Simulate data and fit a 2-species static (aka single-season) occupancy model à la Rota et al. (2016)

We consider a two-species static occupancy model à la Rota et al. (2016). We simulate data from this model, and fit the model to these data using `Unmarked`.

Setting the scene

Ignoring the site index, we use the following notation for the occupancy probabilities:

- ψ_{11} is the prob. that species 1 and species 2 are both present;
- ψ_{10} is the prob. that species 1 is present and species 2 is absent;
- ψ_{01} is the prob. that species 1 is absent and species 2 is present;
- ψ_{00} is the prob. that species 1 and species 2 are both absent, with avec $\psi_{11} + \psi_{10} + \psi_{01} + \psi_{00} = 1$.

The marginal probabilities of occupancy are:

- $\Pr(z_1 = 1) = \Pr(\text{species 1 is present}) = \psi_{10} + \psi_{11}$
- $\Pr(z_2 = 1) = \Pr(\text{species 2 is present}) = \psi_{01} + \psi_{11}$
- $\Pr(z_1 = 0) = \Pr(\text{species 1 is absent}) = \psi_{01} + \psi_{00}$
- $\Pr(z_2 = 0) = \Pr(\text{species 2 is absent}) = \psi_{10} + \psi_{00}$

And the conditional probabilities (reminder: $\Pr(\text{A}|\text{B}) = \Pr(\text{A and B})/\Pr(\text{B})$):

- $\Pr(z_1 = 1|z_2 = 0) = \psi_{10}/(\psi_{10} + \psi_{00}) = \Pr(\text{species 1 is present given species 2 is absent})$;
- $\Pr(z_1 = 1|z_2 = 1) = \psi_{11}/(\psi_{11} + \psi_{01}) = \Pr(\text{species 1 is present given species 2 is present})$;
- $\Pr(z_2 = 1|z_1 = 0) = \psi_{01}/(\psi_{01} + \psi_{00}) = \Pr(\text{species 2 is present given species 1 is absent})$;
- $\Pr(z_2 = 1|z_1 = 1) = \psi_{11}/(\psi_{11} + \psi_{10}) = \Pr(\text{species 2 is present given species 1 is present})$.

Data simulation

We will use the package `mipfb` to simulate occupancy state as a multivariate Bernoulli random variable; more about the multivariate Bernoulli can be found in Dai et al. (2013):

```
library(mipfp)
```

For reproducibility, we set the seed:

```
set.seed(2020)
```

Choose the number of species, the number of sites, and the number of visits:

```
S <- 2 # nb species
N <- 500 # nb sites
J <- 5 # nb visits
```

Let's consider a scenario in which species 2 avoids species 1 while species 1 does not care about species 2 and its presence or absence. To specify this scenario, we will work out the conditional probabilities with, for example:

- $\Pr(z_2 = 1|z_1 = 0) = 0.6$, species 2 is present with high probability whenever species 1 is absent
- $\Pr(z_2 = 1|z_1 = 1) = 0.1$, species 2 avoids species 1 when it is present

- $\Pr(z_1 = 1|z_2 = 0) = \Pr(z_1 = 1|z_2 = 1) = 0.4$, species 1 does not care about presence/absence of species 2

Now we need to go back to the probabilities of occupancy. Let $x = \psi_{01}$, $y = \psi_{10}$ et $z = \psi_{11}$ soit $1 - x - y - z = \psi_{00}$, then we have a system of 3 equations with 3 unknowns:

$$0.6 = x/(x + 1 - x - y - z) \Leftrightarrow x + 0.6y + 0.6z = 0.6$$
$$0.1 = z/(z + y) \Leftrightarrow -0.1y + 0.9z = 0$$
$$0.4 = y/(y + 1 - x - y - z) \Leftrightarrow 0.4x + y + 0.4z = 0.4$$

which can be solved with the Mathematica online solver:

```
psi01 <- 81/175
psi10 <- 36/175
psi11 <- 4/175
psi00 <- 1 - (psi01 + psi10 + psi11) # 54/175
```

We then obtain the marginal occupancy probabilities:

```
psiS1 <- psi10 + psi11
psiS2 <- psi01 + psi11
```

Now we're ready to simulate data from a multivariate Bernoulli (check out `?RMultBinary` and `?ObtainMultBinaryDist`).

First, we calculate the odds ratios:

```
or <- matrix(c(1, (psiS1*(1-psiS2))/(psiS2*(1-psiS1)),
               (psiS2*(1-psiS1))/(psiS1*(1-psiS2)), 1), nrow = 2, ncol = 2, byrow = TRUE)
rownames(or) <- colnames(or) <- c("sp1", "sp2")
```

Then the marginal probabilities:

```
marg.probs <- c(psiS1, psiS2)
```

And we estimate the joint probability:

```
p.joint <- ObtainMultBinaryDist(odds = or, marg.probs = marg.probs)
```

At last, we generate N random samples from a bivariate Bernoulli (2 species) with relevant parameters

```
z <- RMultBinary(n = N, mult.bin.dist = p.joint)$binary.sequences
```

Now we add on top the observation. First, we fix the detection probability for each species:

```
ps <- c(0.5,0.9)
```

Then we generate the detection and non-detections for each species, which we store in a list:

```
y <- list()
for (i in 1:S){
  y[[i]] <- matrix(NA,N,J)
  for (j in 1:N){
    for (k in 1:J){
      y[[i]][j,k] <- rbinom(1,1,z[j,i]*ps[i])
    }
  }
}
names(y) <- c('sp1','sp2')
```

Model fitting

Now let us fit a 2-species static occupancy model to the data we have simulated. We need to load the package `unmarked`:

```
library(unmarked)
```

We format the data as required:

```
data <- unmarkedFrameOccuMulti(y=y)
```

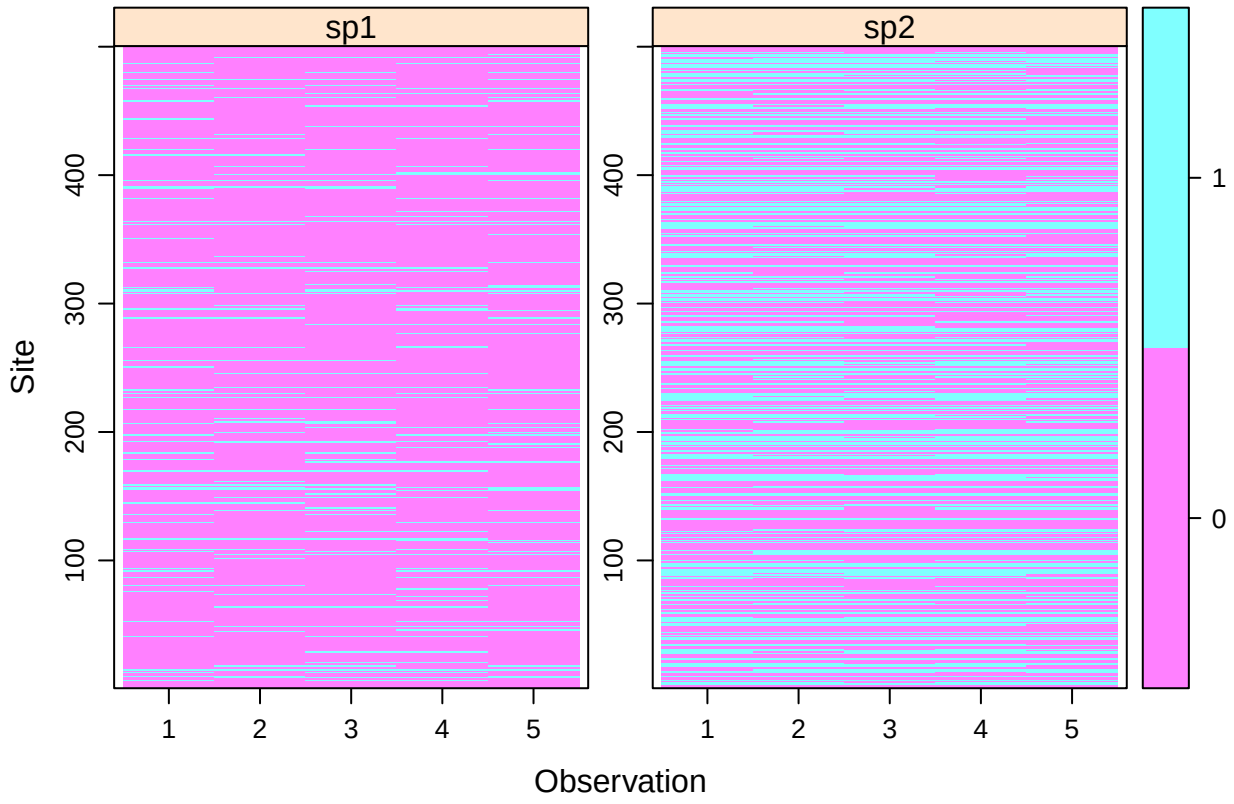
Let's have a look to the data:

```
summary(data)
```

```
## unmarkedFrame Object
##
## 500 sites
## 2 species: sp1 sp2
## Maximum number of observations per site: 5
## Mean number of observations per site:
## sp1: 5  sp2: 5
## Sites with at least one detection:
## sp1: 113  sp2: 238
## Tabulation of y observations:
## sp1:
##    0    1
## 2210  290
## sp2:
##    0    1
## 1425 1075
```

And in particular the detections and non-detections:

```
plot(data)
```



Now we specify the effects we would like to consider on the occupancy and detection probabilities. The thing is that the function `occuMulti` doesn't work directly on the occupancy probabilities but on the so-called natural parameters (in that specific order):

- $f_1 = \log(\psi_{10}/\psi_{00})$;
- $f_2 = \log(\psi_{01}/\psi_{00})$;
- $f_{12} = \log(\psi_{00}\psi_{11}/\psi_{10}\psi_{01})$,

that is:

- $\psi_{11} = \exp(f_1 + f_2 + f_{12})/\text{den}$;
- $\psi_{10} = \exp(f_1)/\text{den}$;
- $\psi_{01} = \exp(f_2)/\text{den}$, where $\text{den} = 1 + \exp(f_1) + \exp(f_2) + \exp(f_1 + f_2 + f_{12})$:

```
occFormulas <- c('~1','~1','~1')
```

To specify the effects on detection, there is no difficulty:

```
detFormulas <- c('~1','~1')
```

We fit a model with constant natural parameters and constant detection probabilities

```
fit <- occuMulti(detFormulas,occFormulas,data)
```

Display the result:

```
fit
```

```
##
## Call:
## occuMulti(detformulas = detFormulas, stateformulas = occFormulas,
##     data = data)
##
```

```
## Occupancy:
##                         Estimate    SE      z  P(>|z|)
## [sp1] (Intercept)       -0.8862 0.139 -6.355 2.09e-10
## [sp2] (Intercept)        0.0636 0.103  0.619 5.36e-01
## [sp1:sp2] (Intercept)   -0.7042 0.225 -3.136 1.71e-03
##
## Detection:
##                    Estimate     SE      z   P(>|z|)
## [sp1] (Intercept)  -0.0132 0.0906 -0.146  8.84e-01
## [sp2] (Intercept)   2.2351 0.0981 22.777 7.72e-115
##
## AIC: 2758.236
```

Get the natural parameter and detection estimates:

```
mle <- fit@opt$par
names(mle) <- c('f1','f2','f12','lp1','lp2')
```

Get the occupancy estimates:

```
den <- 1 + exp(mle['f1'])+exp(mle['f2'])+exp(mle['f1']+mle['f2']+mle['f12'])
psi11hat <- exp(mle['f1']+mle['f2']+mle['f12'])/den
psi10hat <- exp(mle['f1'])/den
psi01hat <- exp(mle['f2'])/den
```

I do it by hand to understand how `unmarked` works. The easy way is to use `predict(fit,'state')`.

Get the detection estimates:

```
p1hat <- plogis(mle['lp1'])
p2hat <- plogis(mle['lp2'])
```

Again I do it by hand, but `unmarked` can do it for you with `predict(fit,'det')`.

Now compare the parameters we used to simulate the data (left column) to the parameter estimates (right column)

```
res <- data.frame(real = c(psiS1,
                           psiS2,
                           psi01,
                           psi10,
                           psi11,
                           ps[1],
                           ps[2]),
                  estim = c(psi10hat+psi11hat,
                            psi01hat+psi11hat,
                            psi01hat,
                            psi10hat,
                            psi11hat,
                            p1hat,
                            p2hat))
rownames(res) <- c('marginal_occ1','marginal_occ2','psi01','psi10','psi11','det1','det2')
res
```

```
##                     real      estim
## marginal_occ1 0.22857143 0.23354481
## marginal_occ2 0.48571429 0.47600454
## psi01         0.46285714 0.39540621
## psi10         0.20571429 0.15294648
```

```
## psi11          0.02285714 0.08059832
## det1           0.50000000 0.49669782
## det2           0.90000000 0.90335332
```

If you just want to get the parameter estimates directly:

```
# detection
predict(fit,'det',species=1)[1,]
```

```
##   Predicted         SE     lower     upper
## 1 0.4966978 0.02264026 0.4523237 0.5410719
```

```
predict(fit,'det',species=2)[1,]
```

```
##   Predicted          SE     lower     upper
## 1 0.9033533 0.008567074 0.8865622 0.9201445
```

```
# marginal occupancy
predict(fit,'state',species=1)[1,]
```

```
## Bootstrapping confidence intervals with 100 samples
```

```
##   Predicted         SE     lower     upper
## 1 0.2335448 0.02867384 0.1867353 0.2919506
```

```
predict(fit,'state',species=2)[1,]
```

```
## Bootstrapping confidence intervals with 100 samples
```

```
##   Predicted        SE     lower     upper
## 1 0.4760045 0.0268335 0.4320817 0.5302099
```

```
# conditional occupancy
predict(fit,'state',species=1,cond='sp2')[1,] # species 1 | species 2 present
```

```
## Bootstrapping confidence intervals with 100 samples
```

```
##   Predicted         SE     lower     upper
## 1 0.1693226 0.03534907 0.1130592 0.2308998
```

```
predict(fit,'state',species=1,cond='-sp2')[1,] # species 1 | species 2 absent
```

```
## Bootstrapping confidence intervals with 100 samples
```

```
##   Predicted         SE     lower     upper
## 1 0.2918851 0.02951294 0.2430333 0.3499646
```

```
predict(fit,'state',species=2,cond='sp1')[1,] # species 2 | species 1 present
```

```
## Bootstrapping confidence intervals with 100 samples
```

```
##   Predicted         SE     lower     upper
## 1 0.3451086 0.05456903 0.2511465 0.4491831
```

```
predict(fit,'state',species=2,cond='-sp1')[1,] # species 2 | species 1 absent
```

```
## Bootstrapping confidence intervals with 100 samples
```

```
##   Predicted         SE     lower     upper
## 1 0.5158895 0.02033788 0.4754084 0.5544719
```

R version used

```
sessionInfo()
```

```
## R version 3.6.2 (2019-12-12)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Catalina 10.15.4
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] fr_FR.UTF-8/fr_FR.UTF-8/fr_FR.UTF-8/C/fr_FR.UTF-8/fr_FR.UTF-8
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
## [1] unmarked_0.13-2     Rcpp_1.0.4          lattice_0.20-38
## [4] mipfp_3.2.1         numDeriv_2016.8-1.1 Rsolnp_1.16
## [7] cmm_0.12
##
## loaded via a namespace (and not attached):
##  [1] codetools_0.2-16 digest_0.6.25    MASS_7.3-51.4    truncnorm_1.0-8
##  [5] plyr_1.8.6       grid_3.6.2       magrittr_1.5     evaluate_0.14
##  [9] rlang_0.4.5      stringi_1.4.6    sp_1.4-1         raster_3.0-12
## [13] Matrix_1.2-18    rmarkdown_2.1    tools_3.6.2      stringr_1.4.0
## [17] xfun_0.12        yaml_2.2.1       compiler_3.6.2   htmltools_0.4.0
## [21] knitr_1.28
```