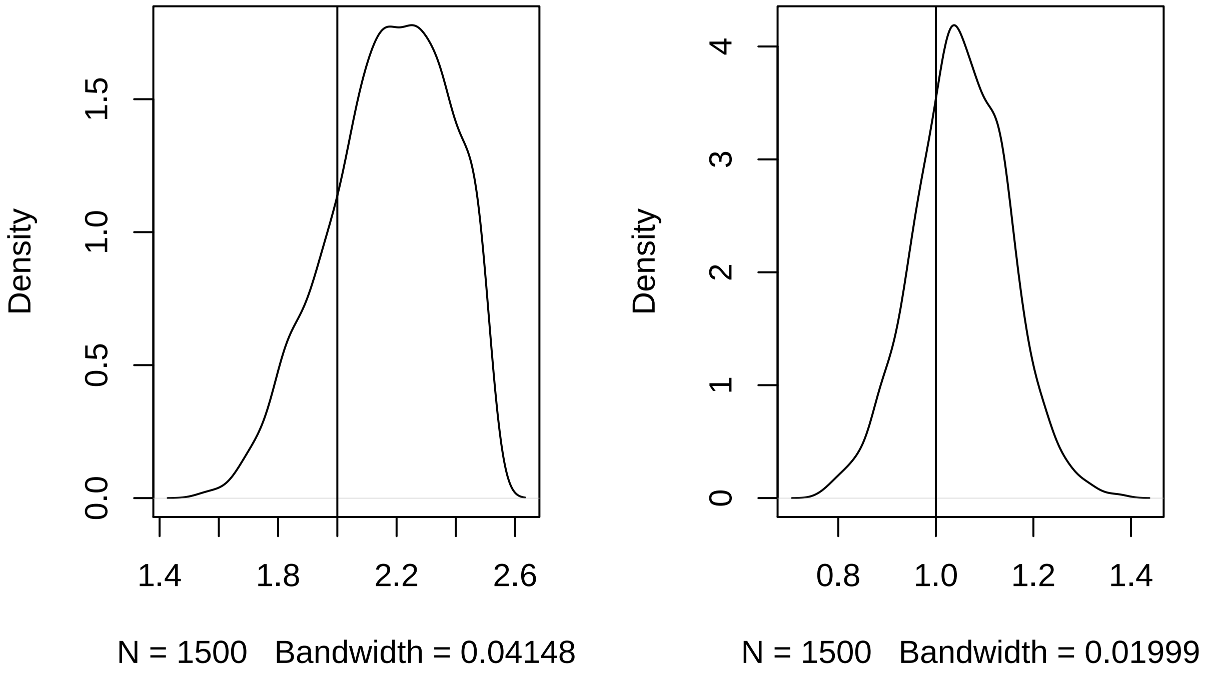
```
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())

fit <- stan(file = 'dist.stan',
            data = datos,
            pars = pars,
            include = TRUE,
            iter = 1000, thin = 1, chains = 3)
```

```
## Warning in readLines(file, warn = TRUE): incomplete final line found on '/
## Users/juanmanuelmorales/distance/dist.stan'

## hash mismatch so recompiling; make sure Stan code ends with a blank line

## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#bulk-ess
```

```r
print(fit)
```

```
## Inference for Stan model: dist.
## 3 chains, each with iter=1000; warmup=500; thin=1;
## post-warmup draws per chain=500, total post-warmup draws=1500.
##
##          mean se_mean   sd    2.5%     25%     50%     75%  97.5% n_eff
## D        2.18    0.01  0.2    1.77    2.05    2.19    2.34   2.49   379
## psi      0.83    0.01  0.1    0.63    0.77    0.84    0.91   0.99   379
## sigma    1.05    0.00  0.1    0.87    0.99    1.05    1.12   1.24  1329
## lp__  -120.11    0.99 14.9 -150.15 -129.79 -120.18 -109.88 -92.51   228
##       Rhat
## D     1.01
## psi   1.01
## sigma 1.00
## lp__  1.02
##
## Samples were drawn using NUTS(diag_e) at Sun May 10 09:48:51 2020.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```r
samples = extract(fit)

op = par(mfrow=c(1,2))
plot(density(samples$D), main = "")
abline(v = n/(2*B*L))
plot(density(samples$sigma), main = "")
abline(v=sigma)
```


Density
0.0
0.5
1.0
1.5
1.4
1.8
2.2
2.6
N = 1500   Bandwidth = 0.04148
Density
0
1
2
3
4
0.8
1.0
1.2
1.4
N = 1500   Bandwidth = 0.01999

```
par(op)
```