ADDENDUM 1

TECHNICAL DESCRIPTION OF ARTIFICIAL NEURAL NETWORKS, GEOSTATISTICS AND MAXIMUM ENTROPY MODELLING

ADDENDUM 1

TECHNICAL DESCRIPTION OF ARTIFICIAL NEURAL NETWORKS, GEOSTATISTICS AND MAXIMUM ENTROPY MODELLING

ARTIFICIAL NEURAL NETWORKS

Artificial neural networks (ANNs) are non-linear mapping structures based on the functioning of the human brain. They have been shown to be highly flexible approximators for any data. ANNs make powerful tools for modelling complex relationships, especially when the underlying data relationships are unknown (Lek and Guégan, 1999).

Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological nervous systems. As in nature, the network function is determined largely by the connections between elements. By adjusting the values of the connections (weights) between elements, neural networks can be trained to perform a particular function. Commonly neural networks are adjusted or trained, to assure that a particular input leads to a specific target output. During the learning process, network outputs and targets are compared; networks are adjusted until the output matches the target. Typically, many such input/target pairs are used in this supervised learning to train a network (Demuth and Beale, 1998). In this research the input variables are the environmental variables and the target values are the diversity indices. The relationships between variables in ecology are often very complicated and highly non-linear. Therefore, neural networks are considered to be powerful tools to investigate these relationships. ANNs have indeed the capacity to predict the output variable but the mechanisms that occur within the network are often ignored. Therefore, ANNs are often considered as black boxes (Gevrey *et al.*, 2003). Several methods have been described to unravel these connections.

Architecture of an Artificial neural network

A Simple Neuron

A neuron with a single scalar input is shown in Fig. A1.1. The input p is multiplied by the weight w (this is a so called 'connection' in the neural network), to form the product $w.p$. This forms the scalar output n of the neuron and n is the argument of the transfer function$f$, which produces the output a.

The neuron in the right panel of Fig. A1.1 has a bias b. This bias is simply added to the product $w.p$. The bias is like a weight, except that it is multiplied with 1. The input of the transfer function is the sum of the weighted input $w.p$ and the bias b. This sum, called n, is the argument of the transfer function *f.* Here f is a transfer function that takes the argument n and produces the output a.

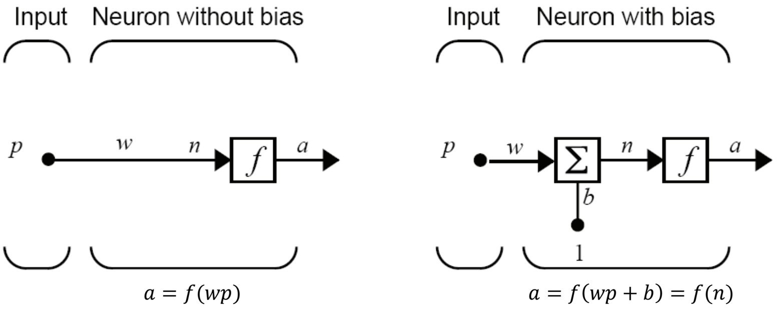

*Fig. A1.1. A neuron with a single scalar input and with and without bias (*from *Demuth and Beale, 1998)*

Note that w and b are both adjustable parameters of the neuron. The central idea of neural networks is that these parameters can be adjusted so that the network exhibits a desired behaviour. Thus, the network can be trained to do a particular job by adjusting the weight or bias parameters, or the network itself (Demuth and Beale, 1998).

Transfer Functions

Three of the most commonly used transfer functions are shown in Fig. A1.2. The hard limit transfer function limits the output of the neuron to either 0, if the net input argument n is less than 0, or 1, if n is greater than or equal to 0. The linear transfer function allows the output to take any value. The log-sigmoid transfer function transforms the input to any value between 0 and 1. It is commonly used in back propagation networks, in part because it is differentiable (Beale *et al.*, 2010).

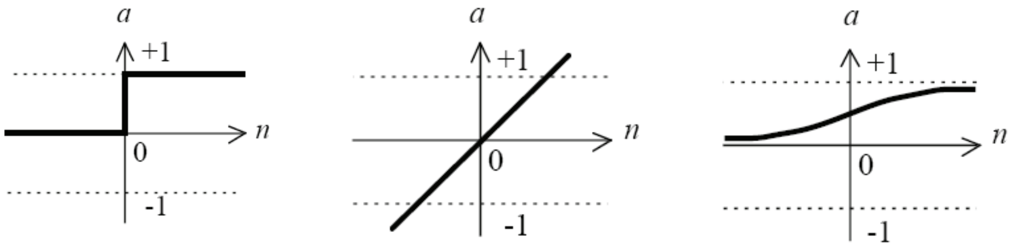

*Fig. A1.2. Three types of transfer functions: Hard limit transfer function, linear transfer function and log-sigmoid transfer function (*from *Beale* et al.*, 2010).*

Neuron with Vector Input

A neuron with an input vector with R elements is shown below (Fig. A1.3).The scalar product of the (single row) matrix $\boldsymbol{W}$ and the vector $\boldsymbol{p}$ results in $\boldsymbol{W.p}$. The neuron has a bias b, which is summed with $\boldsymbol{W.p}$ to form the input n which is the argument of the transfer function f. Two or more of the neurons shown above may be combined in a layer, and a particular network might contain one or more such layers. First consider a single layer of neurons.

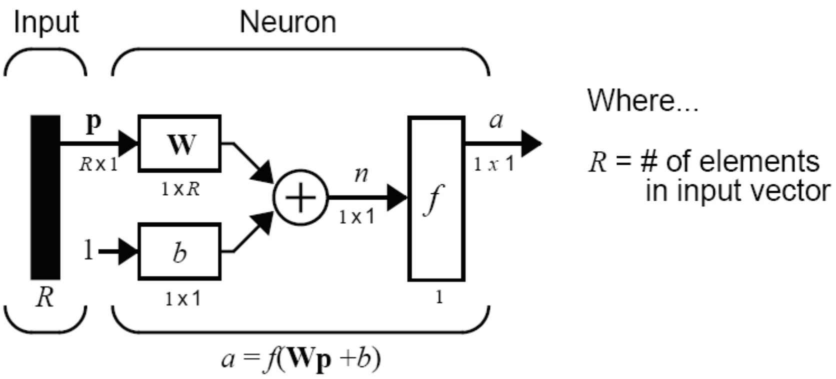

*Fig. A1.3. One neuron with an input vector with R elements (*from *Beale* et al.*, 2010).*

One layer of Neurons

A one layer network with R input elements and S neurons is shown in Fig. A1.4. A layer includes the combination of the weights, the multiplication and summing operation, the bias $\boldsymbol{b}$, and the transfer function f. The array of inputs, vector $\boldsymbol{p}$, is not included in a layer.

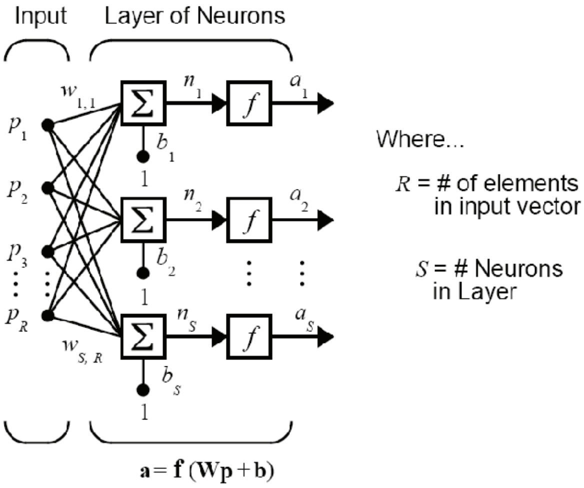

*Fig. A1.4. One layer network with R input elements and S neurons (*from *Demuth and Beale, 1998).*

In this network, each element of the input vector $\boldsymbol{p}$ is connected to each neuron input through the weight matrix $\boldsymbol{W}$, which has the dimensions SxR. The i^{th} neuron has a summation operator that gathers its weighted inputs and bias to form its own scalar output n_i. The n_i form a vector with S elements, called $\boldsymbol{n}$, which is fed through the transfer function and results in the neuron layer output: a vector $\boldsymbol{a}$ with dimension $Sx1$. A layer is not constrained to have the number of its inputs and the same number of neurons.

Multiple Layers of Neurons

A network can have several layers. Each layer has a weight matrix $\boldsymbol{W}$, a bias vector $\boldsymbol{b}$, and an output vector $\boldsymbol{a}$.

The layers of a multilayer network play different roles. A layer that produces the network output is called an output layer. All other layers are called hidden layers. The three layer network shown in Fig. A1.5 has one output layer (layer 3) and two hidden layers (layer 1 and layer 2). Some authors may refer to the inputs as a fourth layer. Weight matrices connected to inputs are called input weights and weight matrices coming from layer outputs are called layer weights (Demuth and Beale, 1998).

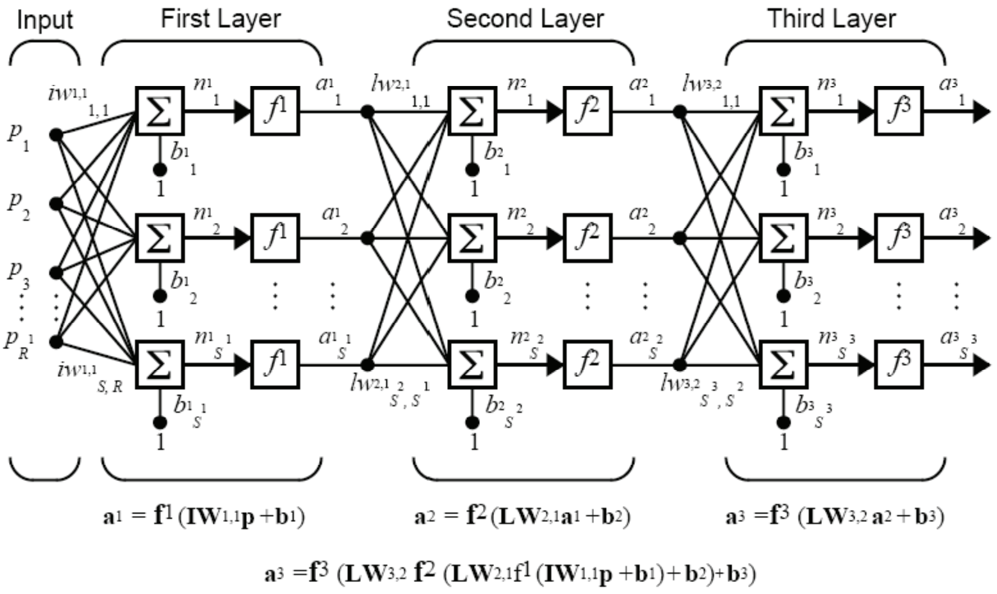
Input
First Layer
Second Layer
Third Layer
$a^1 = f^1(IW^{1,1}p + b^1)$
$a^2 = f^2(LW^{2,1}a^1 + b^2)$
$a^3 = f^3(LW^{3,2}a^2 + b^3)$
$a^3 = f^3(LW^{3,2}f^2(LW^{2,1}f^1(IW^{1,1}p + b^1) + b^2) + b^3)$

*Fig. A1.5. Three layer neural network (*from *Demuth and Beale, 1998)*

Multiple layer networks are quite powerful. For instance, a network of two layers, where the first layer is sigmoid and the second layer is linear, can be trained to approximate any function (with a finite number of discontinuities) arbitrarily well (Beale *et al.*, 2010).

Training a neural network

The process of optimizing the connection weights is known as 'training' or 'learning'. This is equivalent to the parameter estimation in conventional statistical models (Maier and Dandy, 2000). The initial network weights and biases are randomly initialised. After this initialisation, the network is ready for training. The training process requires a set of examples of proper network behaviour - network inputs p and target outputs t. During training the weights and biases of the network are iteratively adjusted to minimise the network performance function. The default performance function is the mean square error (MSE) which is the average squared error between the network outputs a and the target outputs t (Demuth and Beale, 1998). In many cases there are practical difficulties in optimizing the error function, because the error surface may be complicated and have many local minima (Cheng and Titterington, 1994).

There are many variations of the back propagation algorithm. The simplest implementation of back propagation learning updates the network weights and biases in the direction in which the performance function decreases most rapidly - the negative of the gradient. One iteration of this algorithm can be written as $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - a_k\boldsymbol{g}_k$ where $\boldsymbol{x}_k$ is a vector of current weights and biases, $\boldsymbol{g}_k$ is the current gradient, and a_k is the learning rate (or step size).

There are different training algorithms. In batch mode, the weights and biases of the network are updated only after the entire training set has been applied to the network. The gradients calculated at each training example are added together to determine the change in the weights and biases. For this research we applied the Levenberg-Marquardt training algorithm. The Levenberg-Marquardt algorithm is faster than other algorithms by a factor 10 to 100 (Beale *et al.*, 2010). For most situations, the Levenberg-Marquardt algorithm is recommended. The only drawback of this algorithm is that it may require too much computer memory. If this is a problem, one can make use of a variety of other fast algorithms available (Beale *et al.*, 2010). One iteration of this algorithm can be written as:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - [\boldsymbol{J}^T\boldsymbol{J} + \mu\boldsymbol{I}]^{-1}\boldsymbol{J}^T\boldsymbol{e} \qquad \text{(Eq. A1.1)}$$

where μ is the learning rate and $\boldsymbol{I}$ the identity matrix, $\boldsymbol{J}$ is the Jacobian matrix which contains first derivatives of the network errors with respect to the weight and biases, and $\boldsymbol{e}$ is a vector of network errors. μ is decreased after each successful step and is increased when an individual step increases the performance function (Karul *et al.*, 2000). Since the weights and biases are initialised before training, training the network several times, may result in different resulting networks.

Improving Generalisation

One of the problems occurring during neural network training is called overfitting. An overly complex neural network may have too many adjustable parameters and the error on the training set is driven to a very small value, but when new data is presented to the network, the error is large. The network has memorised the training examples, but it has not learned to generalise to new situations.

One method for improving network generalisation is to use a network which is just large enough to provide an adequate fit. The larger a network is, the more complex the functions that the network can create. If a small enough network is used, it will not have enough power to overfit the data. Overfitting is linked to the ratio of the number of training samples to the number of connection weights. Amari *et al.* (1997) show that overfitting does not occur if the above ratio exceeds 30. When the above condition is not met, there are clear benefits in using cross-validation.

Early Stopping with validation

A method for improving generalisation is called early stopping. In this technique the available data is divided into three subsets: the training set, the validation set and the test set. The first subset is the training set which is used for computing the gradient and updating the network weights and biases. The second subset is the validation set. The error on the validation set is monitored during the training process. The validation error will normally decrease during the initial phase of training, as does the training set error. However, when the network begins to overfit the data, the error on the validation set will typically begin to rise. When the validation error increases for a specified number of iterations, the training is stopped, and the weights and biases at the minimum of the validation error are returned (Fig. A1.6).

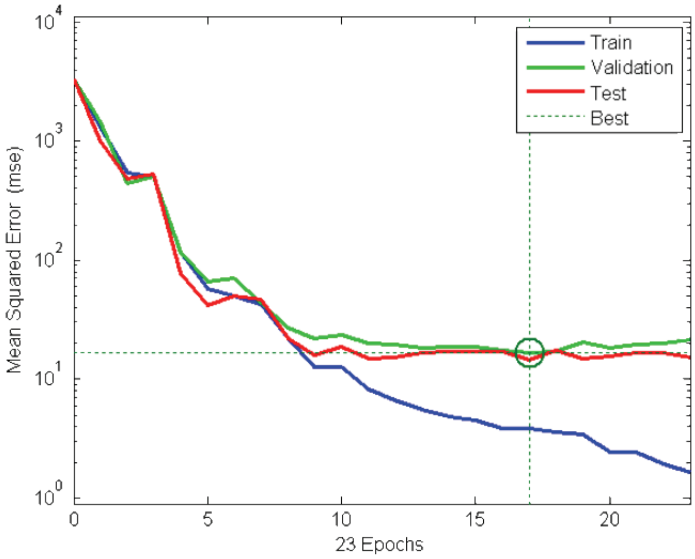

*Fig. A1.6. Evolution of the mean squared error of the training set, the validation set and the test set during training (*from *Beale* et al.*, 2010). The training is stopped when the error on the validation set reaches a minimum.*

It is vital that the third set, the test set, is not used during the training process (Maier and Dandy, 2000). The test set error is only used to compare the performance of different models, so it is a truly independent dataset. It is also useful to plot the test set error during the training process. If the error in the test set reaches a minimum at a significantly different iteration number than the validation set error, this may indicate that the model has been overfitted or the two datasets (test and validation) are not representative of the same population (Masters, 1993). A solution to the latter problem is using stratified datasets, this means that the classes of the predicted variable are evenly distributed over the subsets. In some cases also stratification of the independent variables can be useful (Goethals, 2005).

Cross-validation

Cross-validation is a technique that is frequently used in ANN modelling. In cross-validation a validation set is used to assess the performance of the model at various stages of learning. A different test set is needed to assess the generalisation ability of the different models (Maier and Dandy, 2000). Fig. A1.7 shows how data is divided in a 10-fold cross-validation. The model is built by using 80% of the data as training data. Early stopping of the iterations is done by the use of a validation set. The generalisation performance of the model is screened by applying the test set on the resulting model and calculate the error function. The error function is then averaged over the ten models. By repeating this for different neural network architectures, the optimal architecture can be found. This is the architecture with the lowest average error.

train	train	train	train	train	train	train	train	val	test	→model1
test	train	train	train	train	train	train	train	train	val	→model2
val	test	train	train	train	train	train	train	train	train	→model3
train	val	test	train	train	train	train	train	train	train	→model4
train	train	val	test	train	train	train	train	train	train	→model5
train	train	train	val	test	train	train	train	train	train	→model6
train	train	train	train	val	test	train	train	train	train	→model7
train	train	train	train	train	val	test	train	train	train	→model8
train	train	train	train	train	train	val	test	train	train	→model9
train	train	train	train	train	train	train	val	test	train	→model10

Fig. A1.7. Data division in a tenfold cross-validation with validation and independent test set. The eight individual training sets are joined in one training set (val is the validation set, train is the training set).

Preprocessing

Neural network training can be made more efficient if certain preprocessing steps are performed on the network inputs and targets. Below only those optimisation methods used in this research are mentioned.

The network inputs

Generally, different input variables span different ranges. In order to ensure that all variables receive equal attention during the training process, they should be standardised (Maier and Dandy, 2000). The network inputs can be scaled by normalizing the mean and standard deviation of the training set. It normalises the inputs so that they will have zero mean and unity standard deviation.

In some situations the dimension of the input vector is large, and the components of the vectors may be highly correlated (redundant variables). It is useful in this situation to reduce the dimension of the input vectors. An effective procedure for performing this operation is principal component analysis (PCA). This technique has three effects: it orthogonalises the components of the input vectors (so that they are uncorrelated with each other); it orders the resulting orthogonal components (principal components) so that those with the largest variation come first; and it eliminates those components which contribute the least to the variation in the dataset (Demuth and Beale, 1998).

The targets

As the output of the logistic transfer function is between 0 and 1, the data are generally scaled between 0.1-0.9 or 0.2 and 0.87. If the data are scaled to the extreme limits of the logistic transfer function, the size of the weight updates is extremely small and flat spots in training are likely to occur (Maier and Dandy, 2000). When the transfer function in the output layer is unbounded, as is the case for a linear transfer function, scaling is not strictly required. However, scaling to uniform ranges is still recommended (Masters, 1993). In this research only a linear transfer function was used in the output layer and the targets were scaled by scaling minimum and maximum values to [-1 1].

Input variables contribution methods

Neural networks are generally considered to be a ‘black box’. Gevrey *et al.* (2003) gives an overview of six different methods to unravel this ‘black box’. These methods can be mainly divided in three groups:

1) Changing the input variables and monitoring the effect on the output (PaD method, Profile and Perturb method). The techniques we applied in Chapter 3 belong to this group;
2) removing input variables and monitor the effect on the output (Stepwise method, Improved stepwise method);
3) reveal the relative importance of the various inputs on the connection weights (Weights method).

GEOSTATISTICS

The term geostatistics refers to the statistical analysis of phenomena which vary in a continuous and spatial way. Classical statistical methods, such as linear regression analysis and analysis of variance, are used intensively for geo-referenced data as well. However, there are limitations associated with these methods due to their underlying assumption of independency of observations. Therefore, there should be no spatial autocorrelation between the observations or between the residuals of the model. However, this condition is rarely met in geographical information. Such data is almost always correlated to some degree in relationship with the distance between observations (Van Meirvenne, 2007).

Geostatistics is related to interpolation methods, but extends beyond simple interpolation problems. It consists of a collection of numerical and mathematical techniques dealing with the characterisation of spatial phenomena. Geostatistical techniques model the uncertainty associated with spatial estimation (Van Meirvenne, 2007).

Regionalised variables

A regionalised variable Z is a variable which exists in a spatial continuous way, thus its value is a function of its spatial position $\mathbf{x}$. This variable has a unique value at every location $Z = f(\mathbf{x})$. Assume that Z was observed at a number of locations and a prediction is needed at the unvisited location $\mathbf{x}_0$. In classical statistics such a prediction is modelled as a combination of two components: a mean m plus a random error ε which represents the spatially independent fluctuations around the mean: $Z(\mathbf{x_0}) = m + \varepsilon$ with ε normally distributed and zero mean. In the case of a regionalised variable the hypothesis that the error term is spatially independent is unrealistic. Therefore the error term can be split into two: $\varepsilon'(\mathbf{x})$, an error term which represents the spatial structured component of the spatial variance and ε" a term which can be considered to be spatially uncorrelated:

$$Z(\mathbf{x}_0) = m + \varepsilon'(\mathbf{x}) + \varepsilon" \qquad \text{(Eq. A1.2)}$$

In geostatistics the aim is to characterise $\varepsilon'(\mathbf{x})$ as complete as possible in order to use it for improving the prediction of m (Van Meirvenne, 2007).

Variogram modelling

Variogram modelling is the key to geostatistical modelling. It is a summarizing function and has a strong descriptive and interpretative power about the structure of the spatial variability of a regionalised variable. The variogram is estimated by (Journel and Huijbregts, 1978):

$$\gamma(\mathbf{h}) = \frac{1}{2\mathrm{N}(\mathbf{h})} \sum_{\alpha=1}^{\mathrm{N}(\mathbf{h})} \{\mathrm{z}(\mathbf{x}_\alpha + \mathbf{h}) - \mathrm{z}(\mathbf{x}_\alpha)\}^2 \qquad \text{(Eq. A1.3)}$$

with $\gamma(\mathbf{h})$ the variogram for a distance vector (lag) $\mathbf{h}$ between observations $\mathrm{z}(\mathbf{x}_\alpha)$ and $\mathrm{z}(\mathbf{x}_\alpha + \mathbf{h})$ of the variable at the locations $\mathbf{x}_\alpha$ and $\mathbf{x}_\alpha + \mathbf{h}$, and with $N(\boldsymbol{h})$ the number of pairs separated by $\mathbf{h}$.

A plot of the calculated $\gamma(\mathbf{h})$ values versus the lag $\mathbf{h}$ is called an experimental variogram. To this experimental variogram a theoretical variogram model is fit yielding a continuous function of $\gamma(\mathbf{h})$ versus $\mathbf{h}$. The fitting of a variogram model is an interactive and iterative process (Webster and Oliver, 2007).

Four important characteristics can be derived (Fig. 4.2): the sill, the range, the nugget and the model type. The 'sill' represents the total variance of the variable and is the maximum of the variogram model. The 'range' is the maximal spatial extent of spatial correlation between observations of the variable. At lags larger than the range, the expected difference between observations is maximal (being the sill) and independent of the distance. At distances smaller than the range a dependency exists between the observations which increases as the observations are situated closer to each other. The extrapolation of the model to lags approaching zero is called the 'nugget variance' and represents sources of random noise such as sampling errors and variability at distances closer than the smallest sampling lag (Van Meirvenne, 2007). The smaller the nugget variance or pure random noise, the smaller ε" (i.e. the part of the error term which is spatially independent (Eq. A1.2), while the difference between sill and nugget relates to the spatially structured error term $\varepsilon'(\mathbf{x})$. Sometimes the nugget equals the sill. This situation is called a pure nugget effect. All variability is purely random and unstructured and has no spatial structure, thus no spatial autocorrelation is observable in the data. This may indicate that the sources of variability are too large and they mask the underlying spatial pattern or the smallest sampling distance is larger than the range.

The theoretical variogram can be composed of nested models or structures. Common models are the spherical model, the exponential model, the Gaussian model and the power model.

In isotropic situations, the regionalised variable shows the same range and variability in all directions. However, in an anisotropic situation the regionalised variable may display different ranges or higher variability in different directions. In that case, directional variograms should be derived (Wackernagel, 2003).

To obtain a reliable representation of the average structure of the spatial variability, it is necessary to have sufficient observation points. About 100 observations are a minimum in isotropic situations (Webster and Oliver, 2007). This represents some limitation to the applicability of geostatistical analysis. The minimum number of pairs $N(\mathbf{h})$ that is required to obtain a reliable estimation of $\gamma(\mathbf{h})$ is 30 to 50 per $\mathbf{h}$ class. For each $\mathbf{h}$ class one point in the experimental variogram is derived.

Kriging

Kriging is a collection of generalised linear regression techniques for minimizing an estimation variance defined from a prior covariance model (Deutsch and Journel, 1992).

A local interpolation can be written as a weighted linear combination of measurements points located within a neighbourhood around $\mathbf{x}_0$:

$Z^*(\mathbf{x}_0) = \sum_{\alpha=1}^{n(\mathbf{x}_0)} \lambda_\alpha Z(\mathbf{x}_\alpha)$ (Eq. A1.4)

with λ_α being the weight attributed to the observation $Z(\boldsymbol{x}_\alpha)$ to estimate the value of $Z^*(\mathbf{x}_0)$.

In geostatistics the focus is on the stochastic part of a regionalised variable, the part which is modelled by the variogram: $\varepsilon'(\mathbf{x}) = Z(\mathbf{x}) - m$. Hence the equation is modified to:

$\varepsilon'(\mathbf{x}_0) = \sum_{\alpha=1}^{n(\mathbf{x}_0)} \lambda_\alpha \varepsilon'(\mathbf{x}_\alpha)$ (Eq. A1.5)

Yielding the general kriging equation:

$Z^{*}(\mathbf{x}_0) - m(\mathbf{x}_0) = \sum_{\alpha=1}^{n(\mathbf{x}_0)} \lambda_\alpha \, [Z(\mathbf{x}_\alpha) - m(\mathbf{x}_\alpha)]$ (Eq. A1.6)

Based on different assumptions and objectives, a number of variants of kriging are available. Here ordinary kriging, simple kriging and regression kriging are handled shortly (*modified from* Van Meirvenne, 2007).

Around $\mathbf{x}_0$ an interpolation window, called search neighbourhood , must be specified. Usually this neighbourhood has the shape of a circle in an isotropic situation or an ellipse in anisotropic situations. The search radius is chosen in respect to the range, since observations taken at a larger distance than the range are considered to be uncorrelated with $\mathbf{x}_0$ and will not attribute to its value.

Ordinary kriging (OK)

OK applies to the situation where $m(\mathbf{x})$ is unknown. An underlying assumption for OK is that the mean, although unknown, is locally stationary. Thus, it has a constant value inside the interpolation window.

The algorithm can be written as:

$Z^{*}_{OK}(\mathbf{x}_0) = \sum_{\alpha=1}^{n(\mathbf{x}_0)} \lambda_\alpha \, Z(\mathbf{x}_\alpha)$ with $\sum_{\alpha=1}^{n(\mathbf{x}_0)} \lambda_\alpha = 1$ (Eq. A1.7)

with $n(\mathbf{x}_0)$ being the total number of observations in the interpolation window around $\mathbf{x}_0$ and λ_α being the interpolation weight attributed to the observation $Z(\mathbf{x}_\alpha)$. The weights λ_α are obtained by solving a set of equations involving knowledge of the variogram and they are chosen in such a way that the prediction error variance is minimised (Webster and Oliver, 2007). Observations closer to $\mathbf{x}_0$ get a higher weight than observations further away.

Simple kriging (SK)

Simple kriging differs from OK in the way $m(\mathbf{x})$ is handled. In SK $m(\mathbf{x})$ is supposed to be known and globally stationary, thus the estimated value of $Z(\mathbf{x})$ equals m. Since m is known, SK works on the residuals R: $R(\mathbf{x}_\alpha) = Z(\mathbf{x}_\alpha) - m$

$R^{*}_{SK}(\mathbf{x}_0) = \sum_{\alpha=1}^{n(x_0)} \lambda_\alpha \, R(\mathbf{x}_\alpha)$ (Eq. A1.8)

Thus SK starts with subtracting the global mean from all the observations. Then, the residuals are interpolated using the kriging system. And finally the global mean is added back to the estimations of the kriging system. OK is more often used then SK because the global mean m is rarely known (Van Meirvenne, 2007). However, in case of regression kriging (see 2.3.3) the mean is deduced from external secondary information and the mean of the residuals can be considered to equal zero (Hengl *et al.*, 2007).

Regression kriging (RK)

An alternative to ordinary kriging is regression kriging. Predictions by RK involve fundamentally two steps: first the relationship between the dependent variable and the independent environmental variables at the sampling locations are modelled by a linear regression and this model is then applied to the unseen locations using the environmental variables at this location. Secondly, the residuals of this linear model are subjected to simple kriging with an expected mean of zero (Deutsch and Journel, 1992).

First, the linear model can be written as a linear combination of the environmental variables:

$$\hat{Z}(\mathbf{x}_0) = \sum_{k=0}^{p} \hat{\beta}_k \, q_k(\mathbf{x}_0) \qquad \text{with } q_0(\boldsymbol{x}_0) \equiv 1 \qquad \text{(Eq. A1.9)}$$

where $q_k(\mathbf{x}_0)$ is the value of the independent variable k at the location $\mathbf{x}_0$, $\hat{\beta}_k$ is the estimated regression coefficient of the variable k and p is the number of dependent variables.

Secondly, simple kriging with expected mean 0 is used to fit the residuals of the linear model. This results in (Hengl *et al.*, 2007):

$$\hat{Z}(\mathbf{x}_0) = \sum_{k=0}^{p} \hat{\beta}_k \, q_k(\mathbf{x}_0) + \sum_{\alpha=1}^{n(s_0)} \lambda_\alpha \, e(\mathbf{x}_\alpha) \qquad \text{with } q_0(\boldsymbol{x}_0) \equiv 1 \qquad \text{(Eq. A1.10)}$$

where $e(\mathbf{x}_\alpha)$ is the residual of the linear model at location $\mathbf{x}_\alpha$. The first term describes the linear regression, and the second term describes the simple kriging algorithm of the residuals of the linear regression.

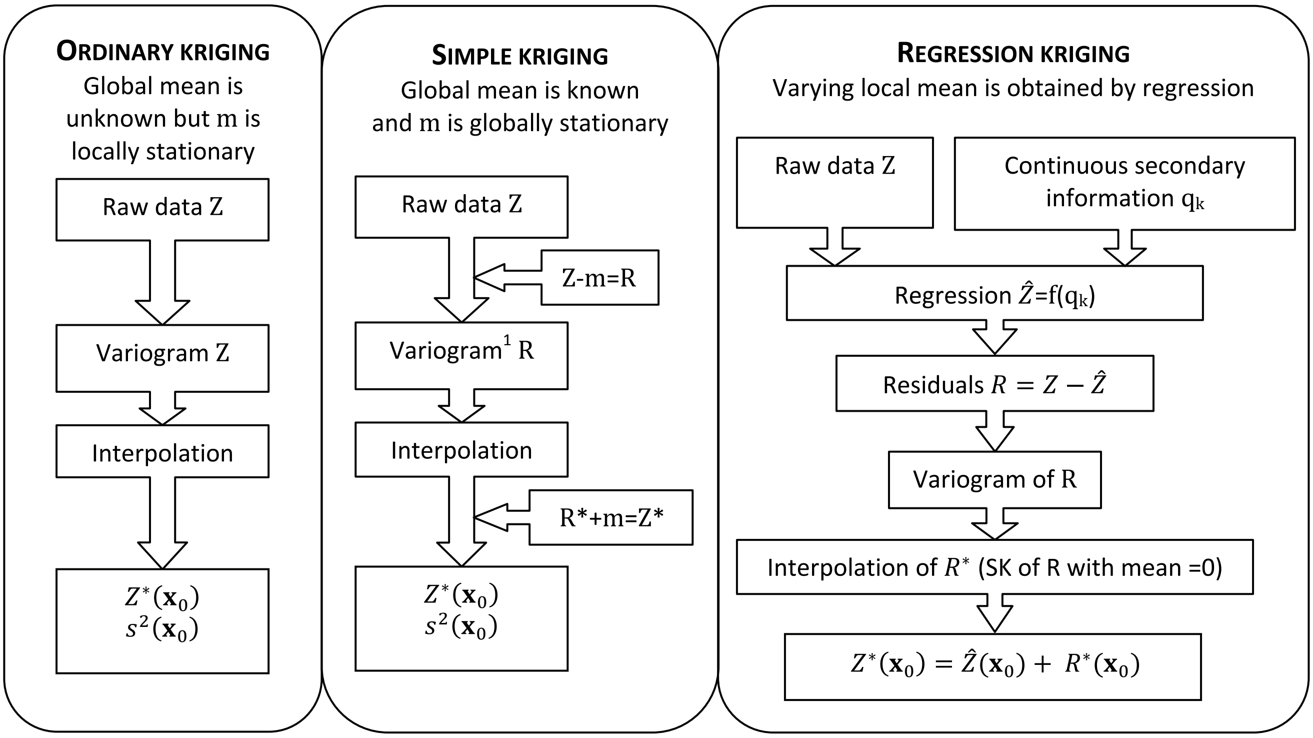
ORDINARY KRIGING
Global mean is unknown but m is locally stationary
Raw data Z
Variogram Z
Interpolation
$Z^*(\mathbf{x}_0)$
$s^2(\mathbf{x}_0)$
SIMPLE KRIGING
Global mean is known and m is globally stationary
Raw data Z
Z-m=R
Variogram[1] R
Interpolation
R*+m=Z*
$Z^*(\mathbf{x}_0)$
$s^2(\mathbf{x}_0)$
REGRESSION KRIGING
Varying local mean is obtained by regression
Raw data Z
Continuous secondary information q_k
Regression $\hat{Z}$=f(q_k)
Residuals $R = Z - \hat{Z}$
Variogram of R
Interpolation of R^* (SK of R with mean =0)
$Z^*(\mathbf{x}_0) = \hat{Z}(\mathbf{x}_0) + R^*(\mathbf{x}_0)$

*Fig. A1.8. Schematic representation of the three kriging techniques: ordinary kriging, simple kriging and regression kriging (*modified from *Van Meirvenne, 2007).*

[3] In case of SK the variogram is converted into the covariance of the residuals using the relationship $C(\boldsymbol{h}) = C(0) - \gamma(\boldsymbol{h})$ with $C(0)$ the covariance at lag 0.

Comparison of the three kriging techniques

Fig. A1.8 shows a schematic overview of the three kriging techniques. As mentioned before, the main difference between these three kriging techniques lies within the assumption concerning the mean of the regionalised variable. In case of OK and SK both a map of the regionalised variable and a map of the variance of this variable can be deduced. In case of regression kriging only a map of the regionalised variable can be inferred, an estimation of the local variance is not possible.

MAXIMUM ENTROPY MODELLING

Maximum entropy modelling is based on the second law of thermodynamics, stating that in systems without outside influences, processes move towards maximum entropy. This theory can be applied in habitat suitability modelling (HSM): in the absence of influences other than those included as constraints in the model, the geographic distribution of a species will evolve to a maximum entropy distribution (Phillips *et al.*, 2006), thus the target is to find the probability distribution of maximum entropy (i.e., the distribution that is most spread out, or closest to uniform), but subject to a set of constraints.

Maxent is a software program for maximum entropy modelling of species' geographic distributions. When Maxent is applied to presence-only species distribution modelling, the pixels of the study area are the space where the Maxent probability distribution is defined. Pixels with known species occurrence records constitute the sample points (Phillips *et al.*, 2006).

Data

Species data

Common statistical methods generally use presence/absence data, however in most cases data can only be considered as presence-only data, e.g. data from inconspicuous species such as nematodes, species with patchy distributions, in case of invasive species which have not yet occupied their potential niche or data from natural history museums (Phillips *et al.*, 2004). In such cases techniques working with presence-only data, such as Maxent, can be very useful. Earlier research pointed out that Maxent is a reliable presence-only modelling technique and it performs well compared to other presence-only modelling techniques (Hernandez *et al.*, 2006; Ortega-Huerta and Peterson, 2008) and it may compete with or even outcompete presence/absence modelling techniques (Elith *et al.*, 2006; Wisz *et al.*, 2008).

Environmental data

The environmental layers should be in raster format all pertaining to the same geographic area (i.e. the study area) which has been partitioned into a grid of pixels with raster cells having the same resolution. The environmental variables or functions thereof are called the ‘features’.

Six feature types are used in Maxent:

1) A continuous variable *f* is a ‘linear feature’ (Fig. A1.9).
2) The square of a continuous variable *f* is a ‘quadratic feature’. It models the species’ tolerance for variation from its optimal conditions (Fig. A1.9) (Phillips *et al.*, 2006).
3) The product of two continuous environmental variables f_i and f_j is a ‘product feature’. Product features incorporate interactions between predictor variables (Phillips *et al.*, 2006).
4) For a continuous environmental variable *f*, a ‘threshold feature’ is equal to 1 when *f* is above a given threshold, and 0 otherwise (Phillips *et al.*, 2006).
5) The forward hinge feature is 0 if $f(x) \leq h$ and then increases linearly to 1 at the maximum value of $f(x)$. In a similar way, a reverse hinge feature is defined , which is 1 at the minimum value of f. It drops linearly to 0 at $f(x) = h$ and remains 0. Forward and reverse hinge features are collectively referred to as hinge features (Fig. A1.9) (Phillips and Dudík , 2008).
6) Category indicator features are derived from categorical variables. Specifically, if a categorical variable has k categories, it is used to derive k categories of indicator features. For each of the k categories, the corresponding category indicator equals 1 if the variable has the corresponding value and 0 if it has any of the remaining $k - 1$ values (Phillips and Dudík , 2008). The category indicator was not used in this research, since all our environmental variables were continuous variables.

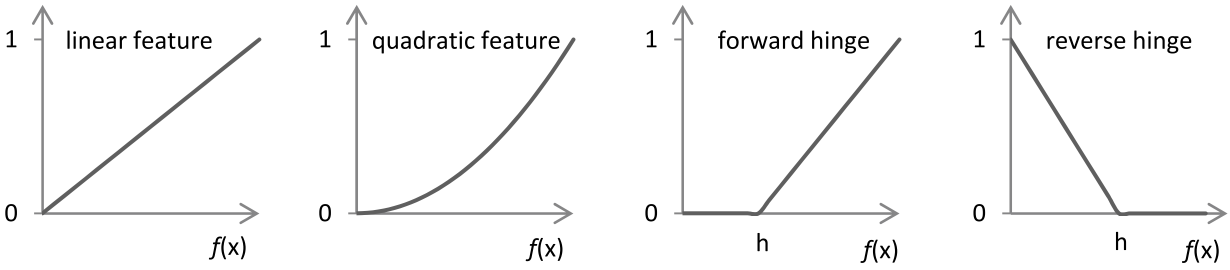

Fig. A1.9. Schematic representation of the features: linear, quadratic, forward and reverse hinge.

Maxent modelling

The basic idea of maximum entropy modelling

A space X represents the geographic region of interest. Typically, X is a set of discrete grid cells. Also a set of points $x_1, \ldots, x_m$ in X are given, each representing a locality where the species has been observed and recorded. In addition, a set of environmental variables defined on X is known. Based on this information, the goal is to estimate the range of the given species (Phillips *et al.*, 2004).

The unknown probability distribution is denoted π over X. The distribution π assigns a non-negative probability $\pi(x)$ to each point x of the area, and these probabilities sum to 1. The approximation of π is also a probability distribution, and is denoted as $\hat{\pi}$. The entropy of the set of probabilities $\hat{\pi}(x)$ is defined as (Phillips *et al.*, 2004):

$$H(\hat{\pi}) = -\sum_{x \in X} \hat{\pi}\,(x).\log_{\mathrm{e}}\big(\hat{\pi}(x)\big) \qquad \text{(Eq. A1.11)}$$

The entropy is nonnegative and is at most the natural log of the number of elements in X. Entropy is a fundamental concept in information theory. The quantity H has a number of interesting properties:

- H = 0 if and only if all the $\hat{\pi}(x)$ but one are zero, this one having the value unity. Thus only when the outcome is certain, there is no entropy and H vanishes. Otherwise H is positive.
- H reaches the maximum when all $\hat{\pi}(x)$ are equal. This is the most uncertain situation (Shannon, 1948). This is the case when 'a species' shows maximum entropy and has the same likelihood across the whole region.
- Entropy is thus a measure of how much 'choice' is involved in the selection of the event or of how uncertain we are of the outcome (Shannon, 1948). Thus a distribution with higher entropy involves more choices (i.e., it is less constrained) (Phillips *et al.*, 2006). Therefore, the maximum entropy principle can be interpreted as saying that no unfounded constraints should be placed on $\hat{\pi}$, or alternatively, it agrees with everything that is known, but carefully avoids assuming anything that is not known (Jaynes, 1990). At the foundation of the Maxent approach lays the premise that the distribution (or a thresholded version of it) coincides with the biologists' concept of the species' potential distribution (Phillips *et al.*, 2006). However, it ignores the fact that some localities are more likely to have been visited than others. Preferential sampling may bias the model towards areas and environmental conditions that have been better sampled (Phillips *et al.*, 2004).

The problem becomes one of density estimation: given $x_1, \ldots, x_m$ chosen independently from some unknown distribution π, a distribution $\hat{\pi}$ that approximates π has to be constructed. The maximum entropy principle consists of defining the constraints on the unknown probability distribution π in the following way. A set of n environmental variables

$f_1, \ldots, f_n$ on X, the so called 'features', are known for the entire area. The information known about π is characterised by the expectations (averages) of the features under π. Here, each feature f_j assigns a real value $f_j(x)$ to each point x in X. The expectation of the feature f_j under π is defined as $\sum_{x \in X} \pi(x). f_j(x)$ and denoted by $\pi[f_j]$ (Phillips *et al.*, 2006).

The feature expectations $\pi[f_j]$ can be approximated using a set of sample points $x_1, \ldots, x_m$ drawn independently from X (with replacement) according to the probability distribution π. The empirical average of f_j is $\frac{1}{m}\sum_{i=1}^{m} f_j(x_i)$, which can be written as $\tilde{\pi}[f_j]$ where $\tilde{\pi}$ is the uniform distribution on the sample points, and is an estimation of $\pi[f_j]$. The probability distribution $\hat{\pi}$ of maximum entropy is subject to the constraint that each feature f_j has the same mean under $\hat{\pi}$ as observed empirically, i.e. $\hat{\pi}[f_j] = \tilde{\pi}[f_j]$ (Eq. A1.12)

for each feature f_j. It can be shown that this characterisation uniquely determines $\hat{\pi}$ (Phillips *et al.*, 2006).

Consider all probability distributions of the form:

$$q_\lambda(x) = \frac{e^{\lambda . f(x)}}{Z_\lambda} \qquad \text{(Eq. A1.13)}$$

where λ is a vector of n real-valued coefficients or feature weights, f denotes the vector of all n features, and Z_λ is a normalizing constant that ensures that q_λ sums to 1. Such distributions are known as Gibbs distributions. It can be shown that the Maxent probability distribution $\hat{\pi}$ is exactly equal to the Gibbs probability distribution q_λ that maximises the likelihood (i.e., the probability) of the m sample points. Equivalently, it minimises the negative log likelihood of the sample points $\tilde{\pi}[-\ln(q_\lambda)]$ which can be written as

$$\log_e Z_\lambda - \frac{1}{m}\sum_{i=1}^{m} \lambda . f(x_i) \qquad \text{(Eq. A1.14)}$$

and is named the 'log loss' (Phillips *et al.*, 2006).

Maxent can severely overfit training data when the constraints on the output distribution are based on feature expectations as described above, especially if there are a large number of features (Dudík *et al.*, 2004). The problem derives from the fact that the empirical feature means will typically not equal the true means; they will only approximate them. Therefore the means under $\hat{\pi}$ should only be restricted to be close to their empirical values. One way this can be done is to relax the constraint in Eq. A1.12, replacing it with

$$|\hat{\pi}[f_j] - \tilde{\pi}[f_j]| \leq \beta_j \qquad \text{(Eq. A1.15)}$$

for each feature f_j for some constants β_j resulting in a form of ℓ_1-regularisation. The Maxent distribution can now be shown to be the Gibbs distribution that minimises

$$\tilde{\pi}[-\log_e n(q_\lambda)] + \sum_{j=1}^{n} \beta_j |\lambda_j| \qquad \text{(Eq. A1.16)}$$

where the first term is the log loss, while the second term penalises the use of large values for the weights λ_j. Regularisation forces Maxent to focus on the most important features, and ℓ_1-regularisation tends to produce models with few nonzero λ_j values. Such models are less likely to overfit, because they have fewer parameters. As a general rule, the simplest explanation of a phenomenon is usually best (the principle of parsimony, Occam's Razor) (Phillips *et al.*, 2006).

The Maxent probability distribution is found by starting from the uniform probability distribution, for which $\boldsymbol{\lambda} = (0, \ldots, 0)$, then repeatedly make adjustments to one or more of the weights in such a way that the regularised log loss decreases. It can be shown that the regularised log loss is a convex function of the weights, so no local minima exist and the weights can be adjusted in a way that guarantees convergence to the global minimum (Phillips *et al.*, 2006). Practically, the 'regularisation multiplier' has a default value of one in the software. This value can be changed but a smaller value than the default of 1 will result in a more localised output distribution that is a closer fit to the given presence records; however this can result in overfitting. A larger regularisation multiplier will give more spread out, thus less localised, predictions (Phillips, 2010).

Quality measures of habitat suitability models

A range of techniques for measuring error in presence/absence models exists. Most of these accuracy measures are calculated from a confusion matrix (Fig. A1.10).

		actual	
predicted		+	−
	+	a	b
	−	c	d

Fig. A1.10. A confusion matrix

In the confusion matrix four parameters are available:

1. a represents the number of true positives;
2. b represents the number of false positives. This error is related to Type I error (the error of rejecting a null hypothesis when it is actually true);
3. c represents the number of false negatives. This error is related to Type II error (the error of failing to reject a null hypothesis when it should be rejected);
4. d represents the number of true negatives.

Several parameters can be calculated from this matrix. In Table A1.1 the most relevant ones are shown. These parameters serve different purposes. If the aim is to assess the effectiveness of the model a parameter such as Kappa, which assesses improvement over chance, is appropriate. This can be important because in case of very high or very low prevalence, the overall accuracy may be high. For instance, if the species is found in 95% of the cases, a model which predicts the species over the complete area will classify 95% of the data correctly. Sensitivity measures the proportion of actual positives which are correctly identified, while specificity measures the proportion of negatives which are correctly identified. An optimal prediction would achieve 100% sensitivity and 100% specificity.

However, in reality there is usually a trade-off between the measures. The use of receiver operating characteristic (ROC) plots visualises this trade-off (Fig. A1.11) (Fielding and Bell, 1997).

Parameter	Formula
Prevalence	$\frac{a+c}{N}$
Overall diagnostic power	$\frac{b+d}{N}$
Sensitivity	$\frac{a}{a+c}$
Specificity	$\frac{d}{b+d}$
Kappa	$\frac{\mathrm{a+d}-\frac{\mathrm{(a+c)(a+b)+(b+d)(c+d)}}{\mathrm{N}}}{N-\frac{(a+c)(a+b)+(b+d)(c+d)}{N}}$

*Table A1.1. parameters of classification accuracy derived from a confusion matrix (*from *Fielding and Bell, 1997). For description of the parameters a, b, c, d see Fig. A1.10. N is the total number of samples and equals $a+b+c+d$.*

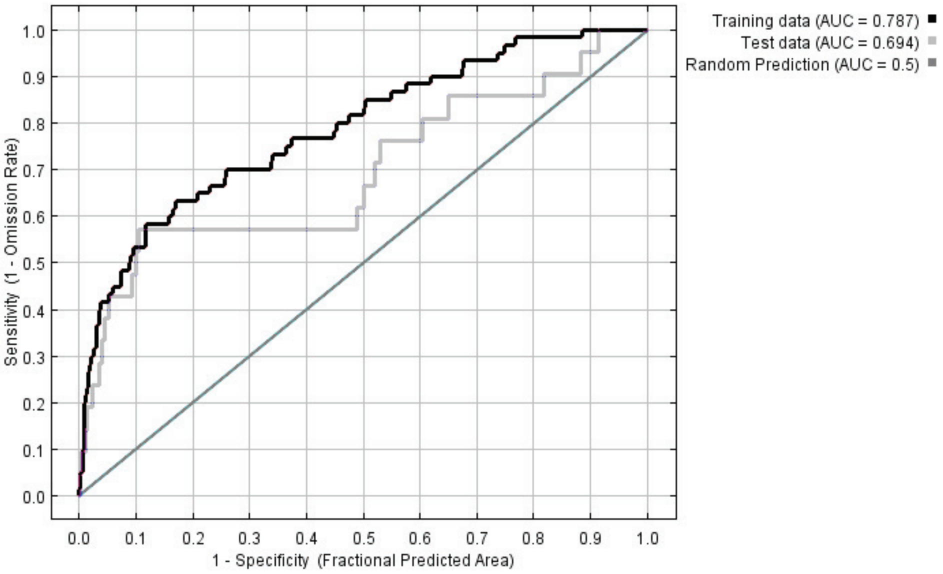

Fig. A1.11. ROC plot for test and training set. The straight line indicates a random prediction.

All of the measures described in Table A1.1 depend on the values assigned to *a*, *b*, *c* and *d* in the confusion matrix. These values are obtained by applying a threshold to the continuous output of the model. One problem with the threshold dependent measures is their failure to use all of the information provided by the model, although dichotomous classifications can be convenient for decision making (Fielding and Bell, 1997). A threshold independent measure is a ROC plot. It is obtained by plotting all sensitivity values (true positive fraction) on the *y* axis against their equivalent (1 - specificity) values (false positive fraction) for all available thresholds on the *x* axis, as in the example shown in Fig. A1.11. The area under the ROC function (AUC) is an important index because it provides a single parameter of overall accuracy that is not dependent of a particular threshold. The value of the AUC is between 0.5 and 1.0. If the value is 0.5, the model is a random model, while a score of 1.0 indicates a perfect model. A value of 0.8 for the AUC means that for 80% of the time a random selection from the positive group will have a score greater than a random selection from the negative class (Fielding and Bell, 1997). The AUC has thus an intuitive interpretation, namely the probability that a random positive instance and a random negative instance are correctly ordered by the model.

Maxent uses presence-only data and it would appear that ROC curves are inapplicable, since there are no absences, and thus it seems impossible to calculate specificity. However, this problem can be circumvented by considering a different classification problem, namely, distinguishing presence from random, rather than presence from absence. More formally, for each pixel x in the study area, a negative instance x_{random} is defined. Similarly, for each pixel x that is included in the species' true geographic distribution, a positive instance $x_{presence}$is defined. The species distribution model can then make predictions for the pixels corresponding to these instances, without seeing the labels random or presence. Thus, predictions are made for both a sample of positive instances ($x_{presence}$) and a sample of negative instances (x_{random} which are background pixels chosen uniformly at random). Together these are sufficient to define an ROC curve. This process can be interpreted as using pseudo-absences instead of real absences in the ROC analysis. For each ROC analysis, all the test localities for the species are used as presences, and a sample of 10 000 pixels drawn randomly from the study region as random instances (Phillips *et al.*, 2006), called the 'fractional predicted area' (the fraction of the total study area predicted present). AUC values tend to be higher for species with narrow ranges, relative to the study area. This does not necessarily mean that the models are better; instead this behaviour is an artefact of the AUC statistic (Phillips, 2010).

The above treatment differs from the use of ROC analysis on presence/absence data in one important respect: with presence-only data, the maximum achievable AUC is less than 1 (Wiley *et al.*, 2003). If the species' distribution covers a fraction a of the pixels, then the maximum achievable AUC is exactly $1 - a/2$. Unfortunately, the value of a is most of the time not known, so it is impossible to say how close to optimal a given AUC value is. Random prediction still corresponds to an AUC of 0.5.

Feature contribution

While the Maxent model is trained, each step of the Maxent algorithm increases the gain of the model by modifying the coefficient for each feature. The program registers for each environmental variable(s) the increase in gain. At the end of the training process this is converted to percentages. These contribution values are heuristically determined: thus they depend on the path that the Maxent algorithm followed to find the optimal solution, and a different algorithm could get to the same solution through a different path, thus resulting in different contribution values. In addition, when the environmental variables are highly correlated, the percent contributions should be interpreted with caution. To get alternate estimates of which variables are most important in the model, a jackknife test can be run. Several models are created: each variable is excluded in turn, and a model is created with the remaining variables and alternatively, a model is created using each variable in isolation. The contribution of each variable to the original model can be monitored in this way (Phillips, 2010).

Relationships with other modelling approaches

The Maxent modelling technique is an 'unconditional' maximum entropy model, it uses only presence data. 'Conditional' models require both presence and absence data. Maxent has strong similarities to some existing methods for modelling species distributions, in particular, generalised linear models (GLMs) and generalised additive models (GAMs) (Phillips *et al.*, 2006). When GLM/GAMs are used to model probability of occurrence, absence data are required. When applied to presence-only data, background pixels must be used instead of true absences (Ferrier and Watson, 1996). However, the interpretation of the result is less clear-cut. It must be interpreted as a relative index of environmental suitability. In contrast, Maxent generates a probability distribution over the pixels in the study region, and in no sense are pixels without species records interpreted as absences. In addition, Maxent is a generative approach, whereas GLM/GAMs are discriminative. The latter approach is generally preferred. However, generative methods may give better predictions when the amount of training data is small (Ng and Jordan, 2001).