Hierarchical Bayesian species distribution models with the **hSDM** R Package

Ghislain Vieilledent[⋆,1]

[⋆] **Corresponding author:** \E-mail: ghislain.vieilledent@cirad.fr \Phone: +33.(0)4.67.59.37.51 \Fax: +33.(0)4.67.59.39.09
[1] **Cirad** – UPR BSEF, F–34398 Montpellier, France

Abstract

Work in progess...

Keywords: R

1 Introduction

2 Species distribution models

2.1 Binomial model

2.1.1 Data generation

For data generation, we can import virtual altitude data in R. Altitude will be used as an explicative variable to determine the habitat suitability of a virtual species through a probability of presence. Altitudinal data are available on the website of the **hSDM** R package hosted on Sourceforge (`http://hsdm.sourceforge.net/altitude.csv`).

These data can be transformed into a raster using function `rasterFromXYZ()` of the **raster** package. The raster has 2500 cells (50 columns and 50 rows) and the altitude is comprised roughly between 100 and 600 m (Fig. 1). For logistic regression, explicative variables are usually centered and scaled to facilitate the inference of model parameters.

```r
# Import altitudinal data
library(raster)
fname <- "http://hsdm.sourceforge.net/altitude.csv"
alt.df <- read.csv(fname,header=TRUE)
alt.orig <- rasterFromXYZ(alt.df)
plot(alt.orig)
# Center and scale altitudinal data
alt <- scale(alt.orig,center=TRUE,scale=TRUE)
plot(alt)
```

Using the altitude data we can simulate the presence of the virtual species using a Binomial distribution. We arbitrarily set the trials of the Binomial distribution to 1. Thus, the random variable y representing the presence of the species, follows a Bernoulli distribution of probability θ. A quadratic effect of the altitude (variable denoted x) is used to compute the probability of presence of the species using a logit link function (Eq. 1).

$$
\begin{aligned}
y_i &\sim \mathcal{B}ernoulli(\theta_i) \\
\text{logit}(\theta_i) &= \beta_0 + \beta_1 x_i + \beta_2 x_i^2
\end{aligned}
\tag{1}
$$

Using matrix notation, the linear model can be written $\text{logit}(\theta_i) = X\beta$. We fix the parameters to $\beta_0 = 1$, $\beta_1 = 2$ and $\beta_2 = -4$. The species has a higher probability of presence at intermediate altitude (Fig. 2).

```r
# Load hSDM library
library(hSDM)
# Target parameters
```

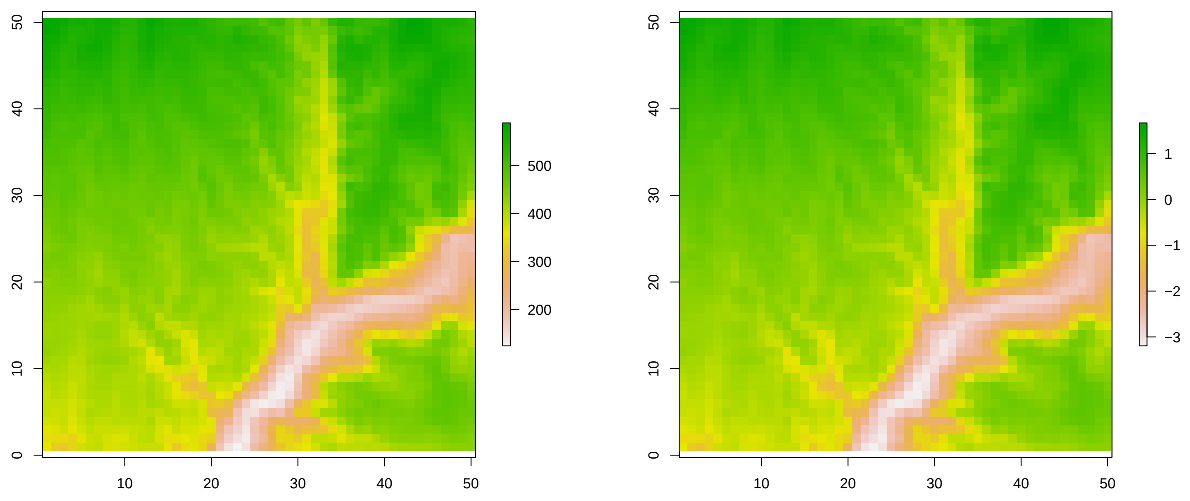

Figure 1: **Altitudinal data**. Original values (in m) on the left. Centered and scaled values on the right.

```r
beta.target <- matrix(c(1,2,-4),ncol=1)
# Matrix of covariates (including the intercept)
X <- cbind(rep(1,ncell(alt)),values(alt),(values(alt))^2)
# Probability of presence as a quadratic function of altitude
logit.theta <- X %*% beta.target
theta <- inv.logit(logit.theta)
# Transform the probability of presence into a raster
theta <- rasterFromXYZ(cbind(coordinates(alt),theta))
# Plot the probability of presence
plot(theta,col=rev(heat.colors(255)))
```

We can assume a number n of points in the landscape where we have been able to observe or not the presence of the species. We can simulate the presence or absence of the species at these n points given our model (Fig. 3).

```r
# Load dismo library
library(dismo)
# Number of observations
nobs <- 200
# Set seed for repeatability
seed <- 1234
# Sample the observations in the landscape
obs <- randomPoints(alt,nobs)
# Extract altitude data for observations
```

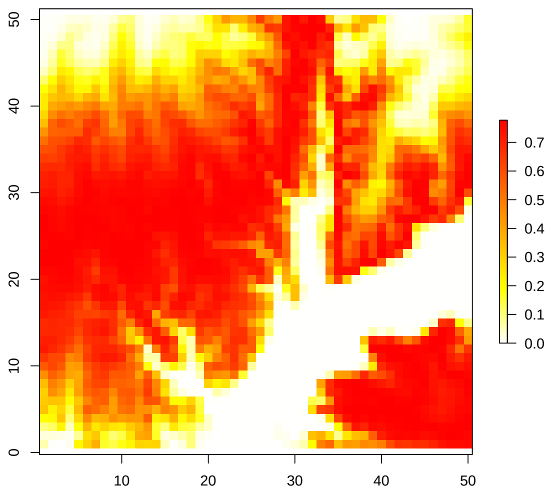

Figure 2: **Probability of presence**.

```
alt.obs <- extract(alt,obs)
# Compute theta for these observations
X.obs <- cbind(rep(1,nobs),alt.obs,alt.obs^2)
logit.theta.obs <- X.obs %*% beta.target
theta.obs <- inv.logit(logit.theta.obs)
# Simulate observations
trials <- rep(1,nobs)
set.seed(seed)
Y <- rbinom(nobs,trials,theta.obs)
# Group explicative and response variables in a data-frame
data.obs.df <- data.frame(Y,trials=trials,alt=alt.obs)
# Transform observations in a spatial object
library(sp)
data.obs <- SpatialPointsDataFrame(coords=coordinates(obs),data=data.obs.df)
# Plot observations
plot(alt.orig)
points(data.obs[data.obs$Y==1,],pch=16)
points(data.obs[data.obs$Y==0,],pch=1)
```

2.1.2 Parameter inference using the `hSDM.binomial()` function

The `hSDM.binomial()` function performs a Binomial logistic regression in a Bayesian framework. Before using this function we need to prepare a bit the data for parameter inference and prediction. For parameter inference, we add the quadratic term for altitude

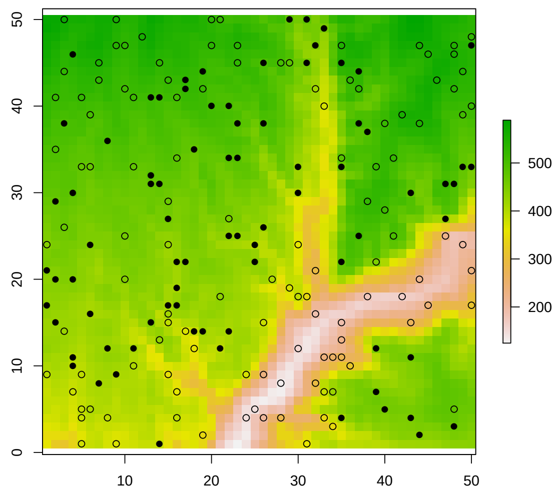

Figure 3: **Observation points**. Presences (full circles) and absences (empty circles) are localized on the altitude map (in m).

in the data frame associated to observations.

```
data.obs$alt2 <- (data.obs$alt)^2
```

We want to have predictions on the whole landscape, not only at observation points. To directly obtain these predictions, we can create a data frame including altitude data on the whole landscape. This data frame will be used for the `suitability.pred` argument. The data frame for predictions must include the same column names as those used in the formula for the `suitability` argument (i.e. 'alt' and 'alt2' in our example).

```
data.pred <- data.frame(alt=values(alt),alt2=(values(alt))^2)
```

We can now call the `hSDM.binomial()` function. Setting parameter `save.p` to 1, we can save in memory the MCMC values for predictions. These values can be used to compute several statistics for each predictions (mean, median, 95% quantiles). For example, mean and 95% quantiles are useful to estimate the uncertainty around the mean predictions.

```
mod.hSDM.binomial <- hSDM.binomial(presences=data.obs$Y,
                                   trials=data.obs$trials,
                                   suitability=~alt+alt2,
                                   data=data.obs,
                                   suitability.pred=data.pred,
                                   burnin=1000, mcmc=1000, thin=1,
                                   beta.start=0,
                                   mubeta=0, Vbeta=1.0E6,
                                   seed=1234, verbose=1, save.p=1)
```

2.1.3 Analysis of the results

The `hSDM.binomial()` function returns an MCMC (Markov chain Monte Carlo) for each parameter of the model and also for the model deviance. To obtain parameter estimates, MCMC values can be summarized through a call to the `summary()` function from the **coda** package. We can check that the values of the target parameters $\beta_0 = 1$, $\beta_1 = 2$ and $\beta_2 = -4$ are within the 95% confidence interval of the parameter estimates.

```
summary(mod.hSDM.binomial$mcmc)

##
## Iterations = 1001:2000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 1000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                      Mean    SD Naive SE Time-series SE
## beta.(Intercept)   0.887 0.226  0.00715         0.0236
## beta.alt           2.164 0.457  0.01445         0.0583
## beta.alt2         -4.133 0.606  0.01917         0.0958
## Deviance         189.821 2.172  0.06870         0.1433
##
## 2. Quantiles for each variable:
##
##                    2.5%     25%     50%    75%  97.5%
## beta.(Intercept)   0.45   0.739   0.889   1.05   1.28
## beta.alt           1.24   1.855   2.187   2.48   2.98
## beta.alt2         -5.22  -4.575  -4.153  -3.73  -2.93
## Deviance         187.38 188.271 189.234 190.77 195.52
```

Parameters estimates can be compared to results obtained with the `glm()` function.

```
#== glm results for comparison
mod.glm <- glm(cbind(Y,trials-Y)~alt+alt2,family="binomial",data=data.obs)
summary(mod.glm)

##
```

```
## Call:
## glm(formula = cbind(Y, trials - Y) ~ alt + alt2, family = "binomial",
##     data = data.obs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.710  -0.787  -0.002   0.810   2.208
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    0.925      0.249    3.71  0.00021 ***
## alt            2.147      0.548    3.91  9.0e-05 ***
## alt2          -4.137      0.727   -5.69  1.3e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 269.20  on 199  degrees of freedom
## Residual deviance: 187.19  on 197  degrees of freedom
## AIC: 193.2
##
## Number of Fisher Scoring iterations: 8
```

MCMC can also be graphically summarized with a call to the `plot.mcmc()` function, also in the **coda** package. MCMC are plotted with a trace of the sampled output and a density estimate for each variable in the chain (Fig. 4). This can plot can be used to visually check that the chains have converged.

```
plot(mod.hSDM.binomial$mcmc)
```

The `hSDM.binomial()` function also returns two other objects. The first one, `prob.p.latent`, is the predictive posterior mean of the latent variable θ (the probability of presence) for each observation.

```
str(mod.hSDM.binomial$prob.p.latent)

##  num [1:200] 0.34 0.689 0.14 0.271 0.76 ...

summary(mod.hSDM.binomial$prob.p.latent)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.111   0.414   0.397   0.689   0.761
```

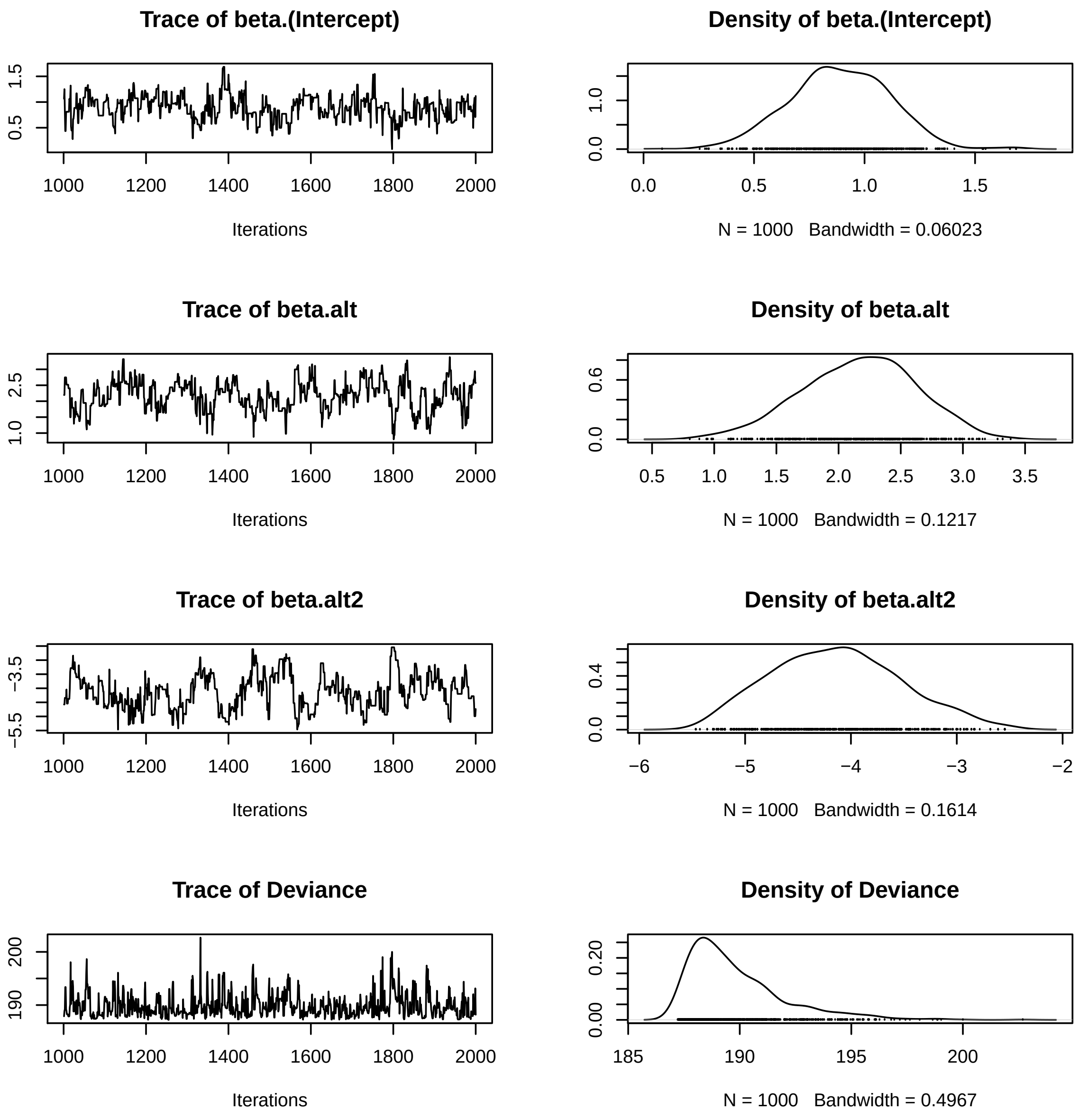

Figure 4: **Trace and density estimate for each variable of the MCMC**.

The second one, `prob.p.pred` is the set of sampled values from the predictive posterior (if parameter `save.p` is set to 1) or the predictive posterior mean (if `save.p` is set to 0) for each prediction. In our example, `save.p` is set to 1 and `prob.p.pred` is an `mcmc` object. Values in `prob.p.pred` can be used to plot the predicted probability of presence on the whole landscape and the uncertainty associated to the predictions (Fig 5).

```r
# Create a raster for predictions
theta.pred.mean <- raster(theta)
# Create rasters for uncertainty
theta.pred.2.5 <- theta.pred.97.5 <- raster(theta)
# Attribute predicted values to raster cells
theta.pred.mean[] <- apply(mod.hSDM.binomial$prob.p.pred,2,mean)
theta.pred.2.5[] <- apply(mod.hSDM.binomial$prob.p.pred,2,quantile,0.025)
theta.pred.97.5[] <- apply(mod.hSDM.binomial$prob.p.pred,2,quantile,0.975)
# Plot the predicted probability of presence and uncertainty
plot(theta.pred.mean,col=rev(heat.colors(255)))
plot(theta.pred.2.5,col=rev(heat.colors(255)))
plot(theta.pred.97.5,col=rev(heat.colors(255)))
```

In our example, we can compare the predictions to the initial probability of presence computed from our model to check that our predictions are correct (Fig. 6).

```r
# Comparing predictions to initial values
plot(theta[],theta.pred.mean[],cex.lab=1.4)
abline(a=0,b=1,col="red",lwd=2)
```

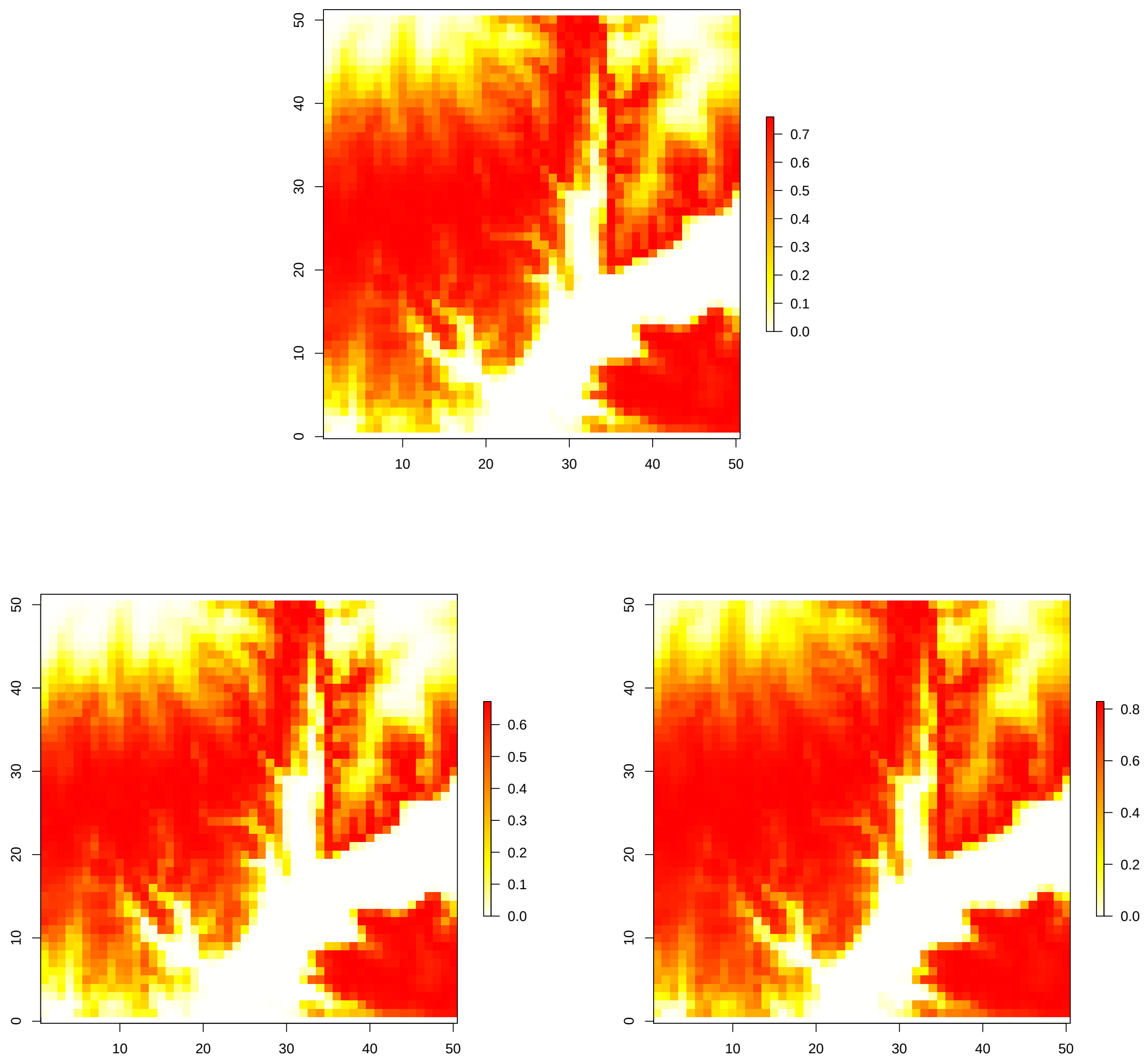

Figure 5: **Predicted probability of presence and uncertainty of predictions**. Mean probability of presence (top), predictions at 2.5% quantile (bottom left) and 97.5% quantile (bottom right) can be plotted from the `mcmc` object `plot.p.pred` returned by function `hSDM.binomial()`.

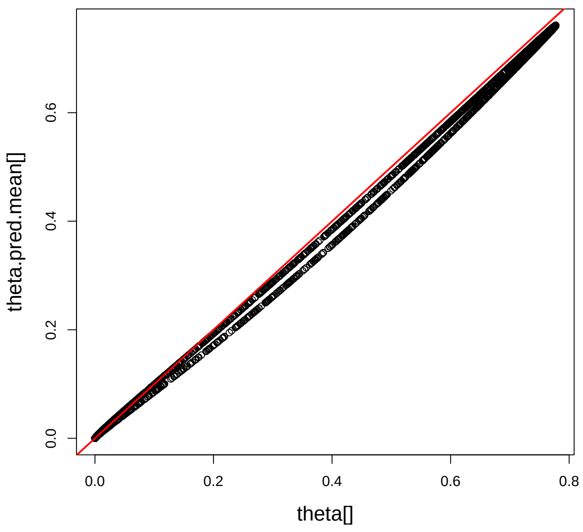

Figure 6: **Predicted vs. initial probabilities of presence**. Initial probabilities of presence are computed from the Binomial logistic regression model with fixed parameters.