Chapter 7
Spatial Analysis of Ecological Data

7.1 Objectives

Spatial analysis of ecological data is a huge field that could fill several books by itself. To learn about general approaches in spatial analysis with **R**, readers may consult the recent book by Bivand et al. (2008). The present chapter has a more restricted scope. After a short general introduction, it deals with several methods that were specifically developed for the analysis of scale-dependent structures of ecological data; these methods can, of course, be applied to other domains. These methods are based on sets of variables describing spatial structures in various ways, derived from the coordinates of the sites or from the neighbourhood relationships among sites. These variables are used to model the spatial structures of ecological data by means of multiple regression or canonical ordination, and to identify significant spatial structures at all spatial scales that can be perceived by the sampling design. As you will see, the whole analytical process uses many of the techniques covered in the previous chapters.

Practically, you will:

- Learn how to compute spatial correlation measures and draw spatial correlograms
- Learn how to construct spatial descriptors derived from site coordinates and from links between sites
- Identify, test and interpret scale-dependent spatial structures
- Combine spatial analysis and variation partitioning
- Assess spatial structures in canonical ordinations by computing variograms of explained and residual ordination scores

D. Borcard et al., *Numerical Ecology with R*, Use R,
DOI 10.1007/978-1-4419-7976-6_7, © Springer Science+Business Media, LLC 2011

7.2 Spatial Structures and Spatial Analysis: A Short Overview

7.2.1 Introduction

As mentioned in Chap. 6, spatial structures play a very important role in the analysis of ecological data. Living communities are spatially structured at many scales, and these structures are the result of several classes of processes. On the other hand, beta diversity is the spatial variation in community composition; so, a study of the factors that can explain the spatial variation of community composition is in every respect an analysis of beta diversity. The environmental control model advocates that external forces (climatic, physical, chemical) control living communities. If these factors are spatially structured, their patterns will reflect on the living communities (examples: patches of desert where the soil is humid enough to support vegetation; gradient of successive communities through an intertidal zone). The biotic control model predicts that intra- and interspecific interactions within communities (examples: social groups of animals; top-down or bottom-up processes), as well as neutral processes such as ecological drift and limited dispersal, may result in spatial patterns which are the cause of spatial autocorrelation in the strict sense. Historical events (e.g. past disturbances like fire or human settlements) may have structured the environment in a way that still influences present-day communities.

In all, ecological data are a combination of many structures, spatial or not:

- The overall mean of each response variable; if the whole sampling area is under the influence of an all-encompassing process that changes the mean in a gradient across the area, then a *trend* is present. The trend may be due to a process operating at a scale larger than the sampling area.
- Spatial structures at regional scales: ecological processes of various kinds (biotic or abiotic) influence the data at scales finer than the overall sampling area, producing identifiable spatial patterns.
- Local deterministic structures with no recognizable spatial component because the sampling design is not fine enough to identify such fine-scale patches.
- Random noise (error): this is the residual (stochastic) component of the variation. It can be attributed to local effects operating independently at each sampling site.

One of the aims of spatial analysis is to discriminate between these sources of variation and model the relevant ones separately.

7.2.2 Induced Spatial Dependence and Spatial Autocorrelation

An important distinction must be made here. As we wrote above, a spatial structure in a response matrix **Y** can result from two main origins: either from the forcing of external (environmental) factors that are themselves spatially structured, or as the result of processes internal to the community itself. In the first case, one speaks of *induced spatial dependence*, in the second case of *spatial autocorrelation*.

For value y_j of a response variable **y** observed at site *j*, the model for *induced spatial dependence* is the following:

$$y_j = \mu_y + f(\mathbf{X}_j) + \varepsilon_j \tag{7.1}$$

where μ_y is the overall mean of variable **y**, **X** is a set of explanatory variables, and ε_j is an error term that varies randomly from location to location. The additional term [$f(\mathbf{X}_j)$] states that y_j is influenced by external processes represented in the model by explanatory variables. The spatial structure of these variables is reflected in *y*. When they form a gradient, they represent what Legendre (1993) called "true gradients", that is, gradient-like deterministic structures generated by external forces, whose error terms are not autocorrelated.

The model for *spatial autocorrelation* is:

$$y_j = \mu_y + \sum f(y_i - \mu_y) + \varepsilon_j \tag{7.2}$$

This equation states that y_j is influenced by the values of **y** at the surrounding sites *i*. This influence is modelled by a weighted sum of the (centred) values y_i at these sites. The biological context dictates the radius of the zone influencing a given point, as well as the weight to be given to the neighbouring points. These weights are generally dependent on the distance. The spatial interpolation method called *kriging* (Isaaks and Srivastava 1989; Bivand et al. 2008) is based on this model. Kriging is a family of interpolation methods that is not discussed further in this book. Kriging functions are available in package **geoR**.

Spatial autocorrelation may mimic gradients if the underlying process has a range of influence larger than the sampling area. Legendre (1993) called the resulting structures "false gradients". There is no statistical way to distinguish false from true gradients. One must rely upon biological hypotheses: in some cases, one has a strong hypothesis about the processes generating spatial structures, and therefore whether these processes may have produced autocorrelation in the data. In other cases, an opinion can be formed by comparing the processes detected at the scale

of the study area with those that are likely to occur at the scale of the (larger) target population (Legendre and Legendre 1998).

Spatial correlation measures the fact that near points in space have either more similar (positive correlation) or more dissimilar values (negative correlation) than randomly selected pairs. This phenomenon, which is generated either by true autocorrelation (7.2) or by spatial structures resulting from spatial dependence (7.1), has noxious effects on statistical tests. In spatially correlated data, values at any given site can be predicted, at least partially, from the values at other sites, if the researcher knows the biological process and the locations of the sites. This means that the values are not stochastically independent of one another. The assumption of independence of errors is violated in such cases. In other words, each new observation does not bring with it a full degree of freedom. While the fraction is difficult to determine, the fact is that the number of degrees of freedom used for a parametric test is often overestimated, thereby biasing the test on the "liberal" side: the null hypothesis is rejected too often. Numerical simulations have shown, however, that this statistical problem only occurs when both the response (e.g. species) and the explanatory variables (e.g. environmental) are spatially correlated (Legendre et al. 2002).

7.2.3 Spatial Scale

The term *scale* is used in many senses across different disciplines. It encompasses several properties of sampling designs and spatial analysis.

A sampling design has three characteristics pertaining to spatial scale (Legendre and Legendre 1998, Section 13.0):

- *Grain size*: size of the sampling units (diameter, surface or volume depending on the study).
- *Sampling interval*, sometimes called *lag*: average distance between neighbouring sampling units.
- *Extent* (sometimes called *range*): total length of the transect, surface area or volume (e.g. air, water) included in the study.

These three properties of a sampling design have an influence on the type and size of the spatial structures that can be identified and measured. (1) Sampling units *integrate* the structures occurring in them: one cannot identify structures of sizes equal to or smaller than the grain of the study. (2) The sampling interval determines the size of the finest spatial structures that can be identified (by *differentiation* among sampling units). (3) The extent of the study area sets an upper limit to the size of the measurable patterns. It is therefore essential to match each of these three

elements to the hypotheses to be tested and to the characteristics of the system under study (Dungan et al. 2002).

The ecological context of the study dictates the optimal grain size, sampling interval and extent. The optimal grain size (size of the sampling units) should match the size of unit entities of the study (e.g. objects like individual plants or animals, patches of vegetation, lakes, or areas affected by fine-scale processes). The average distance between unit objects or unit processes should be matched by the sampling interval. The extent should encompass the range of the broadest processes targeted by the study. These recommendations are detailed in Dungan et al. (2002).

Note that the expressions "large scale" and "small scale" are somewhat ambiguous because their meanings in ecology and cartography are opposite. In ecology "small scale" refers to the fine structures and "large scale" to the broadest structures, contrary to cartography where a large-scale map (e.g. 1:25000) is more detailed than a small-scale map (e.g. 1:1000000). Therefore, we advocate the use of "broad scale" (phenomena with large grains, large extents) and "fine scale" in ecology (Wiens 1989). Although these terms are not strict antonyms, we feel that they are less ambiguous than "large" and "small scale".

Finally, ecological processes occur at a variety of scales, resulting in complex, multiscale patterns. Therefore, identifying the scale(s) of the patterns and relating them to the appropriate processes are goals of paramount importance in modern ecology. To reach them, the researcher must rely on appropriate sampling designs and powerful analytical methods. The approaches presented in this chapter have been devised for the latter purpose.

7.2.4 Spatial Heterogeneity

A process or a pattern that varies across an area is said to be *spatially heterogeneous*. Many methods of spatial analysis are devoted to the measurement of the magnitude and extent of this heterogeneity and testing for the presence of spatial correlation (in other words, spatial structures of any kind). The latter may be done either to support the hypothesis that no spatial correlation (in the broad sense) is present in the data (if the researcher has classical parametric tests in mind) or, on the contrary, to show that correlation is present and use that information in conceptual or statistical models (Legendre and Legendre 1998).

Spatial heterogeneity in relation to inter-site distance is most often studied by means of *structure functions*. Examples of these are correlograms, variograms and periodograms. While it is not the purpose of this book to discuss these various functions, it is useful to devote a section to correlograms, since the main underlying measures of spatial correlation are used later in this chapter.

7.2.5 Spatial Correlation or Autocorrelation Functions and Spatial Correlograms

The two main statistics used to measure spatial correlation of univariate quantitative variables are Moran's *I* (Moran 1950) and Geary's *c* (Geary 1954). The first is constructed in much the same way as the Pearson correlation coefficient:

$$I(d) = \frac{\frac{1}{W}\sum_{h=1}^{n}\sum_{i=1}^{n} w_{hi}(y_h - \overline{y})(y_i - \overline{y})}{\frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y})^2} \quad \text{for} \quad h \neq i \tag{7.3}$$

The expected value of Moran's *I* for no spatial correlation is

$$E(I) = \frac{-1}{n-1} \tag{7.4}$$

Values below $E(I)$ indicate negative spatial correlation, and values above $E(I)$ indicate positive correlation. $E(I)$ is close to 0 when n (the total number of observations) is large.

Geary's *c* is more akin to a distance measure:

$$c(d) = \frac{\frac{1}{2W}\sum_{h=1}^{n}\sum_{i=1}^{n} w_{hi}(y_h - y_i)^2}{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \overline{y})^2} \quad \text{for} \quad h \neq i \tag{7.5}$$

The expected value of Geary's *c* for no spatial correlation is $E(c) = 1$. Values below 1 indicate *positive* spatial correlation, and values above 1 indicate *negative* correlation.

y_h and y_i are the values of variable y at pairs of sites h and i. To compute spatial correlation coefficients, one first constructs a matrix of geographical distances among sites. These distances are then converted to classes d. Both formulas show the computation of the index value for a class of inter-site distance d. The weights w_{hi} have value $w_{hi} = 1$ for pairs of sites belonging to distance class d, and $w_{hi} = 0$ otherwise. W is the number of pairs of points used to compute the coefficient for the distance class considered, i.e., the sum of the w_{hi} weights for that class.

A *correlogram* is a plot of the spatial correlation values against the distance classes. Combined with statistical tests, a correlogram allows a quick assessment of the type and range of the spatial correlation structure of a variable. A typical case is spatial correlation that is positive at short distances, decreases to negative values, and levels out to a point where it becomes non-significant. The corresponding distance class sets the distance beyond which a pair of values can be considered as spatially independent. It is important to note that spatial correlograms display any kind of spatial correlation, generated by (7.1) (induced spatial dependence) or (7.2) (spatial autocorrelation); so the name "spatial autocorrelogram" which is often given to these plots is somewhat misleading.

Univariate spatial correlograms can be computed using the function **`sp.correlogram()`** of package **`spdep`**. We can apply this function to the variable "Substrate density" of the oribatid mite data set. We first define neighbourhoods of size ≤ 0.7 m around the points using the function **`dnearneigh()`**. These links can be visualized using our function **`plot.links()`**. Following that, the function **`sp.correlogram()`** finds successive lag orders of contiguous neighbours and computes Moran's *I* for each of these lag orders. A lag order is the number of links, or steps in the linkage graph, between two points. It can be construed as a generalized form of distance between points. For instance, if sites A and C are connected through site B, two links (A–B and B–C) are needed to connect A and C. They are connected at lag order 2.

Note: Cartesian coordinates can be obtained from latitude–longitude (sometimes abbreviated to Lat/Lon or LatLon) data using the function **`geoXY()`** of package **`SoDA`**.

```
# Load the required packages
library(ape)
library(spdep)
library(vegan)
library(ade4)

# Packages available at the following URL:
# https://r-forge.r-project.org/R/?group_id=195
library(packfor)
library(spacemakeR)
library(AEM)
library(PCNM)
source("plot.links.R")  # Function must be in working directory
source("sr.value.R")    # Function must be in working directory
```

```
# Import the data

mite <- read.table("mite.txt")
mite.env <- read.table("mite_env.txt")
mite.xy <- read.table("mite_xy.txt")

mite.h <- decostand (mite, "hellinger")
mite.xy.c <- scale(mite.xy, center=TRUE, scale=FALSE)

# Spatial correlogram (based on Moran's I)
# ****************************************
# Search for neighbours of all points within a radius of 0.7 m
# and multiples (i.e., 0 to 0.7m, 0.7 to 1.4m and so on). The
# points do not form a connected graph at 0.7 m.
plot.links(mite.xy, thresh=0.7)
nb1 <- dnearneigh(as.matrix(mite.xy), 0, 0.7)
summary(nb1)

# Correlogram of substrate density
subs.dens <- mite.env[,1]
subs.correlog <- sp.correlogram(nb1, subs.dens, order=14,
method="I", zero.policy=TRUE)
print(subs.correlog, p.adj.method="holm")
plot(subs.correlog)
```

Hint *We use the* **`print()`** *function to display the correlogram results because it allows for correction of the p values for multiple testing. In a correlogram, a test is performed for each lag (distance class), so that without correction, the overall risk of type I error is greatly increased. The Holm (1979) correction is applied here.*

This correlogram has a single significant distance class: there is positive spatial correlation at distance class 1 (i.e. 0.0–0.7 m). Negative spatial correlation at distance class 4 (i.e. 2.1–2.8 m) is hinted at, but the coefficient is not significant after Holm (1979) correction for multiple testing (see Sect. 7.2.6). Beyond this mark, no significant spatial correlation is identified, which means that for practical purposes measurements taken more than 0.7 m, or (conservatively) 2.8 m apart (the upper limit of class 4), can be considered as spatially independent with respect to substrate density.

Spatial correlation in the multivariate domain can be assessed and tested for by means of a *Mantel correlogram* (Sokal 1986; Oden and Sokal 1986). Basically, one computes a normalized Mantel statistic r_M (analogous to a Pearson's r coefficient) between a dissimilarity matrix among sites and a matrix where pairs of sites belonging to the same distance class receive value 0 and the other pairs, value 1. The process

is repeated for each distance class. Each r_M value can be tested by permutations. The expectation of the Mantel statistic for no spatial correlation is $r_M = 0$.

A Mantel correlogram can be computed, tested and plotted (Fig. 7.1) by using **vegan**'s function **mantel.correlog()**. The only data necessary are a response distance matrix and either the geographical coordinates of the sites or a matrix of geographical distances among sites. Here is an example of a Mantel correlogram for the oribatid mite data, which is first detrended (Sect. 7.3.2) to make the data second-order stationary (Sect. 7.2.6).

```
# Mantel correlogram of the oribatid mite data
# ********************************************

# The species data are first detrended; see Section 7.3
mite.h.det <- resid(lm(as.matrix(mite.h) ~ ., data=mite.xy))

mite.h.D1 <- dist(mite.h.det)
(mite.correlog <- mantel.correlog(mite.h.D1, XY=mite.xy,
nperm=99))
summary(mite.correlog)
plot(mite.correlog)
```

Hint *In this run, the number of classes has been computed automatically using Sturge's rule. Use argument* `n.class` *to provide a user-determined number of classes.*

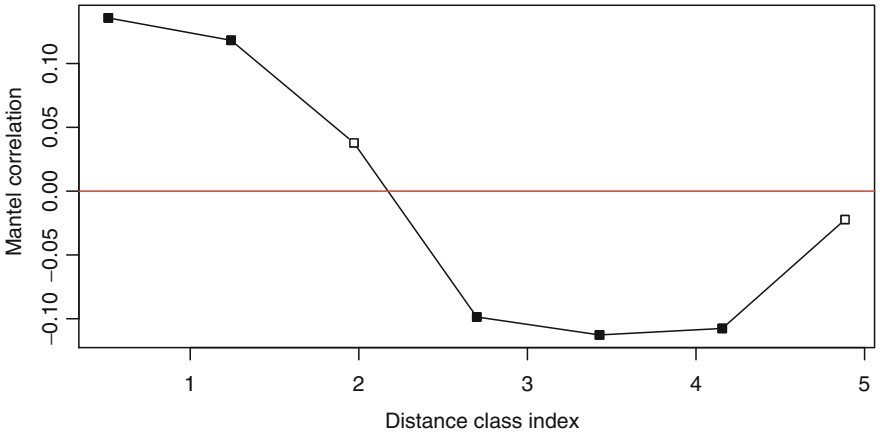

Fig. 7.1 Mantel correlogram of the Hellinger-transformed and detrended oribatid mite species data. *Black squares* indicate significant multivariate spatial correlation after Holm correction for multiple testing. The abscissa is labelled in metres since this is the unit of the data used to construct the distance classes

In this simple run, most default settings have been applied, including Holm's correction for multiple testing (see Sect. 7.2.6). The number of classes has been computed using Sturge's rule [number of classes = $1 + (3.3219 \times \log_{10} n)$, where n is the number of elements, here the number of pairwise distances]. The resulting number of classes and the corresponding break points can be read in the result object:

```
# Number of classes
mite.correlog$n.class       # or: mite.correlog[2]
# Break points
mite.correlog$break.pts     # or: mite.correlog[3]
```

Hint *The default option* `cutoff=TRUE` *limits the correlogram to the distance classes including all points (the first seven distance classes in this example); the results for the last five distance classes (computed on fewer and fewer points) are not shown.*

The result shows significant positive spatial correlation in the first two distance classes (i.e. between 0.15 and 1.61 m; see the break points) and negative significant correlation in the fourth to sixth classes (between 2.34 and 4.52 m). Examining the environmental variables allows some speculation about the ecological reasons behind these structures. Close sites tend to show similar communities because the soil conditions are rather similar. On the other hand, any pair of sites whose members are about 2.7 m apart (class index of distance class 4) falls into contrasting soil conditions, which in turn explains why their mite communities are different.

7.2.6 Testing for the Presence of Spatial Correlation: Conditions

As shown above, spatial correlation coefficients can be tested for significance. However, conditions of application must be respected. The condition of normality can be relaxed if the test is carried out by permutations. To test the significance of coefficients of spatial correlation, however, the condition of *second-order stationarity* must be met. That condition states that the mean of the variable and its spatial covariance (numerator of (7.3)) are the same over the study area, and that its variance (denominator of (7.3)) is finite. This condition tells us, in other words, that the spatial variation of the data should be adequately described by the same single spatial correlation function in all portions of the study area. Spatial correlation coefficients cannot be tested for significance if an overall trend is present in the data ("true gradient"), or if the variable has been measured in a region where several distinct structures should be modelled by different spatial correlation

functions. Data displaying simple trends can often be *detrended* by means of a first- or higher-degree function of the site geographical coordinates (Sect. 7.3).

Another, relaxed form of stationarity is called the *intrinsic assumption*, a short form for "hypothesis of intrinsic stationarity of order 2" (Wackernagel 2003). This condition considers only the increments of the values of the variable; it states that the differences $(y_h - y_i)$ for any distance d (in the numerator of (7.5)) have zero mean and constant and finite variance over the study area, independently of the location (Legendre and Legendre 1998). This condition allows one to compute and examine correlograms but without tests of significance.

Legendre and Legendre (1998, p. 721) show how to interpret all-directional correlograms (i.e. correlograms built on distance classes defined the same way in all directions) as well as directional correlograms.

A word is needed here about *multiple testing*. In Sect. 7.2.5 several spatial correlation values were tested simultaneously for significance. In such cases, the probability of type I error increases with the number of tests. If k tests are carried out, the binomial law tells us that the overall probability of type I error (technically called the "experimentwise error rate") is equal to $1-(1-\alpha)^k$ where α is the nominal value for a single test. For instance, in the Mantel correlogram shown in Fig. 7.1, seven tests are carried out simultaneously. Without correction, the *overall* probability of obtaining *at least* one type I error is equal to $1-(1-0.05)^7 = 0.302$ instead of the nominal $\alpha = 0.05$. Several methods have been proposed to achieve a correct level of type I error in multiple tests (reviewed in Legendre and Legendre 1998; Wright 1992). The most conservative solution for k independent tests is to divide the significance level by the number of simultaneous tests: $\alpha' = \alpha/k$ and compare the p values to α'. Conversely, one can multiply the p values by k (i.e. $p' = kp$) and compare the resulting values to the unadjusted α. For non-independent tests, Holm's procedure (Holm 1979) is more powerful. The reason is that Holm's correction consists in applying Bonferroni's correction sequentially by progressively relaxing the correcting factor as follows. First, order the (uncorrected) p values in increasing order from top to bottom. Then, multiply the smallest p value by k, the second smallest by $k-1$, and so on. If an adjusted p value is smaller than the previous one, make it equal to it. Compare the resulting values to the unadjusted alpha.

Other corrections have been proposed in addition to the two presented above. Several are available in a function called **`p.adjust()`** in package **`stats`**. This function can be called whenever one has run several simultaneous tests of significance. The data submitted to that function must be a vector.

7.2.7 Modelling Spatial Structures

Beyond the methods described above, there are other, more modelling-oriented approaches to spatial analysis. Finding spatial structures in ecological data indicates that some process has been at work to generate them; the most important are environmental forcing (past or present) and biotic processes. Therefore, it is interesting to identify the spatial structures in the data and model them. Spatial structures can then either be related to explanatory variables representing hypothesized causes, or help generate new hypotheses as to which processes may have generated them.

Spatial structures can be present at many different scales. Identifying these scales and modelling the corresponding spatial structures separately is a long-sought goal for ecologists. A first, rather coarse approach in multivariate analysis is the adaptation of trend-surface analysis to canonical ordination. As suggested by ter Braak (1987) and demonstrated by Legendre (1990), response data may be explained by a polynomial function of the (centred) site coordinates. Borcard et al. (1992) have shown how to integrate this method into variation partitioning to identify, among other fractions, the pure spatial component of the ecological variation of species assemblages.

Multivariate trend-surface analysis only allows one to extract rather simple spatial structures because polynomial terms become rapidly cumbersome, and highly correlated if one uses raw polynomials. In practice, their use is restricted to third-degree polynomials. A breakthrough came with the development of principal coordinates of neighbour matrices (PCNM) and other forms of eigenvector-based spatial functions, which are described in Sect. 7.4, after a short example of trend-surface analysis.

7.3 Multivariate Trend-Surface Analysis

7.3.1 Introduction

Most ecological data have been sampled on geographic surfaces. Therefore, the crudest way to model the spatial structure of the response data is to regress them on the *X*–*Y* coordinates of the sampling sites. Of course, this will only model a *linear trend*; a plane will be fitted through the data in the same way as a straight line would be fitted to data collected along a transect by regressing them on their *X* coordinates.

A way of allowing curvilinear structures to be modelled is to add polynomial terms of the coordinates to the explanatory data. Second- and third-degree terms are often applied. It is better to centre (but not standardize, lest one distort the aspect-ratio

of the sampling design) the X and Y coordinates before computing the polynomial terms, to make at least the second-degree terms less correlated. The first-, second- and third-degree functions are:

$$\hat{z} = f(X,Y) = b_0 + b_1 X + b_2 Y \tag{7.6}$$

$$\hat{z} = b_0 + b_1 X + b_2 Y + b_3 X^2 + b_4 XY + b_5 Y^2 \tag{7.7}$$

$$\hat{z} = b_0 + b_1 X + b_2 Y + b_3 X^2 + b_4 XY + b_5 Y^2 + b_6 X^3 + b_7 X^2 Y + b_8 XY^2 + b_9 Y^3 \tag{7.8}$$

An alternative method is to compute orthogonal polynomial terms using the function **poly()** with the default option `raw=FALSE`, which produces orthogonal polynomials. For a set of X–Y coordinates, the monomials X, X^2, X^3 and Y, Y^2, Y^3 have a norm of 1 and are orthogonal to their respective lower order terms. X monomials are not orthogonal to Y monomials, however, except when the points form a regular orthogonal grid; terms containing both X and Y are not orthogonal to one another and their norms differ from 1. Orthogonal polynomials produce the exact same R^2 in regression and canonical analysis as raw polynomials. The orthogonality of orthogonal polynomials presents an advantage when selection of explanatory variables is used to find a parsimonious spatial model.

Trend-surface analysis can be applied to multivariate data by means of RDA or CCA. The result is a set of independent spatial models (one for each canonical axis). One can also use forward selection to reduce the model to its significant components only.

7.3.2 Trend-Surface Analysis in Practice

Our first step in spatial modelling is to produce some monomials of the X and Y coordinates on a grid just to become familiar with the shapes they produce through visualization. We then proceed to apply this technique to the oribatid mite data. As a courtesy to our readers, we have modified **ade4**'s **s.value()** function to draw round instead of square bubbles in some plots. The modified function is called **sr.value()**.

```
# Trend-surface analysis
# **********************

# Simple models on a square, regularly sampled surface

# Construct and plot a 10 x 10 grid
xygrid <- expand.grid(1:10, 1:10)
plot(xygrid)
```

```
xygrid.c <- scale(xygrid, scale=FALSE)  # Centring
X <- xygrid.c[,1]
Y <- xygrid.c[,2]

# Plot some first, second and third-degree functions of X and Y
par(mfrow=c(3,3))
s.value(xygrid,(X))
s.value(xygrid,(Y))
s.value(xygrid,(X+Y))
s.value(xygrid,(X^2+Y^2))
s.value(xygrid,(X^2-X*Y-Y^2))
s.value(xygrid,(X+Y+X^2+X*Y+Y^2))
s.value(xygrid,(X^3+Y^3))
s.value(xygrid,(X^3+X^2*Y+X*Y^2+Y^3))
s.value(xygrid,(X+Y+X^2+X*Y+Y^2+X^3+X^2*Y+X*Y^2+Y^3))

  # Try other combinations, for instance with minus signs or with
  # coefficients not equal to 1.

# Trend-surface analysis of the mite data
# ***************************************
# Computation of the standard (non-orthogonal) third-degree
polynomial
# function on the previously centred X-Y coordinates
mite.poly <- poly(as.matrix(mite.xy.c), degree=3, raw=TRUE)
colnames(mite.poly) <-
  c("X","X2","X3","Y","XY","X2Y","Y2","XY2","Y3")

  # Function poly produces the polynomial terms in the following
  # sequence: X, X^2, X^3, Y, XY, X^2Y, Y^2, XY^2, Y^3).
  # The original column names give the degree for the two
  # variables.
  # For instance, "1.2" means X^1*Y^2.
  # Here raw polynomials have been computed. For orthogonal
  # polynomials, which is the default, raw=FALSE.

# RDA with all 9 polynomial terms
mite.trend.rda <- rda(mite.h ~ ., data=as.data.frame(mite.poly))

# Computation of the adjusted R^2
(R2adj.poly <- RsquareAdj(mite.trend.rda)$adj.r.squared)

# RDA using a third-degree orthogonal polynomial of the
# geographic coordinates
mite.poly.ortho <- poly(as.matrix(mite.xy), degree=3)
```

```
colnames(mite.poly.ortho) <-
 c("X","X2","X3","Y","XY","X2Y","Y2","XY2","Y3")
mite.trend.rda.ortho <- rda(mite.h~.,
data=as.data.frame(mite.poly.ortho))
(R2adj.poly <- RsquareAdj(mite.trend.rda.ortho)$adj.r.squared)

# Forward selection using Blanchet et al. (2008a) double
# stopping criterion
mite.trend.fwd <- forward.sel(mite.h, mite.poly.ortho,
adjR2thresh=R2adj.poly)

# New RDA using the 6 terms retained
(mite.trend.rda2 <- rda(mite.h ~ .,
data=as.data.frame(mite.poly)[,mite.trend.fwd[,2]]))

# Overall test and test of the canonical axes
anova.cca(mite.trend.rda2, step=1000)
anova.cca(mite.trend.rda2, step=1000, by="axis")

# Plot of the three independent significant spatial structures
# (canonical axes). For square bubbles type "s.value" instead of
# "sr.value".
mite.trend.fit <- scores.cca(mite.trend.rda2, choices=c(1,2,3),
display="lc", scaling=1)
par(mfrow=c(1,3))
sr.value(mite.xy,mite.trend.fit[,1])
sr.value(mite.xy,mite.trend.fit[,2])
sr.value(mite.xy,mite.trend.fit[,3])

# ------------------------------------------------------------

 # If you want to construct a raw polynomial function directly
 # within the rda call, here is the syntax (2nd degree):
 # mite.trend.rda <- rda(mite.h ~ Xm + Ym + I(Xm^2) + I(Xm*Ym)
 # + I(Ym^2))
 # Notice how squared variables and product variables are
 # requested to be treated "as they are" by function I().
 # Otherwise R would consider them as ANOVA terms.
```

Hint *Note that the **fitted site scores in scaling 1** have been used in the plots. We want to display the "pure" spatial model, i.e. the linear combination of spatial variables, in a projection preserving the Euclidean distances among sites.*

This analysis shows that the oribatid mite community is significantly spatially structured, and that three significant independent models can be obtained. The first one (first canonical axis, 73.8% of the *explained* variance) displays a strong difference between the upper and the lower half of the area. The two other models (12.1 and 8.5% of the explained variance, respectively) display finer-scale structures (Fig. 7.2).

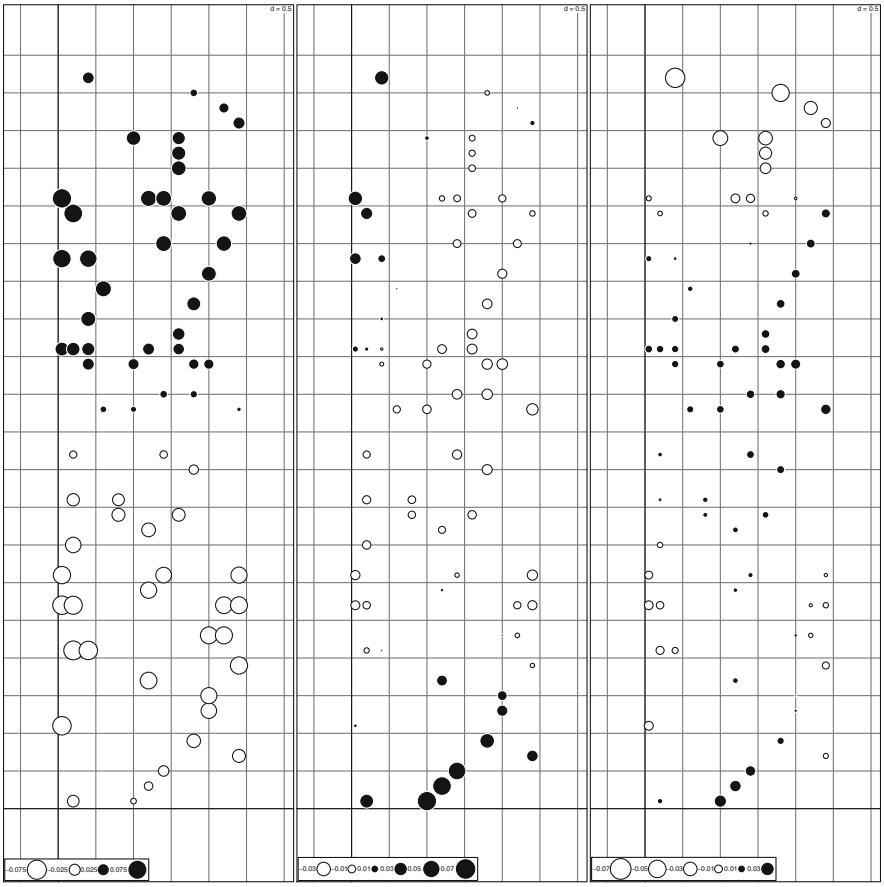

Fig. 7.2 Cubic trend-surface analysis of the Hellinger-transformed oribatid mite data. Three significant RDA axes have been retained, representing linearly independent spatial structures

These models could now be interpreted by regressing them on environmental variables. But we postpone that step until we can implement it in another spatial modelling framework.

Nowadays, the most useful application of trend-surface analysis is for *detrending*. We have seen in Sect. 7.2.6 that data have to be detrended before spatial correlograms can be tested. We will also see later that eigenvector-based spatial analyses are best applied to detrended data. Therefore, a handy procedure is to test for linear trends and detrend the data if the trend surface is significant. This means to regress all variables on the *X*–*Y* coordinates and retain the residuals. This can most easily be done using the function **`lm()`**.

```
# Detrending the mite data
# ************************
anova(rda(mite.h, mite.xy))  # Result: significant trend
# Computation of linearly detrended mite data
mite.h.det <- resid(lm(as.matrix(mite.h) ~ ., data=mite.xy))
```

This detrended data set is now ready for more complex spatial analyses and modelling.

7.4 Eigenvector-Based Spatial Variables and Spatial Modelling

7.4.1 Introduction

Trend-surface analysis is a rather coarse method of spatial modelling. The multi-scale nature of ecological processes and data calls for other approaches that can identify and model structures at all scales that can be perceived by a sampling design. Practically, this means methods that could model structures at scales ranging from the broadest, encompassing the whole sampled area, down to the finest, whose sizes are of the order of magnitude of the sampling interval. To achieve this in the context of canonical ordination, we must construct spatial variables representing structures of all relevant scales. This is what the PCNM method (principal coordinates of neighbour matrices; Borcard and Legendre 2002; Borcard et al. 2004) and its offsprings do. These methods will now be studied in detail.

As will be shown in Sect. 7.4.3, the PCNM method is actually a special case of a wider family of methods that are now called MEM (Moran's eigenvector maps; Dray et al. 2006). However, since many published papers cite this variant under its

original name, we use it here for explanatory purposes. Bear in mind, however, that the acronym PCNM is likely to be short-lived in the literature. New papers use the generic acronym MEM.

7.4.2 Classical Distance-Based MEM, Formerly Called Principal Coordinates of Neighbour Matrices

7.4.2.1 Introduction

The PCNM method works as follows:

- Construct a matrix of Euclidean (geographic) distances among sites.
- Truncate this matrix to retain only the distances among close neighbours. The threshold depends on the data. In most cases, it is chosen to be as short as possible, but all points must remain connected by links smaller than or equal to the truncation distance. Otherwise, different groups of eigenfunctions are created, that model the spatial variation within separate subgroups of points but not among these groups. How to choose the truncation threshold distance is discussed below. All pairs of points more distant than the threshold receive an arbitrary "large" distance value corresponding to four times the threshold.
- Compute a PCoA of the truncated distance matrix.
- In most studies, retain the eigenvectors that model positive spatial correlation (Moran's I larger than $E(I)$, (7.4)).
- Use these eigenvectors as spatial explanatory variables in multiple regression or RDA.

The PCNM method presents several advantages over trend-surface analysis. It produces *orthogonal* (linearly independent) spatial variables over a *much wider range* of spatial scales. It allows the modelling of *any type* of spatial structures, as Borcard and Legendre (2002) have demonstrated through extensive simulations.

The PCNM method can work for any sampling design, although the spatial variables are easier to interpret in the case of regular designs, as shown below. When the design is irregular, it may happen that a large truncation value must be chosen to allow all points to remain connected. A large truncation value means a loss of the finest spatial structures. Therefore, ideally, even an irregular sampling design should ensure that the minimum distance allowing all points to be connected is as short as possible. In cases where this distance is too large, Borcard and Legendre (2002) suggested (1) to add a limited number of supplementary points to the spatial data to cut down the threshold distance, (2) compute the PCNM variables, and (3) remove the supplementary points from the PCNM matrix. This ensures that

the finest scales are better modelled. The trade-off is that the resulting PCNM variables are no longer totally orthogonal to one another, but if the number of supplementary points is small with respect to the number of true points, the departure from orthogonality remains small.

The classical PCNM method produces eigenfunctions for all positive eigenvalues. However, some of these eigenfunctions display negative spatial correlation. In most studies, one is primarily interested in patterns produced by spatially contagious processes, which display positive spatial correlation. Therefore, it is generally preferable to retain only the eigenfunctions with Moran's $I > E(I)$, or to run separate analyses with the eigenfunctions with positive and negative spatial correlation. The relationship between the sign of the eigenvalues and the sign of the spatial correlation is not simple for PCNM, whereas the value of Moran's I is a linear function of the eigenvalue in the case of standard MEM eigenfunctions. So it is advised to compute Moran's I in all cases. Function **`PCNM()`** of the package **`PCNM`** presented in Sect. 7.4.2.3 can do that automatically (argument `moran`).

7.4.2.2 PCNM Variables on Regular Sampling Designs

When the spatial coordinates correspond to points that are equispaced along a transect or across a surface, the resulting PCNM variables represent a series of sinusoids of decreasing periods. For a transect with n regularly spaced points and sampling interval s, the wavelength λ_i of the eigenfunction with rank i is: $\lambda_i = 2(n+s)/(i+1)$ (Guénard et al. 2010, eq. 3).[1] Let us construct and illustrate a one-dimensional (Fig. 7.3) and a two-dimensional (Fig. 7.4) example, both equispaced.

```
# Constructing PCNM variables step by step
# ****************************************

# 1. One-dimensional sampling: transect with 100 equispaced
#    points. The distance between adjacent points is 1

tr100 <- seq(1:100)             # Generate transect points
tr100.d1 <- dist(tr100)         # Euclidean distance matrix
thresh <- 1                     # truncation distance set to 1
# Truncation to threshold 1
tr100.d1[tr100.d1 > thresh] <- 4*thresh
# PCoA of truncated matrix
tr100.PCoA <- cmdscale(tr100.d1, eig=TRUE, k=length(tr100)-1)
```

[1] A simple function to find the wavelength for an intersite distance $s = 1$ is: `wavelength <- function(i,n) {2*(n+1)/(i+1)}`.

```
# Count the positive eigenvalues
(nb.ev <- length(which(tr100.PCoA$eig > 0.0000001)))
# Matrix of PCNM variables
tr100.PCNM <- tr100.PCoA$points[,1:nb.ev]

# Plot some PCNM variables modelling positive spatial
# correlation (Fig. 7.3)
par(mfrow=c(4,2))
somePCNM <- c(1, 2, 4, 8, 15, 20, 30, 40)
for(i in 1:length(somePCNM)){
 plot(tr100.PCNM[,somePCNM[i]], type="l", ylab=c("PCNM",
  somePCNM[i]))
}

# 2. Two-dimensional sampling: equispaced grid
#    The truncation distance is set to 1. It could also be
#    chosen to be the diagonal distance within a small square
#    of 4 points, sqrt(2)

xygrid2 <- expand.grid(1:20, 1:20)
xygrid2.d1 <- dist(xygrid2)
thresh <- 1                    # truncation distance set to 1
# Truncation to threshold 1
xygrid2.d1[xygrid2.d1>thresh] <- 4*thresh
# PCoA of truncated matrix
xygrid2.PCoA <- cmdscale(xygrid2.d1, eig=TRUE,
  k=nrow(xygrid2)-1)
# Count the positive eigenvalues
(nb.ev2 <- length(which(xygrid2.PCoA$eig > 0.0000001)))
# Matrix of PCNM variables
xygrid2.PCNM <- xygrid2.PCoA$points[,1:nb.ev2]

# Plot some PCNM variables modelling positive spatial
# correlation (Fig. 7.4)
par(mfrow=c(4,2))
somePCNM2 <- c(1, 2, 5, 10, 20, 50, 100, 150)
for(i in 1:length(somePCNM2)){
 sr.value(xygrid2, xygrid2.PCNM[,somePCNM2[i]],
 method="greylevel", csize=0.35, sub=somePCNM2[i],
 csub=2)
}
```

Hint *Functions* **`s.value()`** *and* **`sr.value()`** *propose two representations of values on maps: by symbols with sizes proportional to the values (default argument* `method = "squaresize"`*) and by symbols of constant size and values represented by shades of grey (*`method = "greylevel"`*).*

Figures 7.3 and 7.4 show that these variables are periodical and ranging from broadest to finest scales. As discussed above, this does not imply that only periodical structures can be modelled by PCNM analysis, however. Even short-range spatial correlation can be modelled by fine-scale PCNM variables. This topic is addressed later.

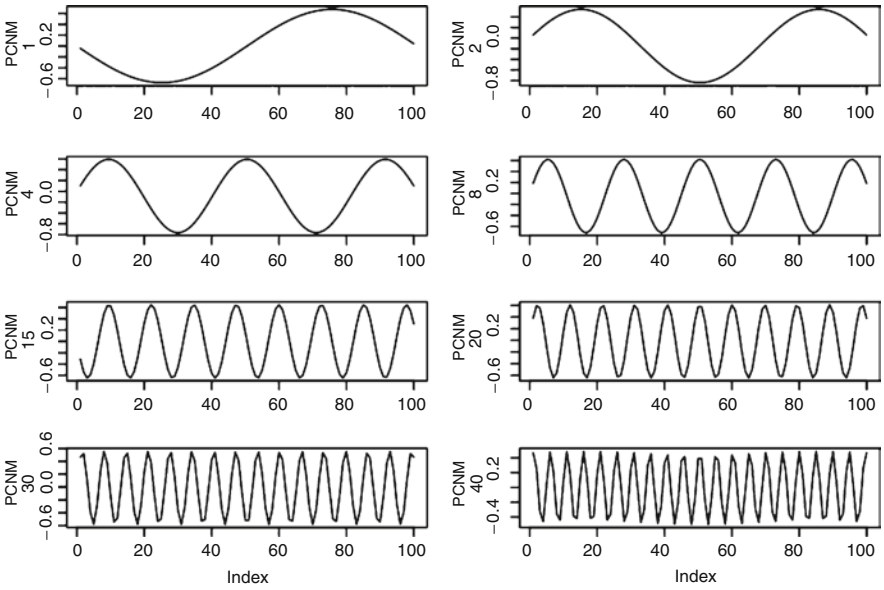

Fig. 7.3 Some of the 67 PCNM variables with positive eigenvalues built from a transect of 100 equispaced points. The first 49 of them have Moran's I larger than $E(I)$, showing that they model positive spatial correlation

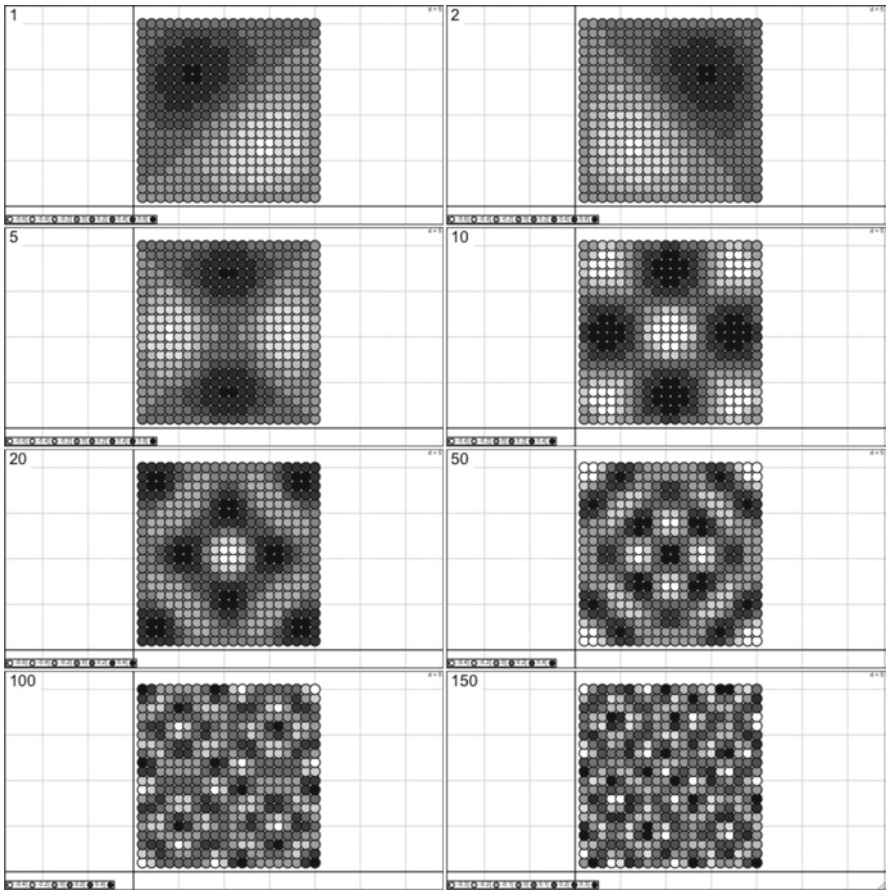

Fig. 7.4 Some of the 279 PCNM variables with positive eigenvalues built from a grid of 20 by 20 equispaced points. The first 209 of them have Moran's I larger than $E(I)$

7.4.2.3 PCNM Analysis of the Mite Data

PCNM analysis is not restricted to regular sampling designs. The drawback in the case of irregular designs is that the PCNM variables lose the regularity of their shapes, sometimes making the assessment of scale more difficult.

Now, it is time to try PCNM analysis on real data. You will first run the analysis "by hand", i.e. by separately coding every step. After that, automated functions will be presented that simplify the analysis.

In the code below, PCNM variables are constructed using a package dedicated to this task (called **PCNM**). Function **PCNM()** of this package provides an immediate assessment of the spatial correlation (Moran's *I*, (7.3)) displayed by the computed eigenfunctions. Moran's *I* gives a criterion to decide which eigenfunctions should be used for modelling; see Sect. 7.4.4. Otherwise, the user can apply the simpler function **pcnm()** of the **vegan** package.

```
# PCNM analysis of the oribatid mite data
# ***************************************

# 1a. Construct the matrix of PCNM variables step by step...
# ----------------------------------------------------------

xy.d1 <- dist(mite.xy)
  # Search for the truncation threshold: maximum value of the
  # minimum spanning tree of the Euclidean distance matrix
  # using vegan's function spanntree. Use that distance, or any
  # other distance larger than that, as the truncation distance.
spanning <- spantree(xy.d1)
dmin <- max(spanning$dist)

# Truncate the distance matrix
xy.d1[xy.d1 > dmin] <- 4*dmin
# PCoA of truncated distance matrix
xy.PCoA <- cmdscale(xy.d1, k=nrow(mite.xy)-1, eig=TRUE)
# Count the positive eigenvalues (PCNM with positive AND
negative spatial
# correlation)
(nb.ev <- length(which(xy.PCoA$eig > 0.0000001)))
# Construct a data frame containing the PCNM variables
mite.PCNM <- as.data.frame(xy.PCoA$points[1:nrow(mite.xy),
1:nb.ev])

# 1b. ... or construct the PCNM variables automatically
# -----------------------------------------------------

# library(PCNM) # If not already loaded
xy.d1 <- dist(mite.xy)
mite.PCNM.auto <- PCNM(xy.d1)
summary(mite.PCNM.auto)
# Plot the minimum spaning tree used to find the truncation
# distance
plot.spantree(mite.PCNM.auto$spanning, mite.xy)
(dmin <- mite.PCNM.auto$thresh) # Truncation distance
(nb.ev <- length(mite.PCNM.auto$values)) # Number of eigenvalues
```

```
# Moran's I of the PCNM variables (in the first distance class,
# 0 to truncation threshold); also see figure generated by the
# PCNM() function (not reproduced here).

# Expected value of I, no spatial correlation
mite.PCNM.auto$expected_Moran
mite.PCNM.auto$Moran_I

# Eigenfunctions with positive spatial correlation
(select <- which(mite.PCNM.auto$Moran_I$Positive == TRUE))
length(select)  # Number of PCNM with I > E(I)
mite.PCNM.pos <- as.data.frame(mite.PCNM.auto$vectors)[,select]

# 1c. ... or use vegan's function pcnm()
# ---------------------------------------
mite.PCNM.vegan <- pcnm(dist(mite.xy))
mite.PCNM <- as.data.frame(mite.PCNM.vegan$vectors)
# dmin <- mite.PCNM.vegan$threshold
nb.ev <- length(which(mite.PCNM.vegan$values > 0.0000001))
 # The eigenvectors obtained by this function are divided by the
 # square root of their eigenvalue. This is not important for
 # their use as spatial variables.
 # Contrary to function PCNM(), vegan's pcnm() does not provide
 # Moran's I,which must be computed separately if one chooses to
 # retain only the eigenfunctions with positive spatial
 # correlation.
```

Hint *The truncation distance can be chosen by the user to be either the value proposed by the PCNM function (longest link along the minimum spanning tree drawn on the map of the points), or any other value larger than that. For example, for a regular two-dimensional grid of points with spacing of 1, one may choose a value slightly larger than the distance between diagonal neighbours, sqrt(2) = 1.4142, as the truncation distance. The chosen truncation distance may be 1.4143 in that case.*

As one can see, the first 17 PCNMs have significant positive spatial correlations at the 5% significance level, while significant negative spatial correlations are found in PCNMs 35–43. The test of significance of Moran's *I* may not be a reliable criterion to eliminate PCNMs from the analysis, however, so we will keep the 23 PCNMs with positive spatial correlation. We will apply forward selection with the Blanchet et al. (2008a) double stopping criterion.

```
# 2. Run the global PCNM analysis on the *detrended* mite data
# ------------------------------------------------------------

mite.PCNM.rda <- rda(mite.h.det, mite.PCNM.pos)
anova.cca(mite.PCNM.rda)

# 3. Since the analysis is significant, compute the adjusted R2
#    and run a forward selection of the PCNM variables

(mite.R2a <- RsquareAdj(mite.PCNM.rda)$adj.r.squared)
(mite.PCNM.fwd <- forward.sel(mite.h.det,
as.matrix(mite.PCNM.pos), adjR2thresh=mite.R2a))
 # According to the R2a criterion, if we retain PCNM 5 we get a
 # model with a R2adj slightly higher than that of the complete
 # model. This slight excess is not too serious, however.

(nb.sig.PCNM <- nrow(mite.PCNM.fwd)) # Number of signif. PCNM

# Identity of significant PCNMs in increasing order
(PCNM.sign <- sort(mite.PCNM.fwd[,2]))
# Write the significant PCNMs to a new object
PCNM.red <- mite.PCNM.pos[,c(PCNM.sign)]

# 4. New PCNM analysis with 10 significant PCNM variables
#    Adjusted R-square after forward selection: R2adj=0.2713

mite.PCNM.rda2 <- rda(mite.h.det ~ ., data=PCNM.red)
(mite.fwd.R2a <- RsquareAdj(mite.PCNM.rda2)$adj.r.squared)
anova.cca(mite.PCNM.rda2)
axes.test <- anova.cca(mite.PCNM.rda2, by="axis")
(nb.ax <- length(which(axes.test[,5] <= 0.05))) # Number of
significant axes

# 5. Plot the two significant canonical axes
mite.PCNM.axes <- scores.cca(mite.PCNM.rda2, choices=c(1,2),
display="lc", scaling=1)
par(mfrow=c(1,2))
sr.value(mite.xy, mite.PCNM.axes[,1]) # ade4 function: s.value
sr.value(mite.xy, mite.PCNM.axes[,2]) # ade4 function: s.value
```

Hint *In the* **`scores.cca()`** *call above, be careful to set* **`display="lc"`**. *The default is* **`"wa"`**, *but here we want the fitted site scores.*

PCNM analysis of the detrended mite data explained 27.13% of the variance (see **`mite.fwd.R2a`**, the adjusted R^2 obtained with the ten variables retained by forward selection, slightly exceeding the total R^2_{adj}). Two canonical axes, explaining $27.13 \times 0.7527^2 = 20.4\%$ of the total variance, are significant; their fitted site

[2] The value 0.7527 is found in the section "Accumulated constrained eigenvalues" of the RDA output. It is the proportion of variance explained by the first two canonical axes with respect to the total *explained* variance.

scores have been plotted on a map of the sites. These two plots (Fig. 7.5) represent the spatially structured variation of the detrended oribatid mite data. Now, is this variation related to any of the environmental variables? A simple way to assess this is to regress the fitted site scores of these two canonical axes on the environmental data.

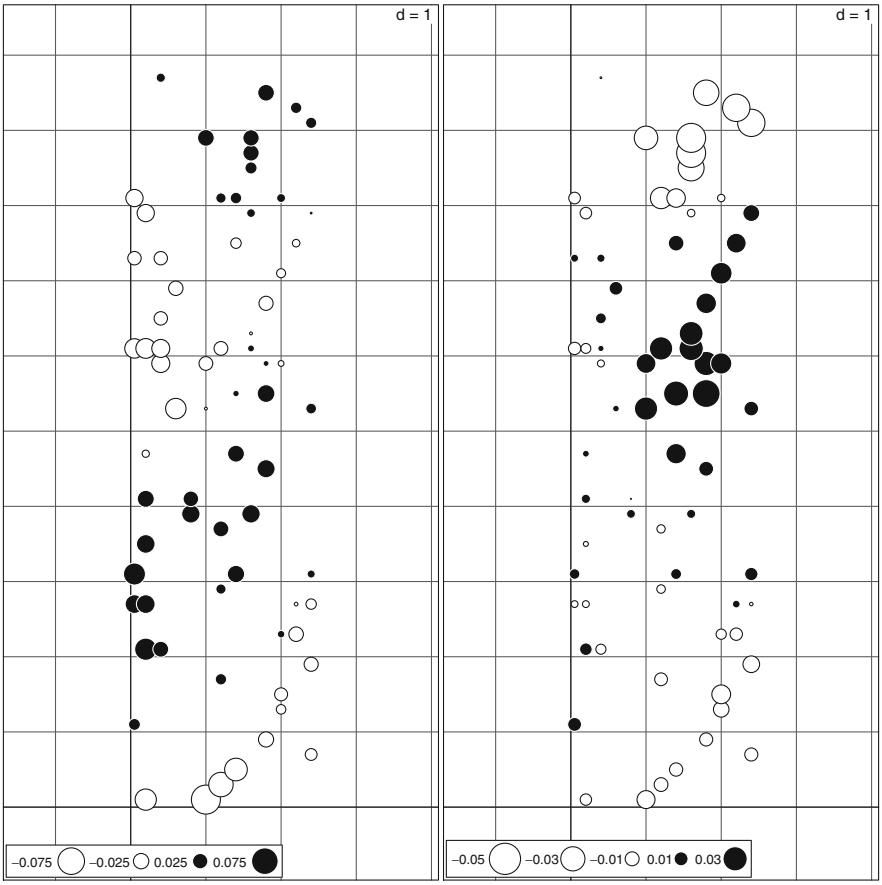

Fig. 7.5 Manual PCNM analysis of the detrended oribatid mite data, ten significant PCNM variables. Maps of the fitted site scores of the first two canonical axes

```
# Interpreting the spatial variation: regression of the two
# significant canonical axes on the environmental variables
# (after normality tests)

shapiro.test(resid(lm(mite.PCNM.axes[,1] ~ ., data=mite.env)))
mite.PCNM.axis1.env <- lm(mite.PCNM.axes[,1] ~ ., data=mite.env)
summary(mite.PCNM.axis1.env)

shapiro.test(resid(lm(mite.PCNM.axes[,2] ~ ., data=mite.env)))
mite.PCNM.axis2.env <- lm(mite.PCNM.axes[,2] ~ ., data=mite.env)
summary(mite.PCNM.axis2.env)

  # As one can see, the two spatial axes are not related to the
  # same environmental variables (except for shrubs). They are
  # not fully explained by them either. A precise assessment of
  # the portions explained will require variation partitioning.
```

This PCNM analysis produced spatial models combining all the PCNM variables forward-selected from the set of 23 classical PCNMs with positive spatial correlation. Here, both significant canonical axes are a combination of PCNM variables ranging from broad (PCNM1) to fine scale (PCNM23). While this may be interesting if one is interested in the global spatial structure of the response data, it does not allow one to discriminate between broad, medium and fine-scale structures since all significant PCNMs are combined.

Another approach consists in computing *separate* RDAs constrained by subsets of the significant PCNM variables. The PCNM variables being linearly independent of one another, any submodel defined with a subset of PCNMs is also independent of any other submodel defined with another subset. These subsets can be defined in such a way as to model different scales. The choices are arbitrary: there is no general rule defining what is broad, medium or fine scale. One can either predefine these limits, using the sizes of the patterns corresponding to the PCNM variables, or run forward selection and define submodels corresponding to groups of more or less consecutive PCNM variables retained by the procedure. One can also draw maps of the significant PCNM variables (Fig. 7.6) and group them according to the scales of the patterns they represent:

```
# Maps of the 10 significant PCNM variables
# ****************************************
par(mfrow=c(2,5))
for(i in 1:ncol(PCNM.red)){
 sr.value(mite.xy, PCNM.red[,i], sub=PCNM.sign[i], csub=2)
}
```

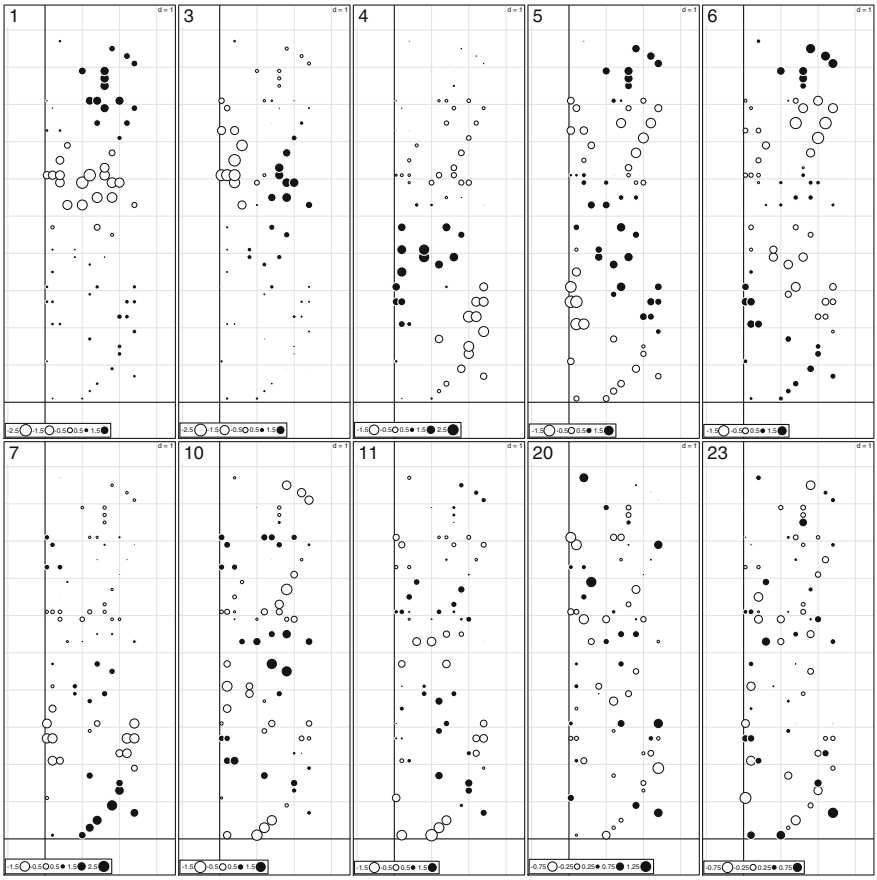

Fig. 7.6 The ten significant PCNM variables with positive spatial correlation used in the manual PCNM analysis of the detrended oribatid mite data

On this basis, one could for instance define PCNMs 1, 3 and 4 as "broad scale", PCNMs 5, 6, 7, 10 and 11 as "medium scale", and PCNMs 20 and 23 as "fine scale" descriptors. Separate RDAs with these subsets model broad, medium and fine scale patterns, respectively.

```
# PCNM analysis of the mite data - broad scale
# ********************************************

mite.PCNM.broad <- rda(mite.h.det ~ .,
 data=mite.PCNM.pos[,c(1,3,4)])
anova.cca(mite.PCNM.broad)
axes.broad <- anova.cca(mite.PCNM.broad, by="axis")
nb.ax.broad <- length(which(axes.broad[,5] <= 0.05))
nb.ax.broad     # Number of significant axes

# Plot of the two significant canonical axes
mite.PCNMbroad.axes <- scores.cca(mite.PCNM.broad,
 choices=c(1,2), display="lc", scaling=1)
par(mfrow=c(1,2))
s.value(mite.xy, mite.PCNMbroad.axes[,1])
s.value(mite.xy, mite.PCNMbroad.axes[,2])

# Interpreting spatial variation: regression of the two
# significant spatial canonical axes on the environmental
# variables

mite.PCNMbroad.ax1.env <- lm(mite.PCNMbroad.axes[,1] ~ .,
 data=mite.env)
summary(mite.PCNMbroad.ax1.env)

mite.PCNMbroad.ax2.env <- lm(mite.PCNMbroad.axes[,2] ~ .,
 data=mite.env)
summary(mite.PCNMbroad.ax2.env)
 # The broad-scale relationships are clearly related to
 # microtopography and the absence of shrubs.


# PCNM analysis of the mite data - medium scale
# *********************************************

mite.PCNM.med <- rda(mite.h.det ~ .,
data=mite.PCNM.pos[,c(5,6,7,10,11)])
anova.cca(mite.PCNM.med)
axes.med <- anova.cca(mite.PCNM.med, by="axis")
nb.ax.med <- length(which(axes.med[,5] <= 0.05))
nb.ax.med      # Number of significant axes

# Plot of the significant canonical axes (the second axis is
# marginally significant)
mite.PCNMmed.axes <- scores.cca(mite.PCNM.med, choices=c(1,2),
 display="lc", scaling=1)
```

```
par(mfrow=c(1,2))
s.value(mite.xy, mite.PCNMmed.axes[,1])
s.value(mite.xy, mite.PCNMmed.axes[,2])

# Interpreting spatial variation: regression of the significant
# spatial canonical axes on the environmental variables

mite.PCNMmed.ax1.env <- lm(mite.PCNMmed.axes[,1] ~ .,
 data=mite.env)
summary(mite.PCNMmed.ax1.env)
mite.PCNMmed.ax2.env <- lm(mite.PCNMmed.axes[,2] ~ .,
 data=mite.env)
summary(mite.PCNMmed.ax2.env)
 # The medium-scale features correspond to several types of soil
 # coverage and to soil moisture (variable WatrCont).


# PCNM analysis of the mite data - fine scale
# ******************************************

mite.PCNM.fine <- rda(mite.h.det ~ .,
data=mite.PCNM.pos[,c(20,23)])
anova.cca(mite.PCNM.fine)
axes.fine <- anova.cca(mite.PCNM.fine, by="axis")
nb.ax.fine <- length(which(axes.fine[,5] <= 0.05))
nb.ax.fine      # Number of significant axes

# Plot of the significant canonical axis
mite.PCNMfine.axes <- scores.cca(mite.PCNM.fine, choices=1,
display="lc", scaling=1)
par(mfrow=c(1,2))
s.value(mite.xy, mite.PCNMfine.axes)

# Interpreting spatial variation: regression of the significant
# spatial canonical axis on the environmental variables

mite.PCNMfine.ax1.env <- lm(mite.PCNMfine.axes ~ .,
data=mite.env)
summary(mite.PCNMfine.ax1.env)
 # The fine-scale structure is weakly related to the
 # environmental variables. Only one soil coverage class (plant
 # litter) is significant.
```

Something has occurred here, which is often found in fine-scale PCNM analysis. The only convincing correlation is with the presence of bare peat, a feature that was very localized in the study area. Otherwise, in most cases, fine scale PCNM

variables cannot be related to environmental descriptors and are mostly the signature of local spatial correlation generated by community dynamics. This topic will be addressed later.

7.4.2.4 Hassle-Free PCNM Analysis: Function `quickPCNM()`

A single-step PCNM analysis can be performed easily with the function **`quickPCNM()`**. This function, available in the **`PCNM`** package, requires only two arguments: a response data table (pre-transformed if necessary) and a table containing the site geographic coordinates (which can be one- or two-dimensional). The function performs a complete PCNM analysis: it checks whether the response data should be detrended and does it if a significant trend is identified; it constructs the PCNM variables and tests the global analysis; it runs forward selection, using the PCNMs with positive spatial correlation; it runs RDA with the retained PCNM variables and tests the canonical axes; it delivers the RDA results (including the set of PCNM variables) and plots maps of the significant canonical axes.

```
# Single-step PCNM analysis using function quickPCNM()
# ****************************************************

mite.PCNM.quick <- quickPCNM(mite.h, mite.xy)
summary(mite.PCNM.quick)
mite.PCNM.quick[[2]]   # Eigenvalues
mite.PCNM.quick[[3]]   # Results of forward selection

  # quickPCNM prevents forward.sel() from admitting a variable
  # when the resulting R2adj exceeds that of the complete model.
  # Therefore, PCNM5 is not among the selected PCNMs in the
  # quickPCNM results.
```

Function **`quickPCNM()`** provides several arguments to fit various needs. For instance, detrending is done by default if a significant trend is found, but this option can be disabled (`detrend=FALSE`). The truncation threshold is computed automatically unless the user provides another value (e.g. `thresh=1.234`). Computation of the PCNM variables is overridden if the user provides a ready-made set of spatial regressors (`myPCNM=userdataset`).

`quickPCNM()` provides a composite output object containing many results. The summary shows all the components. To draw a biplot of the RDA results, the code is the following:

```
# Extract and plot RDA results from a quickPCNM output
# (scaling 2)
# ************************************************

plot(mite.PCNM.quick$RDA, scaling=2)
sp.scores2 <- scores(mite.PCNM.quick$RDA,  choices=1:2,
  scaling=2, display="sp")
arrows(0, 0, sp.scores1[,1], sp.scores1[,2], length=0, lty=1
   col="red")

  # The scaling 2 shows the relationship of some species with
  # some PCNM variables. These correlations can be explored to
  # reveal at which scale the species distributions are spatially
  # structured.
```

7.4.2.5 Combining PCNM Analysis and Variation Partitioning

A clever and global approach to assess the environmental variation related to all scales of spatial variation is to perform a *variation partitioning* with an environmental data set and up to three subsets of spatial variables. Function **`varpart()`** can only handle numeric variables (not factors), however, so that we have to recode environmental variables 3–5 into dummy binary variables.

Variation partitioning aims at quantifying the various unique and combined fractions of variation explained by several sources. In this context, a linear trend can be considered as a source of variation like any other. The trend is likely to act on the response as well as the explanatory variables. Therefore, in this application we advocate *not* to detrend the response data prior to variation partitioning, but rather to test for a linear trend and incorporate it explicitly in the partitioning procedure if it is significant.

In this example, we independently forward-select the *X*–*Y* coordinates, the environmental variables and the PCNM variables before variation partitioning. The significant PCNM variables are split into a broad and a fine-scale fraction. The partitioning results are presented in Fig. 7.7.

```
# Mite - trend - environment - PCNM variation partitioning
# ********************************************************

# 1. Test trend. If significant, forward selection of
#    coordinates
# -----------------------------------------------------

mite.XY.rda <- rda(mite.h, mite.xy)
anova.cca(mite.XY.rda)
(mite.XY.R2a <- RsquareAdj(mite.XY.rda)$adj.r.squared)
(mite.XY.fwd <- forward.sel(mite.h, as.matrix(mite.xy),
  adjR2thresh=mite.XY.R2a))
XY.sign <- sort(mite.XY.fwd$order)
# Write the significant coordinates to a new object
XY.red <- mite.xy[,c(XY.sign)]

# 2. Test and forward selection of environmental variables
# -------------------------------------------------------

# Recode environmental variables 3 to 5 into dummy binary
# variables
substrate <- model.matrix(~mite.env[,3])[,-1]
shrubs <- model.matrix(~mite.env[,4])[,-1]
topo <- model.matrix(~mite.env[,5])[,-1]
mite.env2 <- cbind(mite.env[,1:2], substrate, shrubs, topo)

# Forward selection of the environmental variables
mite.env.rda <- rda(mite.h, mite.env2)
(mite.env.R2a <- RsquareAdj(mite.env.rda)$adj.r.squared)
mite.env.fwd <- forward.sel(mite.h, mite.env2,
adjR2thresh=mite.env.R2a, nperm=9999)
env.sign <- sort(mite.env.fwd$order)
env.red <- mite.env2[,c(env.sign)]
colnames(env.red)

# 3. Test and forward selection of PCNM variables
# -----------------------------------------------

# Run the global PCNM analysis on the *undetrended* mite data
mite.undet.PCNM.rda <- rda(mite.h, mite.PCNM.pos)
anova.cca(mite.undet.PCNM.rda)
# Since the analysis is significant, compute the adjusted R2
#    and run a forward selection of the PCNM variables
(mite.undet.PCNM.R2a <-
 RsquareAdj(mite.undet.PCNM.rda)$adj.r.squared)
(mite.undet.PCNM.fwd <- forward.sel(mite.h,
```

```
  as.matrix(mite.PCNM.pos),
  adjR2thresh=mite.undet.PCNM.R2a))

  # According to the R2a criterion, if we retain 12 PCNMs
  # we get a model with a R2adj slightly higher than that of the
  # complete model. This slight excess is not too serious,
  # however.

# Number of signif. PCNM
(nb.sig.PCNM <- nrow(mite.undet.PCNM.fwd))
# Identity of significant PCNMs in increasing order
(PCNM.sign <- sort(mite.undet.PCNM.fwd$order))
# Write the significant PCNMs to a new object
PCNM.red <- mite.PCNM.pos[,c(PCNM.sign)]

# 4. Arbitrary split of the significant PCNMs into broad and
# fine scale
# ------------------------------------------------------------
# Broad scale: PCNMs 1, 2, 3, 4, 6, 7, 8, 9, 10, 11
PCNM.broad <- PCNM.red[,1:10]
# Fine scale: PCNMs 16, 20
PCNM.fine <- PCNM.red[,11:12]

# 5. Mite - environment - trend - PCNM variation partitioning
# ------------------------------------------------------------

(mite.varpart <- varpart(mite.h, env.red, XY.red, PCNM.broad,
  PCNM.fine))
par(mfrow=c(1,2))
showvarparts(4)
plot(mite.varpart, digits=2)
# The default options leave the fractions with negative R2a
# empty. To display the negatives values, set the argument
# cutoff = -Inf.

# Tests of the unique fractions [a], [b], [c] and [d]
# ***************************************************
# Fraction [a], pure environmental
anova.cca(rda(mite.h, env.red, cbind(XY.red, PCNM.broad,
  PCNM.fine)))
# Fraction [b], pure trend
anova.cca(rda(mite.h, XY.red, cbind(env.red, PCNM.broad,
  PCNM.fine)))
# Fraction [c], pure broad scale spatial
anova.cca(rda(mite.h, PCNM.broad, cbind(env.red, XY.red,
   PCNM.fine)))
```

```
# Fraction [d], pure fine scale spatial
anova.cca(rda(mite.h, PCNM.fine, cbind(env.red, XY.red,
  PCNM.broad)))
 # Only the pure environmental and the pure broad scale
 # spatial fractions are significant.
```

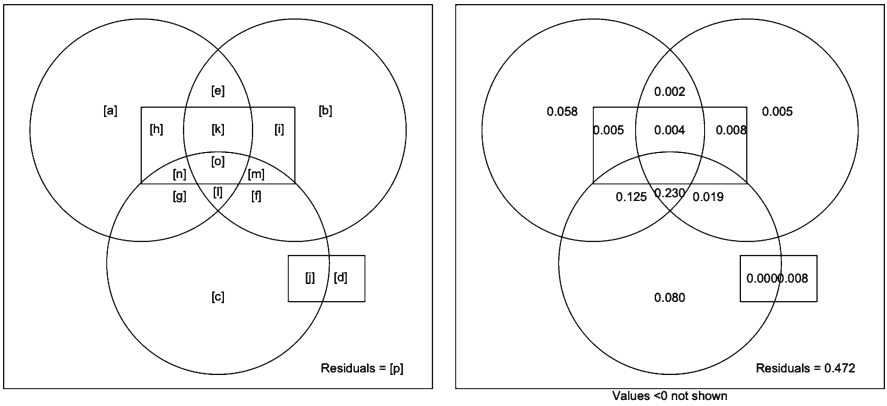

Fig. 7.7 Variation partitioning of the undetrended oribatid mite data into an environmental component (*upper left-hand circle*), a linear trend (*upper right-hand circle*), a broad scale (*lower circle*) and fine scale (*disjoined rectangles*) PCNM spatial components. The empty fractions in the plots have small negative R^2_{adj} values

When interpreting such a complex variation partitioning diagram, keep in mind that the R^2 adjustment is done for each fraction that can be fitted without resorting to partial RDA or multiple regression (here, the first 15 rows of the table of results), and that the individual fractions [a] to [p] are then computed by subtraction. Very small negative R^2_{adj} values frequently appear in this process. Negative R^2_{adj} values correspond to explanatory variables that explain less of the response variables' variation than would be expected by chance; so, for all practical purposes, they can be interpreted as zeros and neglected during interpretation, although they must be taken into account for the sum of all fractions to be 1.

The whole set of environmental and spatial variables explains 52.8% of the variation of the undetrended mite data (see the R^2_{adj} for "All" fractions). The environmental variables alone (matrix **X**1 in the partitioning results) explain 40.8% of the variation, of which a mere 5.8% is not spatially structured (fraction [a]). This fraction represents species–environment relationships associated with local environmental conditions.

The remaining fractions involving environmental and spatial variables (essentially fractions [g] and [l]) represent spatially structured environmental variation. Fraction [g] (12.5% variance explained) is common to the environmental and broad scale PCNM variables. Fraction [l] (23.0%) represents a strong spatial component that is jointly explained by the environmental variables, the *Y* coordinate of the sampling sites and the broad scale PCNM variation. This is a typical case of *induced spatial variation*, where the spatial structure of environmental factors produces a similar spatial structure in the response data. In this example, fraction [l], which represents two-thirds of that joint structure, corresponds to the linear gradient in the north–south direction of the map represented in the analysis by variable mite.xy[,2], showing that broad-scale PCNM variables can indeed model a linear gradient. On the other hand, the common fractions corresponding to the environment and the fine-scale PCNM structure ([h+k+n+o], $R^2_{adj} = -0.006\%$) is negligible.

When some variance is explained commonly by the environmental and spatial variables, one should be careful when inferring causal species–environment relationships: the correlations may be due to a direct influence of the environmental variables on the species (direct induced spatial variation), or to some unmeasured underlying process that is spatially structured and is influencing both the mite community and the environmental variables (e.g. spatial variation induced by a historical causal factor).

The variation partitioning also shows that the four sources of variation have unequal unique contributions: the environment alone ([a], 5.8%), as well as the broad scale ([c], 8.0%) variation, are significant, while the trend alone ([b], 0.5%) and the fine scale variation ([d], 0.8%) are not.

There is also some variation explained by spatial variables independently of the environment. This variation is represented by fractions [b], [c], [d], [f], [i], [j] and [m]. Together these fractions explain 12% of the variation. Most likely, some of this variation, especially at broad and medium scales, could be explained by unmeasured environmental variables, although one cannot exclude the influence of past events that could still show their marks in the mite community (Borcard and Legendre 1994). Fine scale structures are more likely explainable by spatial correlation produced by neutral biotic processes. Neutral processes include ecological drift (variation in species demography due to random reproduction and random survival of individuals due to competition, predator–prey interactions, etc.) and random dispersal (migration in animals, propagule dispersion in plants). Controlling for spatial correlation by means of PCNM variables when testing species–environment relationships is briefly addressed in Sect. 7.4.3.4.

Finally, note that the broad and fine-scale PCNM variables have a non-null intersection despite the fact that the PCNM variables are orthogonal: fraction [j+m+n+o] totals −1.7%. This occurs because other variables (environment and trend), which are not orthogonal to the PCNMs, are involved in the partitioning, and also because the variation partitioning procedure involves subtractions of R^2 that have been adjusted on the basis of different numbers of explanatory variables.

7.4.3 MEM in a Wider Context: Weights Other than Geographic Distances

7.4.3.1 Introduction

The PCNM method provides an elegant way of constructing sets of linearly independent spatial variables. Since its publication, it has gained a wide audience and has been applied in several research papers. But it is not the end of the story.

Dray et al. (2006) have greatly improved the mathematical formalism of PCNM analysis by showing that it is a particular case of a wider family of methods that they called *Moran's eigenvector maps* (MEM). They demonstrated the link between the eigenvalues of the MEM eigenvectors and Moran's spatial correlation index, *I* (7.3).

They reasoned that the relationship among sites, which is the basis for any spatial eigenvector decomposition, actually has two components: (1) a list of *links* among objects, represented by a *connectivity matrix* and (2) a matrix of *weights* to be applied to these links. In the simplest case, the weights are binary (i.e. either two objects are linked, or they are not). In more complex models, non-negative weights can be placed on the links; these weights represent the easiness of exchange (of organisms, energy, information, etc.) between the points connected by the links. For instance, link weights can be made to be inversely proportional to the squared Euclidean distance among sites.

Furthermore, Dray et al. (2006) showed that (1) by using similarities instead of distances among sites, (2) setting the relationship of the sites with themselves to null similarity and (3) avoiding a square-root standardization of the eigenvectors within the PCoA procedure, one obtains a family of flexible methods (MEM) that bear an immediate connexion with Moran's *I* and can be modulated to optimize the construction of spatial variables. The MEM method produces $n-1$ spatial variables with positive *and* negative eigenvalues, allowing the construction of a wide range of variables modelling positive and negative spatial correlation. The eigenvectors maximize Moran's *I* index, the eigenvalues being equal to Moran's *I* multiplied by a constant. Therefore, the spatial structures of the data are extracted in such a way

that the axes first optimally display the positively autocorrelated structures in decreasing order of importance, and then the negatively autocorrelated structures in increasing order.

The MEM method consists in defining two matrices describing the relationships among the sites:

- A binary **connectivity** matrix **B** defining which pairs of sites are connected (1) and which are not (0)
- A **weighting** matrix **A** providing the intensity of the connexions

The final *spatial weighting matrix* **W** results from the Hadamard (i.e. term-by-term) product of these two matrices, **B** and **A**.

The connectivity matrix **B** can be constructed on the basis of distances (by selecting a distance threshold and connecting all points that are within that distance) or by other connexion schemes, such as Delaunay triangulation, Gabriel graph or others (described by Legendre and Legendre 1998, Section 13.3). The connexion matrix can of course be customized to fit special needs – for instance, by only allowing connexions among sites along the littoral zone of a lake (not across water) or along the shoreline of an island.

Matrix **A** is not mandatory, but is often used to weight the connexions according to distance, e.g. by inverse distance or inverse squared distance, since it is ecologically realistic to assume that a process influences a community with an intensity decreasing with distance. The choice of both matrices is very important because it greatly affects the structure of the spatial variables obtained. These variables, in turn, condition the results of the spatial analysis, especially in the case of irregular sampling: "*In the case of **regular** sampling (e.g. a regular grid), structures defined by eigenvectors are roughly similar for different definitions of **W**. For **irregular** distributions of sites, however, the number of positive/negative eigenvalues and the spatial structures described by their associated eigenvectors are greatly influenced by the spatial relationships defined in **W***" (Dray et al. 2006). These authors provide the following general recommendations:

> *The choice of the spatial weighting matrix **W** is the most critical step in spatial analysis. This matrix is a model of the spatial interactions recognized among the sites, all other interactions being excluded. In some cases, a theory-driven specification can be adopted, and the spatial weighting matrix can be constructed based upon biological considerations […]. In most situations, however, the choice of a particular matrix may become rather difficult and a data-driven specification could then be applied. Under this latter approach, the objective is to select a configuration of **W** that results in the optimal performance of the spatial model.*

For data-driven model specification, the authors proposed a procedure starting with a user-defined set of possible spatial weighting matrices. For *each* candidate, one computes the MEM eigenfunctions, reorders them according to their explanatory power, enters them one by one into the model and retains the model with the lowest corrected Akaike information criterion (AIC_c). When this is done for all candidates, one retains the **W** matrix yielding the lowest AIC_c.

The AIC_c-based selection is but one possibility. One could also forward-select the MEM within each candidate model using Blanchet et al.'s (2008a) double stopping criterion and retain the model with the highest R^2_{adj}. This alternative, which had not yet been devised when the Dray et al. (2006) paper was published, addresses the concerns raised by these authors in their conclusion about the drawbacks of forward selection procedures.

7.4.3.2 MEM Analysis of the Mite Data

Dray et al. (2006) used the oribatid mite data to illustrate MEM analysis. As an example, we duplicate their analysis, exploring some choices along the steps of the method. Several packages are used. The following example is based on Stéphane Dray's tutorial on MEM analysis, with our thanks to the author. It relies heavily on the **`spacemakeR`** package that Dray devised especially for this purpose.

The **`spacemakeR`** functions used below should make model selection relatively easy. Of course, the final result depends upon a proper choice of a class of models. The function **`test.W()`** is particularly useful as it combines the construction of MEM variables and model selection; examine the help file of that function.

We experiment with three classes of models:

- The first class is based on Delaunay triangulation with binary weights.
- The second class starts from the same connectivity matrix, to which weights are added. The weighting function is based on Euclidean distances among the sites: $f_2 = 1 - (d/d_{max})^{\alpha}$ where d is a distance value and d_{max} is the maximum value in the distance matrix.
- The third class evaluates a series of models based on a range of distances around the points. All pairs of points within the distance considered are linked, the others not. What should the range of distances be? This can be assessed by means of a multivariate variogram of the response data. Variograms are plots of semivariances against distance classes. Semivariance is a distance-dependent measure of variation, which is used in the same way as in the correlograms presented earlier (e.g. Bivand et al. 2008). A variant of this approach will weight the links by the

function of inverse distance that was used in the second model class above. This last variant duplicates the results presented in the Dray et al. (2006) paper.

First and second classes of MEM models: unweighted (binary) and distance-weighted Delaunay triangulation.

```
# MEM analysis of the detrended oribatid mite data
# ************************************************

# Selection of an optimal spatial weighting matrix
# ************************************************

# 1. Search based on Delaunay triangulation.
#    We use mite.h.det as response data and mite.del as Delaunay
#    triangulation data.
#    No weighting matrix (binary weights); function test.W
#    selects among the MEM variables constructed on the basis of
#    the Delaunay triangulation.

# Delaunay triangulation
(mite.del <- tri2nb(mite.xy))
mite.del.res <- test.W(mite.h.det, mite.del)

  # The screen output says that the best model has an AICc value
  # of -94.2 and is based on 7 MEM variables.

# Summary of the results for the best model
summary(mite.del.res$best)

# Unadjusted R^2 of best model
# This line returns the R^2 of the model with the smallest AICc
(R2.del <-
 mite.del.res$best$R2[which.min(mite.del.res$best$AICc)])

# Adjusted R^2 of best model (n = 70 and m = 7)
RsquareAdj(R2.del,70,7)


# 2. Delaunay triangulation weighted by a function of distance.
#    Distances are ranged to maximum 1, and raised to power
#    alpha

f2 <- function(D, dmax, y) { 1 - (D/dmax)^y }

# Largest Euclidean distance on links belonging to the Delaunay
# triangulation
```

```
max.d1 <- max(unlist(nbdists(mite.del, as.matrix(mite.xy))))
# Power is set from 2 to 10
mite.del.f2 <- test.W(mite.h.det, mite.del, f=f2, y=2:10,
dmax=max.d1, xy=as.matrix(mite.xy))

  # The screen output says that the best model has an AICc value
  # of -95.4 and is based on 6 MEM variables.

# Unadjusted R^2 of best model
(R2.delW <-
  mite.del.f2$best$R2[which.min(mite.del.f2$best$AICc)])
# Adjusted R^2 of best model (n = 70 and m = 6)
RsquareAdj(R2.delW,70,6)
```

Third class of MEM models: connectivity matrix based on distances.

```
# 3a. Connectivity matrix based on a distance (radius around
#     points)

# Assessment of the relevant distances, based on a multivariate
# variogram of the detrended mite data, with 20 distance
# classes.

(mite.vario <- variogmultiv(mite.h.det, mite.xy, nclass=20))
plot(mite.vario$d, mite.vario$var, ty='b', pch=20,
  xlab="Distance", ylab="C(distance)")
```

The multivariate variogram is presented in Fig. 7.8. It consists in the sum of univariate variograms computed for all species. The variance increases from 0 to 4 m. Since the shortest distance to keep all sites connected is 1.0111 m (see PCNM analysis), we explore a range of ten evenly distributed distances ranging from this threshold up to 4.0 m (i.e. approximately four times the threshold, as in PCNM analysis).

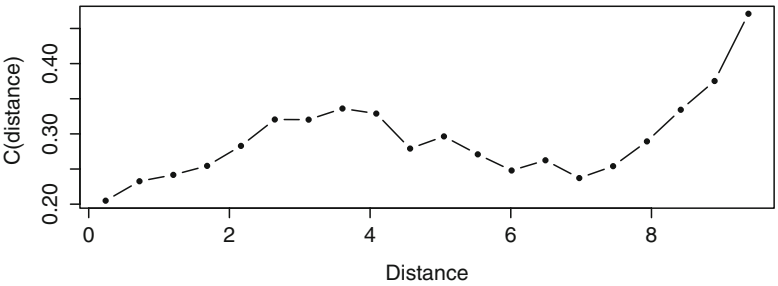

Fig. 7.8 Multivariate variogram of the detrended oribatid mite data. Twenty distance classes

```
# Construction of 10 neighbourhood matrices (class nb)
# Vector of 10 threshold distances
(thresh10 <- seq(give.thresh(dist(mite.xy)), 4, le=10))

# Create 10 neighbourhood matrices.
# Each matrix contains all connexions with lengths ≤ the
# threshold value
list10nb <- lapply(thresh10, dnearneigh, x=as.matrix(mite.xy),
 d1=0)

# Display an excerpt of the first neighbourhood matrix
print(listw2mat(nb2listw(list10nb[[1]], style="B"))[1:10,1:10],
 digits=1)

# Now we can apply the function test.W() to the 10 neighbourhood
# matrices. There are no weights on the links.
mite.thresh.res <- lapply(list10nb, test.W, Y=mite.h.det)

# Lowest AICc, best model, threshold distance of best model
mite.thresh.minAIC <- sapply(mite.thresh.res, function(x)
min(x$best$AICc, na.rm=TRUE))
# Smallest AICc (best model among the 10)
min(mite.thresh.minAIC)
# Number of the model among the 10
which.min(mite.thresh.minAIC)
# Truncation threshold (distance)
thresh10[which.min(mite.thresh.minAIC)]
 # Identify the best model, that is, the model with the lowest
```

```
 # AICc. What is the range of distances in that model? How many
 # MEMs were selected?
```

Hint **dnearneigh()** *requires two geographic dimensions. Add a constant column (e.g. a column of 1) if you only have one dimension, e.g. a transect or a time series.*

This result is more interesting than that of the weighted Delaunay MEM. The AICc of the best model, obtained with a threshold of 2 m, is −100.6 with a model consisting of five MEM variables only. Let us see if we could improve this result by weighting the connexions by an inverse distance function.

```
# 3b. Variant: same as above, but connections weighted by the
#     complement of the power of the distances, 1-(d/dmax)^y

mite.thresh.f2 <- lapply(list10nb, function(x) test.W(x,
  Y=mite.h.det, f=f2, y=2:10, dmax=max(unlist(nbdists(x,
  as.matrix(mite.xy)))), xy=as.matrix(mite.xy)))

# Lowest AIC, best model
mite.f2.minAIC <- sapply(mite.thresh.f2, function(x)
min(x$best$AICc, na.rm=TRUE))

# Smallest AICc (best model among the 10
min(mite.f2.minAIC))

# Number of the model among the 10
(nb.bestmod <- which.min(mite.f2.minAIC))

# Actual dmax of best model
(dmax.best <- mite.thresh.f2[nb.bestmod][[1]]$all[1,2])
```

Hint *The actual d_{max} value found by the function is often smaller than the d_{max} provided to the function by means of the vector of user-selected threshold distances, because the output of the function shows the largest actual distance within the limit provided by each threshold value. In the example, the sixth value in vector* `thresh10`*, which contains the list of user-selected threshold distances, is 2.671639. There is no such distance in the mite geographic distance matrix; the function found that the largest distance smaller than or equal to that threshold is 2.668333.*

With an AIC_c of −102.7, this is the best result of all our attempts in terms of AIC_c. We can therefore extract this champion model, which contains seven MEM variables, from the output object:

```
# Extraction of the champion MEM model
# ************************************
mite.MEM.champ <-
 unlist(mite.thresh.f2[which.min(mite.f2.minAIC)],
 recursive=FALSE)
summary(mite.MEM.champ)

mite.MEM.champ$best$values  # Eigenvalues
mite.MEM.champ$best$ord     # MEM variables by order of added R2

# MEM variables selected in the best model
MEMid <-
 mite.MEM.champ$best$ord[1:which.min(mite.MEM.champ$best$AICc)]
sort(MEMid)
MEM.all <- mite.MEM.champ$best$vectors
MEM.select <- mite.MEM.champ$best$vectors[, sort(c(MEMid))]
colnames(MEM.select) <- sort(MEMid)
# Unadjusted R2 of best model
R2.MEMbest <-
 mite.MEM.champ$best$R2[which.min(mite.MEM.champ$best$AICc)]
# Adjusted R2 of best model
RsquareAdj(R2.MEMbest, nrow(mite.h.det), length(MEMid))

# Plot the links using the function plot.links()
plot.links(mite.xy, thresh=dmax.best)

# Maps of the 7 significant MEM variables
# ***************************************
par(mfrow=c(2,4))
for(i in 1:ncol(MEM.select)){
s.value(mite.xy,MEM.select[,i], sub=sort(MEMid)[i], csub=2)
}
```

The very best MEM model among those tested contains seven MEM variables; four of them are positively spatially correlated (1, 2, 3, 6) and three negatively (9, 11, 57). Interestingly enough, the same result is found by redoing the selection using the forward selection procedure proposed by Blanchet et al. (2008a), with two separate forward selections for the MEM with positive and negative spatial correlation and using only the α value as stopping criterion for the negative MEM.

RDA of the detrended mite data with the seven MEM variables can be computed in a similar fashion as in the PCNM analysis:

```
# RDA of the mite data constrained by the 7 MEM retained, using
# vegan
# ************************************************************
(mite.MEM.rda <- rda(mite.h.det~., as.data.frame(MEM.select)))
(mite.MEM.R2a <- RsquareAdj(mite.MEM.rda)$adj.r.squared)
anova.cca(mite.MEM.rda)
axes.MEM.test <- anova.cca(mite.MEM.rda, by="axis")
# Number of significant axes
(nb.ax <- length(which(axes.MEM.test[,5] <= 0.05)))

# Plot maps of the two significant canonical axes
mite.MEM.axes <- scores.cca(mite.MEM.rda, choices=c(1,2),
 display="lc", scaling=1)
par(mfrow=c(1,2))
s.value(mite.xy, mite.MEM.axes[,1])
s.value(mite.xy, mite.MEM.axes[,2])
```

The R^2_{adj} of the MEM and PCNM models are similar (approximately 0.30), but the PCNM model requires 11 variables to reach this value and is thus less parsimonious than the MEM model. The graphical result (not reproduced here) closely resembles that of the PCNM analysis, showing that the structures revealed by the two analyses are the same.

For the sake of comparison with the PCNM variables, one can plot the seven MEM variables on a map of the sampling area:

```
# Maps of the 7 significant MEM variables
# ***************************************

par(mfrow=c(2,4))
for(i in 1:ncol(MEM.select)){
s.value(mite.xy, MEM.select[,i], sub=sort(MEMid)[i], csub=2)
}
```

The MEMs look similar to the PCNMs at first glance, but a closer look shows that they differ more than one would have expected. Indeed, the two groups of spatial variables are rather weakly correlated:

```
# Correlation of the retained MEM and PCNM variables
# --------------------------------------------------
cor(MEM.select, PCNM.red)
```

These MEM results show that the whole process of selecting and fine-tuning a spatial model, cumbersome as it may seem, can end up with an efficient and parsimonious set of spatial variables.

Note also that **`test.W()`** searches the best model, including MEMs with positive *and* negative spatial correlation. To identify the two groups of eigenfunctions and, if desired, run separate analyses, one must compute Moran's *I* of the selected MEMs found in object `MEM.select`. Another approach could be to compute Moran's *I* of all MEM candidates of the champion model (object `MEM.all`) and run separate forward selections on MEMs with positive and negative spatial correlation.

7.4.3.3 Other Types of Connectivity Matrices

In special cases, when one has a specific spatial model in mind, it is useless to go through the automatic procedure shown above, which finds the best model among multiple possibilities. The present subsection shows how to construct connectivity matrices of several types by hand.

Apart from the Delaunay triangulation used in the example above, the package **`spdep`** offers many possibilities for the definition of connectivity matrices. The ones constructed below and shown in Fig. 7.9 are described in Legendre and Legendre (1998), Section 13.3. They are presented in decreasing order of connectivity and are nested (i.e. the edges (connexions) of a minimum spanning tree are all included in the relative neighbourhood graph and so on).

Depending on the context (hypotheses, data), researchers need connecting schemes that are more or less dense. Some reasons may be technical (e.g. the use of a minimum spanning tree in the PCNM procedure presented in Sect. 7.4.2). Ecological reasons include topographical structure of the sampling area (including possible barriers), dispersion ability of organisms, permeability of some types of substrates and so on.

```
# Connectivity matrices in decreasing order of connectivity
# Delaunay triangulation (as in the previous example)
mite.del <- tri2nb(mite.xy)
# Gabriel graph
mite.gab <- graph2nb(gabrielneigh(as.matrix(mite.xy)), sym=TRUE)
# Relative neighbourhood
mite.rel <- graph2nb(relativeneigh(as.matrix(mite.xy)),
 sym=TRUE)
# Minimum spanning tree
mite.mst <- mst.nb(dist(mite.xy))
 # All these neighbourhood matrices are stored in objects of
 # class nb.
```

```
# Plots of the connectivity matrices
par(mfrow=c(2,2))
plot(mite.del, mite.xy, col="red", pch=20, cex=1)
title(main="Delaunay triangulation ")
plot(mite.gab, mite.xy, col="purple", pch=20, cex=1)
title(main="Gabriel graph")
plot(mite.rel, mite.xy, col="dark green", pch=20, cex=1)
title(main="Relative neighbourhood")
plot(mite.mst, mite.xy, col="brown", pch=20, cex=1)
title(main="Minimum spanning tree")
```

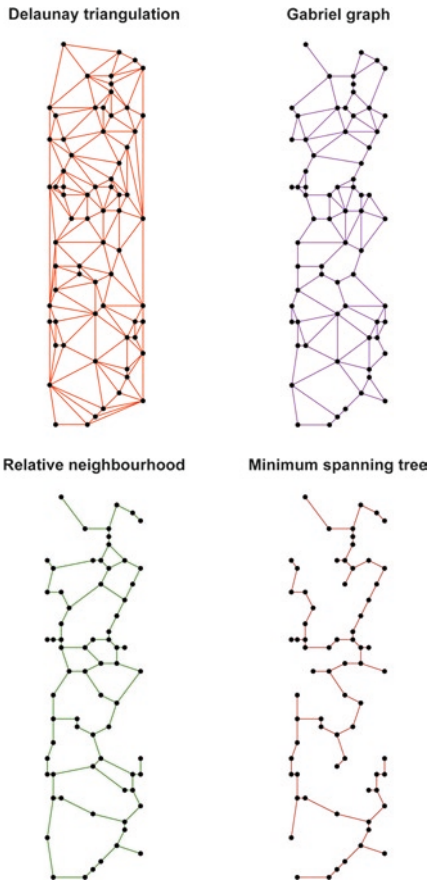

Fig. 7.9 Four types of connectivity matrices applied to the oribatid mite sampling plot. They are presented in decreasing order of connectivity. The links in each model are a subset of the links in the previous one

Some of these matrices may contain unwanted links (for instance, along the borders of the areas). These can be edited either interactively or by command lines:

```
# Link editing
# ************

# 1. Interactive:
plot(mite.del, mite.xy, col="red", pch=20, cex=2)
title(main="Delaunay triangulation ")

mite.del2 <- edit.nb(mite.del,mite.xy)

  # To delete a link, click on its two nodes. Follow on-screen
  # instructions.
  # Wait until you have finished editing before entering the next
  # command line.

# 2. Alternately, links can also be removed by command lines,
# after having converted the nb object into an editable matrix:

mite.del.mat <- nb2mat(mite.del, style="B")
# Remove connection between objects 23 and 35:
mite.del.mat[23,35] <- 0
mite.del.mat[35,23] <- 0
# Back-conversion into nb object:
mite.del3 <- neig2nb(neig(mat01=mite.del.mat))

plot(mite.del3, mite.xy)

# Example: list of neighbours of site 23 for the Delaunay
triangulation:
mite.del[[23]]   # Before editing
mite.del2[[23]]  # After interactive editing
mite.del3[[23]]  # After command line editing
```

The following code shows how to construct connectivity matrices based on a distance: pairs of sites within a given radius are connected, the others not.

```
# Connectivity matrix based on a distance (radius around points)
# Using the same truncation distance dmin as in the PCNM
# example: dmin = 1.011187

mite.thresh4 <- dnearneigh(as.matrix(mite.xy), 0, dmin*4)
nb2mat(mite.thresh4)[1:10, 1:10] # Display of some values
```

```
# Using a shorter distance (1*dmin, 2*dmin)

mite.thresh1 <- dnearneigh(as.matrix(mite.xy), 0, dmin*1)
mite.thresh2 <- dnearneigh(as.matrix(mite.xy), 0, dmin*2)

# Using a longer distance
mite.thresh8 <- dnearneigh(as.matrix(mite.xy), 0, dmin*8)

# Plot of some connectivity matrices
par(mfrow=c(1,2))
plot(mite.thresh1, mite.xy, col="red", pch=20, cex=0.8)
title(main="1 * dmin")
plot(mite.thresh4, mite.xy, col="red", pch=20, cex=0.8)
title(main="4 * dmin")

  # The 1*dmin version shows one disconnected point (7). To avoid
  # such problems, use a slightly larger dmin value. In this case
  # 1.0011188 is enough. The 4*dmin version is very crowded. A
  # lot of links are possible within a little more than 4 meters
  # around each point.
```

These connectivity matrices belong to class "nb". To use them further, we need to convert them into another class called "listw". The function doing this conversion is called **nb2listw()**.

In the simplest case, one of the binary matrices above can be directly converted as follows (including a matrix-class representation of the connectivity matrix for convenience, using function **listw2mat())**:

```
# Conversion of a "nb" object into a "listw" object
# Example: mite.thresh4 created above. "B" is for "binary"

mite.thresh4.lw <- nb2listw(mite.thresh4, style="B")
print(listw2mat(mite.thresh4.lw)[1:10, 1:10], digits=1)
```

This binary (unweighted) matrix could be used directly to create MEM variables using the function **scores.listw()**; see below.

Now, if you want to apply weights (matrix **A**) onto a binary matrix on the basis of Euclidean distances, you need two additional steps: (1) replace all values "1" in the connectivity matrix by the corresponding Euclidean distances [function **nbdists()**] and (2) define weights as a function of inverse distances in this example (weights may be different in other examples):

```
# Creation of a spatial weighting matrix W = Hadamard product of
# B and A
# ------------------------------------------------------------

# Replace "1" by Euclidean distances in the connectivity matrix
mite.thresh4.d1 <- nbdists(mite.thresh4, as.matrix(mite.xy))

# Weights as function of inverse distance
mite.inv.dist <- lapply(mite.thresh4.d1, function(x) 1-
x/max(dist(mite.xy)))

# Creation of spatial weighting matrix W. Argument "B" stands
# for "binary" but concerns the links themselves, not their
# weights
mite.invdist.lw <- nb2listw(mite.thresh4, glist=mite.inv.dist,
  style="B")
print(listw2mat(mite.invdist.lw)[1:10, 1:10], digits=2)

  # All nonzero links have been replaced by weights proportional
  # to the inverses of the Euclidean distances between the
  # points.
```

Now, it is time to compute the MEM spatial variables. This can be done by function **`scores.listw()`** of the package **`spacemakeR`**. We do it on the weighted distance matrix created above. The MEMs are then tested for spatial correlation (Moran's *I*).

```
# Computation of MEM variables (from an object of class listw)
# ------------------------------------------------------------

mite.invdist.MEM <- scores.listw(mite.invdist.lw, echo=TRUE)
summary(mite.invdist.MEM)
mite.invdist.MEM$values
barplot(mite.invdist.MEM$values)

# Test of Moran's I of each eigenvector
(mite.MEM.Moran <- test.scores(mite.invdist.MEM,
mite.invdist.lw, 999))
# MEM with significant spatial correlation
which(mite.MEM.Moran[,2] <= 0.05)
length(which(mite.MEM.Moran[,2] <= 0.05))

  # 31 to 33 MEM variables have significant Moran's I
  # coefficients; among these only MEM 1, 2, 3, 4, 5, 6 and 7
  # (significance of 6 and 7 depending on the permutation result)
  # have positive spatial correlation.
```

```
# Store the MEM vectors in new objects
# All MEM
mite.invdist.MEM.vec <- mite.invdist.MEM$vectors
# MEM with positive spatial correlation
MEM.Moran.pos <- which(mite.MEM.Moran[,1] > -
 1/(nrow(mite.invdist.MEM$vectors)-1))
mite.invdist.MEM.pos <- mite.invdist.MEM.vec[,MEM.Moran.pos]
# MEM with positive *and significant* spatial correlation
MEM.Moran.pos.sig <-
 MEM.Moran.pos[which(mite.MEM.Moran[MEM.Moran.pos,2] <= 0.05)]
mite.invdist.MEM.pos.sig <-
 mite.invdist.MEM.vec[,MEM.Moran.pos.sig]
```

To show that the MEM variables are directly related to Moran's *I*, let us draw a scatterplot of the MEM eigenvalues and their corresponding Moran's *I*:

```
# Plot of MEM eigenvalues vs Moran's I
plot(mite.invdist.MEM$values, mite.MEM.Moran$stat,
 ylab="Moran's I", xlab="Eigenvalues")
text(-1, 0.5, paste("Correlation=", cor(mite.MEM.Moran$stat,
 mite.invdist.MEM$values)))
```

As in the case of the automatic model selection presented before, these MEM variables can now be used as explanatory variables in RDA or multiple regression, in the same way as PCNM variables were.

These are but several examples. We suggest that you explore the manual of the package **spacemakeR**, which presents in great detail the use of many options to construct, present and use various types of connectivity matrices.

7.4.3.4 Controlling for Spatial Correlation Using MEM

Peres-Neto and Legendre (2010) explored the potential use of polynomials and MEM eigenfunctions to control for spatial correlation in statistical tests. Their main conclusion is that MEM, but not polynomials, can adequately achieve this goal. They propose the following procedure: (1) Test for the presence of a spatial structure using all positive MEM variables. (2) If the global test is significant, proceed to forward-select MEM variables, but (a novelty) do this individually for each species, and retain the union of the MEMs selected, i.e. retain all MEMs that have been selected at least once. (3) Proceed to test the species–environment relationships, controlling for spatial correlation by placing the retained MEM variables in a matrix of covariables. The authors demonstrate that this procedure yields correct type I error for tests of significance in linear models, in the presence of spatial correlation.

7.4.3.5 MEM on Sampling Designs with Nested Spatial Scales

The hierarchical structure of many natural entities (e.g. metapopulations or metacommunities; landscapes at various scales) sometimes calls for nested sampling designs. An example is found in Declerck et al. (2011), where the authors studied cladoceran metacommunities in wetland pools found in several valleys of the High Andes. The authors analysed the metacommunity spatial structure among and within valleys by means of a two-level spatial model. The among-valley component was modelled by a set of dummy variables. For the within-valley component, where several pools had been sampled in each valley, a set of MEM variables was computed for each valley. All dummy and MEM variables were assembled into a single staggered matrix. The MEM variables were arranged in blocks corresponding to each valley. Within each block, all pools belonging to other valleys received the value 0, in a way similar to the one presented in Appendix C of Legendre et al. (2010) in the context of space–time analysis. Declerck et al. (2011) provide a function called **`create.MEM.model()`** to construct the staggered spatial matrix from a set of Cartesian coordinates and information about the number of groups and number of sites per group. That function is also available in the electronic material accompanying this book (see Chap. 1).

7.4.4 MEM with Positive or Negative Spatial Correlation: Which Ones Should Be Used?

In the course of the examples above, PCNM and MEM eigenfunctions have been produced, some with positive and some with negative spatial correlation. The question therefore arises: should one use all the (significant) eigenfunctions as explanatory variables in the following regression or canonical analyses, or only those that model positive spatial correlation?

There is no single answer to this question. Ecologically speaking, one is generally more interested in features that are positively correlated at various ranges, simply because they are the signature of contagious processes that are frequent in nature. On the other hand, our experience shows that with real data the significant and negatively correlated variables are either related to very local, almost "accidental" data structures, or they belong to the pure spatial fraction of variation in partitioning, i.e. they correspond to biotic interactions. If these are of interest, then all eigenfunctions should be considered in the analyses.

The original PCNM procedure generates a maximum of $2n/3$ eigenfunctions (n = number of sites), with roughly the first $n/2$ modelling positive spatial correlation, so a forward selection procedure including all variables can be conducted with the Blanchet et al. (2008a) double stopping criterion, which involves the computation

of the R^2_{adj} of the global analysis. In the generalized MEM framework, this is not possible because this method produces $n-1$ spatial variables, which saturate the regression model if they are all considered together. This is why Blanchet et al. proposed to run separate selections on the MEM with positive and negative eigenvalues (usually, the first and the second half of the eigenfunctions), and then apply the Šidák (1967) correction to the probability values: $P_S = 1-(1-P)^k$ where P is the p value to be corrected and k is the number of tests (here $k=2$).

7.4.5 Asymmetric Eigenvector Maps: When Directionality Matters

7.4.5.1 Introduction

The PCNM and MEM analyses presented above are designed for situations where the physical processes generating the response structures (e.g. in communities) do not present any directionality. In other words, the influence of any given point on its surroundings does not depend on the direction.

There are other situations, however, where directionality matters. The most obvious one is the cases of streams or rivers. Consider community effects driven by current: the physical process is geographically asymmetrical, the influence of a site onto another following an upstream–downstream direction. Colonization of the stream network by fish from the river mouth represents a different process, which follows the opposite direction. PCNM or MEM variables are computed on distance or connectivity matrices, where no directionality is specified. Therefore, information about directionality is lost and the modelling, although adequate to reveal major spatial structures, does not exploit all the potential of directional data. Trends do not have to be extracted from the data prior to asymmetric eigenvector maps (AEM) analysis because directional processes are expected to produce trends in the response data; so, a trend is a part of the response data that one wants to model in AEM analysis.

This is the reason why Blanchet et al. (2008b) developed the AEM modelling method. AEM is an eigenfunction-based technique that uses information about the direction of the physical process, plus the same information as MEM (spatial coordinates of sites, connexion diagram, optional weights) if needed. It works best on tree-like structures like river networks or on two-dimensional sampling designs like series of cross-river traps or sampling sites located in a large river or marine current. Depending on the process under study, the origin(s) or root(s) in a river network may be located upstream (e.g. flow of dissolved chemical substances, plankton dispersal) or downstream (fish invasion routes).

For spatial transects or time series, AEM and MEM regression and canonical models are very similar, and in most cases they explain the response data with similar (although not strictly equal) R^2. The AEM eigenfunctions are cosine-like, just like MEM eigenfunctions, although the AEMs have longer wavelengths than MEMs along transects. If the n observations are regularly spaced along the transect and the sampling interval is s, the wavelength λ_i of the AEM with rank i is $\lambda_i = 2ns/i$. AEM analysis should be preferred when modelling gradients and other spatial structures generated by directional physical processes.

AEM analysis was devised for cases where *physical* forces drive the communities in such a way that the causal relationships are directional. This is not the same as a simple ecological gradient, where an ecological factor is spatially structured but the communities can still interact in any direction. In the latter case, PCNM and MEM modelling are appropriate.

7.4.5.2 Principle of the Method

The basic piece of information needed is a table, where each site is described by the connexions (hereafter called "edges", following graph-theory vocabulary) that it has with other sites located in the direction of the root(s) or origin(s) of the directional structure. The result is a rectangular sites-by-edges table **E**, where the sequence of edges connecting each site to the "root" of the network receive code "1" and the others get code "0".

Legendre and Legendre (1998, Section 1.5.7) give an example for fish dispersal from the river mouth in a group of lakes interconnected by a river arborescence. In other cases, for instance a two-dimensional grid consisting of rows of sampling devices placed across a large river or a marine current at regular or irregular intervals, each sampling point may influence (and hence be connected to) the one directly downstream of it, plus the two adjacent to the latter. If the process is assumed to originate upstream, an imaginary point "0" is created upstream of the sampling area, representing the root of the process, with connexions to each of the points in the first row of sites. All edges present in the network are numbered. In table **E**, the rows (i) are the sites and the columns (j) are the edges. The construction rule for AEM is that $E(i,j) = 1$ for all edges j connecting site i to the root (or site 0) of the graph; otherwise, $E(i,j) = 0$.

The edges (columns) of table **E** may be weighted if deemed necessary, e.g. if the transmission of the directional effects are supposed to be easier through some paths than others.

The next step consists in transforming table **E** into eigenfunctions. This can be done in different ways, but the simplest is to compute a PCA of table **E** and use

the matrix of principal components as explanatory variables. The AEM method produces $n-1$ eigenvectors with positive eigenvalues and none with negative eigenvalues. The corresponding eigenfunctions, however, are also divided in two groups depicting positive or negative spatial correlation so that the selection of significant variables must also be run separately for these two groups, in the same way as for MEM variables.

A more detailed explanation about AEM construction is provided by Blanchet et al. (2008b). The authors address the various issues related to edge definition and weighting, which can greatly influence the results of AEM analysis. Blanchet et al. (2011) present three applications to real ecological data.

As a first example, let us construct a fictitious set of AEM variables based on the river arborescence shown by Legendre and Legendre (1998, Section 1.5.7). This example shows how to construct AEM variables in the simplest case, when one can easily produce a matrix of edges by hand. The construction is done by function **aem()** of the package **AEM**.

```
# AEM analysis
# ************

# Coding of a river arborescence.
# See Legendre and Legendre (1998, p. 47).
lake1 <- c(1,0,1,1,0,0,0,0)
lake2 <- c(1,0,1,0,0,0,0,0)
lake3 <- c(1,1,0,0,0,0,0,0)
lake4 <- c(0,0,0,0,1,0,1,1)
lake5 <- c(0,0,0,0,0,1,1,1)
lake6 <- c(0,0,0,0,0,0,0,1)
arbor <- rbind(lake1, lake2, lake3, lake4, lake5, lake6)

# AEM construction
(arbor.aem <- aem(binary.mat=arbor))
arbor.aem.vec <- arbor.aem$vectors

# AEM eigenfunctions can also be obtained directly by singular
# value decomposition (function svd()), which is what the
# function aem() does:
arbor.c = scale(arbor, center=TRUE, scale=FALSE)
arbor.svd = svd(arbor.c)
# Singular values of the previous construction
arbor.svd$d[1:5]
# AEM eigenfunctions of the previous construction
arbor.svd$u[,1:5]
```

Let us now construct AEM variables in a case where the number of data points and edges is too large to allow the use of the simple procedure presented above. The sampling design consists of ten cross-river transects with four traps per transect, and the edges are weighted proportional to inverse squared distance (Fig. 7.10). The procedure involves function **`cell2nb()`** of the package **`spdep`** to construct a list of neighbours from a grid of predefined dimensions.

```
# Coding of sampling design: 10 cross-river transects, 4 traps
# per transect. Edges weighted proportional to inverse squared
# distance.

# X-Y coordinates
xy <- cbind(1:40, expand.grid(1:4, 1:10))

# Object of class nb (spdep) containing links of chess type
# "queen"
nb <- cell2nb(4, 10, "queen")

# Site-by-edges matrix (produces a fictitious object "0")
edge.mat <- build.binary(nb, xy)

# Matrix of Euclidean distances
D1.mat <- as.matrix(dist(xy))

# Extract the edges, remove the ones directly linked to site 0
edges.b <- edge.mat$edges[-1:-4,]

# Construct a vector giving the length of each edge
length.edge <- vector(length=nrow(edges.b))
for(i in 1:nrow(edges.b)){
 length.edge[i] <- D1.mat[edges.b[i,1], edges.b[i,2]]
}

# Weighting of edges based on inverse squared distance
weight.vec <- 1-(length.edge/max(length.edge))^2

# Construction of AEM eigenfunctions from edge.mat,
# of class build.binary
example.AEM <- aem(build.binary=edge.mat, weight=weight.vec,
 rm.link0=TRUE)
example.AEM$values
ex.AEM.vec <- example.AEM$vectors
```

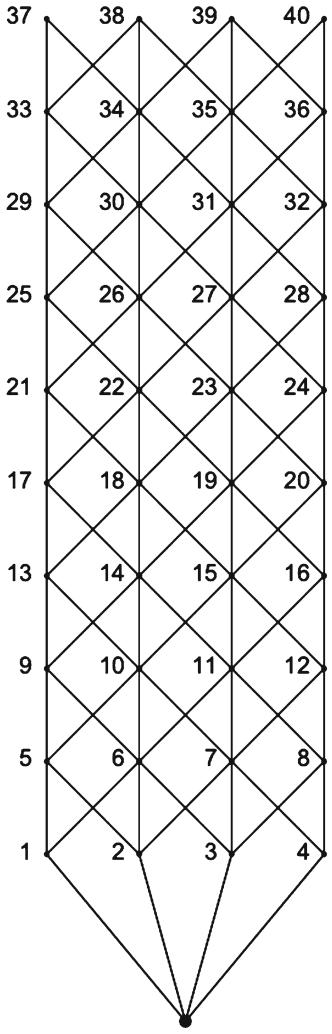

Fig. 7.10 Fictitious directional sampling design for AEM analysis: ten rows of four capture devices along a stream. A site "0" has been added upstream (*bottom* of the figure) to set the direction of the flow

Let us now construct a set of five fictitious species observed at these 40 sites:

```
# Construction of 5 fictitious species

# Two randomly distributed species
sp12 <- matrix(trunc(rnorm(80, 5, 2), 0), 40)
```

```
# One species restricted to the upper half of the stream
sp3 <- c(trunc(rnorm(20, 8, 2.5), 0), rep(0, 20))

# One species restricted to the left-hand half of the transect
sp4 <- t(matrix(c(trunc(rnorm(20, 8, 3), 0), rep(0, 20)), 10))
sp4 <- c(sp4[,1], sp4[,2], sp4[,3], sp4[,4], sp4[,5], sp4[,6],
sp4[,7], sp4[,8], sp4[,9], sp4[,10])

# One species restricted to the 4 upper left-hand sites
sp5 <- c(4, 7, 0, 0, 3, 8, rep(0, 34))

sp <- cbind(sp12, sp3, sp4, sp5)
```

We are ready to proceed with the AEM analysis, using the first half (20, with positive spatial correlation) of the AEM variables generated earlier. Note that four out of five species have a random component, so the actual result of the following AEM analysis will vary from run to run.

```
# Global AEM analysis with 20 first AEM variables (for
# computation of R2a)
# ************************************************
AEM.20 <- rda(sp ~ ., as.data.frame(ex.AEM.vec[,1:20]))
R2a.AEM <- RsquareAdj(AEM.20)$adj.r.squared

AEM.fwd <- forward.sel(sp,ex.AEM.vec, adjR2thresh=R2a.AEM)
(AEM.sign <- sort(AEM.fwd[,2]))
# Write significant AEM in a new object
AEM.sign.vec <- ex.AEM.vec[,c(AEM.sign)]
(sp.AEMsign.rda <- rda(sp ~ .,
 data=as.data.frame(AEM.sign.vec)))  # RDA with signif. AEM
anova.cca(sp.AEMsign.rda)
AEM.rda.axes.test <- anova.cca(sp.AEMsign.rda, by="axis")
# Number of significant axes
(nb.ax.AEM <- length(which(AEM.rda.axes.test[,5] <= 0.05)))

# Plot of the significant canonical axes
AEM.rda.axes <- scores.cca(sp.AEMsign.rda, choices=c(1,2),
 display="lc", scaling=1)
par(mfrow=c(1, nb.ax.AEM))
for(i in 1:nb.ax.AEM) s.value(xy[,c(2,3)], AEM.rda.axes[,i])
```

In most of the runs, this small example shows that AEM analysis reveals the patterns formed by the species present only in the upper half of the stream, as well as the left–right contrast created by the species present only in the left-hand part of

the stream. The pattern of the more restricted species # 5 is less obvious. A PCNM analysis (not shown here) reveals less of the structure and has a lower R^2_{adj}. This stresses the importance of modelling directional processes adequately.

7.5 Another Way to Look at Spatial Structures: Multiscale Ordination

7.5.1 Principle

Wagner (2003, 2004) took an entirely different path towards integration of spatial information into canonical ordination. Under the well-known argument that autocorrelated residuals can alter the results of statistical tests, she introduced geostatistical methods to devise diagnostic tools allowing the partitioning of ordination results into distance classes, the distinction between induced spatial dependence and spatial autocorrelation, and the use of variograms to check important assumptions, such as independence of residuals and stationarity. The principle of multiscale ordination (MSO) is the following[3]:

- Analyse the species by RDA. The explanatory variables can be of any kind (environmental, spatial, …). This provides the matrix of fitted values and its eigenvectors, as well as the matrix of residuals and its eigenvectors.
- By way of a variogram matrix computed for the fitted values, obtain the spatial variance profiles of the canonical ordination axes (see below).
- By way of a variogram matrix computed for the residuals, obtain the spatial variance profiles of the residual ordination axes.
- Plot the variograms of the explained and residual variances. Permutation tests may be used to identify significant spatial correlation in the distance classes.

A variogram matrix is a three-dimensional array containing a separate variance–covariance matrix for each distance class (Wagner, 2003, Fig. 2). The diagonal of each matrix quantifies the contribution of the corresponding distance class to the variance of the data. MSO computes a variogram matrix on the fitted values of a constrained ordination, thereby allowing its spatial decomposition. Multiplying this variogram matrix with the matrix of constrained eigenvectors provides the spatial decomposition of each eigenvalue (variance profiles). The same holds for the residuals.

[3] Wagner (2004) describes the method for CCA, but the principle is the same for RDA.

7.5.2 Application to the Mite Data: Exploratory Approach

Let us use the oribatid mite data as an example. Wagner (2004) also used these data, but in a CCA context so that the results will differ. MSO can be computed using function **mso()** of the package **vegan**. This function uses a result object produced by functions **cca()** or **rda()**, plus the table of geographical coordinates and a value for the interval size (argument "grain") of the distance classes of the variograms. The first example applies MCO in the exploratory way proposed by Wagner. An MSO plot of direct ordination can show whether the spatial structure in the response data can be explained by the explanatory (environmental) variables alone. In such a case, no detrending is necessary (H. Wagner, pers. comm.), but the confidence interval of the variogram is indicative only, since a variogram should be computed on stationary data.

Hereunder, MSO is run using the RDA result of the Hellinger-transformed oribatid mite data explained by the environmental variables. The "grain" of the variogram (size of a distance class) is chosen to be the truncation threshold used in the PCNM analysis, 1.011187.

```
# Multiscale ordination (MSO)
# ***************************

# MSO of the undetrended mite data vs environment RDA
# ---------------------------------------------------
mite.undet.env.rda <- rda(mite.h, mite.env2)

mite.env.rda.mso <- mso(mite.undet.env.rda, mite.xy, grain=dmin,
  perm=999)
msoplot(mite.env.rda.mso, alpha=0.05/7)
mite.env.rda.mso
```

The resulting plot (Fig. 7.11) provides several informations. In the upper part of the diagram, the dashed line with the crosses represents the sum of the explained and residual empirical variograms. The continuous lines represent the confidence envelopes of the variogram of the data matrix. The monotonic increase of the dashed line is the signature of the strong linear gradient present in the data. Note, however, that the variogram of the residuals (bottom of the graph) shows no distance class with significant spatial correlation (after a global Bonferroni correction for seven simultaneous tests: rejection threshold divided by the number of classes), and that variogram is essentially flat. This means that the broad-scale linear gradient is well explained by the environmental variables.

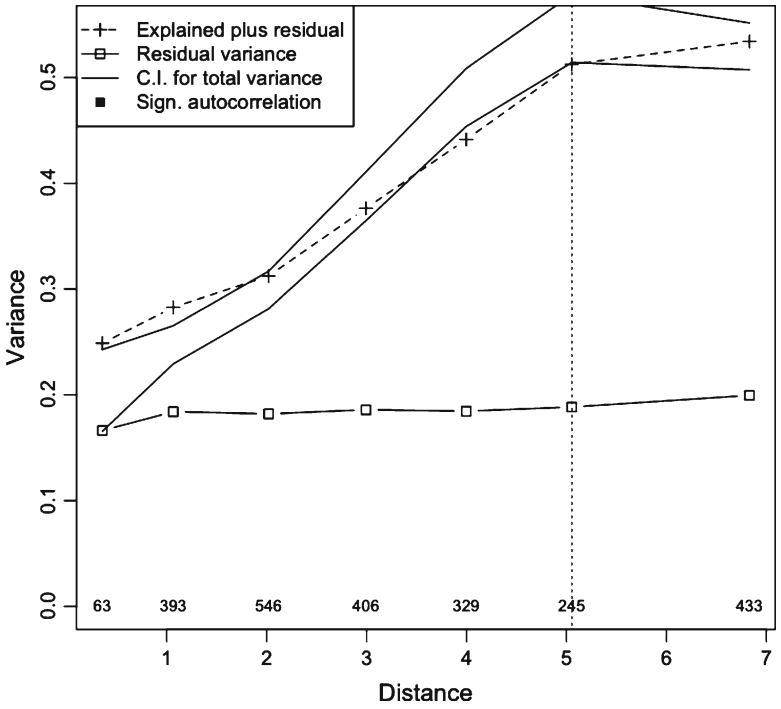

Fig. 7.11 Plot of the MSO of a RDA of the Hellinger-transformed oribatid mite data explained by the environmental variables. Explanations: see text

However, an intriguing feature appears. When the species–environment correlations do not vary with scale, the dashed line remains within the boundaries of the confidence envelopes. This is not the case here (see classes 1, 2 and 5), suggesting that it is not appropriate to run a non-spatial, global species–environment analysis with the implicit assumption that the relationships are scale-invariant. On the contrary, we can expect the regression parameters to vary with scale, so that a global estimation is meaningless unless one controls for the regional scale spatial structure causing the problem.

As an attempt in this direction, let us run an MSO on a partial RDA of the mite species explained by the environment, controlling for the spatial structure, here represented by the seven MEM variables of our best model.

```
# MSO of the undetrended mite data vs environment RDA,
# controlling for MEM
# ----------------------------------------------------
mite.undet.env.MEM <- rda(mite.h, mite.env2,
as.data.frame(MEM.select))
mite.env.MEM.mso <- mso(mite.undet.env.MEM, mite.xy, grain=dmin,
  perm=999)
msoplot(mite.env.MEM.mso, alpha=0.05/7)
mite.env.MEM.mso
```

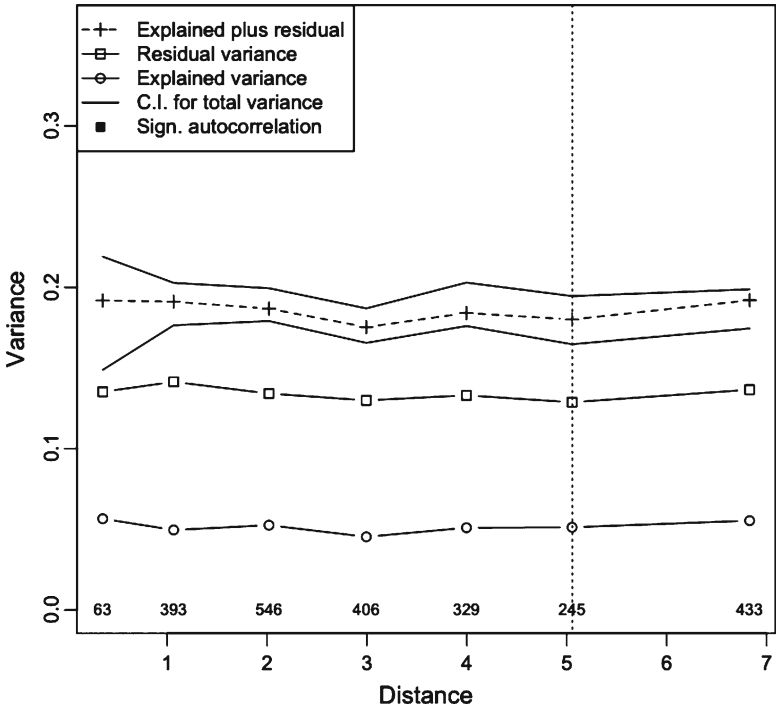

Fig. 7.12 Plot of the MSO of a RDA of the Hellinger-transformed oribatid mite data explained by the environmental variables, controlling for spatial structure (seven MEM variables)

Figure 7.12 shows that the problem of scale-dependence in the model has been properly addressed. There is no spatial correlation in the residuals, and the variogram of the residual species–environment relationship (after taking the MEM spatial structure into account) stays within the confidence interval across all scales. Furthermore, the MEM variables have also removed the major gradient from the data, resulting in a globally flat empirical variogram. The console message stating that the "Error variance of regression model [is] underestimated by -Inf percent"

actually refers to the difference between the total residual variance and the sill of the residual variance. When the value is negative (and extreme in this case), the absence of significant autocorrelation causes an underestimation of the global error value of the regressions. A positive value (e.g. 10%), which could occur if the residuals were significantly autocorrelated, would act as a warning that the condition of independent residuals is violated, thereby invalidating the statistical tests (see Sect. 7.2.2).

7.5.3 *Application to the Detrended Mite and Environmental Data*

Let us apply an MSO analysis on detrended data, as an effort to meet the conditions of application of the calculation of the variogram confidence intervals. We know from Sect. 7.4.2.5 that there is a significant spatial structure only in the *Y* direction. We shall therefore detrend the mite and environmental data on the *Y* coordinate before running the RDA.

```
# MSO on detrended mite and environmental data
# -------------------------------------------

# Detrend mite data on Y coordinate
mite.h.det2 <- resid(lm(as.matrix(mite.h) ~ mite.xy[,2]))

# Detrend environmental data on Y coordinate
env2.det <- resid(lm(as.matrix(mite.env2) ~ mite.xy[,2]))

# RDA and MSO
mitedet.envdet.rda <- rda(mite.h.det2, env2.det)
miteenvdet.rda.mso <- mso(mitedet.envdet.rda, mite.xy,
  grain=dmin, perm=999)
msoplot(miteenvdet.rda.mso, alpha=0.05/7)
miteenvdet.rda.mso
```

The result (Fig. 7.13) tells us a similar story, less the broad-scale gradient which has been removed prior to the analysis by detrending. The residual variance shows no spatial correlation, and the second, fourth and fifth class of the variogram of explained plus residual data fall outside the confidence interval. So the overall variogram shows no trend, but some regional spatial variance is present. Can the MEM control successfully for this spatial variance?

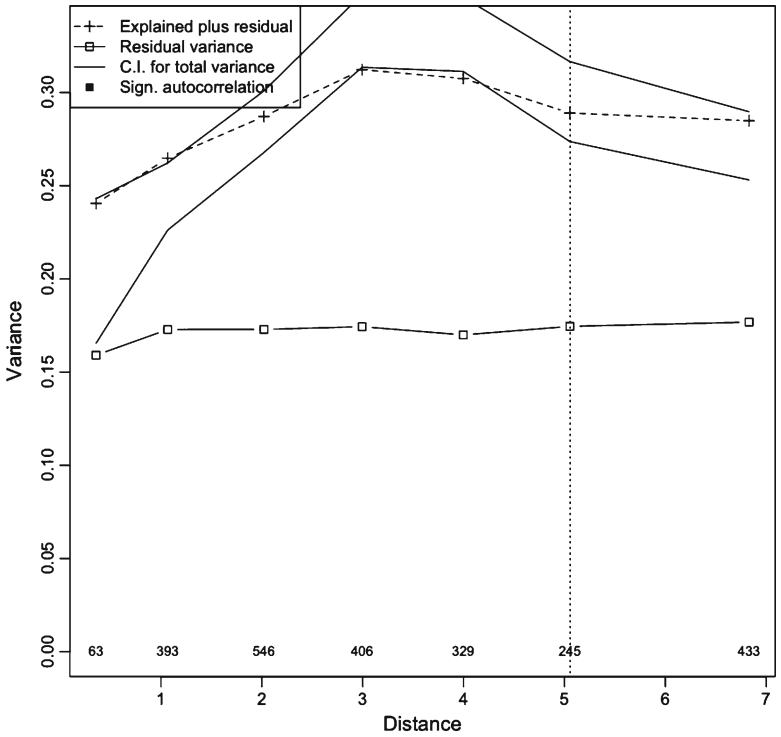

Fig. 7.13 Plot of the MSO of a RDA of the Hellinger-transformed and detrended oribatid mite data explained by the detrended environmental variables. Explanations: see text

```
# MSO of the detrended mite data vs environment RDA, controlling
# for MEM
# ------------------------------------------------------------------

mite.det.env.MEM <- rda(mite.h.det2, env2.det,
as.data.frame(MEM.select))
mite.env.MEM.mso <- mso(mite.det.env.MEM, mite.xy, grain=dmin,
  perm=999)
msoplot(mite.env.MEM.mso, alpha=0.05/7)
mite.env.MEM.mso
```

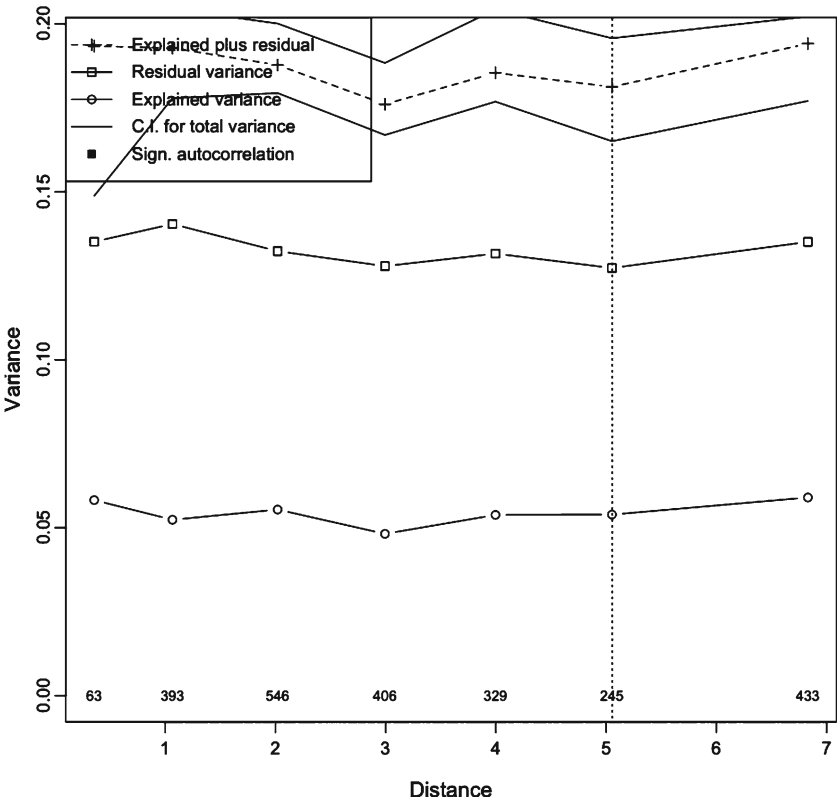

Fig. 7.14 Plot of the MSO of a RDA of the Hellinger-transformed and detrended oribatid mite data explained by the detrended environmental variables, controlling for spatial structure (seven MEM variables). Further explanations: see text

The answer is "yes" (Fig. 7.14). As in the undetrended example, one can see no spatial variance in the residuals or in the data. Compare with Fig. 7.12: the variograms are very similar (although the default graphical output provides a different ordinate scale). The MEM variables have successfully controlled for the spatial variance unexplained by the environmental data.

This example shows the potential of combining multivariate geostatistical methods with canonical ordination when the aim of the study is to test for and model species–environment relationships while discriminating between the two major sources of concern related to spatial structures: spatial dependence (7.1) and spatial autocorrelation (7.2). Some aspects of this approach remain to be explored, however. Wagner (2004) notes "*an important discrepancy between the results presented*

here and those by Borcard et al. (1992). Borcard found that 12.2% of the total inertia was spatially structured but could not be explained by the environmental variables. In the spatial partitioning of CCA results by multi-scale ordination (MSO), however, spatial autocorrelation appeared to be limited to distances smaller than 0.75 m, and there was no evidence of any cyclic pattern that could account for such a large portion of inertia. The large portion of nonenvironmental spatial structure identified by Borcard et al. (1992) may partly be due to a confounding of the effects of space and environment (Méot et al. 1998)". Arguing from an opposite point of view, we believe that the pure spatial structures revealed by canonical ordination (and especially in the PCNM and MEM framework which would give an even larger pure spatial fraction) are real and not due to confounding effects. In the latter case, they would have shown up in the common fraction of variation, not the pure spatial fraction. The question is rather: why did the MSO *not* reveal these structures? This may be due to the fact that no formal way of quantifying variance components in MSO has been devised as yet (H. Wagner, pers. comm.). However, this does by no means invalidate the MSO approach which, combined with the powerful tools developed in this chapter, increases our control over the complex process of extracting meaningful spatial information from ecological data.

7.6 Conclusion

Spatial analysis of ecological data has undergone huge developments during the last three decades. The paradigm shift announced by Legendre (1993) has been accompanied by an increasing awareness, not only of the importance of spatial structures per se, but also of the need for refined modelling tools to identify, represent and explain the complex structures by which ecological interactions manifest themselves in living communities. While an entire family of techniques aimed at prediction and mapping has been developed in the field of geostatistics and some of them can be applied to ecological problems, the specific questions and data in ecology demanded other approaches more directly related to the multivariate structure of communities and their relationship to the environment. We have presented the most important among them in this chapter, encouraging readers to apply them to their own data in a creative way. The multiscale nature of ecological problems can now be addressed in a much deeper way than before, and the authors of the methods are themselves constantly surprised at the range of applications ecologists make of their statistical offsprings. Many more developments will certainly be made in the forthcoming years, and we wish to conclude by inviting the readers to participate in this effort, both by asking new and challenging ecological questions and by devoting themselves to the exciting task of methodological development.