8

Species' occurrence records (bei)

8.1 Introduction

A special group of Species Distribution Models (SDMs) focuses on the so-called **occurrence-only records** — pure records of locations where a species occurred (Tsoar et al., 2007). Although such point data set can be considered of interest to geostatistics, standard geostatistical techniques cannot be used to generate (realized) species' distributions using occurrence-only data, mainly for two reasons: (1) absence locations are missing ('1's only), so that it is not possible to analyze the data using e.g. indicator geostatistics; and (2) the sampling is purposive and points are often clustered in both geographical and feature spaces, which typically causes difficulties during the model estimation.

Spatial statisticians (e.g. Diggle (2003); Bivand et al. (2008)) generally believe that geostatistical techniques are suited only for modeling of features that are inherently continuous (spatial fields); discrete objects (points, lines, polygons) should be analyzed using point pattern analysis and similar methods. This exercises tries to bridge this gap. It demonstrates how to combine geostatistical techniques with conceptually different techniques — point pattern analysis and Niche analysis — to allow prediction of species' distributions using regression-kriging. For more info about the theoretical basis for this approach see section 2.6. This chapter is based on an article published in Ecological Modeling journal (Hengl et al., 2009b).

We will use the data set `bei`, distributed together with the `spatstat`[1] package, and used in school books on point pattern analysis by Baddeley (2008) and many other authors. This data set consists of a point map showing observed locations of trees of the species *Beilschmiedia pendula Lauraceae*. This is only a small part of the Barro Colorado Island 50 Hectare Plot[2] Data set (Panama) that contains about 2 million records — complete inventory of all plant species and numerous plant and environmental attributes for six time periods (Leigh et al., 2004).

8.1.1 Preparation of maps

To run the `bei.R` script, you will first need to load the following packages:

```
> library(spatstat)
> library(adehabitat)
> library(gstat)
> library(splancs)
> library(rgdal)
> library(RSAGA)
```

We load the `bei` data set directly from R:

[1] http://spatstat.org
[2] http://www.stri.org/english/research/facilities/terrestrial/barro_colorado/

```
> data(bei)
> str(bei)
```

```
List of 5
 $ window    :List of 5
  ..$ type  : chr "rectangle"
  ..$ xrange: num [1:2] 0 1000
  ..$ yrange: num [1:2] 0 500
  ..$ units :List of 3
  .. ..$ singular  : chr "metre"
  .. ..$ plural    : chr "metres"
  .. ..$ multiplier: num 1
  .. ..- attr(*, "class")= chr "units"
  ..$ area  : num 5e+05
  ..- attr(*, "class")= chr "owin"
 $ n         : int 3604
 $ x         : num [1:3604]  11.7 998.9 980.1 986.5 944.1 ...
 $ y         : num [1:3604] 151 430 434 426 415 ...
 $ markformat: chr "none"
 - attr(*, "class")= chr "ppp"
```

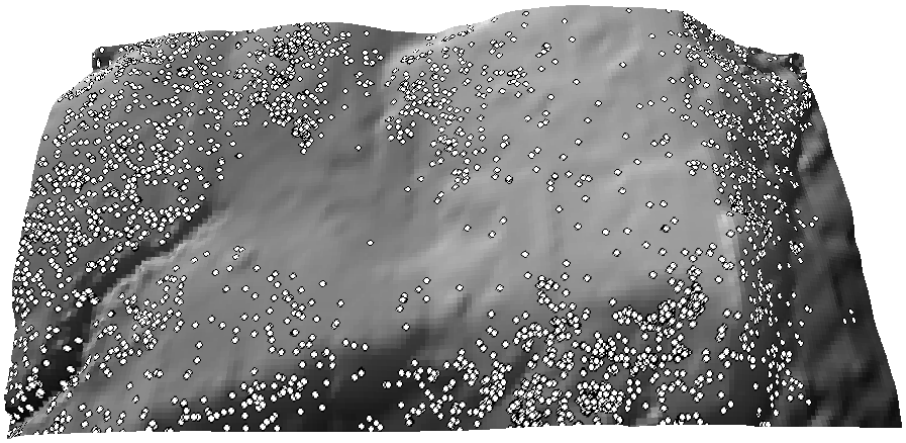

Fig. 8.1: The `bei` data set — locations of *Beilschmiedia pendula Lauraceae* trees, and a 5 m DEM used as environmental predictor. Viewed from south to north.

This shows locations of individual trees (a total of 3604 trees) observed in a 1000 m by 500 m rectangular window[3]. The beginning of the rectangle is at Lat. 9.15125°, Long. -79.8553°. The `bei` object is of type `ppp`, which is the native `spatstat` format (point pattern data set in the two-dimensional plane). If you would try to analyze this data set using some geostatistical package, you would soon find out that not much can be done because this is not even a binary variable.

8.1.2 Auxiliary maps

In addition to the point map a map of elevation and slope gradient is provided in the `bea.extra` data set:

```
> str(bei.extra, max.level=1)
```

```
List of 4
 $ elev :List of 11
  ..- attr(*, "class")= chr "im"
 $ grad :List of 11
  ..- attr(*, "class")= chr "im"
```

[3] What makes this data set especially suitable for this exercise is the fact that the complete population of the trees has been mapped for the area of interest.

We can extend the initial list of covariates and attach two more maps that we can derive in SAGA and then import back into R — the wetness index and vertical distance from the channel network (Fig. 8.2):

```
> grids <- as(bei.extra[[1]], "SpatialGridDataFrame")
> names(grids)[1] <- "elev"
> grids$grad <- as(bei.extra[[2]], "SpatialGridDataFrame")$v
> write.asciigrid(grids["elev"], "dem.asc")
> write.asciigrid(grids["grad"], "grad.asc")
# Generate the wetness index and vertical distance from the channel network:
> rsaga.esri.to.sgrd(in.grids="dem.asc", out.sgrds="dem.sgrd", in.path=getwd())
# Filter the spurious sinks:
> rsaga.geoprocessor(lib="ta_preprocessor", module=2,
+   param=list(DEM="dem.sgrd", RESULT="dem_f.sgrd"))
> rsaga.geoprocessor(lib="ta_hydrology", module=15, param=list(DEM="dem_f.sgrd",
+   C="catharea.sgrd", GN="catchslope.sgrd", CS="modcatharea.sgrd", SB="twi.sgrd", T=10))
> rsaga.geoprocessor(lib="ta_channels", module=0, param=list(ELEVATION="dem.sgrd",
+   CHNLNTWRK="chnlntwrk.sgrd", CHNLROUTE="channel_route.sgrd", SHAPES="channels.shp",
+   INIT_GRID="dem_f.sgrd", DIV_CELLS=10, MINLEN=30))
> rsaga.geoprocessor(lib="ta_channels", module=3, param=list(ELEVATION="dem.sgrd",
+   CHANNELS="chnlntwrk.sgrd", ALTITUDE="achan.sgrd", THRESHOLD=0.1, NOUNDERGROUND=TRUE))
> rsaga.sgrd.to.esri(in.sgrds=c("twi.sgrd","achan.sgrd"),
+   out.grids=c("twi.asc","achan.asc"), prec=1, out.path=getwd())
> grids$achan <- readGDAL("achan.asc")$band1
> grids$twi <- readGDAL("twi.asc")$band1
```

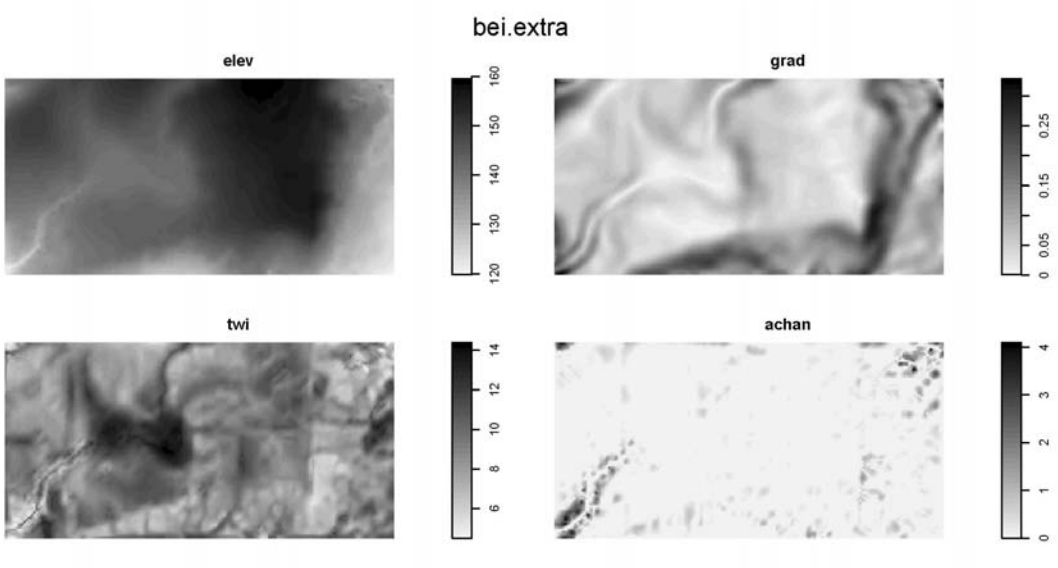

Fig. 8.2: Auxiliary (environmental) maps used to explain species' distribution: Elevation (`elev`), Slope in % (`grad`), Topographic Wetness Index (`twi`) and Altitude above channel network in m (`achan`) derived in SAGA GIS.

where `twi.sgrd` is the Topographic Wetness Index, and `achan.sgrd` is the Altitude above channel network. For more info about the SAGA syntax, see section 3.1.2 or refer to Conrad (2007).

We will now implement all steps described in §2.6 to predict the spatial density of trees over the area of interest (M=20301 grid nodes). To test our algorithm we will use a 20% sub-sample of the original population and then validate the accuracy of our technique versus the whole population.

8.2 Species distribution modeling

8.2.1 Kernel density estimation

We start by estimating a suitable bandwidth size for kernel density estimation (Eq.2.6.3). For this we use the method of Berman and Diggle (1989), as described in Bivand et al. (2008, p.166–167), that looks for the smallest Mean Square Error (MSE) of a kernel estimator:

```
> gridbbox <- as.points(list(x=c(grids@bbox[1,1], grids@bbox[1,2], grids@bbox[1,2],
+   grids@bbox[1,1]), y=c(grids@bbox[2,1], grids@bbox[2,1], grids@bbox[2,2], grids@bbox[2,2])))
> mserw <- mse2d(as.points(coordinates(bei.pnt)), gridbbox, 100, 10*bei.pixsize)
> bw <- mserw$h[which.min(mserw$mse)]
> plot(mserw$h,mserw$mse, type="l")
```

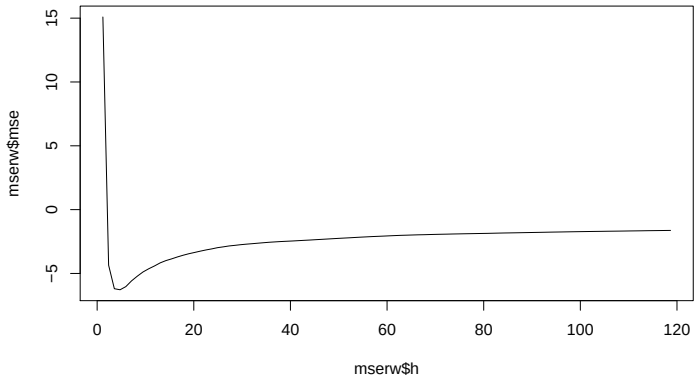

Fig. 8.3: Selection of the optimal bandwidth using the method of Berman and Diggle (1989).

This shows that the optimal bandwidth size is about 4 m. But since our grid cell size is 5 m this bandwidth is not really suited for this scale of work. Based on the plot from above we only know that we should not use a bandwidth finer/smaller than 5 m; coarser bandwidths are all plausible. We can also consider the least squares cross validation method to select the bandwidth size using the method of Worton (1995), as implemented in the adehabitat package:

```
> bei.pnt <- data.frame(x=bei$x, y=bei$y, no=rep(1, length(bei$x)))
> coordinates(bei.pnt) <- ~ x+y
> dem.asc <- import.asc("dem.asc")
> bei.kdens <- kernelUD(xy=as.data.frame(bei.pnt@coords), id=NULL, h="LSCV", grid=dem.asc)

  Warning message:
  In kernelUD(xy = as.data.frame(bei.pnt@coords), id = NULL, h = "LSCV",  :
    The algorithm did not converge within the specified range of hlim: try to increase it
```

This does not converge either[4], hence we need to set the bandwidth size using some *ad hoc* method. As a rule of thumb we can start by estimating the smallest suitable range as the average size of block:

$$p = \sqrt{area(B_{HR})/N} \tag{8.2.1}$$

and then set the bandwidth size at two times this value. There are 3605 trees (N) in the area of size 507,525 m^2, which means that we could use a bandwidth of 24 m (H):

[4] This is unfortunately a very common problem with many real point patterns.

```
> bei.pixsize <- sqrt(areaSpatialGrid(grids)/length(bei$x))
> bei.pixsize

  [1] 11.86687
```

We next derive a relative kernel density map using the standard methods in the spatstat package (Eq.2.6.7). The resulting density map is shown in Fig. 8.4:

```
> bei.ppp <- ppp(coordinates(bei.pnt)[,1], coordinates(bei.pnt)[,2],
+      marks=bei.pnt$no, window=as(grids[1], "owin"))
# reformat the point pattern so it fits the grids:
> summary(bei.ppp)

  Marked planar point pattern: 3604 points
  Average intensity 0.0071 points per square unit
  marks are  numeric,  of type 'double'
  Summary:
     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
        1       1       1       1       1       1

  Window: binary image mask
  101 x 201 pixel array (ny, nx)
  pixel size: 5 by 5 units
  enclosing rectangle: [-2.5, 1002.5] x [-2.5, 502.5] units
  Window area =  507525 square units


> bei.dens <- density(bei.ppp, sigma=2*bei.pixsize)
> grids$dens <- as(bei.dens, "SpatialGridDataFrame")$v
# relative density:
> grids$densr <- grids$dens/(max(grids$dens, na.rm=T))
> bei.pnt.plot <- list("sp.points", bei.pnt, pch="+", cex=.7, col="black")
> plt.dens1 <- spplot(grids["densr"], scales=list(draw=F), at=seq(0,1,0.025),
+      col.regions=grey(rev(seq(0,1,0.025))), sp.layout=bei.pnt.plot)
```

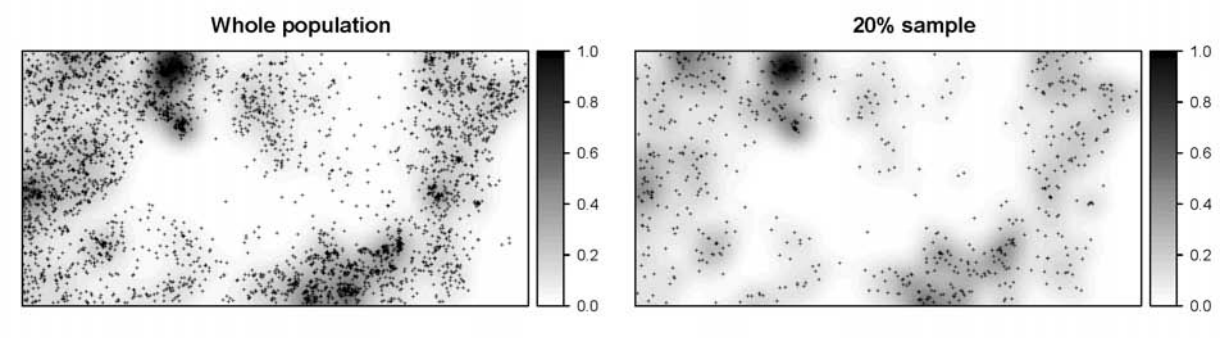

Fig. 8.4: Relative intensity estimated for the original bei data set (left), and its 20% sub-sample (right). In both cases the same bandwidth was used: H=24 m.

If we randomly subset the original occurrence locations and then re-calculate the relative densities, we notice that the spatial pattern of the two maps does not differ significantly, and neither do their histograms:

```
# Randomly subset points:
> sel <- runif(length(bei.pnt$no))<0.2
> bei.sub.pnt <- bei.pnt[sel,]
> bei.sub <- ppp(coordinates(bei.sub.pnt)[,1], coordinates(bei.sub.pnt)[,2],
+       marks=bei.sub.pnt$no, window=as(grids[1], "owin"))
# plot(bei.sub)
# Derive kernel density:
```

```
> bei.sub.dens <- density(bei.sub, sigma=2*bei.pixsize)
> grids$sub.dens <- as(bei.sub.dens, "SpatialGridDataFrame")$v
> grids$sub.densr <- grids$sub.dens/(max(grids$sub.dens, na.rm=T))
# Plot the second map:
> bei.sub.pnt.plot <- list("sp.points", bei.sub.pnt, pch="+", cex=.7, col="black")
> plt.dens2 <- spplot(grids["sub.densr"], scales=list(draw=F), at=seq(0,1,0.025),
+       col.regions=grey(rev(seq(0,1,0.025))), sp.layout=bei.sub.pnt.plot)
> print(plt.dens1, split=c(1,1,1,2), more=T)
> print(plt.dens2, split=c(1,2,1,2), more=T)
```

then check if the two maps follow the same distribution:

```
> t.test(grids$sub.densr, grids$densr)
> t.test(grids$sub.densr, grids$densr)

         Welch Two Sample t-test

data:  grids$sub.densr and grids$densr
t = -6.5351, df = 40567.5, p-value
= 6.432e-11
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.010186940 -0.005486209
sample estimates:
mean of x mean of y
0.1125212 0.1203578
```

which confirms that the two maps basically follow the same distribution. This supports our assumption that the relative density map (Eq.2.6.7) can be indeed reproduced also from a representative sample (n=721).

We proceed with preparing the environmental predictors and testing their correlation with the density values. We can extend the original single auxiliary map (DEM) by adding some hydrological parameters: slope, topographic wetness index and altitude above channel network. We want to attach the `SpatialGridDataFrame` class maps to the original `im` class `bei.extra`, which means that we need to 'coerce' the objects to format of interest:

```
> bei.extra$twi <- as.im(as.image.SpatialGridDataFrame(grids["twi"]))
> bei.extra$achan <- as.im(as.image.SpatialGridDataFrame(grids["achan"]))
> plot.im(bei.extra, col=grey(rev(seq(0,1,0.025))))
```

which will produce the map shown in Fig. 8.2. The result of fitting a non-stationary point process with a log-linear density using the `ppm` method of `spatstat` shows that density is negatively correlated with wetness index, and positively correlated with all other predictors (Baddeley, 2008):

```
> bei.sub.ppm <- ppm(bei.sub, ~ elev+grad+twi+achan, covariates=list(elev=bei.extra$elev,
+     grad=bei.extra$grad, twi=bei.extra$twi, achan=bei.extra$achan))
> summary(bei.sub.ppm)

Point process model
fitted by maximum pseudolikelihood (Berman-Turner approximation)
Call:
ppm(Q = bei.sub, trend = ~ elev + grad + twi + achan,
covariates = list(elev = bei.extra$elev,...
Edge correction: "border"
--------------------------------------------------------
FITTED MODEL:

Nonstationary Poisson process

 ---- Intensity: ----

Trend formula: ~ elev + grad + twi + achan
Model involves external covariates
```

```
Fitted coefficients for trend formula:
(Intercept)        elev        grad         twi       achan
-9.00309481  0.01675857  4.24494406 -0.07240637  0.03017035
```

We can actually use this model to predict the species density by using the generic `predict` method:

```
> bei.ppm.trend <- predict(bei.sub.ppm, type="trend", ngrid=c(grids@grid@cells.dim[2],
+       grids@grid@cells.dim[1]), window=as(grids[1], "owin"))
> plot(bei.ppm.trend)
> plot(bei.sub, add=T, pch=".")
```

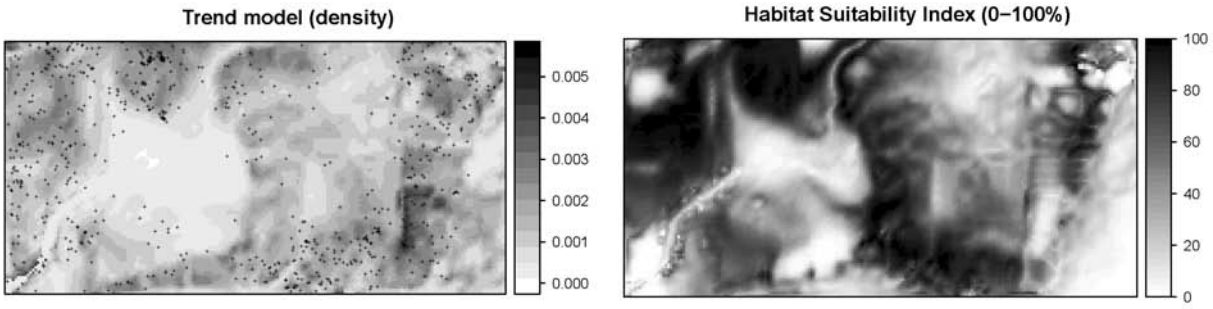

Fig. 8.5: Trend model "ppm" predicted using elevation, slope, topographic wetness index and altitude above channel network as environmental covariates (left); Habitat Suitability Index (0-100%) derived in the adehabitat package using the same list of covariates (right).

Visually (Fig. 8.5), we can see that the predicted trend seriously misses some hot-spots i.e. clusters of points. It seems that using point pattern analysis techniques to map (realized) species' distribution with covariates is of limited use. A comparison between the *Akaike Information Criterion* (AIC) for a model without predictors and with predictors shows that there is a slight gain in using the covariates to predict the spatial density:

```
> fitnull <- ppm(bei.sub, ~ 1)
> AIC(fitnull)

  [1] 10496.56


> AIC(bei.sub.ppm)  # this is a slightly better model

  [1] 10289.79
```

Just for a comparison, we can also fit a GLM model[5] using all grid nodes:

```
> bei.ppp.lm <- glm(sub.densr ~ grad+elev+twi+achan, grids@data, family=poisson())

  There were 50 or more warnings (use warnings() to see the first 50)


> summary(bei.ppp.lm)
```

This shows a similar picture: the best predictor of the density is `TWI`.

[5] We need to use the log as the link function because the density values are heavily skewed.

8.2.2 Environmental Niche analysis

We proceed with the Environmental Niche Factor Analysis (Hirzel and Guisan, 2002), as implemented in the adehabitat[6] package (Calenge, 2007). Our objective is to derive the Habitat Suitability Index (0–100%) which basically shows potential spreading of a species in the feature space (environmental preference). We need to prepare the maps in the adehabitat native format:

```
> beidata <- data2enfa(as.kasc(list(dem=import.asc("dem.asc"), grad=import.asc("grad.asc"),
+     twi=import.asc("twi.asc"), achan=import.asc("achan.asc"))), bei.sub.pnt@coords)
# run ENFA and make predictions of habitat suitability index:
> enfa.bei <- enfa(dudi.pca(beidata$tab, scannf=FALSE), beidata$pr, scannf=FALSE, nf=2)

 Warning message:
 In predict.enfa(enfa.bei, beidata$index, beidata$attr) :
   the enfa is not mathematically optimal for prediction:
  please consider the madifa instead


> bei.dist <- predict(enfa.bei, beidata$index, beidata$attr)
> grids$bei.dist <- as.SpatialGridDataFrame.im(asc2im(bei.dist))$v
# Convert to 0-100 scale:
> sum.dist <- summary(grids$bei.dist)
> grids$rankv <- rank(grids$bei.dist, ties.method="first")
> grids$hsit <- ifelse(grids$bei.dist<sum.dist[["Median"]],
+     (1-grids$rankv/max(grids$rankv))*100, (1-grids$rankv/max(grids$rankv))*100)
> grids$hsi <- 100*round((grids$hsit-min(grids$hsit, na.rm=T))/(max(grids$hsit,
+     na.rm=T)-min(grids$hsit, na.rm=T)), 3)
> plt.HSI <- spplot(grids["hsi"], at=seq(0,100,2.5), col.regions=bpy.colors())
```

The resulting HSI map shows that this species generally avoids the areas of high wetness index, i.e. it prefers ridges/dry positions (Fig. 8.5, right). This spatial pattern is now more distinct (compare with the trend model in Fig. 8.5, left). This demonstrates the power of ENFA, which is in this case more suited for analysis of the occurrence-only locations than regression analysis and/or the point pattern analysis.

8.2.3 Simulation of pseudo-absences

In the next section we will use the results of Niche analysis to generate pseudo-absences. Recall (§2.6) that insertion of pseudo-absences is an important step because it will allow us to fit regression models. We use two maps as weight maps to randomize allocation of the pseudo-absences: the geographical distance from the occurrence locations and habitat suitability index. We first derive the buffer distance to the points in SAGA GIS:

```
# first the buffer distance:
> rsaga.geoprocessor(lib="grid_gridding", module=3, param=list(GRID="bei_buffer.sgrd",
+    INPUT="bei_sub.shp", FIELD=0, LINE_TYPE=0, USER_CELL_SIZE=grids@grid@cellsize[[1]],
+    USER_X_EXTENT_MIN=grids@bbox[1,1]+grids@grid@cellsize[[1]]/2,
+    USER_X_EXTENT_MAX=grids@bbox[1,2]-grids@grid@cellsize[[1]]/2,
+    USER_Y_EXTENT_MIN=grids@bbox[2,1]+grids@grid@cellsize[[1]]/2,
+    USER_Y_EXTENT_MAX=grids@bbox[2,2]-grids@grid@cellsize[[1]]/2))
# now extract a buffer distance map and load it back into R:
# (the parameters DIST and IVAL are estimated based on the grid properties)
> rsaga.geoprocessor(lib="grid_tools", module=10, param=list(SOURCE="bei_buffer.sgrd",
+   DISTANCE="bei_dist.sgrd", ALLOC="bei_alloc.sgrd", BUFFER="bei_bdist.sgrd",
+   DIST=sqrt(areaSpatialGrid(grids))/3, IVAL=grids@grid@cellsize[[1]]))
> rsaga.sgrd.to.esri(in.sgrds="bei_dist.sgrd", out.grids="bei_dist.asc",
+   out.path=getwd(), prec=1)
> grids$buffer <- readGDAL("bei_dist.asc")$band1
# relative distance:
> grids$bufferr <- grids$buffer/max(grids$buffer, na.rm=T)
```

[6] http://cran.r-project.org/web/packages/adehabitat/

By combining HSI and buffer map around the occurrence locations (Eq.2.6.11) we can generate the same amount of pseudo-absence locations:

```
# weight map:
> grids$weight <- ((100*grids$bufferr+(100-grids$hsi))/2)^2
> dens.weight <- as.im(as.image.SpatialGridDataFrame(grids["weight"]))
> bei.absences <- rpoint(length(bei.sub.pnt$no), f=dens.weight)
> bei.absences <- data.frame(x=bei.absences$x, y=bei.absences$y,
+    no=rep(0, length(bei.sub.pnt$no)))
> coordinates(bei.absences) <- ~ x+y
# combine the occurences and absences:
> bei.all <- rbind(bei.sub.pnt["no"], bei.absences["no"])
```

One realization of the simulated pseudo-absences (overlaid over the weight map) can be seen in Fig. 8.10a. Correct insertion of pseudo-absences is crucial for the success of regression analysis (see Chefaoui and Lobo (2008) for discussion), hence it also a good idea to visually validate each simulated pseudo-absence and remove 'suspicious' candidates.

8.2.4 Regression analysis and variogram modeling

Before we run any regression analysis we will convert the original four predictors to principal components (in order to reduce their dimensions and the multicolinearity effect):

```
# derive PCs from the original predictors:
> pc.grids <- prcomp( ~ grad+elev+twi+achan, scale=TRUE, grids)
> pc.comps <- as.data.frame(pc.grids$x)
> pointgrids <- as(grids, "SpatialPointsDataFrame")
> pc.comps$X <- coordinates(pointgrids)[,1]
> pc.comps$Y <- coordinates(pointgrids)[,2]
> coordinates(pc.comps) <- ~ X+Y
> gridded(pc.comps) <- TRUE
> pc.comps <- as(pc.comps, "SpatialGridDataFrame")
> spplot(pc.comps, c("PC1","PC2","PC3","PC4"),
+    col.regions=grey(c(rep(0,10),seq(0,1,0.05),rep(1,10))), cuts=40)
```

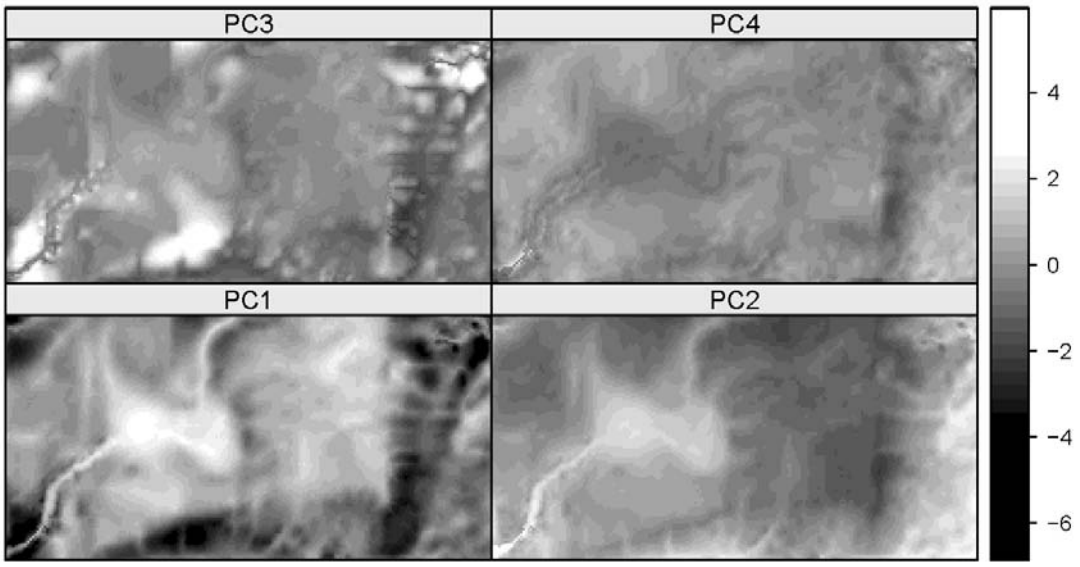

Fig. 8.6: Principal Components generated using the original predictors.

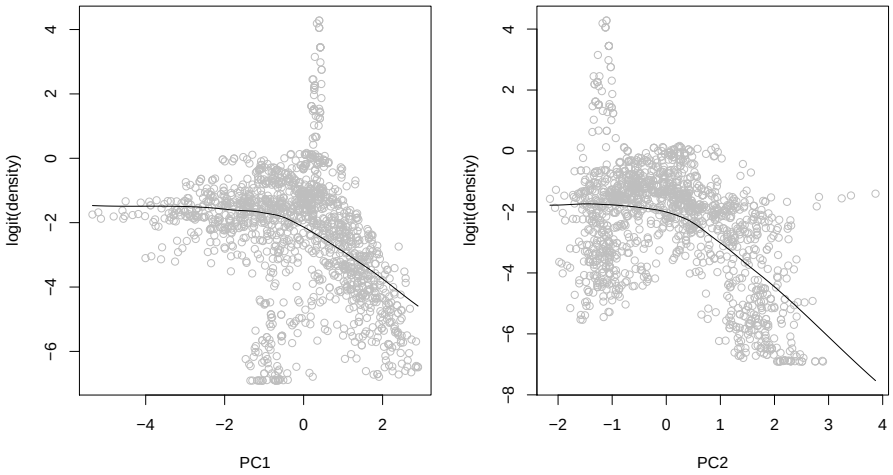

Fig. 8.7: Correlation plot PC1/PC2 versus the *logit*-transformed density.

which shows that the first component reflects pattern in the `twi` and `grad`; the third component reflects pattern in `achan` (Fig. 8.6). Next, we can overlay the predictors and the estimated intensities at occurrence and absence locations:

```
> bei.ov2 <- overlay(pc.comps, bei.all)
> bei.ov2$no <- bei.all$no
# convert the original values to logits:
> bei.ov2$log.densr <- log((bei.ov2$densr+0.001)/(1-(bei.ov2$densr-0.001)))
> summary(bei.ov2$log.densr)

    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-6.908  -3.378  -1.882  -2.297  -1.197   4.253


> var(bei.ov2$log.densr, na.rm=T)

[1] 3.666165
```

The benefit of converting the target variable to logits and the predictors to PCs is that the relationship between the two is now *close* to linear (Fig. 8.7):

```
> scatter.smooth(bei.ov2$PC1, bei.ov2$log.densr, span=19/20, col="grey",
+    xlab="PC1", ylab="logit(density)")
> lm3.bei <- lm(log.densr ~ PC1+PC2+PC3+PC4, bei.ov2@data)
> summary(lm3.bei)

Call:
lm(formula = log.densr ~ PC1 + PC2 + PC3 + PC4, data = bei.ov2@data)

Residuals:
     Min       1Q   Median       3Q      Max
-4.09463 -0.98535  0.04709  0.99504  5.52267

Coefficients:
            Estimate Std. Error t value
(Intercept) -2.35896    0.04247 -55.546
PC1         -0.40174    0.02864 -14.025
PC2         -0.79147    0.03684 -21.482
PC3         -0.34820    0.04629  -7.522
PC4          0.34360    0.07423   4.629
```

```
            Pr(>|t|)
(Intercept)  < 2e-16 ***
PC1          < 2e-16 ***
PC2          < 2e-16 ***
PC3         9.65e-14 ***
PC4         4.02e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.559 on 1379 degrees of freedom
Multiple R-squared: 0.3441,     Adjusted R-squared: 0.3422
F-statistic: 180.9 on 4 and 1379 DF,  p-value: < 2.2e-16
# the model explains  34% of variation
```

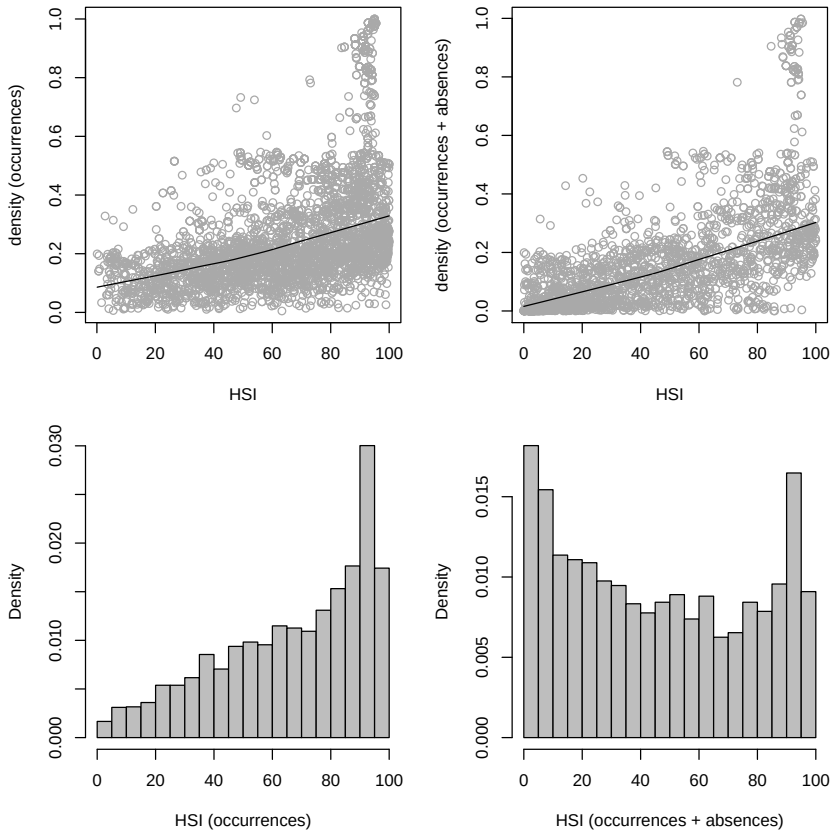

Fig. 8.8: Correlation plot HSI vs relative density with occurrence-only locations (left) and after the insertion of the pseudo-absence locations (right). Note that the pseudo-absences ensure equal spreading around the feature space (below).

The correlation between the HSI and density is now clearer and the spreading of the points around the HSI feature space is symmetric:

```
# Glue occurrences and pseudo-absences:
> bei.all <- rbind(bei.sub.pnt["no"], bei.absences["no"])
> scatter.smooth(bei.ov2$hsi[bei.ov2$no>0], bei.ov2$densr[bei.ov2$no>0],  span=19/20,
+   col="darkgrey", pch=21, cex=.5, xlab="HSI", ylab="density (occurrences-only)")
> scatter.smooth(bei.ov2$hsi, bei.ov2$densr, span=19/20, col="darkgrey", pch=21,
+   cex=.5, xlab="HSI", ylab="density (+ absences)")
> hist(bei.ov2$hsi[bei.ov2$no>0], col="grey", freq=F, breaks=30,
+   xlab="HSI (occurrences)", ylab="frequency", main="")
> hist(bei.ov2$hsi, col="grey", freq=F, breaks=30,
+   xlab="HSI (occurrences + absences)", ylab="frequency", main="")
```

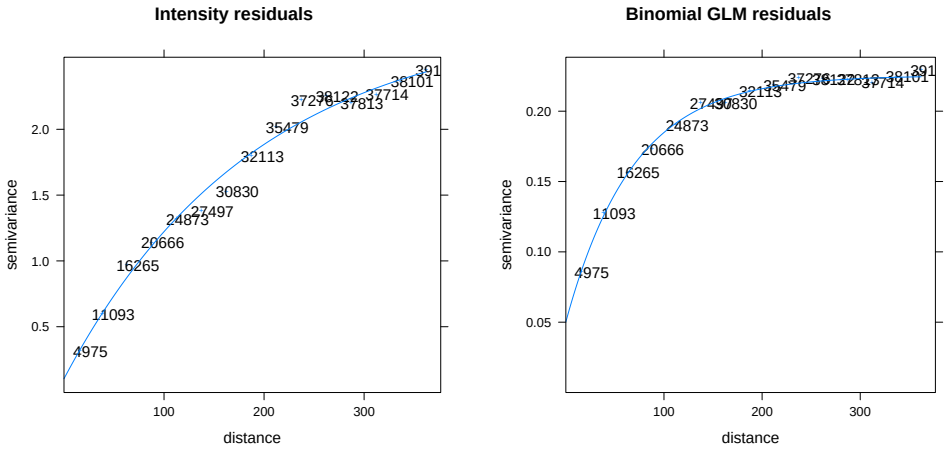

Fig. 8.9: Variogram models for residuals fitted in gstat using occurrence-absence locations: (left) density values (logits), and (right) probability values.

which produces the plot shown in Fig. 8.8 (right). Consequently, the model fitting is more successful: the adjusted R-square fitted using the four environmental predictors jumped from 0.07 to 0.34. This demonstrates the benefits of inserting the pseudo-absence locations using a feature-space design model. It can be further demonstrated that, if we would randomly insert the pseudo-absences, the model would not improve or would become even noisier.

We proceed with analyzing the point data set indicated in Fig. 8.10b using standard geostatistical tools. We can fit a variogram for the residuals and then predict relative densities using regression-kriging as implemented in the gstat package. For a comparison, we also fit a variogram for the occurrence-absence data but using the residuals of the GLM modeling with the logit link function, i.e. using the 0/1 values (Fig. 8.9):

```
> glm.bei <- glm(no ~ PC1+PC2+PC3+PC4, bei.ov2@data, family=binomial())
```

We can now deal with the residuals by using some standard geostatistical techniques. To fit a variogram for the residuals, we run:

```
> res.var <- variogram(residuals(lm3.bei) ~ 1, bei.ov2)
> res.vgm <- fit.variogram(res.var, vgm(nugget=0, model="Exp",
+   range=sqrt(areaSpatialGrid(grids))/3, psill=var(residuals(lm3.bei), na.rm=T)))
> plot(res.var, res.vgm, plot.nu=T, pch="+", main="Intensity residuals")
```

As with any indicator variable, the variogram of the binomial GLM will show higher nugget and less distinct auto-correlation than the variogram for the (continuous) density values (Fig. 8.9). This is also because the residuals of the density values will still reflect kernel smoothing, especially if the predictors explain only a small part of variation in the density values.

8.3 Final predictions: regression-kriging

We can finally generate predictions using a regression-kriging model[7]:

```
# regression part:
> vt.gt <- gstat(id=c("dens"), formula=log.densr ~ PC1+PC2+PC3+PC4, data=bei.ov2)
# residuals:
> vt.gr <- gstat(id=c("dens"), formula=residuals(lm3.bei) ~ 1,
+      data=bei.ov2, model=res.vgm)
> vt.reg <- predict.gstat(vt.gt, pc.comps) # regression part
> vt.ok <- predict.gstat(vt.gr, pc.comps, nmax=80, beta=1, BLUE=FALSE,
+      debug.level=-1)
```

[7] Residuals and the deterministic part are predicted separately.

```
# Back-transform:
> vt.reg$dens <- exp(vt.reg$dens.pred+vt.ok$dens.pred)/
+      (1+exp(vt.reg$dens.pred+vt.ok$dens.pred))
> vt.reg$densA <- vt.reg$dens * length(bei.pnt$no)/sum(vt.reg$dens, na.rm=F)
> sum(vt.reg$densA) # check if the sum of counts equals the population!

  [1] 3604


> bei$n

  [1] 3604
```

To predict the probability of species' occurrence we run similar steps:

```
> bin.reg <- predict(glm.bei, newdata=pc.comps, type="response", na.action=na.omit)
> bin.gr <- gstat(id=c("no"), formula=glm.bei$residual ~ 1, data=bei.ov2, model=res.bvgm)
> bin.ok <- predict.gstat(bin.gr, pc.comps, nmax=80, beta=1, BLUE=FALSE, debug.level=-1)
> bin.ok$rk.bin <- bin.reg$fit + bin.ok$no.pred
> bin.ok$rk.binf <- ifelse(bin.ok$rk.bin<0, 0, ifelse(bin.ok$rk.bin>1, 1, bin.ok$rk.bin))
```

The resulting map of density predicted using regression-kriging (shown in Fig. 8.10c) is indeed a hybrid map that reflects kernel smoothing (hot spots) and environmental patterns, thus it is a map richer in content than the pure density map estimated using kernel smoothing only (Figs. 8.4), or the Habitat Suitability Index (Fig. 8.5, right). Note also that, although the GLM-kriging with a binomial link function (Fig. 8.10d) is statistically a more straight forward procedure (it is completely independent from point pattern analysis), its output is limited in content because it also fails to represent some hot-spots. GLM-kriging of 0/1 values in fact only shows the areas where a species' is likely to occur, without any estimation of how dense will the population be. Another advantage of using the occurrences+absences with attached density values is that we are able not only to generate predictions, but also to generate geostatistical simulations, map the model uncertainty, and run all other common geostatistical analysis steps.

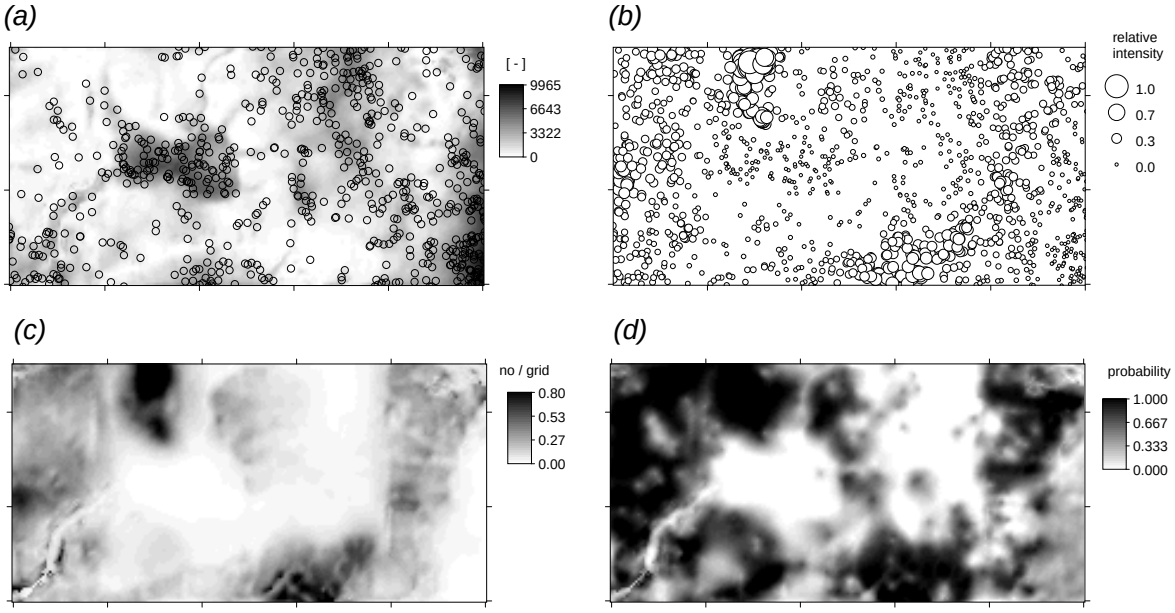

Fig. 8.10: Spatial prediction of species distribution using the `bei` data set (20% sub-sample): (a) the weight map and the randomly generated pseudo-absences using the Eq.(2.6.11); (b) input point map of relative intensities (includes the simulated pseudo-absences); (c) the final predictions of the overall density produced using regression-kriging (showing number of individuals per grid cell as estimated using Eq.(2.6.8); and (d) predictions using a binomial GLM.

In the last step of this exercises we want to validate the model performance using the ten-fold cross validation (as implemented in the `gstat` package):

```
> vt.cross <- krige.cv(log.densr ~ PC1+PC2+PC3+PC4, bei.ov2, res.vgm, nfold=10)
# 10-fold cross-validation
# MPE, ideally 0 (unbiased estimate)
> mean(vt.cross$residual)
```

```
[1] -0.004776853
```

```
# MSPE, ideally small
> var(vt.cross$residual, na.rm=T)
```

```
[1] 0.01412567
```

```
# portion of variation explained by the model:
> 1-var(vt.cross$residual, na.rm=T)/var(bei.ov2$log.densr, na.rm=T)
```

```
[1] 0.9961782
```

which shows that the model is highly precise — it explains over 99% of the variance in the training samples. Further comparison between the map shown in Fig. 8.10c and Fig. 8.4 shows that with 20% of samples and four environmental predictors we are able to explain 96% of the pattern in the original density map (R-square=0.96). Fig. 8.11 indeed confirms that this estimator is unbiased. These are rather high values[8] and you will possibly not expect to achieve such high accuracy with your own data.

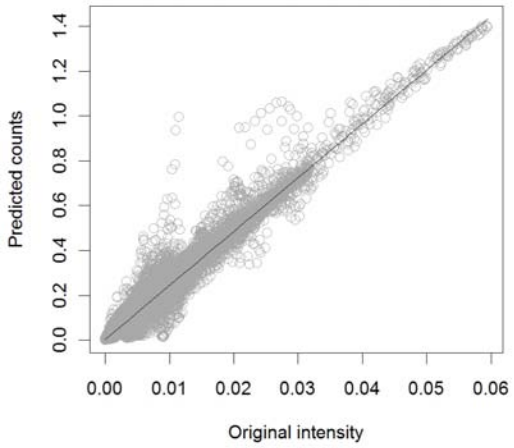

Fig. 8.11: Evaluation of the mapping accuracy for the map shown in Fig. 8.10c versus the original mapped density using 100% of samples (Fig. 8.4).

One last point: although it seems from this exercise that we are recycling auxiliary maps and some analysis techniques (we use auxiliary maps both to generate the pseudo-absences and make predictions), we in fact use the HSI map to generate the pseudo-absences, and the original predictors to run predictions, which do not necessarily need to reflect the same features. Relative densities do not have to be directly correlated with the HSI, although a significant correlation will typically be anticipated. Likewise, we use kernel smoother to estimate the intensities, but we then fit a variogram, which is controlled by the amount of smoothing, i.e. value of the bandwidth, hence the variogram will often show artificially smooth shape (as shown in Fig. 8.9). The only way to avoid this problem is to estimate the bandwidth using some objective technique (which we failed to achieve in this example), or to scale the variogram fitted for the indicator variable (Fig. 8.9; right) to the density values scale.

8.4 Niche analysis using MaxEnt

In section §8.2.2 we have derived Habitat Suitability Index map using the adehabitat package. An alternative approach to estimate species ecological preference is by using the MaxEnt package (Phillips and Dudík, 2008), which is by many considered to be the most robust approach to species distribution modeling (see §2.6). For example, MaxEnt can work with both continuous and categorical predictors and has very extensive and flexible possibilities for analysis of biodiversity data. With MaxEnt, we can request cross-validation and investigate which predictors are most significant, which is not as simple with adehabitat.

MaxEnt is not available as an R package, therefore you will first need to request and download it from the MaxEnt homepage[9]. The complete algorithm is contained in a single `maxent.jar` (**J**ava **AR**chive) file, which is basically a zipped Java (class file) code[10]. Then, define location of MaxEnt in the R session:

[8] This is an ideal data set where all individual trees have been mapped and the species' distribution is systematic.
[9] `www.cs.princeton.edu/~schapire/maxent/`
[10] Obviously, before you can run MaxEnt, you will need to install **Java** software on your machine.

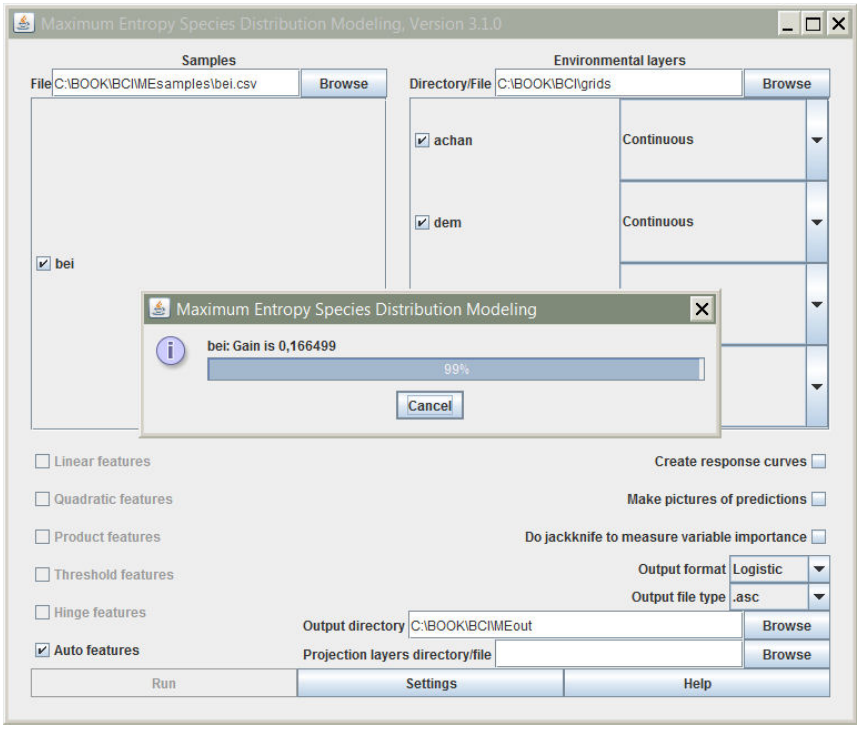

Fig. 8.12: Extraction of Habitat Suitability Index using MaxEnt.

```
# Location of MaxEnt and directories:
> Sys.chmod(getwd(), mode="7777")  # write permissions
> MaxEnt <- "C:\\MaxEnt\\maxent.jar"
> dir.create(path="MEout"); dir.create(path="MEsamples")
> MaxEnt.layers <- paste(gsub("/", "\\\\", getwd()), "\\grids", sep="")
> MaxEnt.out <- paste(gsub("/", "\\\\", getwd()), "\\MEout", sep="")
> MaxEnt.samples <- paste(gsub("/", "\\\\", getwd()), "\\MEsamples", sep="")
```

where `MEout` is the directory where MaxEnt will write the results of analysis (plots, grids and table data), and `MEsamples` is a directory containing the input samples. Next, copy the grids of interest to some working directory e.g. `/grids`:

```
> dir.create(path="grids")
> for(j in c("dem.asc", "grad.asc", "twi.asc", "achan.asc")) {
>   file.copy(j, paste("grids/", j, sep=""), overwrite=TRUE)
> }
> asc.list <- list.files(paste(getwd(), "/grids", sep=""),
+      pattern="\\.asc$", recursive=TRUE, full=FALSE)
> asc.list
```

```
  [1] "achan.asc" "dem.asc"   "twi.asc"
```

Before we can run MaxEnt, we still need to prepare the occurrence records in the required format (`.csv`):

```
# Write records to a csv file (species, longitude, latitude):
> bei.csv <- data.frame(sp=rep("bei", bei$n), gx=bei$x, gy=bei$y)
> write.csv(bei.csv[,c("sp","gx","gy")], "MEsamples/bei.csv", quote=FALSE, row.names=FALSE)
```

We can now run MaxEnt using the `system` command in R:

```
# Run a batch process (opens a process window):
> system(command=paste("java -mx1000m -jar ", MaxEnt, " environmentallayers=",
+    MaxEnt.layers, " samplesfile=", MaxEnt.samples, "\\bei.csv", " outputdirectory=",
+    MaxEnt.out, " randomtestpoints=25 maximumiterations=100 redoifexists
+    autorun nowarnings notooltips", sep=""))
```

where randomtestpoints=25 will randomly take 25% points for cross-validation, redoifexists will replace the existing files, autorun, nowarnings and notooltips will force MaxEnt to run without waiting. For more information about MaxEnt batch mode flags, look in the Help file.

After the analysis, open the /MEout directory and browse the generated files. We are most interested in two files: bei.asc — the Habitat Suitability Index map (0–1), and bei.html — the complete report from the analysis. Optionally, you can also read the result of analysis directly to R by using the XML package:

```
> library(XML)
> bei.html <- htmlTreeParse("MEout/bei.html")
# summary result:
> bei.html[[3]][[1]][[2]][[65]]; bei.html[[3]][[1]][[2]][[52]];
+     bei.html[[3]][[1]][[2]][[56]]; bei.html[[3]][[1]][[2]][[67]]

  12703 points used to determine the Maxent distribution
  (background points and presence points).
  Regularized training gain is 0.167, training AUC is 0.712,
   unregularized training gain is 0.193.
  Test AUC is 0.704, standard deviation is 0.008
  Environmental layers used (all continuous): achan dem grad twi
```

which shows that AUC (Area Under Curve) statistics for cross-validation locations is 0.704 (maximum AUC is 1.0). Note that the HSI map derived using ENFA and MaxEnt are in fact very similar. MaxEnt also shows the most important predictors that can be used to explain distribution of bei trees are twi (40.6%) and dem (32.5%).

Self-study exercises:

(1.) Delineate all areas with >50 trees per ha. How big is this area in ha?

(2.) Randomly allocate 300 points and derive an HSI map using these points. Is there a significant difference between the map you derived and the map shown in Fig. 8.5 (right)?

(3.) Derive additional four DEM parameters in SAGA: two based on the lightning module and two based on the residual analysis (see p.230) and repeat the regression modeling explained in §8.2.4 using the extended list of predictors. Which predictor is now the most significant?

(4.) Produce a factor type map showing: (1) actual spreading (predicted probability exceeds 0.5), (2) potential spreading (HSI> 50%), and (3) no spreading using the original data set.

(5.) Run 4–6 geostatistical simulations using the model fitted in §8.2.4. Where is the distribution of the species most variable? Relate to both geographical and state space.

(6.) Try to allocate the pseudo-absences (section 8.2.3) using even higher spreading at the edges of the feature space. Determine the optimal spreading based on the improvement in the R-square.

(7.) Compare HSI maps derived in adehabitat and MaxEnt (using the same inputs) and derive correlation coefficient between the two maps.

Further reading:

★ Baddeley, A., 2008. **Analysing spatial point patterns in R**. CSIRO, Canberra, Australia.

★ Calenge, C., 2007. Exploring Habitat Selection by Wildlife with adehabitat. Journal of Statistical Software 22(6), 2–19.

★ Diggle, P. J., 2003. **Statistical Analysis of Spatial Point Patterns**, 2nd Edition. Arnold Publishers.

★ Hirzel A. H., Hausser J., Chessel D., Perrin N., 2002. Ecological-niche factor analysis: How to compute habitat-suitability maps without absence data? Ecology, 83, 2027–2036.

★ Pebesma, E. J., Duin, R. N. M., Burrough, P.A., 2005. Mapping sea bird densities over the North Sea: spatially aggregated estimates and temporal changes. Environmetrics 16(6), 573–587.

★ Phillips, S. J., Dudík, M., 2008. Modeling of species distributions with Maxent: new extensions and a comprehensive evaluation. Ecography, 31(2): 161-175.

★ Tsoar, A., Allouche, O., Steinitz, O., Rotem, D., Kadmon, R., 2007. A comparative evaluation of presence-only methods for modeling species distribution. Diversity & Distributions 13(9), 397–405.