10 · *Regression-Based Approaches*

10.1 Concepts

Regression-based approaches are by far the most commonly used in ecology and other disciplines, and particularly in habitat suitability modeling (Guisan et al., 2002). They usually rely on robust statistical theories (e.g. sum of squares, maximum likelihood) and are treated in detail in textbooks.

Regression relates a response variable (e.g. presence–absence, abundance, biomass) to a set of pre-selected environmental predictors (e.g. climate, land use, resource). The predictors can be used as untransformed environmental variables or, in order to prevent multicollinearity in the data, as orthogonal components derived from the environmental variables through multivariate analyses. As seen in section 6.4.2, one diagnostic to test for multicollinearity is the VIF (Montgomery and Peck, 1982; see Part II) and its derivation to test for various combinations of variables. The classical ordinary least-square (OLS) linear regression approach (often simply called linear model, LM) is theoretically valid only when the response variable is normally distributed (i.e. Gaussian) and the variance does not change as a function of the mean (homoscedasticity). In other words, homoscedasticity relates to the specific case in which the error term (i.e. the random effect in the relationship between the predictors and the response variable) is constant across all values of the predictor variables. GLMs constitute a more flexible family of regression models, which allow the response variable to follow other distributions and non-constant variance functions to be modeled. In GLMs, the combination of predictors (the linear predictors) is related to the mean of the response variable through a link function. Using such link functions makes it possible to both transform the response to linearity and maintain the predicted values within the original range of values allowed for the response variable. By doing so, the GLMs can handle Gaussian (e.g. biomass), Poisson (species abundance,

species richness), binomial (e.g. presence–absence), or gamma distributions – with link functions set to identity, logarithm, logit, and inverse respectively, for example.

If the response shape is not a linear function of predictors, a transformed (higher-order polynomial) term of the latter can be included in the model (Hastie et al., 2009). This type of regression is called a polynomial regression. Second order polynomial regressions simulate unimodal symmetric responses (e.g. a hypothetical bell-shaped relationship between species abundance and a given environmental variable; Austin, 1985), whereas third-order or higher terms make it possible to simulate skewed and bimodal responses, or even a combination of both. Fitting complex curves should, however, be done carefully, since there is then a high risk of obtaining undesired shapes for the resulting response curve outside of the calibration range of the model (e.g. rising again exponentially outside the possible range of the species, which will make problem when projecting the model on these values; Thuiller et al., 2004b). See Merow et al. (2014) for a discussion of fitting simple versus complex response shapes.

Alternative regression techniques for relating the distribution of biological entities to environmental gradients are based on non-parametric smoothing functions of predictors. GAMs are commonly used to implement non-parametric smoothers in regression models (Wood, 2006; Wood et al., 2015), making them semi-parametric approaches. This technique applies smoothers independently to each predictor and additively calculates the component response.

The major difference between GLMs and their extensions (e.g. GAM, MARS, BRUTO) thus lies in the choice of model-driven versus data-driven response shapes. Indeed, to properly use a GLM, one should have some expectation regarding the shape of the response variable along the predictors. When a highly limiting factor is expected, a linear relationship could be sufficient, whereas when a unimodal response along a wide continuous gradient is expected, a bell-shaped (quadratic) curve is required (see Part I). If there is no expectation regarding shape, then various shapes would need to be tested, which could become tedious when several predictors are used together. Data-driven approaches, such as GAM, are slightly more flexible in this regard, but other choices have to be decided up front (e.g. type of smoother, degrees of freedom). We will see later what impact the decision to select one approach over another can have on the predictions.

Table 10.1 *Examples of commonly used distributions, associated families and links for GLM. A classical ecological example is also given.*

Distribution	Family or/and usefulness	Link	Example
Normal	Ordinary linear model	Identity	Biomass (usually log-transformed)
Poisson	Log-linear model	Log or square root	Species richness
Binomial	Logistic regression, probit	Logit or probit	Presence–absence
Gamma	Alternative to lognormal model	Log or inverse link	Species abundance distribution
Negative binomial	Account for overdispersion	log	Frequency count data

10.2 Generalized Linear Models

As touched on earlier, GLMs generalize OLS regression by allowing the linear model to be related to the response variable via a link function, and by allowing the magnitude of the variance for each measurement to act as a function of its predicted value.

There are a number of distributions in addition to the normal distribution that leads to a GLM (Table 10.1).

The linear predictor determines the mean of the response (McCullagh and Nelder, 1989a). It is unbounded, but the mean of some of these distributions (e.g. binomial) is restricted. The mean is supposed to be a (monotone) function of the linear predictor and the inverse of this function is called the inverse-link function. As stated above, this function ensures that the reversely transformed predictions remain within the original scale of the response variable. Users need to define the link before running any models (see Table 10.1).

If we go back to our red fox species modeled in Chapter 9, a simple GLM with a number of predictor variables can be easily implemented:

```
> glm1 <- glm(VulpesVulpes ~ 1 + bio3 + bio7 + bio11 + bio12,
data = mammals_data, family = "binomial")
> glm2 <- glm(VulpesVulpes ~ 1 + poly(bio3, 2) + poly(bio7,
2) + poly(bio11, 2) + poly(bio12, 2), data = mammals_data,
family = "binomial")

> library(biomod2)
> par(mfrow = c(2, 2))
```

```
> level.plot(mammals_data$VulpesVulpes, XY = mammals_data[,
c("X_WGS84", "Y_WGS84")], color.gradient = "grey",
cex = 0.3, show.scale = F, title = "Original data")
> level.plot(fitted(glm1), XY = mammals_data[, c("X_WGS84",
"Y_WGS84")], color.gradient = "grey", cex = 0.3,
show.scale = F, title = "GLM with linear terms")
> level.plot(fitted(glm2), XY = mammals_data[, c("X_WGS84",
"Y_WGS84")], color.gradient = "grey", cex = 0.3,
show.scale = F, title = "GLM with quadratic terms")
```

The two models *glm1* and *glm2* mostly differ in terms of the hypotheses used regarding the shape of the relationship between all variables and the presence of the species. In *glm1*, one assumes that linear predictors are sufficient, in *glm2* one expects quadratic relationships, (i.e. non-symmetric, unimodal or sigmoidal relationships). The poly function in glm2 is an effective way of dealing with correlation between x and x^2 and provide a more flexible response (i.e. non-symmetric unimodal) that simply uses x + I(X)^2 in the formula.

In this particular example, the spatial distributions of the probability of occurrence from the two different models appear rather similar at first glance (Figure 10.1). However, let's examine how the modeled responses differ in environmental space by analysing the response curves of the species along the environmental gradients fitted in the models (Figure 10.2).

There are several ways of visualizing the response curves of a species for the different models. One possibility is to use a function in the `biomod2` package, which implements the evaluation strip method proposed by Elith et al. (2005). This method has the advantage of being independent of the algorithm used. For building the predicted response curves, n-1 variables are set as constants to a fixed value (mean, median, min or max, i.e. `fixed.var.metric` argument) and only the remaining one (remaining two for three-dimensional response plots) varies across its whole range (given by `Data`). The variations observed and the curve thus obtained shows the sensibility of the model to that specific variable (Figure 10.2.).

```
> library(ggplot2)
> ## create the response plot
> rp <- response.plot2(models = c("glm1", "glm2"),
Data = mammals_data[,
c("bio3", "bio7", "bio11", "bio12")],
show.variables = c("bio3", "bio7", "bio11", "bio12"), fixed.var.
metric = "mean",
plot = FALSE, use.formal.names = TRUE)
> ## define a custom ggplot2 theme
```

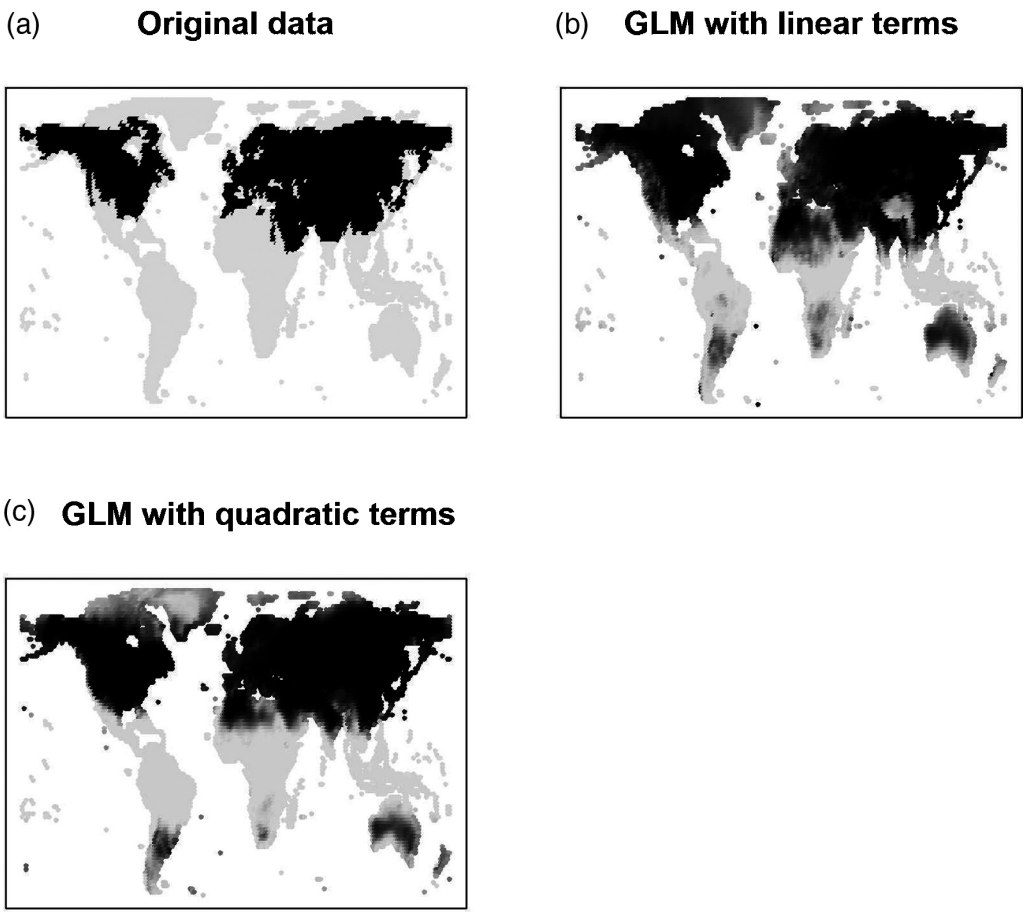

Figure 10.1 Observed (black = presence, light gray = absence) and potential distribution of species Sp290 modeled by different GLM differing by the complexity of the parameters (linear, quadratic, and second-order polynomials). The gray scale of predictions (b, c) shows habitat suitability values between 0 (light, unsuitable) and 1 (dark, highly suitable).

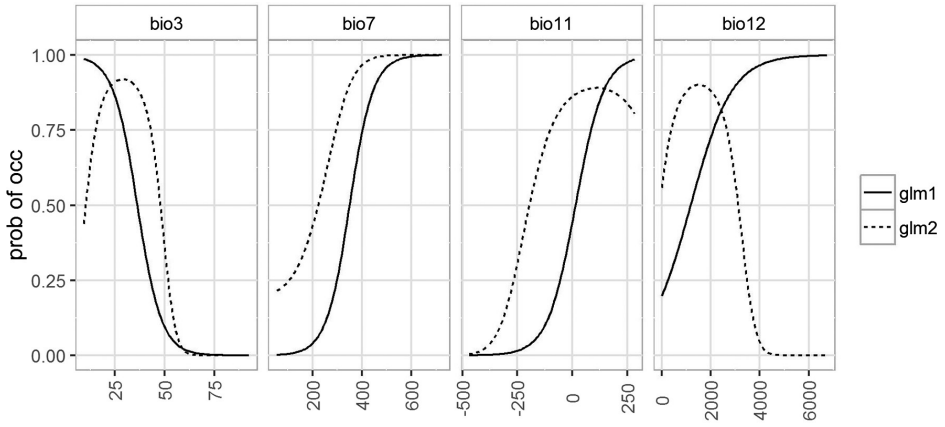

Figure 10.2 Response curves of model `glm1` (linear terms) and `glm2` (quadratic terms). Plotted are the probabilities of occurrence in function of the bioclimatic variables.

```
> rp.gg.theme <- theme(legend.title = element_blank(),
axis.text.x = element_text(angle = 90, vjust = 0.5),
panel.background = element_rect(fill = NA,
colour = "gray70"), strip.background = element_rect(fill = NA,
colour = "gray70"), panel.grid.major = element_line(colour =,
"grey90"), legend.key = element_rect(fill = NA,
colour = "gray70"))

> ## display the response plot
> gg.rp <- ggplot(rp, aes(x = expl.val, y = pred.val,
lty = pred.name)) + geom_line() + ylab("prob of occ") + xlab("")
+ rp.gg.theme + facet_grid(~expl.name, scales = "free_x")
> print(gg.rp)
```

It is interesting to note that although the response curves differ between the models, the spatial predictions remain relatively similar. This reminds us that slightly different models can yield very similar predictions. This shows the importance of producing these plots in order to analyse the model and decide whether the estimated relationships meet expectations. In this respect, Merow et al. (2014) analysed the pros and cons of simple versus complex response shapes when calibrating SDMs.

Obviously, when one has no idea what the a priori importance of each variable might be and which should be included in the model, there is a need for some sort of variable selection. Stepwise regression – backward, forward, or both – is a traditional method for examining the relative importance of each derived variable to explain presence–absence or abundance of species. Usually, stepwise regressions are based on the Akaike information criteria (AIC; Akaike, 1974) or its Bayesian derivation (BIC, see Chapter 12), but other information criteria also exist (Johnson and Omland, 2004). In both backward and forward stepwise regressions, variables are tested sequentially, and the one producing the lowest AIC or BIC is retained. The method then assesses the contribution of the other variables after accounting for the variable selected. This approach is appealing as it classifies the variables, ranks them based on their contribution to reducing the total AIC, and retains the most parsimonious combination of variables. The backward and forward strategies differ with regards to the starting model. In the latter case, model selection will start from the intercept (null) model including no variable, while, for the former, it will start from the saturated (full) model, including all the initial variables.

Although stepwise regression is certainly appealing and used to be one of the most commonly used means of reducing complexity in regression-like methods, it is often deemed to be a high-variance exercise since the slightest disturbance in the response data can sometimes lead to vastly different subsets of the variables (Johnson and Omland, 2004; Whittingham et al., 2006). This is especially the case when the number of predictor variables is large (over 10) and the variables correlated with each other. We highly recommend, at least, reducing the number of variables first with PCA, VIF analyses, or simple pairwise correlation tests, to ultimately select a series of non-correlated, ecologically relevant variables (see Part II and Dormann et al., 2013).

The last few years have also seen the development of penalized regression and shrinkage rules as alternatives to stepwise regression. Penalizing algorithms such as "lasso" or "ridge" have gained momentum in the statistical literature, but also in the habitat suitability modeling literature (Hastie et al., 2009; Renner and Warton, 2013; and see Chapter 11). Lasso (Tibshirani, 1996, 1997) and ridge (Hoerl and Kennard, 1970; Le Cessie and van Houwelingen, 1992) provide alternative algorithms that shrink the estimates of the regression coefficients toward zero relative to the maximum likelihood estimates. The overarching goal of the penalty (or shrinkage) is to accurately estimate the parameters while avoiding overfitting either due to multicollinearity of the predictors or overly high dimensionality (i.e. too many predictors). The ridge penalty generally leads to many small but non-zero regression coefficients, while the lasso penalty results in few regression coefficients with little shrinkage and the remaining ones shrunk to zero. However, as in any optimization process, one has to decide a priori what criteria should be used to optimally shrink the parameters. This is determined by tuning a shrinkage parameter (usually called λ) that takes values between zero (i.e. no shrinkage, maximum likelihood estimation) and infinity (i.e. infinite shrinkage, all regression coefficients set to zero). The penalized package offers interesting tools to perform lasso and ridge regressions and select the optimal λ by means of cross-validation.

Here, we provide an example of stepwise selection using the `stepAIC()` function (in the `MASS` package). Let's start by running an intercept model that will serve as the starting model. Then, the `stepAIC()` function will sequentially add and remove the different variables. There are three important parameters in that function: `scope`, `direction` and `k`. Scope can be used to specify the form of the different variables to be

tested (e.g. linear, quadratic, interactions). Direction can be used to specify the direction of the variable selection. Starting with the intercept model means only the forward direction can be used, but one can also use the more advanced both option, combining forward and backward. If glmStart is the saturated model, then backward or both would be the two choices proposed. *k* is the multiple of the number of degrees of freedom used for the penalty. If the `k` parameter is set to `2`, then AIC is used for variable selection. If `k = log(n)`, then variable selection is based on BIC.

The `scope` argument is rather tedious to write, but it can be done using the formula function.

```
> library(MASS)
> glmStart <- glm(VulpesVulpes ~ 1, data = mammals_data,
family = binomial)

> glm.formula <- formula("VulpesVulpes ~ 1 + poly(bio3,2) +
poly(bio7,2) + poly(bio11,2) + poly(bio12,2) + bio3:bio7 +
bio3:bio11 + bio3:bio12 + bio7:bio11 + bio7:bio12 + bio11:bio12")


> glm.formula <- formula("VulpesVulpes ~ 1 + poly(bio3, 2) +
poly(bio7, 2) + poly(bio11, 2) + poly(bio12, 2) + bio3:bio7
+ bio3:bio11 + bio3:bio12 + bio7:bio11 + bio7:bio12 +
bio11:bio12")

> glmModAIC <- stepAIC(glmStart, glm.formula, data = mammals_
data, direction = "both", trace = FALSE, k = 2, control = glm.
control(maxit = 100))

> glmModBIC <- stepAIC(glmStart, glm.formula, direction = "both",
trace = FALSE, k = log(nrow(mammals_data)),
control = glm.control(maxit = 100))


> rp <- response.plot2(models = c("glm1", "glm2", "glmModAIC",
"glmModBIC"), Data = mammals_data[, c("bio3", "bio7", "bio11",
"bio12")], show.variables = c("bio3", "bio7", "bio11", "bio12"),
fixed.var.metric = "mean", plot = FALSE, use.formal.
names = TRUE)
> gg.rp <- ggplot(rp, aes(x = expl.val, y = pred.val, lty = pred.
name)) +
geom_line() + ylab("prob of occ") + xlab("") + rp.gg.theme +
facet_grid(~expl.name, scales = "free_x")
> print(gg.rp)
```

We can see now the effects of the variable selection on the retained best model (Figure 10.3).

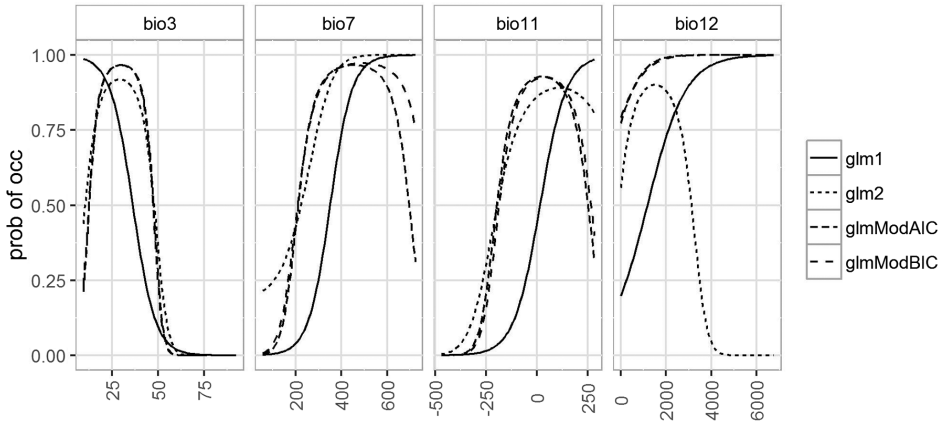

Figure 10.3 Two-dimensional response curves for the different fitted models.

Another way to look at a species' response is to generate and visualize bivariate response curves. Here, we illustrate the example of the best glm selected according to the AIC scores.

```
> rp.2D <- response.plot2(models = c("glmModAIC"),
Data = mammals_data[, c("bio3", "bio7", "bio11", "bio12")],
show.variables = c("bio3", "bio7", "bio11", "bio12"),
fixed.var.metric = "median",
do.bivariate = T, plot = FALSE, use.formal.names = TRUE)
> gg.rp.2D <- ggplot(rp.2D, aes(x = expl1.val, y = expl2.val,
fill = pred.val)) + geom_raster() + rp.gg.theme + ylab("") +
xlab("") + theme(legend.title = element_text()) + scale_fill_
gradient(name = "prob of occ.",
low = "#f0f0f0", high = "#000000") + facet_grid(expl2.name ~
expl1.name, scales = "free")
> print(gg.rp.2D)
```

These bivariate plots allow analysing the joint effects of two variables on the modeled probability of presence (Figure 10.4). For instance, the probability of occurrence is high for high values of bio3 and low values of bio7. When both bio3 and bio7 are both low, the probability of occurrence of the red fox is also low.

The variable rankings can be easily extracted using the `anova()` function.

```
> anova(glmModAIC)
     Analysis of Deviance Table
     Model: binomial, link: logit
     Response: VulpesVulpes
     Terms added sequentially (first to last)
```

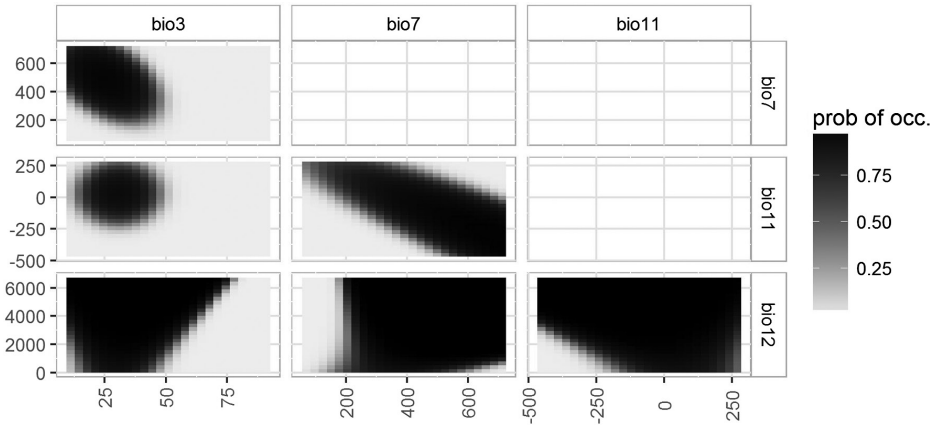

Figure 10.4 Bivariate response curves from the model `glmModAIC` for four predictor variables.

```
Df Deviance Resid. Df Resid. Dev
    NULL                          8541    11839.6
    poly(bio3, 2)    2   5581.8    8539     6257.8
    poly(bio7, 2)    2   1247.0    8537     5010.8
    poly(bio11, 2)   2    299.1    8535     4711.7
    poly(bio12, 2)   2    136.1    8533     4575.7
    bio3:bio7        1    338.3    8532     4237.4
    bio7:bio11       1    129.4    8531     4107.9
    bio11:bio12      1     53.9    8530     4054.0
    bio7:bio12       1     53.0    8529     4001.1
    bio3:bio12       1      3.6    8528     3997.5
```

Variable `bio3` is by far the best explaining variable, followed by `bio7` and `bio11` which strongly influences the distribution of our model species. Although they are less influential than formal variables, adding interaction terms can slightly improve the model's performance.

```
> par(mfrow = c(2, 2))

> level.plot(mammals_data$VulpesVulpes, XY = mammals_data[,
c("X_WGS84", "Y_WGS84")], color.gradient = "grey", cex = 0.3,
level.range = c(0, 1), show.scale = F, title = "Original data")
> level.plot(fitted(glmModAIC), XY = mammals_data[, c("X_WGS84",
"Y_WGS84")], color.gradient = "grey", cex = 0.3, level.
range = c(0, 1), show.scale = F, title = "Stepwise GLM
with AIC")
> level.plot(fitted(glmModBIC), XY = mammals_data[, c("X_WGS84",
"Y_WGS84")], color.gradient = "grey", cex = 0.3, level.
range = c(0, 1), show.scale = F,
title = "Stepwise GLM with BIC")
```

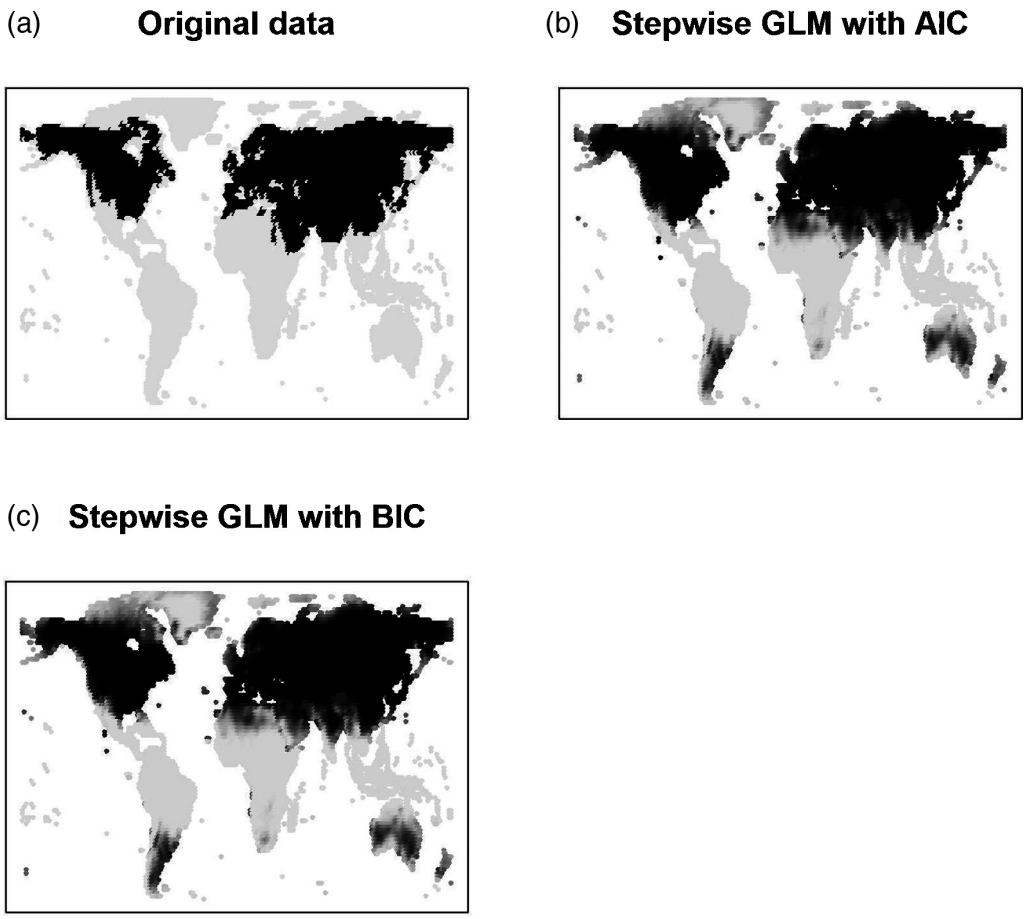

Figure 10.5 (a) Observed (black = presence, light gray = absence) and potential distribution of red fox extracted from (b) `glmModAIC` and (c) `glmModBIC` models. The gray scale of predictions shows habitat suitability values between 0 (light, unsuitable) and 1 (dark, highly suitable).

The potential distribution of the red fox does not differ significantly between the two stepwise procedures (Figure 10.5). We would have expected larger differences primarily when using small sample sizes, but not when using big datasets as is the case here.

10.3 Generalized Additive Models

GAMs are techniques designed to capitalize on the strengths of GLMs but which do not require postulating a shape for the response curve from a specific parametric function. GAMs use algorithms called "smoothers" that automatically fit response curves "as closely as possible" to the data given the permitted level of smoothing. GAMs are therefore useful when

the relationship between the variables is expected to be of a more complex form, not easily fitted with standard parametric functions of the predictors (e.g. GLM with a linear or quadratic response), or where there is no a priori reason for using a particular shape (Hastie and Tibshirani, 1990). If one wants to remain within a parametric scheme, GAM can also be used in complement to GLM, firstly to explore the general shape of the response function and then to implement it in the best possible way in a GLM (Guisan et al., 2006b). Link and family in GAM are the same as in GLM.

There are now several packages, which can be used to fit GAMs in R (e.g. `gam`, `mgcv`, `gamair`, `GAMBoost`). The `gam` package iteratively fits weighted additive models using backfitting (i.e. iteratively smoothing partial residuals (Hastie et al., 2009). There are different smoothers available, but the most commonly used is the cubic-spline smoother, a collection of polynomials of degree less than or equal to 3, defined on subintervals. A separate polynomial model is fitted in each neighborhood (using a moving window algorithm), thus enabling the fitted curve to connect all the points. Nevertheless, the user has to predetermine the degree of smoothing applied when fitting the curve (or select it through cross-validation). In the SDMs field, researchers have generally used degrees lower than 4, which corresponds roughly to a polynomial of degree 3 (Hastie et al., 2009). Higher degrees will generate more locally complex curves.

The syntax is exactly the same as for a GLM, except that the user needs to specify the smoother (below, a cubic-spline called *s*) and the degree of smoothing (below 2 and 4). Note that the degree of smoothing can change across the variables in a model (i.e. a different smoothing level can be specified for each variable).

```
> if (is.element("package:mgcv", search()))
detach("package:mgcv")  ## make sure the mgcv package is not
loaded to avoid conflicts between packages

> library(gam)
> gam1 <- gam(VulpesVulpes ~ s(bio3, 2) + s(bio7, 2) + s(bio11,
2) + s(bio12, 2), data = mammals_data, family = "binomial")
> gam2 <- gam(VulpesVulpes ~ s(bio3, 4) + s(bio7, 4) + s(bio11,
4) + s(bio12, 4), data = mammals_data, family = "binomial")
```

The `gam` package provides its own function (`plot.gam()`) to extract the response curves, which works in exactly in the same way as the `response.plot2()` function in the `biomod2` package. However, it is important to note that these responses are expressed in the transformed unit (here, the logit scale; Figure 10.6). A very nice feature of `plot.`

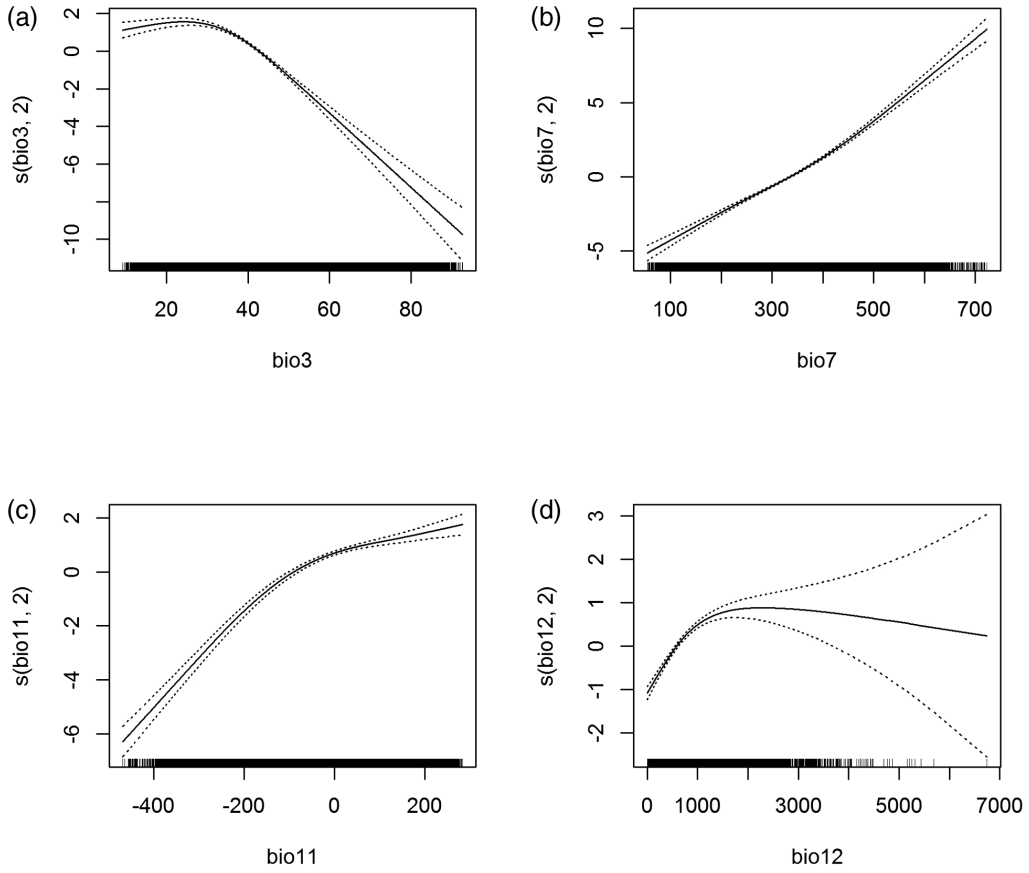

Figure 10.6 Response curves of model gam1 expressed in logit scale (function `plot.gam()` from the `gam` package).

`gam()` is the possibility to include upper and lower point-wise ±2 standard error curves.

```
> par(mfrow = c(2, 2))
> plot(gam1, se = T)
```

We can compare the influence of the degree of smoothing on the response curves expressed in the original unit (between 0 and 1) using the `response.plot2()` function in the `biomod2` package.

```
> rp <- response.plot2(models = c("gam1", "gam2"),
Data = mammals_data[,
c("bio3", "bio7", "bio11", "bio12")],
show.variables = c("bio3", "bio7", "bio11", "bio12"),
fixed.var.metric = "mean", plot = FALSE, use.formal.names = TRUE)
> gg.rp <- ggplot(rp, aes(x = expl.val, y = pred.val, lty = pred.
name)) + geom_line() + ylab("prob of occ") + xlab("") + rp.gg.
theme + facet_grid(~expl.name, scales = "free_x")
> print(gg.rp)
```

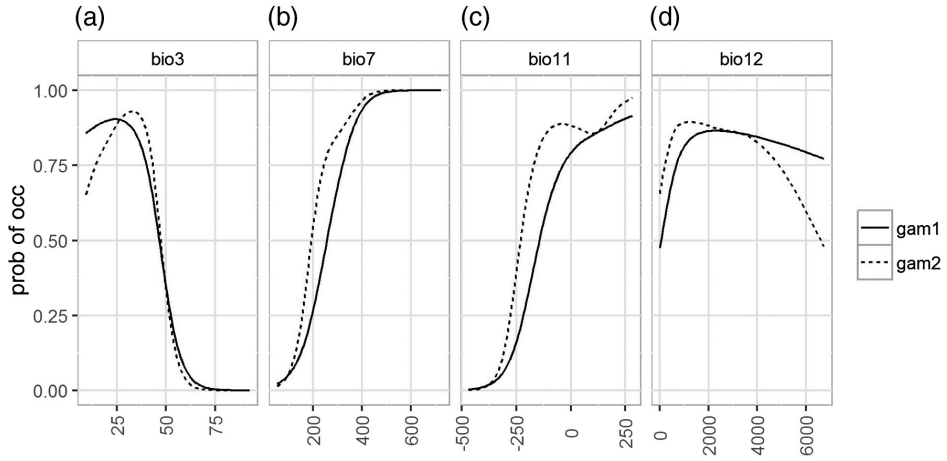

Figure 10.7 Response curves of the `gam1` (degree of smoothing = 2) and `gam2` (degree of smoothing = 4) models.

Note that the response curves are quite similar to those obtained from the GLMs (Figure 10.7). Therefore, it is clear that the degree of smoothing has a relatively small effect in this example. However, it is important to carefully check the complexity of models. GAMs are data-driven and thus prone to overfitting the data when highly complex smoothers are used. When modeling species distributions for predictive purposes, we do not recommend using degree of smoothing higher than 4 or 5. Users who want to model more complex relationships, e.g. in order to very closely fit and predict the calibration data, may use a higher degree of smoothing, but at the cost of reduced generalization (Merow et al., 2014).

Similarly to a GLM, the `gam()` function supports various options for variable selection using stepwise procedures or shrinkage rules. These are implemented in the same way as in a GLM. It is also possible to use a custom function for the scope argument from the `biomod2` package (function.`scope()`). Here we will illustrate the use of the stepwise procedure with another function called `step.gam()` (note however that the `stepAIC()` function also works for `gam()` and can be implemented in the same way as previously shown for GLM).

```
> gamStart <- gam(VulpesVulpes ~ 1, data = mammals_data,
family = binomial)
> gamModAIC <- step.gam(gamStart, biomod2:::.scope(mammals_
data[1:3,
c("bio3", "bio7", "bio11", "bio12")], "s", 4), trace = F,
direction = "both")
```

In the step.gam procedure, we will test a single degree of smoothing (here 4), which represents the maximum degree achieved by the model.

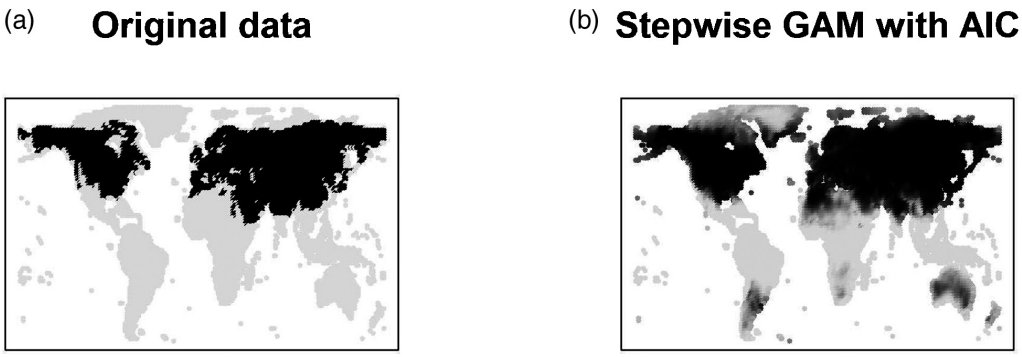

Figure 10.8 (a) Observed (black = presence, light gray = absence) and (b) potential distribution of *Vulpes vulpes* extracted from `gamModAIC`. The gray scale of prediction shows habitat suitability values between 0 (light, unsuitable) and 1 (dark, highly suitable).

In practice, when the observed relationship is linear, the GAM will also fit a linear relationship even if the degree has been pre-set to 4. An alternative would be to test for different degree of smoothing using c(2,3,4) instead of 4.

The spatial prediction can easily be displayed and compared with the observed distribution (Figure 10.8).

```
> par(mfrow = c(1, 2))

> level.plot(mammals_data$VulpesVulpes, XY = mammals_data[,
c("X_WGS84", "Y_WGS84")], color.gradient = "grey",
cex = 0.3, level.range = c(0, 1), show.scale = F,
title = "Original data")

> level.plot(fitted(gamModAIC), XY = mammals_data[, c("X_WGS84",
"Y_WGS84")], color.gradient = "grey", cex = 0.3, level.
range = c(0, 1), show.scale = F,
title = "Stepwise GAM with AIC")
```

Alternatively, the `mgcv` package provides a slightly different version of GAM. Smooth terms are implemented through penalized regression splines with smoothing parameters selected through generalized cross-validation or AIC in the `mgcv` package, or regression splines with fixed degrees of freedom, as in the `gam` package. The most interesting feature is the possibility to explore interactions between variables through multidimensional smoothers using penalized thin-plate regression splines (isotropic) or tensor product splines (when an isotropic smooth is inappropriate) (Wood, 2006).

The default syntax in mgcv is very similar to the `gam` package except that the user does not have to specify the degree of smoothing, which is automatically defined by means of internal cross-validation.

```
> if (is.element("package:gam", search())) detach("package:gam")
## make sure the gam package is not loaded to avoid conflicts
> library(mgcv)
> gam_mgcv <- gam(VulpesVulpes ~ s(bio3) + s(bio7) + s(bio11) +
s(bio12), data = mammals_data, family = "binomial")
> ## see a range of summary statistics
> summary(gam_mgcv)

 Family: binomial
     Link function: logit

 Formula:
     VulpesVulpes ~ s(bio3) + s(bio7) + s(bio11) + s(bio12)

 Parametric coefficients:
                 Estimate Std. Error z value Pr(>|z|)
     (Intercept)  -1.3421     0.5091  -2.636  0.00839 **
     ---
     Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
0.1 ' ' 1

 Approximate significance of smooth terms:
                edf Ref.df Chi.sq  p-value
     s(bio3)  5.888  6.589 491.77  < 2e-16 ***
     s(bio7)  6.601  7.358 732.09  < 2e-16 ***
     s(bio11) 8.740  8.968 347.37  < 2e-16 ***
     s(bio12) 7.013  8.022  56.49 2.34e-09 ***
     ---
     Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

 R-sq.(adj) =  0.712   Deviance explained = 65.8%
     UBRE = -0.51977  Scale est. = 1         n = 8542
> gam.check(gam_mgcv)
```

The mgcv package provides a lot of summary statistics that can be very useful when carefully examined (see `gam.check()`). Additionally, response curves can also be plotted using the internal functions of `mgcv` (Figure 10.9).

```
> plot(gam_mgcv, pages = 1, seWithMean = TRUE)
```

This makes it possible to compare the response curves from the `mgcv` implementation of GAM to those from the `gam` package (Figure 10.10).

```
> rp <- response.plot2(models = c("gam1", "gam2"),
Data = mammals_data[,
c("bio3", "bio7", "bio11", "bio12")],
show.variables = c("bio3", "bio7", "bio11", "bio12"),
fixed.var.metric = "mean", plot = FALSE, use.formal.names = TRUE)
> gg.rp <- ggplot(rp, aes(x = expl.val, y = pred.val,
lty = pred.name)) + geom_line() + ylab("prob of occ") + xlab("")
+ rp.gg.theme + facet_grid(~expl.name, scales = "free_x")
> print(gg.rp)
```

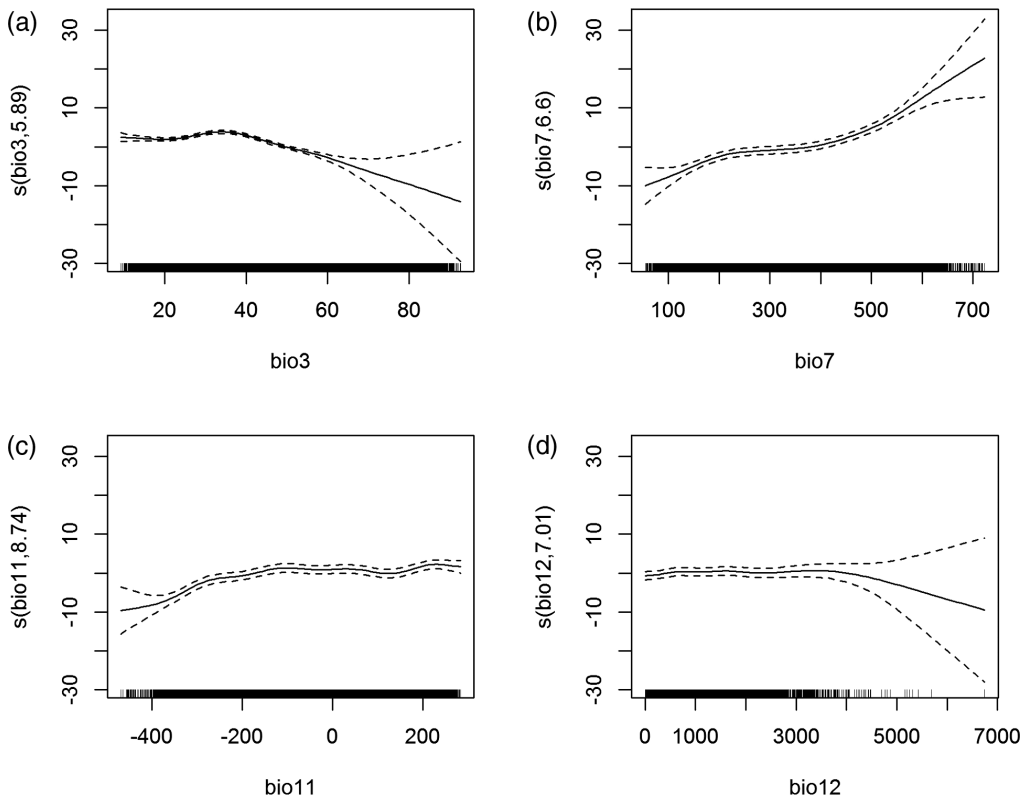

Figure 10.9 Response curves of model `gam_mgcv` plotted using the internal function of `mgcv()`.

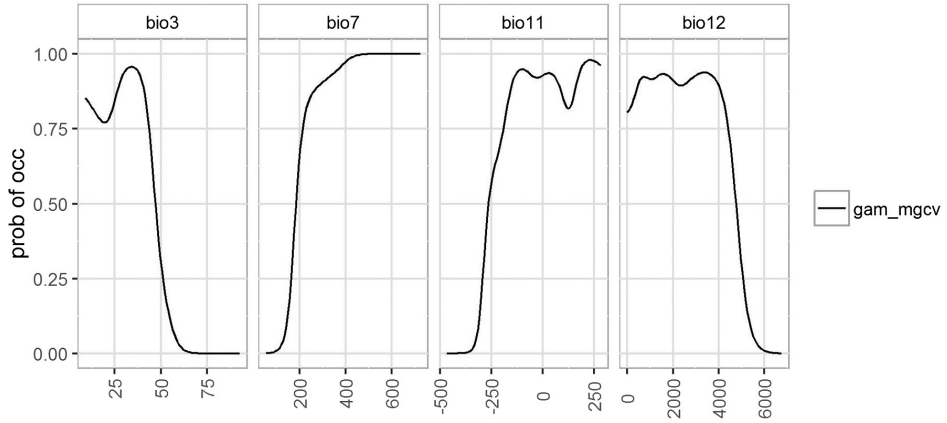

Figure 10.10 The response curves from the model calibrated with the `mgcv` package (`gam_mgcv`).

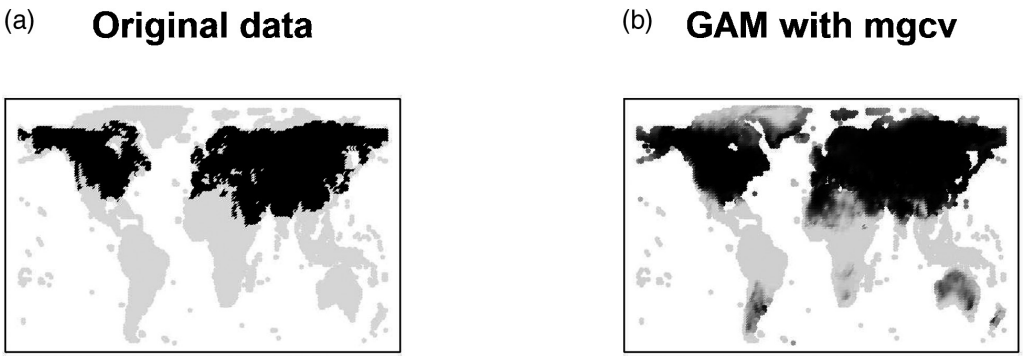

Figure 10.11 (a) Observed (black = presence, light gray = absence) and (b) potential distribution of *Vulpes vulpes* extracted from the `gam_mgcv` object. The gray scale of predictions illustrates habitat suitability values between 0 (light, unsuitable) and 1 (dark, highly suitable).

Despite these slight differences between models calibrated with the gam algorithm from `gam` package (`gam1`, `gam2` and `gamModAIC`) and the model calibrated with the `mgcv` package (`gam_mgcv`), the resulting spatial predictions (see Part V) are similar to those obtained from the `gam` package (Figure 10.11).

We can see that for this particular species (*V. vulpes*), all the models we have seen so far (except SRE and ENFA) have yielded quite similar potential distributions. We will later learn about (Part IV) different ways of testing the predictive accuracy of different models in order to obtain quantitative metrics and compare their predictive power.

10.4 Multivariate Adaptive Regression Splines

Like GAM, multivariate adaptive regression splines (MARS) constitute a more flexible regression technique than GLM, as they also do not require any assumptions to be made about the underlying functional relationship between the species and the environmental variables. Instead of using a predefined shape, such as polynomial functions in GLMs, MARS fits piecewise functions that together can accommodate nonlinear responses. In this sense, it is quite similar to GAM and the smoothed functions. Knots define the breaks between segments and different regression lines with different slopes are thus fitted between each pair of knots, while the full fitted function is constrained to have no breaks or abrupt steps. Generalized cross-validation is used to assess the effect of adding or removing knots. Backward and forward variable selection is also possible, as in GAM and GLM.

MARS is implemented in R in both the `mda` and `earth` package. Here, we use the `earth` package, which provides additional functions that are not available in `mda`.

Very few parameters are required to fit a MARS model. One important parameter concerns the maximum interaction degree, which determines whether interactions between variables are fitted or not. This is set to one by default, but more complicated response curves are likely to be required in certain instances. In the following examples, we thus use both a degree of 1 (no interactions) and 2 (pairwise interactions).

```
> library(earth)
> Mars_int1 <- earth(VulpesVulpes ~ 1 + bio3 + bio7
+ bio11 + bio12, data = mammals_data, degree = 1,
glm = list(family = binomial))
> Mars_int2 <- earth(VulpesVulpes ~ 1 + bio3 + bio7
+ bio11 + bio12, data = mammals_data, degree = 2,
glm = list(family = binomial))
> ## print the summary of objects
> Mars_int1
     Earth selected 14 of 15 terms, and 4 of 4 predictors
     Termination condition: Reached nk 21
     Importance: bio7, bio11, bio3, bio12
     Number of terms at each degree of interaction: 1 13
(additive model)
     Earth GCV 0.08460021    RSS 718.0938    GRSq 0.6615926
RSq 0.6636498

 GLM null.deviance 11839.56 (8541 dof)   deviance 4267.856 (8528
dof)   iters 11
> Mars_int2
     Earth selected 18 of 21 terms, and 4 of 4 predictors
     Termination condition: Reached nk 21
     Importance: bio7, bio3, bio11, bio12
     Number of terms at each degree of interaction: 1 4 13
     Earth GCV 0.07349056    RSS 621.379    GRSq 0.7060321    RSq
0.7089503

 GLM null.deviance 11839.56 (8541 dof)   deviance 3625.926 (8524
dof)   iters 25 did not converge
```

From the summary statistics in `earth()`, we can visualize that the r-square of Mars_int2 ($R^2 = 0.71$) is slightly better than the simpler version (Mars_int1, $R^2 = 0.66$).

```
> summary(fitted.values(Mars_int1))
       VulpesVulpes
      Min.   :-0.5111
```

```
      1st Qu.: 0.0582
      Median : 0.5105
      Mean   : 0.4920
      3rd Qu.: 0.8625
      Max.   : 1.2911
```

There is no in-built `fitted.value` function in MARS. If a user wants to extract the predictions the `predict` function needs to be used and the `type` argument "`response`" employed to make sure the predictions are converted to the appropriate scale.

```
> pred_Mars_int1 <- predict(Mars_int1, type = "response")
> summary(pred_Mars_int1)
       VulpesVulpes
      Min.   :0.0000001
      1st Qu.:0.0240546
      Median :0.5087504
      Mean   :0.4920393
      3rd Qu.:0.9383953
      Max.   :0.9996739
> pred_Mars_int2 <- predict(Mars_int2, type = "response")
> summary(pred_Mars_int2)
       VulpesVulpes
      Min.   :0.0000
      1st Qu.:0.0227
      Median :0.4513
      Mean   :0.4920
      3rd Qu.:0.9731
      Max.   :1.0000
```

One interesting feature of the `earth` package is that it can be used to plot the distribution of observed presences and absences across classes of predicted values. This allows us to clearly visualize whether species presences are located within high values of predictions, and conversely whether absences are distributed in areas with low prediction values (Figure 10.12).

```
> plotd(Mars_int1, hist = T)
```

Figure 10.12 clearly shows the discriminatory power of MARS for this species, as the probability values are always extremely low when the species is absent and reciprocally high when the species is present. The optimal threshold for transforming the probability values into presence–absence is also easily identifiable in this case (around 0.5–0.6).

We can also see the difference between the two MARS models, which differ in terms of the amount of interaction between variables (one

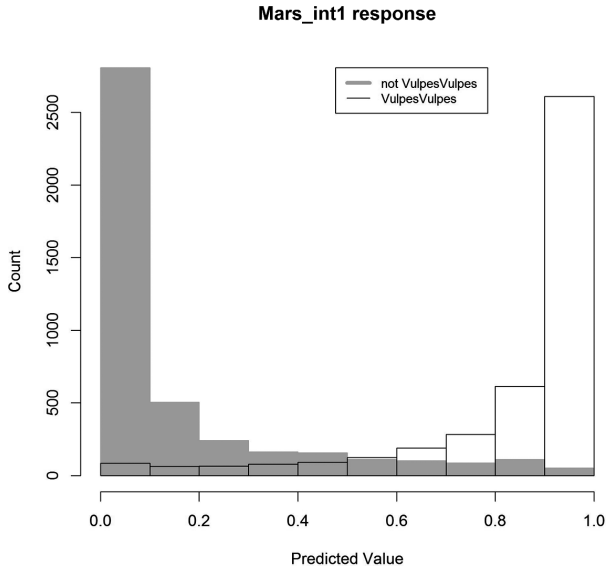

Figure 10.12 The distribution of the predicted values from MARS for both the presence and absence of *Vulpes vulpes*.

model with no interaction, one with pairwise interactions). This can be represented visually by plotting the fitted probabilities against each other (Figure 10.13).

```
> plot(pred_Mars_int1, pred_Mars_int2,
xlab = "MARS with max inter degree 1",
ylab = "MARS with max inter degree 2")
```

Although the two MARS models have similar predictions at and close to 0 and 1, we can see fairly high variability in the predicted probabilities at intermediate values (Figure 10.13). For instance, there are points where the probability of occurrence is close to 1 in MARS with no interactions, whereas it is close to 0 when predicted by MARS with 2 degrees of interactions. This highlights that those kinds of choices are not without consequences, so need to be very carefully evaluated.

The spatial predictions of the two versions of the model look relatively similar overall, except for few regions such like Greenland and some parts of the southern hemisphere (Figure 10.14)

```
> par(mfrow = c(2, 2))
> level.plot(mammals_data$VulpesVulpes, XY = mammals_data[,
c("X_WGS84", "Y_WGS84")], color.gradient = "grey", cex = 0.3,
level.range = c(0, 1), show.scale = F,
title = "Original data")
```

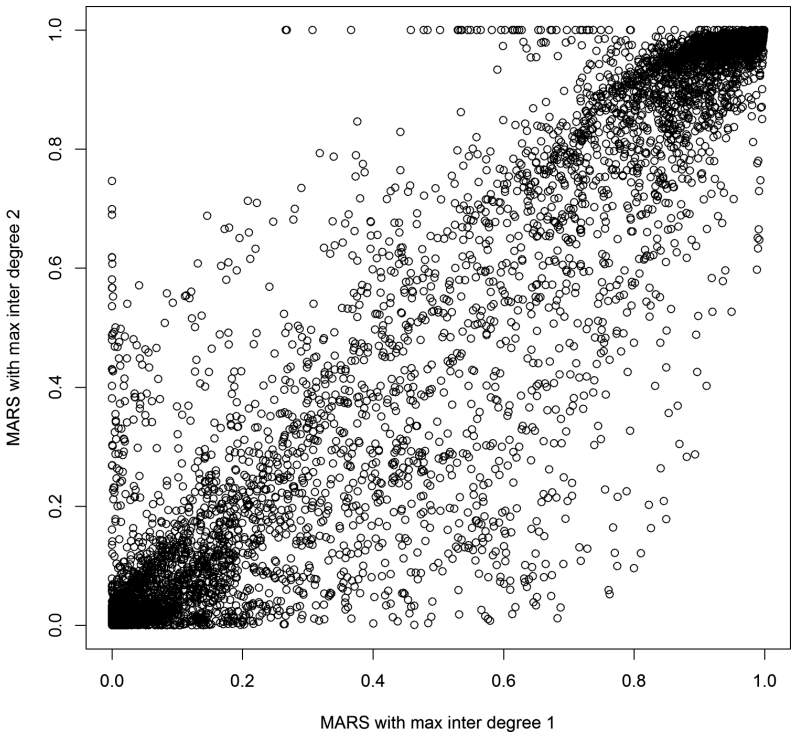

Figure 10.13 Differences between probability of occurrence between a MARS model with a maximum of 1 degree of interaction and a MARS model with a maximum of 2 degrees of interaction.

```
> level.plot(pred_Mars_int1, XY = mammals_data[, c("X_WGS84",
"Y_WGS84")], color.gradient = "grey", cex = 0.3, level.
range = c(0, 1),
show.scale = F, title = "MARS with interaction degree 1")
> level.plot(pred_Mars_int2, XY = mammals_data[, c("X_WGS84",
"Y_WGS84")], color.gradient = "grey", cex = 0.3, level.
range = c(0, 1),
show.scale = F, title = "MARS with interaction degree 2")
```

The response curves for MARS are not shown here, as they can be extracted using the same function as in GLM or GAM, as shown above.

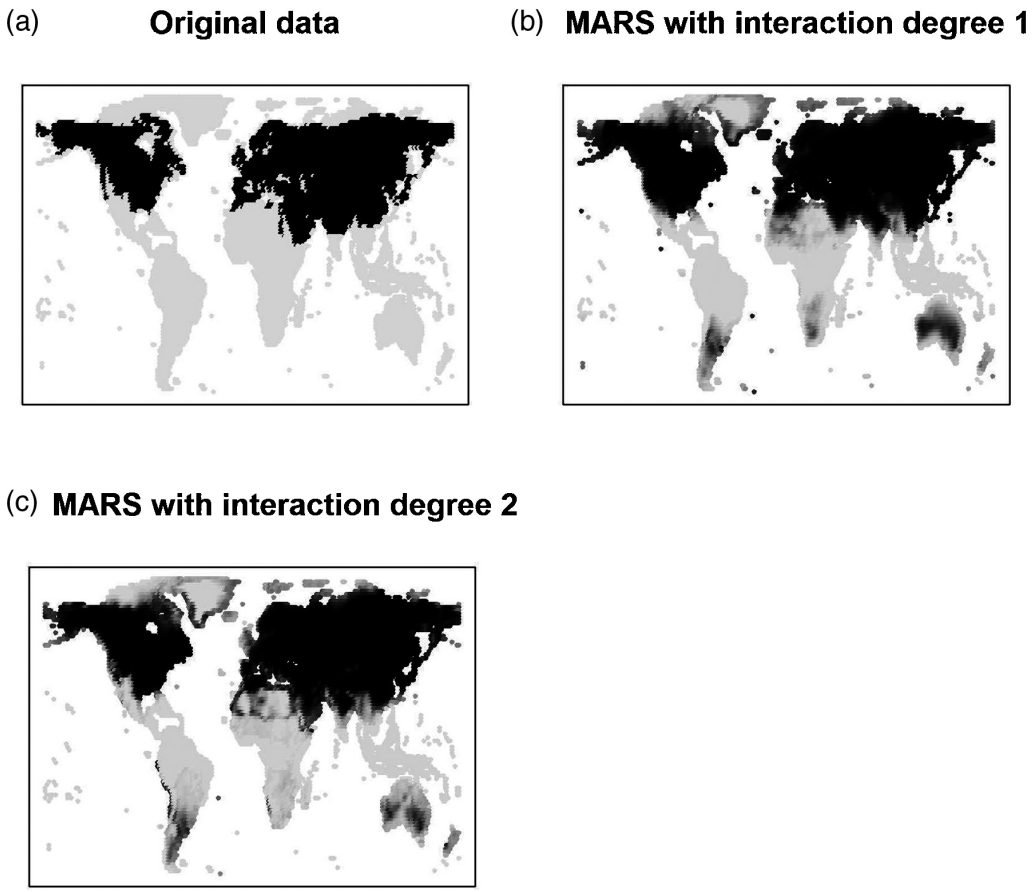

Figure 10.14 (a) Observed (black = presence, light gray = absence) and potential distribution of *Vulpes vulpes* extracted from the (b) MARS 1 and (c) MARS 2 objects. The gray scale of predictions (upper-right and lower-left panels) illustrates habitat suitability values between 0 (light, unsuitable) and 1 (dark, highly suitable).