Addressing non-equilibrium in species distribution modeling

Javier Fernández López

19/06/2023

Non-equilibrium

Animal distribution is often driven by different environmental predictors such as land cover, temperature, precipitation, etc. When using correlative models to predict species distributions, we benefit from those relationships between species presence and environmental cartographic predictors, to build maps about species distribution according to those predictors. Suitability maps define those areas where the predictor values are similar than those places where the species is actually distributed, creating a potential distribution maps.

However, there are many other factors affecting species distributions: biological interactions, dispersal ability, or human interactions can drastically model the species distribution, see Peterson 2009 for an extended explanation. This sitiuation is sometimes called "non-equilibrium", in the sens that the "species is not occupying all its potential area".

Correlative model issues with non-equilibrium species

Though this theory can be discussed, it's true that when performing species distribution models we recognize areas predicted by our models where the species could biologically exists, but from where they were removed by human pressures or other factors. As an example we can think about the distribution of wolves in Spain. Thought they were distributed all across the country in XVIII century, current populations are enclosed in half-north Spain after human pressure and hunting. In many cases, when we perform a correlative distribution model with current presence data, predictions will include southern areas where the wolf is currently absent, since the model just recognize the relationship between presences and environmental conditions. If suitable conditions does exist in southern areas, the model will predict wolf in south of Spain.

Modelling species distributions (equilibrium)

Let's simulate an environmental predictor (temperature) that drives the distribution of our virtual species following the formula:

$$y_i \sim Bernoulli(\Psi_i) \quad \text{(Eq 1)}$$

and Ψ parameter follows:

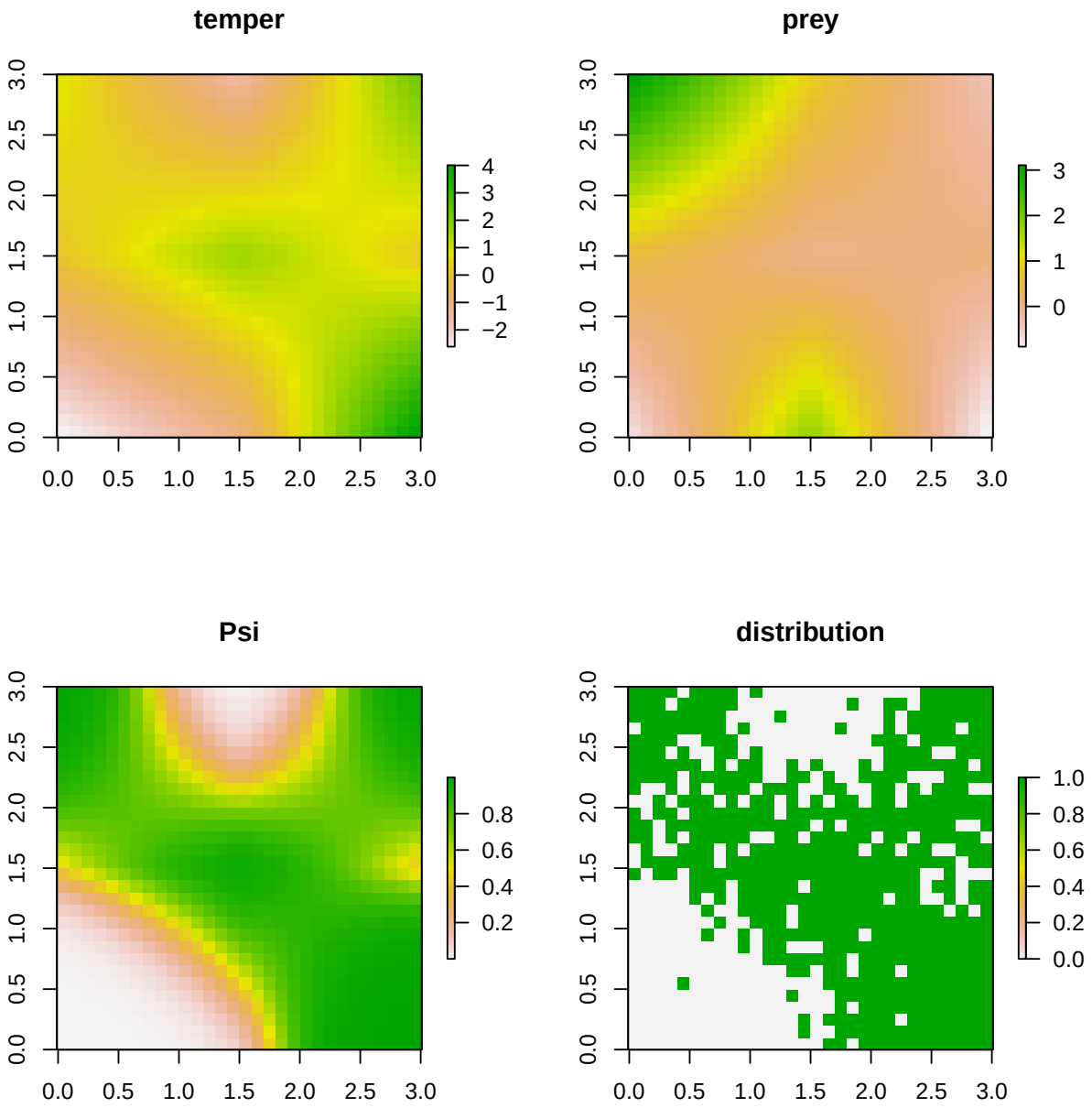
$$logit(\Psi_i) = \beta_0 + \beta_1 * temperature_i \quad \text{(Eq 2)}$$

So we just choose some values for coefficients and run the simulation:

- $\beta_0 = -1.2$
- $\beta_1 = 2.9$

```
library(plgp)
library(mvtnorm)
library(raster)
```

```r
library(spBayes)
library(spOccupancy)
set.seed(4)

# https://bookdown.org/rbg/surrogates/chap5.html
temper <- raster(nrows = 3, ncols = 3, xmn = 0, xmx = 3, ymn = 0, ymx = 3)
temper[] <- rnorm(9)
temper <- disaggregate(temper, 10, "bilinear")
names(temper) <- "temper"

prey <- raster(nrows = 3, ncols = 3, xmn = 0, xmx = 3, ymn = 0, ymx = 3)
prey[] <- rnorm(9)
prey <- disaggregate(prey, 10, "bilinear")
names(prey) <- "prey"

psi <- exp(-1.2 + 2.9*temper + 1.2*prey)/(1+exp(-1.2 + 2.9*temper+ 1.2*prey))
distr <- psi
distr[] <- rbinom(900, 1, psi[])


par(mfrow = c(2,2))
plot(temper, main = "temper")
plot(prey, main = "prey")
plot(psi, main = "Psi")
plot(distr, main = "distribution")
```



Now we can simulate a sampling situation in which we visit 200 places 4 times to account for imperfect detection (probability to detect the species if it is actually present). Fortunately, our species is not difficult to detect, so let's say detection probability is 0.7 per visit. We'll keep this value constant across space and time (same detectability in all sampling visits).

```
n <- 200
sampID <- sample(1:900, n)
coords <- xyFromCell(distr, sampID)
occ.covs <- as.matrix(data.frame(temper = temper[sampID], prey = prey[sampID]))
colnames(occ.covs) <- c("temper","prey")
y <- matrix(NA, n, 4)
```

```r
for (j in 1:4){
  for (i in 1:length(sampID)){
    y[i,j] <- rbinom(1, extract(distr,sampID[i]), 0.7)
  }
}
```

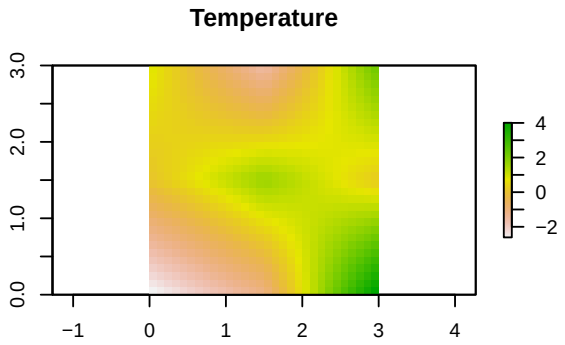
Now we can fit a site-occupancy model. We'll use spOccupancy (fast and flexible Bayesian inference).

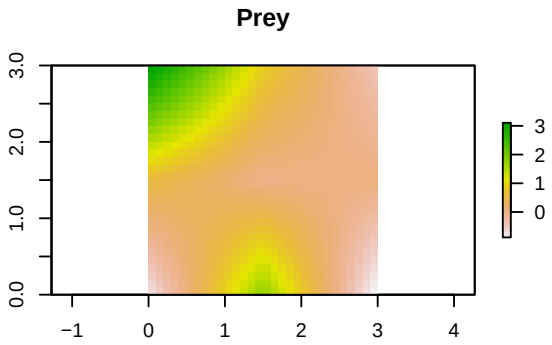
```r
data.list <- list(y = y,
                  occ.covs = occ.covs,
                  coords = coords)

# Number of batches
n.batch <- 10
# Batch length
batch.length <- 25
n.iter <- n.batch * batch.length
# Priors
prior.list <- list(beta.normal = list(mean = 0, var = 2.72),
                   alpha.normal = list(mean = 0, var = 2.72),
                   sigma.sq.ig = c(2, 2),
                   phi.unif = c(3/1, 3/.1))
# Initial values
inits.list <- list(alpha = 0, beta = 0,
                   phi = 3 / .5,
                   sigma.sq = 2,
                   w = rep(0, n), # CHANGED
                   z = apply(y, 1, max, na.rm = TRUE))
# Tuning
tuning.list <- list(phi = 1)

outEq <- PGOcc(occ.formula = ~ temper + prey,
               det.formula = ~ 1,
               data = data.list,
               inits = inits.list,
               n.samples = 5000,
               priors = prior.list,
               n.report = 10,
               n.burn = 50,
               n.chains = 1,
               verbose = F)
summary(outEq)
```

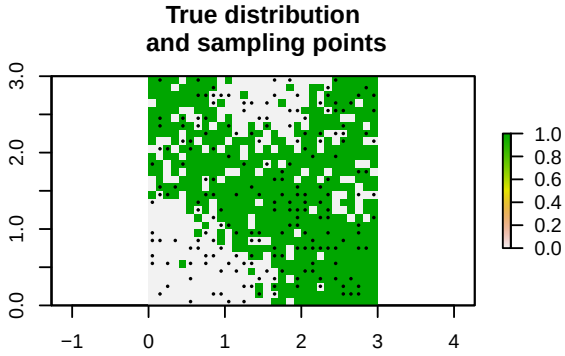
```
##
## Call:
## PGOcc(occ.formula = ~temper + prey, det.formula = ~1, data = data.list,
##     inits = inits.list, priors = prior.list, n.samples = 5000,
##     verbose = F, n.report = 10, n.burn = 50, n.chains = 1)
##
## Samples per Chain: 5000
## Burn-in: 50
## Thinning Rate: 1
## Number of Chains: 1
## Total Posterior Samples: 4950
```

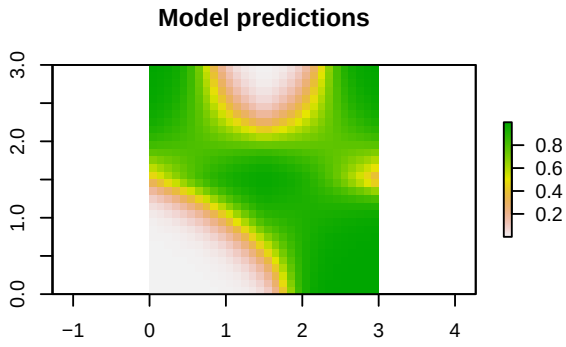
```
## Run Time (min): 0.0225
##
## Occurrence (logit scale):
##                Mean     SD    2.5%     50%   97.5% Rhat  ESS
## (Intercept) -1.7447 0.4634 -2.6835 -1.7353 -0.8672   NA  961
## temper       3.7187 0.5372  2.7165  3.6998  4.8315   NA  639
## prey         1.6059 0.4834  0.6876  1.5866  2.5718   NA 1565
##
## Detection (logit scale):
##                Mean     SD   2.5%   50%  97.5% Rhat  ESS
## (Intercept) 0.7933 0.1012 0.5996 0.792 0.9957   NA 3440
```

```r
# Do prediction.
library(stars)
pred.0 <- cbind(1, temper[], prey[])
coords.0 <- as.matrix(xyFromCell(distr, 1:900))

out.sp.pred_outEq <- predict(outEq, pred.0)
# Produce a species distribution map (posterior predictive means of occupancy)

plot.dat_outEq <- data.frame(x = coords.0[,1],
                      y = coords.0[,2],
                      mean.psi = apply(out.sp.pred_outEq$psi.0.samples, 2, mean),
                      sd.psi = apply(out.sp.pred_outEq$psi.0.samples, 2, sd))
mPred_outEq <- temper
mPred_outEq[] <- plot.dat_outEq$mean.psi

par(mfrow = c(3,2))
plot(temper, main = "Temperature")
plot(prey, main = "Prey")
plot(psi, main = "True Psi")
plot(distr, main = "True distribution\n and sampling points")
points(xyFromCell(distr,sampID), pch = 16, cex = .4)
plot(mPred_outEq, main = "Model predictions")
```









Modelling species distributions (non-equilibrium)

Let's see what happen if we artificially remove southern presences from the south of our study area (simulating hunting pressure or similar). Note that Psi will be the same (that is, the potential distribution of the species would be the same, since the temperature has not change... however distribution will change according to external pressures).

We will repeat the same sampling scheme and we'll applied the same model.

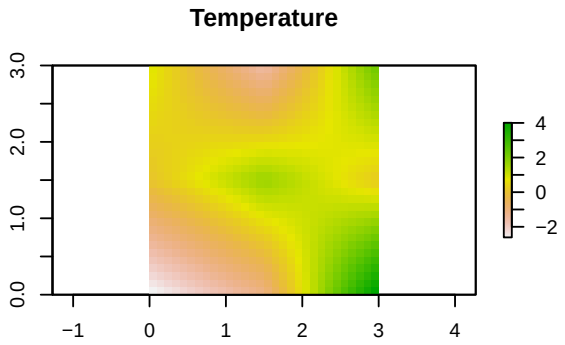
```
# Removing presences from the south of the study area
distr[510:900] <- 0
```

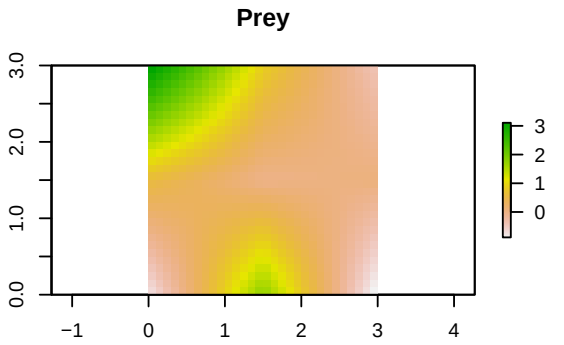
```r
# Sampling again
y <- matrix(NA, n, 4)

for (j in 1:4){
  for (i in 1:length(sampID)){
    y[i,j] <- rbinom(1, extract(distr,sampID[i]), 0.7)
  }
}



data.list <- list(y = y,
                  occ.covs = occ.covs,
                  coords = coords)

# Number of batches
n.batch <- 10
# Batch length
batch.length <- 25
n.iter <- n.batch * batch.length
# Priors
prior.list <- list(beta.normal = list(mean = 0, var = 2.72),
                   alpha.normal = list(mean = 0, var = 2.72),
                   sigma.sq.ig = c(2, 2),
                   phi.unif = c(3/1, 3/.1))
# Initial values
inits.list <- list(alpha = 0, beta = 0,
                   phi = 3 / .5,
                   sigma.sq = 2,
                   w = rep(0, n), # CHANGED
                   z = apply(y, 1, max, na.rm = TRUE))
# Tuning
tuning.list <- list(phi = 1)

outEq <- PGOcc(occ.formula = ~ temper + prey,
               det.formula = ~ 1,
               data = data.list,
               inits = inits.list,
               n.samples = 5000,
               priors = prior.list,
               n.report = 10,
               n.burn = 50,
               n.chains = 1,
               verbose = F)
summary(outEq)
```

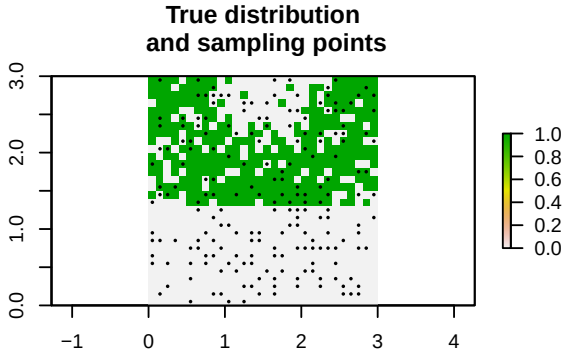
```
##
## Call:
## PGOcc(occ.formula = ~temper + prey, det.formula = ~1, data = data.list,
##     inits = inits.list, priors = prior.list, n.samples = 5000,
##     verbose = F, n.report = 10, n.burn = 50, n.chains = 1)
##
## Samples per Chain: 5000
## Burn-in: 50
```

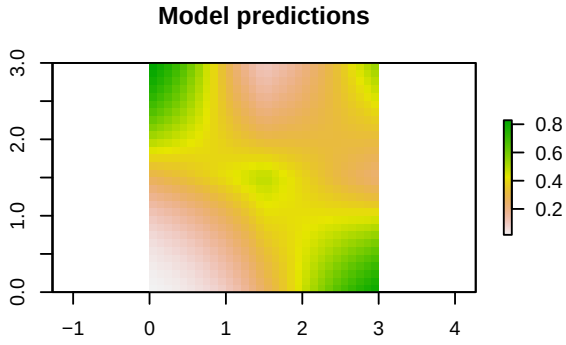
```
## Thinning Rate: 1
## Number of Chains: 1
## Total Posterior Samples: 4950
## Run Time (min): 0.0131
##
## Occurrence (logit scale):
##                Mean     SD    2.5%     50%   97.5% Rhat  ESS
## (Intercept) -1.5400 0.3058 -2.1703 -1.5328 -0.9672   NA 1972
## temper       0.9417 0.2377  0.4957  0.9361  1.4154   NA 2003
## prey         0.8539 0.3321  0.2158  0.8468  1.5187   NA 2692
##
## Detection (logit scale):
##                Mean     SD   2.5%    50% 97.5% Rhat  ESS
## (Intercept) 0.8545 0.1399 0.5753 0.8567 1.131   NA 3474
```

```r
# Do prediction.
library(stars)
pred.0 <- cbind(1, temper[], prey[])
coords.0 <- as.matrix(xyFromCell(distr, 1:900))

out.sp.pred_outEq <- predict(outEq, pred.0)
# Produce a species distribution map (posterior predictive means of occupancy)

plot.dat_outEq <- data.frame(x = coords.0[,1],
                      y = coords.0[,2],
                      mean.psi = apply(out.sp.pred_outEq$psi.0.samples, 2, mean),
                      sd.psi = apply(out.sp.pred_outEq$psi.0.samples, 2, sd))
mPred_outEq <- temper
mPred_outEq[] <- plot.dat_outEq$mean.psi

par(mfrow = c(3,2))
plot(temper, main = "Temperature")
plot(prey, main = "Prey")
plot(psi, main = "True Psi")
plot(distr, main = "True distribution\n and sampling points")
points(xyFromCell(distr,sampID), pch = 16, cex = .4)
plot(mPred_outEq, main = "Model predictions")
```









As we can see, even removing presences from the south and conducting again the sampling, model predictions detect the positive relationship between the species presence and the temperature. Therefore, our model predicts species distribution also in shouthern areas, where the temperature is suitable, but the species is not present. Remember that we conducted the sampling again, so no presences are including in the model for southern areas. The unique effect of these new "zeros" is to decrease the total average suitability (comparing with the previous model). How to deal with this situation? While there are probably many options, one solution could be to recognize a "pure spatial" effect in our species distribution, that is, there are purely spatial processes that we actually don't know that also govern the species distribution. This is actually the same as recognize that there are other variables that we are not including in the model, but that we know that they have a pure spatial origin and therefore they can be modeled by accounting for the spatial location of our sampling sites. There are many ways to do that. Here we will use Gaussian process approach

implemented in the spOccupancy R package.

```
outNonEq <- spPGOcc(occ.formula = ~ temper + prey,
                    det.formula = ~ 1,
                    data = data.list,
                    inits = inits.list,
                    n.batch = n.batch,
                    batch.length = batch.length,
                    priors = prior.list,
                    cov.model = "exponential",
                    tuning = tuning.list,
                    NNGP = FALSE,
                    n.neighbors = 5,
                    search.type = 'cb',
                    n.report = 10,
                    n.burn = 50,
                    n.chains = 1)
```

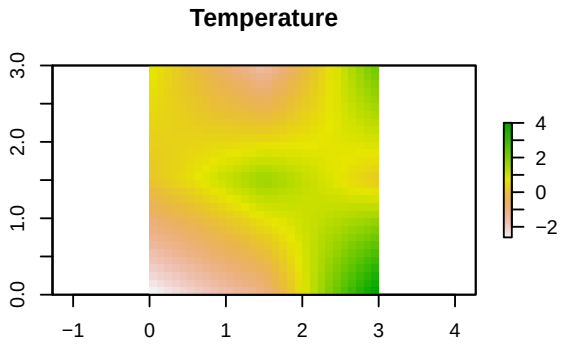
```
## ----------------------------------------
##  Preparing to run the model
## ----------------------------------------
## ----------------------------------------
##  Model description
## ----------------------------------------
## Spatial Occupancy Model with Polya-Gamma latent
## variable fit with 200 sites.
##
## Samples per chain: 250 (10 batches of length 25)
## Burn-in: 50
## Thinning Rate: 1
## Number of Chains: 1
## Total Posterior Samples: 200
##
## Using the exponential spatial correlation model.
##
## Source compiled with OpenMP support and model fit using 1 thread(s).
##
## Adaptive Metropolis with target acceptance rate: 43.0
## ----------------------------------------
##  Chain 1
## ----------------------------------------
## Sampling ...
## Batch: 10 of 10, 100.00%
```

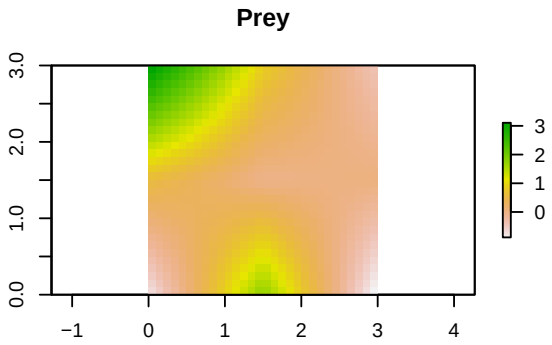
```
summary(outNonEq)
```

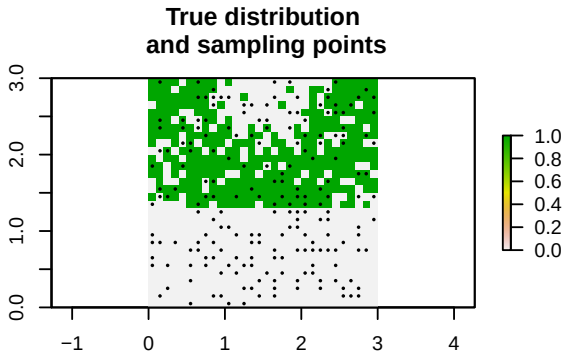
```
##
## Call:
## spPGOcc(occ.formula = ~temper + prey, det.formula = ~1, data = data.list,
##     inits = inits.list, priors = prior.list, tuning = tuning.list,
##     cov.model = "exponential", NNGP = FALSE, n.neighbors = 5,
##     search.type = "cb", n.batch = n.batch, batch.length = batch.length,
##     n.report = 10, n.burn = 50, n.chains = 1)
##
## Samples per Chain: 250
```

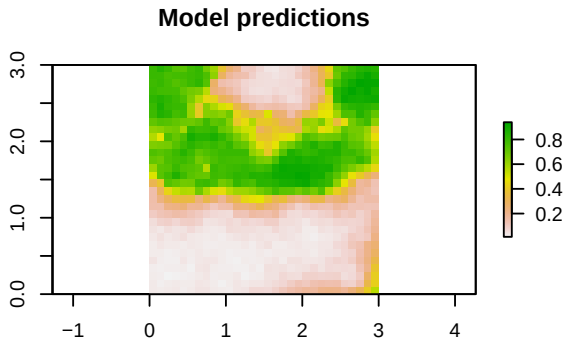
```
## Burn-in: 50
## Thinning Rate: 1
## Number of Chains: 1
## Total Posterior Samples: 200
## Run Time (min): 0.0257
##
## Occurrence (logit scale):
##                Mean     SD    2.5%     50%   97.5% Rhat ESS
## (Intercept) -2.1949 0.8081 -3.9264 -2.1545 -0.7186   NA  15
## temper       1.5729 0.7228 -0.0483  1.5527  3.0673   NA  13
## prey         0.8099 1.1210 -1.5588  0.8087  2.9999   NA  11
##
## Detection (logit scale):
##             Mean     SD   2.5%    50%  97.5% Rhat ESS
## (Intercept) 0.85 0.1353 0.6003 0.8419 1.1098   NA 408
##
## Spatial Covariance:
##             Mean     SD   2.5%    50%   97.5% Rhat ESS
## sigma.sq 10.8054 4.2131 4.9849 9.9208 21.5577   NA   7
## phi       3.1466 0.1749 3.0013 3.1052  3.5847   NA  19
```

```
# Do prediction.
library(stars)
pred.0 <- cbind(1, temper[], prey[])
coords.0 <- as.matrix(xyFromCell(distr, 1:900))

out.sp.pred_outNonEq <- predict(outNonEq, pred.0, coords.0, verbose = FALSE)
# Produce a species distribution map (posterior predictive means of occupancy)

plot.dat_outNonEq <- data.frame(x = coords.0[,1],
                      y = coords.0[,2],
                      mean.psi = apply(out.sp.pred_outNonEq$psi.0.samples, 2, mean),
                      sd.psi = apply(out.sp.pred_outNonEq$psi.0.samples, 2, sd))
mPred_outNonEq <- temper
mPred_outNonEq[] <- plot.dat_outNonEq$mean.psi

par(mfrow = c(3,2))
plot(temper, main = "Temperature")
plot(prey, main = "Prey")
plot(psi, main = "True Psi")
plot(distr, main = "True distribution\n and sampling points")
points(xyFromCell(distr,sampID), pch = 16, cex = .4)
plot(mPred_outNonEq, main = "Model predictions")
```









Not bad! :) :)

```
# Some correlation functions for the multivariate random normal:
gauss <- function(phi, d){
  return(exp(-(phi*d)^2))
}
plot(seq(0,5,0.1), gauss(1,seq(0,5,0.1)), ty = "l", lwd = 3, col = "darkred",
     ylab = "correlation", xlab = "distance")

expo <- function(phi, d){
  return(exp(-(phi*d)))
}
```

```r
lines(seq(0,5,0.1), expo(1,seq(0,5,0.1)), lwd = 3, col = "darkblue")

spher <- function(phi, d){
  r <- rep(NA, length(d))
  for(i in 1:length(d)){
    if(d[i]<1/phi){
      r[i] <- 1-(1.5*phi*d[i]) + (0.5*(phi*d[i])^3)
    }else{r[i] <-  0}
  }
  return(r)
}
lines(seq(0,5,0.1), spher(1,seq(0,5,0.1)), lwd = 3, col = "darkgreen")

mater <- function(phi, d, v){
  return(((1/((2^(v-1))*gamma(v)))*(phi*d)^v)*besselK(x = (phi*d), nu = v))
}
lines(seq(0,5,0.1), mater(1,seq(0,5,0.1),1), lwd = 3, col = "gold2")
legend("topright", legend=c("Gaussian", "Exponential","Spherical", "Matern"),
       col=c("darkred", "darkblue", "darkgreen", "gold2"),lty = 1,lwd = 2,  cex=1.2)
```