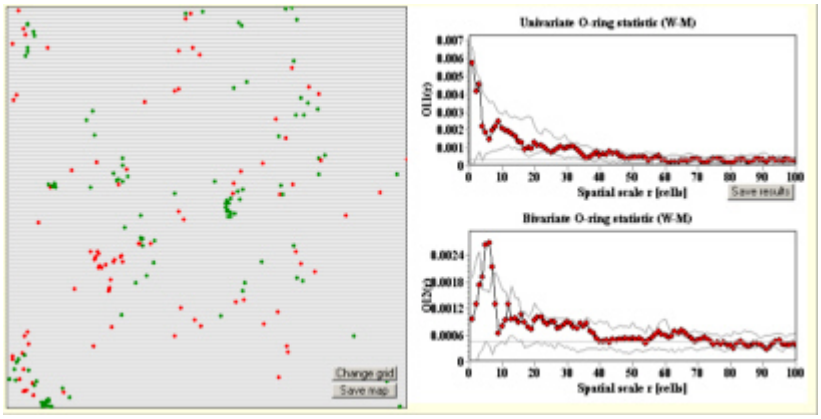
Introduction to Point Pattern Analysis with Ripley's *L* and the *O*-ring statistic using the *Programita* software

A user manual
with an collection of examples for point pattern analysis
using the *Programita* software



Second draft version
8 of March 2004

written by
Thorsten Wiegand
Department of Ecological Modelling,
UFZ-Centre for Environmental Research,
PF 500136,
04301 Leipzig, Germany,

Tel.: (**49) 341-235 2479
Fax: (**49) 341-235 3500
email: towi@oesa.ufz.de

Contents

1. *Programita*

1.1. Abstract

The *Programita* software allows you to perform univariate and bivariate point-pattern analysis with Ripley's *L*-function and the *O*-ring statistic. *Programita* contains standard and non-standard procedures for most practical applications. Procedures for non-standard situations include the possibility to perform point-pattern analyses for arbitrarily shaped study regions and *Programita* offers a range of non-standard null models such as heterogeneous Poisson null models or cluster null models.

The calculation of the *L*-function and the *O*-ring statistic is done within a grid-based framework which greatly simplifies the computation of *L* and *O* for non-standard situations. Both measure are based on the distance between all pairs of points of a pattern and count the number of points within (or at) a certain distance, *r*, of each point, with *r* taking a range of scales. While the *L*-function is basically related to the mean number of neighbours in a circle of radius *r*, the *O*-ring statistic is related to the mean number of neighbours in an annulus of radius *r*.

Programita tests for significance of a given null model by comparing the observed data with Monte Carlo envelopes from multiple simulations of the null model. *Programita* allows for a variety of specific null models for univariate and bivariate point-patterns. The procedures used by *Programita* are described in detail in Wiegand and Moloney 2004.

This document is primarily a manual to the use of *Programita* with extensive examples, but it provides also an introduction to point-pattern analysis.

1.2. Before starting *Programita*

1.2.1. Hardware requirements

Programita is a free unsupported software, developed in Borland Delphi4 under a WindowsXP environment. *Programita* is executable under 32-bit operating systems such as Windows98, Windows 2000, Windows XP or WindowsNT.

Running *Programita* requires little hard drive space. For example, for grid sizes < 200 ×200 cells *Programita* and temporally created files occupy < 10M. How-

ever, analysis of larger grid sizes may be slow for small working memory and low computer speed.

1.2.2. Terms of use and copyright agreement

The *Programita* software is produced by Thorsten Wiegand in his spare time. He is affiliated at the Dept. Ecological Modelling, UFZ Centre for Environmental Research Leipzig-Halle. *Programita* is intended to foster analysis of point patterns in ecology by providing ecologists a tool that contains null models and procedures not supported by most statistical packages, but which are essential for a throughout analysis of point-patterns. The *Programita* software is not a not commercial venture and may be downloaded and used free of charge for purposes of scientific research and teaching. Any commercial application of the program requires the previous permission by the author. Publications must acknowledge use of the *Programita* and cite Wiegand and Moloney (2004) which describes the implementation and the procedures used by *Programita*.

1.2.3. Installation

There is no setup procedure; installation of programita requires only the extraction of all files from the zip file **Progamita.zip**. Make sure that you also access the PDF (**ManualProgramita2004b.pdf**) and HTM versions (**ManualHTM2004b.zip**) of the user manual of Programita. Place the files into a directory of your choice; extracting the zip file will place all files into the subdirectory Programita. Note that you must place all files in the same directory; for simplicity *Programita* does not use a path variable. The zip-file contains the following files and file types:

programita2004b.exe	the executable of *Programita,* version 8 of March 2004
*.asc files	example data file in ArcView raster format
*.dat files	example data files and temporary files
*.fit files	file with results of the fit of a cluster null model
*.res files	results and settings files
*.shp files	files used for defining an irregularly shaped study region

The manual of *Programita* (**ManualProgramita2004b.pdf**) and a HTM version of the manual (**ManualHTM2004bzip**) are provided separately. You can use the HTM version as help because it contains many textmarks and internal links for easy navigation through the document.

1.2.4. Screen size

Programita was designed for a screen of 1024 × 768 pixels, but it can be run as well using a 800 × 600 screen. If you execute *Programita* in the 1024 × 768 pixel mode, it must look like the segment shown in Figure 1. Sometimes windows within *Programita* are truncated and one cannot see all of some buttons or headers. In this case it is as if the window is too small to handle them. To avoid this problem check the default letter size in the settings of your computer. Your computer may scale the letters but not the window sizes and as a consequence, the windows appear too small.

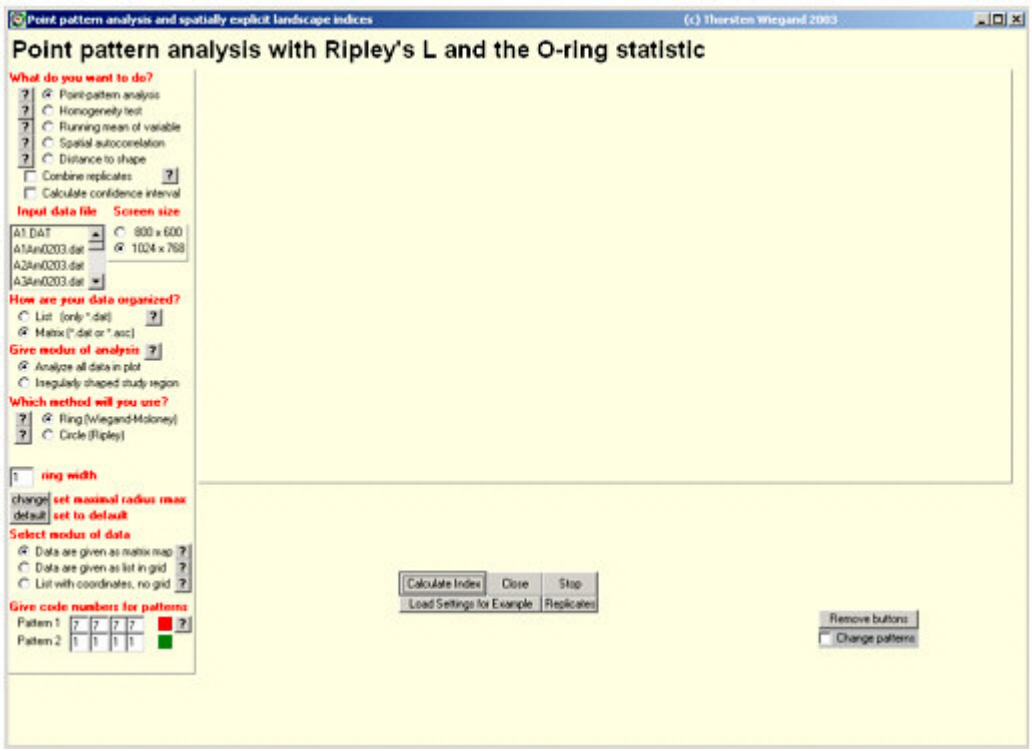


Figure 1. Correct display of the *Programita* interface under the 1024 × 768 pixel mode.

1.3. A quick start

1.3.1. Execute *Programita*

Execute programita2004b.exe and adjust *Programita* to your screen size. Two options are given, a screen of 800 × 600 pixels, and a larger screen of 1024 × 768 pixels.

1.3.2. Load a settings file to redo an analysis

There is a convenient way to quickly start with *Programita* and to learn the settings. You can read a file (a *.res file) that contains all setting of a previous analysis and redo this analysis. For example, you can repeat all analysis show in the figures 3 - 6 in Wiegand and Moloney (2004).

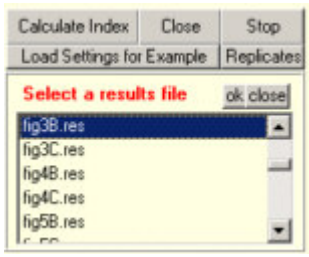


Figure 2. Load an example settings file.

To load a settings file, apply the button "Load Settings for Example" (Fig. 2) and a list with files containing settings of old analysis will appear (Fig. 2). Select a *.res file, for example fig3B.res, press ok, and then the button "Calculate Index". Now *Programita* performs the analysis of figure 3B in Wiegand and Moloney (2004).

1.3.3. What happens on the screen?

After loading the settings file fig3b.res, *Programita* will automatically select all settings for the data and analysis mode and all settings for the null model that was used in the example fig3B.res.

Two plots will appear: on the left a plot showing the original point pattern being analyzed (Fig. 3a), and on the right appear the patterns of the Monte Carlo simulations of the null model used for constructing the confidence envelopes. After termination of the simulations of the null model, the figure with the simulated patterns of the null model disappears, and instead a figure with the result of the analysis appears on the right (Fig. 3b).

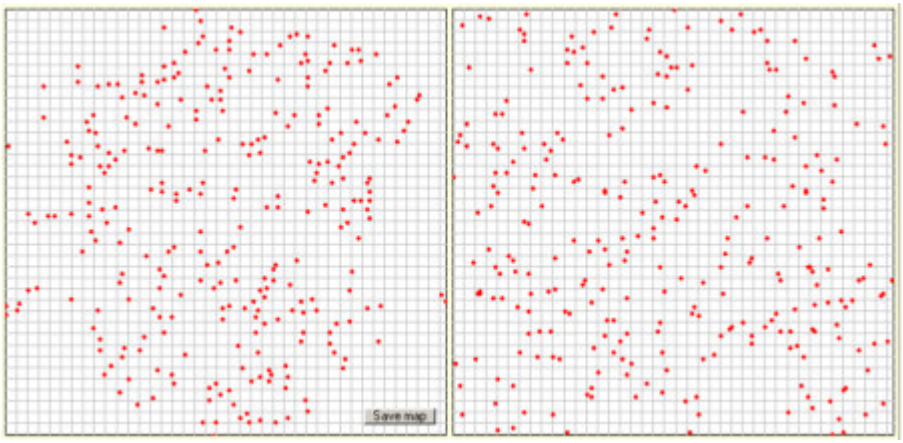
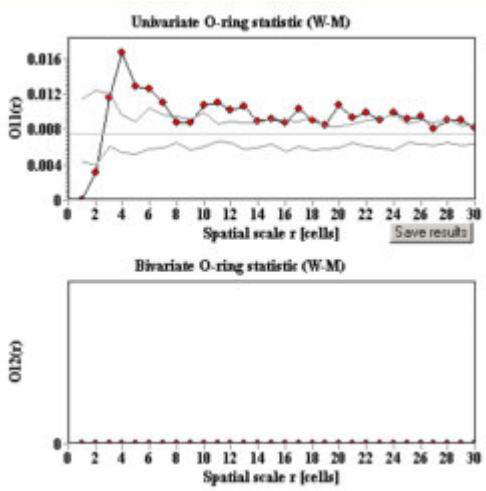


Figure 3a. Left: the point-pattern analysed in fig3B in Wiegand and Moloney (2004). Right: One realization of the Monte Carlo null model (a random pattern, CSR) used to construct the confidence envelopes.

Figure3b. After termination of the Monte Carlo simulations of the null model, a figure with the result appears on the right. The figure shows Wiegand-Moloney's *O*-ring statistics (or Ripley's *L*-function) together with the confidence envelopes for the specific null model chosen. The top figure shows the results of the univariate point pattern analysis; the bottom figure shows the results of the bivariate analysis if a second type of points was specified. In fig3B only one type of points was used, therefore there appears no result for the bivariate analysis.



1.3.4. Save the results of the analysis

To save the results of the analysis press the button “save results” that appears below the graph with the results of the univariate analysis (figure 3b), and insert a name for the result file. The results file will be saved as ASCII file **name.res** in the same directory where programita.exe is located. The results file (figure 4) contains the settings of this analysis and the results of the univariate and the bivariate point-pattern analysis. The results file **name.res** can be used (in the same way as fig3B.res) in the previous section to load the setting and to repeat the analysis.

```
Pointpattern analysis of file C:\THORSTEN\curso\nuevo\marcela.dat.dat
Method Wiegand-Moloney (ring) with  19 replicates for confidence interval
Test Model= 12random
the null assumed homogeneous pattern(s)
Analysis modus= multiple
several points per cell allowed
All cells within the rectangle were considered for calculating the indices
number points of pattern1 =  287
number points of pattern2 =    0
the rectangular area contains 198*191 = 37818   cells  (= dim1*dim2)
 x-grid-size=  198 y-grid-size=  191 cell-size =  1.0000 units
```

Scale	r	WM11(r),	E11-,	E11+,	WM12(r),	E12-,	E12+
0		1.0278746	1.0000000	1.0278746	0.0000000	0.0000000	0.0000000
1	- r	0.0000000	0.0044072	0.0114588	0.0000000	0.0000000	0.0000000
2	- r	0.0030756	0.0040009	0.0123866	0.0000000	0.0000000	0.0000000
3	r r	0.0116136	0.0060520	0.0120503	0.0000000	0.0000000	0.0000000
4	+ r	0.0167229	0.0053619	0.0096111	0.0000000	0.0000000	0.0000000
5	+ r	0.0128600	0.0052102	0.0088624	0.0000000	0.0000000	0.0000000
6	+ r	0.0125429	0.0058450	0.0104407	0.0000000	0.0000000	0.0000000
7	+ r	0.0110644	0.0059284	0.0096626	0.0000000	0.0000000	0.0000000
8	r r	0.0087497	0.0064436	0.0095510	0.0000000	0.0000000	0.0000000
9	r r	0.0087797	0.0056123	0.0092629	0.0000000	0.0000000	0.0000000
10	+ r	0.0108068	0.0061435	0.0099628	0.0000000	0.0000000	0.0000000

Figure 4. The *.res results file (fig3B.res). The **first 11 lines** contain the information on the settings of the analysis; the following part contains a table with the results of the analysis. The **first column** gives the spatial scale *r* of the point-pattern analysis, the second and third column provide a summary of the Monte Carlo significance test of the null model ("-": data at scale *r* below the confidence intervals, "r": inside the confidence envelopes, and "+": above the confidence envelopes; second column for univariate analysis, third column for bivariate analysis), **columns 4, 5, 6**: results of univariate analysis (column 4: univariate *L*-function or *O*-statistic of the data, column 5: lower confidence envelop, column 6: upper confidence envelope), and **columns 7, 8, 9:** results of bivariate analysis (column 7: univariate *L*-function or *O*-statistic of the data, column 8: lower confidence envelop, column 9: upper confidence envelope).

1.3.5. Temporary data files

During the analysis, *Programita* creates a number of temporary data files which are overwritten by a new analysis. Knowing these files you may use the information they contain.

The files tempp1.dat and tempp2.dat.—The file tempp1.dat contains a matrix representation of pattern 1. The first line contains information on the dimensions of the grid: (1, number of lines, 1, number of columns). The following lines are the data matrix with the pattern. The numbers are not code numbers as in the matrix data format but give the number of points of pattern 1 in a given cell. The file tempp1.dat does not contain information on an irregularly shaped study region.

The file tempp2.dat is the analogue to tempp1.dat and gives the number of points of pattern 2 in a given cell.

The files Bi_confidence.dat and Uni_confidence.dat.—Programita uses the lowest and highest $O(r)$ [or $L(r)$] of the different simulations of the null model as confidence envelope. However, it automatically produces two temporally files (Uni_confidence.dat, Bi_confidence.dat) that contain the $O(r)$ [or $L(r)$] for all simulations of the null model. The columns of these files are the scales $r = 1$, r_{max}, and the lines are the different simulations of the null model. You may use this information to construct confidence envelopes with different definitions, for example the 5th highest and 5th lowest $O(r)$ [or $L(r)$] out of 99 replicate simulations of the null model for defining 95% confidence envelopes (e.g., Stoyan and Stoyan 1994).

The file tempshape.dat.—If you analyze an irregularly shaped study region in the mode "Points without grid", *Programita* creates the file tempshape.dat. This file is the version of your data in the mode "Points in grid"

The file temp.fit.—If you fit a Neyman-Scott cluster model to your data programita saves the results of the fit in the temporary file temp.fit. However, the menu of *Programita* allows you to save the results of the fit under any name.

The files RL_join_1.rlb and RL_join_2.rlb.—Programita uses these files to show you, without performing new simulations of the null model, the results of the different variants 1 - 6 of random labeling.

1.4. The input data files (*.dat and *.asc data files)

Programita performs point pattern analysis for two different situations. First, it calculates the *O*-ring statistic and the *L*-function for point pattern which are basically given as a list of points. In a second mode, *Programita* performs point-pattern analysis for categorical maps. In this case the data input is a matrix with categories that can range from 0 to 9. It is important to understand the difference in point pattern analysis between points and categorical maps. In the following we discuss the data input separately for these two modes.

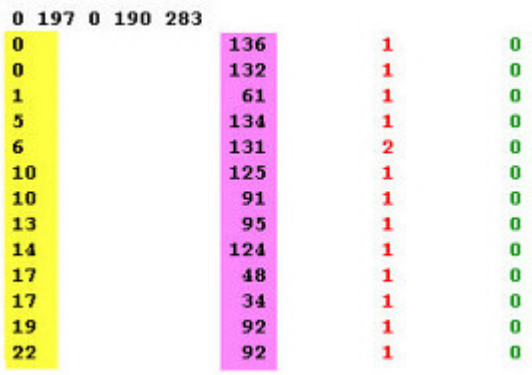
1.4.1. Settings for point-pattern analysis using lists of points

Programita performs common point pattern analysis for patterns which are given as a list of coordinates. In this case enable "List" in menu How are your data organized? (figure 5 left). There are two options for lists: (1) the data are list of coordinates, or (2) the data are transformed to a grid. If your data are a list of coordinates of points select the option "List with coordinates, no grid" in the settings menu Select modus of data (figure 5 right). If the coordinates in the list refer to cells of a grid select "Data are given as list in grid" in the settings menu Select modus of data.

Figure 5. The settings menus for data input.

1.4.2. Preparation of data in grid-mode for lists of points

The data file must be a space (or tab) delimited ASCII file with the *.dat extension (see example in figure 6). You need to provide information on the grid size, the number of cells with data, and the coordinates of cells that contain points. If your study region is of non-rectangular shape you need additionally to include the empty cells of your study region into the list. The numbering of the cells can start with any integer number; however, *Programita* will internally transform the coordinates to integers that start with the coordinates 1. You can read data which are given as numbers of points in a cell (in this case the columns 3 and 4 of the list can contain the values 0, 1, 2, 3, 4 …), or as a list of points with coordinates (in this case columns 3 and 4 may contain only the values 0 and 1). The grid size is automatically set to a value of 1.

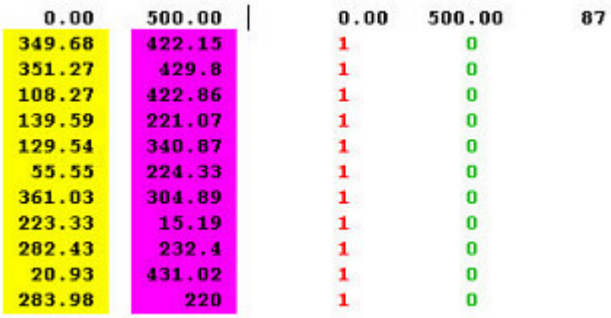


0 197 0 190 283			
0	136	1	0
0	132	1	0
1	61	1	0
5	134	1	0
6	131	2	0
10	125	1	0
10	91	1	0
13	95	1	0
14	124	1	0
17	48	1	0
17	34	1	0
19	92	1	0
22	92	1	0

Figure 6. Example of a data file for "Data age given as list in grid". Shown are the first lines of the file marcela.dat used in fig3B.res. The first line contains information on the grid: it is a 198 × 191 grid with a grid size 1. The numbers of the first line: 0: smallest x-coordinate for a cell, 197: largest x-coordinate, 0: smallest y-coordinate for a cell, 190: largest y-coordinate, 283: total number of cells to read (= number of lines in marcela.dat -1). The first column gives the x-coordinates of the cells, the second column gives the y-coordinates of the cells, the **third column** gives the number of points of pattern 1 in the cell, and the **fourth column** gives the number of points of pattern 2 in the cell. The columns 3 and 4 can contain any integer number 0, 1, 2,… Note that a line "**x y 0 0**" defines an empty cell with coordinates (x, y). You need to include empty cells if you want to analyze a study region of non-rectangular shape.

1.4.3. Preparation of data in points-mode

The data file must be a space (or tab) delimited ASCII file with the *.dat extension (see example in figure 7). In contrast to a data file in the grid-mode, the coordinates in the point mode can be real numbers. This is a convenient feature since many field data may be e.g., in meter units with centimetres as digits. You need to provide information on the edge-coordinates of your study region, and the number of points in the list. If your study region is of non-rectangular shape you need additionally a (*.shp) data file with a list of points that encircle your study region. In this case the edge-coordinates of your study region are the coordinates of a rectangle that contains the entire study region.



0.00	500.00	0.00	500.00	87
349.68	422.15	1	0	
351.27	429.8	1	0	
108.27	422.86	1	0	
139.59	221.07	1	0	
129.54	340.87	1	0	
55.55	224.33	1	0	
361.03	304.89	1	0	
223.33	15.19	1	0	
282.43	232.4	1	0	
20.93	431.02	1	0	
283.98	220	1	0	

Figure 7. Example of a data file for "List with coordinates, no grid". Shown are the first lines of the file adults_real.dat. The first line contains the edges of the study region (xmin, xmax, ymin, ymax), and the number of points in the list. The first column gives the x-coordinates of the points, and the second column gives the y-coordinates of the points. The **third column** contains the indicator 1 if the point is of type 1 and the indicator 0 if the point is of type 2. The **fourth column** contains the indicator 0 if the point is of type 1 and the indicator 1 if the point is of type 2. Note that the third and forth column can only have the values 0 or 1.

1.4.4. Interval definition for points-mode

For transformation of your original coordinates to the internal grid coordinates of *Programita*, you need to provide a cell size. The cell size defines the resolution of the analysis with *Programita*. Selection of an appropriate cell size is constrained by the sampling error of the coordinates of the points that defines a minimum cell size, and by computational time for larger grids. A resolution coarser than the sampling error can be selected; this will depend on the minimum resolution of distance classes necessary for responding to the scientific question.

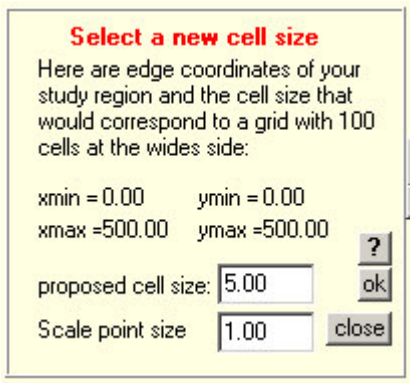


Figure 8. The window to select a cell size.

After selecting a data file and enabling "List with coordinates, no grid" the window Select a new cell size opens and asks you to provide a cell size. To help you in the selection of an appropriate cell size *Programita* shows the edge coordinates of the study region and the cell size that would correspond to a grid with 100 cells in the wide size. You can select any cell size > 0. However, by selecting a cell size be aware that large grids may considerably slow down *Programita*.

1.4.5. Transformation of data to grid for points-mode

Programita uses the following scheme for transformation of your original coordinates to grid coordinates:

grid coordinate	interval of original data
1	[0*cell size, 1*cell size)
2	[1*cell size, 2*cell size)
3	[2*cell size, 3*cell size)
…	…
n	[(n-1)*cell size, n*cell size]

where "[" is the closed interval that includes the left edge, and ")" is the open interval that does not includes the right edge.

1.4.6. Data input for point-pattern analysis using categorical maps

Programita facilitates analysis of categorical maps in raster format. For calculation of Wiegand-Moloney's *O*-ring statistic and Ripley's *L*-function, *Programita* considers four different categories:

- the cell is of type 1 (pattern 1)
- the cell is of type 2 (pattern 2)
- the cell is empty
- the cell is outside the study region (mask)

The procedures for calculation of the *O*-ring statistic and the *L*-function for categorical maps are the same as for point data.

1.4.7. Difference between matrix and point mode

Because the content of a cell is not a number of points, but a category, the Monte Carlo simulation of null models differs slightly. Under the mode "Matrix" the null model does not allow to have the same category two times in a given cell. However, if you enable the checkbox "Only one point per pattern" in the null-model window (figure 9), *Programita* allows having a mixed category where type 1 and type 2 are together in one cell. Thus, *Programita* uses in the matrix mode the same procedures for calculation of the *O*-ring statistic, the *L*-function, and for the null models as in the "Point mode", but in the matrix mode only one point (or one point per pattern) is allowed in a given cell.

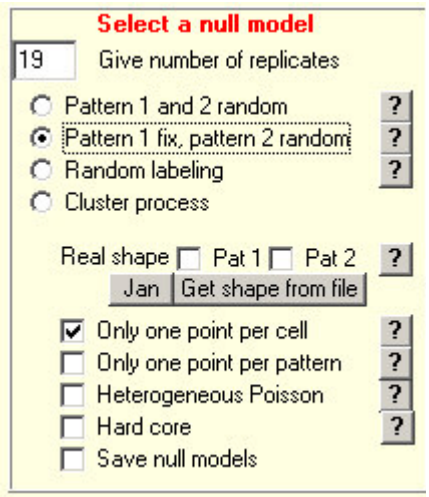


Figure 9. The null-model window for the mode "Matrix".

1.4.8. Preparation of data under matrix-mode

The input data are a matrix that can have the following code numbers:

- 0, 1, 2, ..., 9 if the cell is inside the study region
- -1 (or -9999) if the cell is outside the study region (mask)

Programita reads two different data formats in the matrix-mode:

1. a space (or tab) delimited ASCII file with the *.dat extension with line breaks.
2. the ASCII format of ArcView (a *.asc file) without line breaks. The head of the *.asc file must look like this:

```
ncols 144
nrows 45
xllcorner 1
yllcorner 1
cellsize  1
nodata_value -9999
```

ncols gives the number of columns, nrows the number of rows, xllcorner the smallest x-coordinate, and yllcorner the smallest y-coordinate. The cellsize must be "1" and the value for no data (the mask) must be -9999.

The matrix mode allows you to use a data matrix with different code numbers, however, calculation of Wiegand-Moloney's *O*-ring statistic and Ripley's *L*-function *Programita* requires a reduction of the original code numbers to the four categories:

- the cell is of type 1 (pattern 1)
- the cell is of type 2 (pattern 2)
- the cell is outside the study region (mask)
- the cell is empty

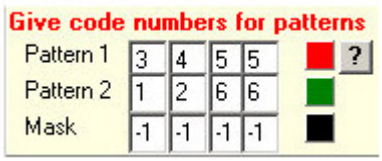


Figure 10. Transformation of the original code numbers of the data matrix to the three categories: pattern 1, pattern 2, and mask outside the study region. All other categories which are not set are automatically defined as empty cells.

If you enable the "Matrix" or "Data are given as matrix" option, the window Give code number for patterns (figure 10) appears and ask you to group your code numbers into the final categories "pattern 1", "pattern 2", and "mask". All other cells with code numbers not defines as pattern 1, pattern 2, or mask are defined automatically as empty cells. You can combine up to four code numbers (but not -1) to define "pattern 1" and "pattern 2", and up to four categories (including -1) to define the area outside the study region.

Thus, you can mask, if required, additionally cells which are part of the original the study region. For example, if you study vegetation maps with category 0: bare ground, category 1: grass tufts (size of one cell) and category 2: shrubs (size of several cells) you may mask the area occupied by shrubs for studying the spatial pattern of the grass tufts. If you do not exclude the area occupied by shrubs (which cover perhaps 10% or so of the study region) a simple null model that randomizes the locations of the grass tufts (CSR) will distribute tufts at locations where they cannot occur in the field. This introduces a bias in the analysis.

The possibility to use up to 10 categories is a convenient feature because you can use the same data for different analyses. Be sure that a given code number does not appear in different categories. A given cell can either be pattern 1, pattern 2, empty, or mask!

1.4.9. Format of the *.dat matrix data file

The *.dat matrix data file is a space (or tab) delimited ASCII file. The first line contains information on the dimensions of the grid: (1, number of lines, 1, number of columns). The following lines are the data matrix with the different code numbers. In contrast to the ArcView ASCII matrix format you need to insert line breaks. Note that the visualization of *Programita* corresponds to the transposed matrix. (figure 11).

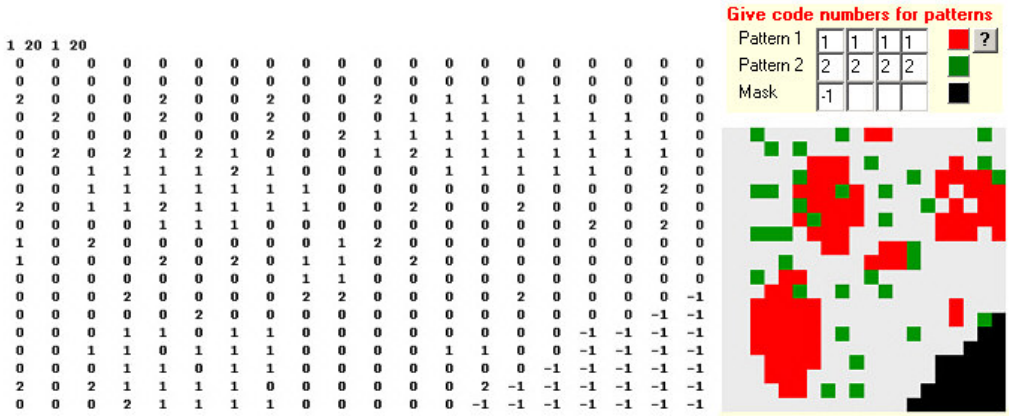


Figure 11. Example of a *.dat input data file for the matrix mode. Shown are the file small_matrix.dat (left) and the visualization in *Programita* (right). Red: cells of pattern 1 (code 1), green: cells of pattern 2 (code 2), grey: empty cells (code 0), black: mask with cells outside the study region (code -1). The first line contains information on the grid: (1, number of lines, 1, number of columns). Note that the visualization of *Programita* corresponds to the transposed matrix.

1.5. The Settings Menu for Point Pattern Analysis

If you select in the menu What do you want to do? "Point-pattern analysis", *Programita* allows you to select different types of analysis, input data, input data formats, etc. If you do not use a *.res settings file that stores the settings from a previous analysis you need to carefully select all settings manually from the settings menu before performing any analysis.

Programita calculates Ripley's *L* function (Circle in Which method will you use) and Wiegand-Moloney's *O*-ring statistic (Ring in Which method will you use) in a grid-based implementation for a given data file (selected in Input data file).

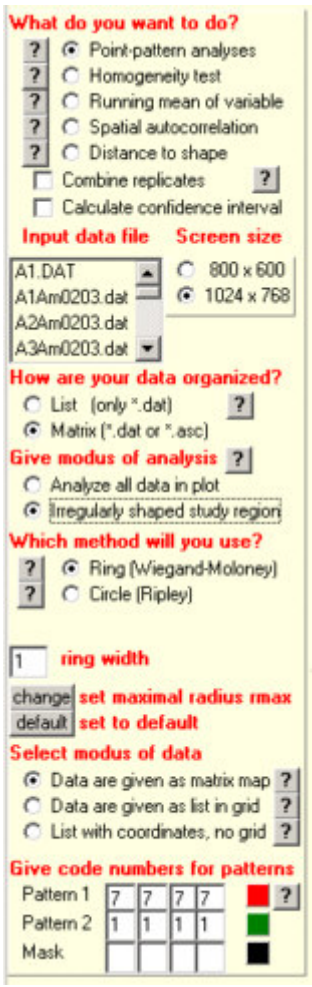


Figure 12. The settings menu for Point-pattern analysis.

In the window How are your data organized? you can select between two types of input data: (1) data which are given as a list of points and (2) categorical data which are organized as a matrix.

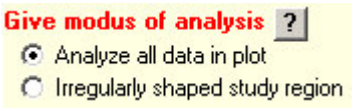
1.5.1. List of points

If your data are organized as list of points, you need to specify in the window Select modus of data whether your data are already transformed to a grid (integer coordinates) or if they are a list of coordinates (integer or real coordinates) without reference to a grid. In the latter case you need to provide a cell size.

1.5.2. Matrix data

If your data are a matrix (categorical data), you need to specify in the window Give code numbers for pattern which code numbers of your data matrix make up pattern 1, pattern 2, and the mask. The mask defines the area outside the study region if your study region is irregularly shaped.

1.5.3. Arbitrarily shaped study region



You can consider any arbitrarily shaped study region supported by the grid structure. If you select in the window Give modus of analysis the option "Irregularly shaped study region" some cells of the rectangular grid are not considered during the Monte Carlo simulation of the null models, and cells outside the study region are not counted for the numerical implementation of the *L*-function and the *O*-ring statistic. In contrast, if you select "Analyze all data in rectangle" the study region is the rectangle defined by your grid and all cells of the rectangle count and all cells are considered for simulation of the Monte Carlo Null models.

1.5.4. Arbitrarily shaped study region for "list in grid"

If your data are organized as list in a grid the list needs to include all cells with points and all empty cells of the study region. All cells that do not appear in the list are automatically defined as mask.

1.5.5. Arbitrarily shaped study region for "list without grid"

If your data are organized as list without grid, a window opens and ask you to provide a file with a list of points that encircle the study region (figure 13). This data file needs to be a space or tab delimited ASCII file with the *.shp extension. The first line gives the number of points in the list, and the following lines give the coordinates of the points that encircle the study region. The *.shp list of points (figure 14) need to define a closed shape. Be sure that the resolution of the line that encircles the study region is in accordance with the minimal cell size you will use.

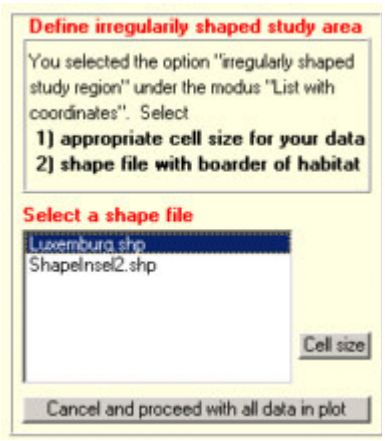


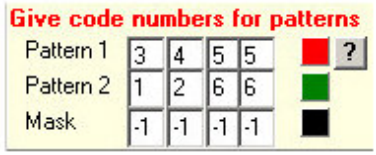
Figure 13. Window to define an irregularly study region if the data are points without grid.

If the resolution of the line is too coarse, *Programita* cannot properly define the study region. Be sure that the points in the *.shp list are defined in the same units as the points in the *.dat file that define your patterns. *Programita* asks you to confirm the cell size and to select the *.shp file.

```
   72
72454    135512
75587    137188
75988    133637
77582    130740
76690    127758
76051    125286
77301    122121
```

Figure 14. Example of the first lines of a *.shp file.

1.5.6. Arbitrarily shaped study region for matrix data



If your data are organized as matrix you can define a mask (cells outside the study region) with the category "-1", but additionally you can use any code number of your data matrix as mask.

1.5.7. Maximum scales *r* and ring width *dr*

The analysis is performed for spatial scale $r = 1, .. r_{max}$. The default value of the maximal scale r_{max} is half of the dimension of the smaller side of the grid; however, r_{max} can be changed with the button set maximal radius rmax.

If you select the *O*-ring statistic, you can change the ring width *dr* in the box ring width. The default ring width *dr* is one cell; however, if the rings are too narrow *Programita* will produce jagged plots for $O(r)$ as not enough points will fall into the different distance classes (figure 15). In this case you may select a larger ring width.

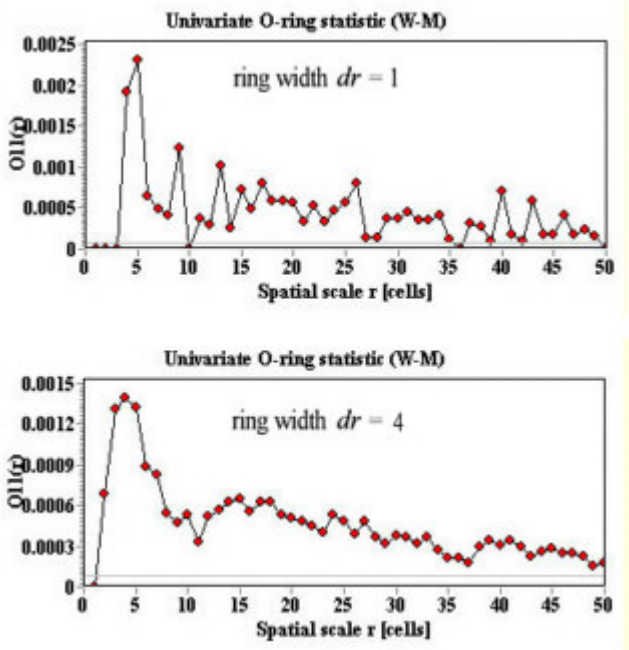


Figure 15. Different ring widths *dr*. The example shows the *O*-ring statistic for the data set adults_real.dat with a cell size 0.5m and a ring width *dr* = 1 (top) and a ring width *dr* = 4(bottom).

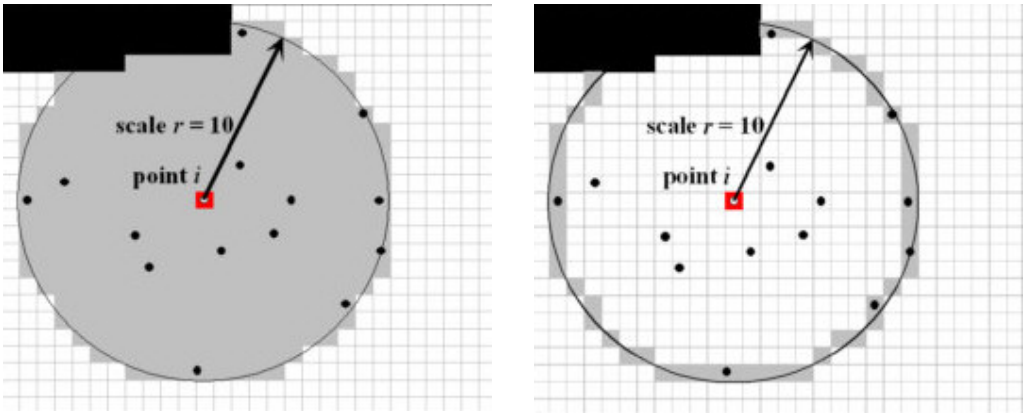


Figure D1. Numerical implementation of the *L*- function and the *O*-ring statistic for an irregularly shaped study region. Points of pattern 2 are represented by closed circles, the focal point *I* of pattern 1 as open circle within the red cell. Note that we approximate circles and rings with the underlying grid structure. Study region: grey and white cells, area outside the study region: black cells. (Left): For numerical implementation of Ripley's bivariate *L*-function we count the number of points of pattern 2 inside the part of the circles around point *I* of pattern 1 which falls inside the study region (i.e., the gray shaded area), and the number of cells within this area. (Right): For implementation of the bivariate *O*-function we count the number of points of pattern 2 inside the part of the ring around point *I* of pattern 1 which falls inside the study region (i.e., the gray shaded area), and the number of cells within this area.

2. Background of second-order statistics

For a homogeneous and isotropic point pattern, the second-order characteristics depend only on distance *r*, but not on the direction or the location of points. An appropriate geometry is therefore to adopt circular shapes (such as the circles of Ripley's *K*-function or the rings of Wiegand-Moloney's *O*-ring statistic) as a basis for the spatial statistics. Using rings instead of circles (Figure D1) has the advantage that one can isolate specific distance classes, whereas the cumulative *K*-function confounds effects at larger distances with effects at shorter distances. Note that the *K*-function and the *O*-ring statistic respond to slightly different biological questions. The accumulative *K*-function can detect aggregation or dispersion *up to* a given distance *r* and is therefore appropriate if the process in question (e.g., the negative effect of competition) may work only up to a certain distance, whereas the *O*-ring statistic can detect aggregation or dispersion *at* a given distance *r*.

2.1. Ripley's *K* and *L*-function

2.1.1. Definition of the bivariate *K*- and *L*-functions

The bivariate *K*-function $K_{12}(r)$ is defined as the expected number of points of pattern 2 within a given distance r of an arbitrary point of pattern 1, divided by the intensity λ_2 of points of pattern 2:

$$\lambda_2 K_{12}(r) = \mathbf{E}[\#(\text{points of pattern 2} \leq r \text{ from an arbitrary point of pattern 1})] \quad \text{(D1)}$$

where # means "the number of", and **E**[] is the expectation operator. Under independence of the two point patterns, $K_{12}(r) = \pi r^2$, without regard to the individual univariate point patterns. It can be difficult to interpret $K_{12}(r)$ visually. Therefore, a square root transformation of $K(r)$, called *L*-function (Besag 1977), is used instead:

$$L_{12}(r) = \left(\sqrt{\frac{K_{12}(r)}{\pi}} - r\right) \quad \text{(D2)}$$

This transformation removes the scale dependence of $K_{12}(r)$ for independent patterns and stabilizes the variance (Ripley 1981). Values of $L_{12}(r) > 0$ indicate that there are on average more points of pattern 2 within distance r of points of pattern 1 as one would expect under independence, thus indicating attraction between the two patterns up to distance r. Similarly, values of $L_{12}(r) < 0$ indicate repulsion between the two patterns up to distance r. The estimated *L*-function $\hat{L}_{12}(r)$ is calculated for a sequence of distances r and the results of $\hat{L}_{12}(r)$ are then plotted against distance.

2.1.2. Confidence envelopes

Because a given data set is only a unique realization of a given stochastic point process within a study region of limited size, the estimators of the *L*-function or the *O*-ring statistic may show small deviations from their theoretical values under a given null model. In order to test a null model against real data it is therefore necessary to take uncertainty due to (1) the stochastic character of the point process and uncertainty due to (2) the limited sample size (the number of points N of the pattern may be small) into account. Theoretically, distribution theory could be used in determining confidence envelopes for null models of point-patterns. However, this approach quickly becomes analytically intractable if edge effects for irregularly shaped study regions are considered, or if null models other than CSR are considered. Therefore, the more practical alternative is to

use Monte Carlo simulations of a realization of the stochastic process underlying the specific null model in constructing confidence envelopes around the null model (Upton and Fingleton 1985; Bailey and Gatrell 1995). Each simulation generates an $\hat{L}_{12}(r)$ function, and approximate $n/(n + 1) \times 100\%$ confidence envelopes are calculated from the highest and lowest values of $\hat{L}_{12}(r)$ taken from n simulations of the null model. For example, a 95% confidence envelope requires $n = 19$ simulations (e.g., Bailey and Gatrell 1995). A more accurate approach is to use the 5th-lowest and 5th highest $\hat{L}_{12}(r)$. In this case, 99 randomizations provide 5% confidence envelopes (e.g., Stoyan and Stoyan 1994). If $\hat{L}_{12}(r)$ has some part outside of that envelope, it is judged to be a significant departure from the null model.

Programita uses the lowest and highest $\hat{L}_{12}(r)$ for determination of the confidence interval, however, it automatically produces two temporally files (Uni_confidence.dat, Bi_confidence.dat) that contain the $\hat{L}_{11}(r)$ and $\hat{L}_{12}(r)$ for all simulations of the null model. The columns are the scales $r = 1$, r_{max}, and the lines are the different simulations of the null model. You can use this data for alternative constructions of confidence envelopes.

2.1.3. The Univariate *K*- and *L*-functions

The univariate K-function $K(r)$ is calculated in a manner analogous to the bivariate K function by setting pattern 1 equal to pattern 2. In this case the focal points of the circles are not counted. For a homogeneous Poisson process (complete spatial randomness CSR), $K(r) = \pi r^2$ and $L(r) = 0$. $L(r) > 0$ indicates aggregation of the pattern up to distance r, while $L(r) < 0$ indicates regularity of the pattern up to distance r.

2.2. Wiegand-Moloney's *O*-ring statistic

2.2.1. Definition of the *O*-ring statistic and the *g*-function

The mark-correlation function $g_{12}(r)$ is the analogue of Ripley's $K_{12}(r)$ when replacing the circles of radius r by rings with radius r, and the O-ring statistic $O_{12}(r) = \lambda_2 g_{12}(r)$ gives the expected number of points of pattern 2 at distance r from an arbitrary point of pattern 1 (Fig. 1B):

$O_{12}(r) = \mathbf{E}$[#(points of pattern 2 at distance r from an arbitrary point of pattern 1)] (D3)

The mark-correlation function $g_{12}(r)$ is related to Ripley's K-function (Ripley 1981; Stoyan and Stoyan 1994):

$$g_{12}(r) = \frac{\mathrm{d}K_{12}(r)}{\mathrm{d}r} / (2\pi r) \qquad \text{(D4)}$$

We obtain $O_{12}(r) = \lambda_2$ for independent patterns, $O_{12}(r) < \lambda_2$ for repulsion, whereas $O_{12}(r) > \lambda_2$ for attraction.

2.2.2. Selection of ring width

In practice, the calculation of the *O*-ring statistic involves a technical decision on the width of the rings. Clearly, the use of rings that are too narrow will produce jagged plots as not enough points will fall into the different distance classes. This problem does not occur for the accumulative *K*-functions. On the other hand, the *O*-ring statistic will lose the advantage that it can isolate specific distance classes if the rings are too wide.

Again, the univariate *O*-ring statistic $O(r)$ is calculated by setting pattern 2 equal to pattern 1. For CSR, $O(r) = \lambda$, $O(r) > \lambda$ indicates aggregation of the pattern at distance r, and $O(r) < \lambda$ regularity.

2.3. The memory in the *K*-function

Departure from the expectation under independence at a given scale r_0 [e.g., repulsion due to non-overlapping tree canopies] yields a *K*-function $K(r_0) \neq \pi r_0^2$. Because the *K*-function is accumulative, the value $K(r_0)$ influences the shape of the *K*-function also at scales $r > r_0$. This can be show mathematically for a pattern with no second-order effects at scales $r > r_0$ [thus $g(r) = 1$ for $r > r_0$], but a second-order effect up to scale r_0. In this case integration of equation D4 yields

$$\begin{aligned} K(r) &= \int_0^{r_0} 2\pi r' g(r') dr' + \int_{r0}^{r} 2\pi r' dr' \\ &= K(r_0) - \pi r_0^2 + \pi r^2 \end{aligned} \qquad \text{(M1)}$$

and the *L*-function becomes:

$$L(r) = -r + \sqrt{\frac{K(r_0)}{\pi} - r_0^2 + r^2}$$

, (M2)

$$L(r) = r\left(\sqrt{1 + \left(\frac{K(r_0) - \pi r_0^2}{\pi r^2}\right)} - 1\right)$$

which collapses back to $L(r) = 0$ if there are no second-order effects [i.e., $K(r_0) = \pi r_0^2$]. For a hard core at $r_0 = \delta$ [i.e., $K(\delta) = 0$], equation M2 collapses to equation HC3. Fig. M1 shows how second-order effects at small scales [i.e., a given $K(r_0) \neq \pi r_1^2$, equation M1)] impact the *L*-function at higher scales if there a no second order effects at higher scales [i.e., $g(r) = 1$ for $r > r_0$].

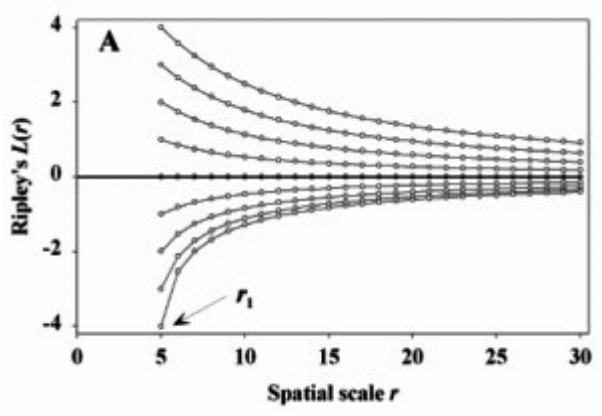


Figure M1. The memory in the accumulative Ripley's *L*-function. True second-order effects up to scale r_0, and no second-order effects for scales $r > r_0$. The different curves show equation M2 for initial values $L(r_0)$ = -4, -3, -2, -1, 0, 1, 2, 3, and 4. The bold line gives the *L*-function without second-order effects [i.e., $L(r) = 0$]

2.4. Virtual aggregation of univariate point patterns

If a pattern is not homogeneous, the null model of CSR is not suitable for exploration of second-order characteristics. This is because large-scale, first-order effects introduce a systematic bias in the univariate *K*-function, not only at larger scales, but also at smaller scales. In this case, an observed departure from CSR could well be due to first order effects rather than to second order effects (Bailey and Gatrell 1995). This can be understood intuitively, when imagining a point pattern that comprises a single internally homogeneous cluster in the center of the study region (e.g., Example CSR_1, figure V1). In this case the local density of points in the cluster will be higher than the overall density of points in the entire study region (figure V1, left). As a consequence, there are always more points in the closer neighborhood of other points than expected under homogeneity, and the *K*-function will indicate aggregation at smaller scales even if the pattern is random inside the cluster (figure V1, right). We call this phenomenon "virtual aggregation."

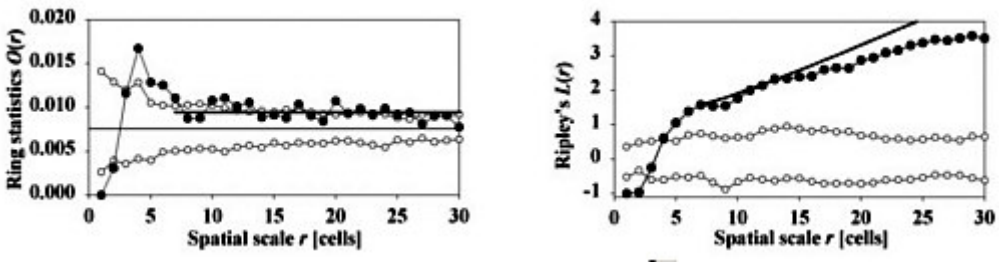


Figure V1. The results of example CSR_1.res for the *O*-ring-statistic (left) and for Ripley's *K* (right).

To demonstrate this intuitive idea mathematically, we imagine a univariate point pattern with overall intensity λ that forms an internally random cluster covering the proportion c of the study region. There are no points outside the cluster. Because sub-regions of the cluster satisfy CSR, the probability $O(r)$ of finding a point at the closer neighborhood r of other points will be approximately constant, i.e. $O(r) = g\,\lambda$ with $g = 1/c$. To obtain the corresponding *K*-function we integrate equation D4 using $g(r) = O(r)/\lambda = g$ and obtain $K(r) = \pi\, g\, r^2$, which yields:

$$L(r) = r(\sqrt{g} - 1) \qquad \text{(V1)}$$

Thus, under virtual aggregation we observe an *L*-function that increases at smaller scales linearly, and the extent of virtual aggregation, given through the

slope $\sqrt{g}$ -1, is inversely related to the fraction c of the study region covered by the cluster. Note that for smaller scales (i.e., scales r below the cluster size) the functional form of $L(r)$ under virtual aggregation is the same as under a Neyman-Scott cluster process (cf. equation V1 and equation C2). This is not surprising because virtual aggregation is caused by larg-scale clustering. The difference is that the cluster size under virtual aggregation is defined to be large, while the Neyman-Scott process can be applied for any cluster size.

The L-function can increase under virtual aggregation only over a limited range of scales; it will start to drop if a notable proportion of circles overlap the part of the study region outside the cluster. Finally, the L-function will approach zero for very large scales r because then all points will be located within each circle, i.e., $K(r) = \pi r^2$, and $L(r) = 0$.

If the pattern shows virtual aggregation but additionally true second-order effects [i.e., a non constant pair-correlation function $g(r)$ at scales $r < r_1$, and $g(r) = g$ for $r > r_1$], integration of equation I3 yields

$$\begin{aligned} K(r) &= \int_0^{r_1} 2\pi r' g(r')dr' + \int_{r_1}^{r} 2\pi r' g dr' \\ &= K(r_1) - \pi g r_1^2 + \pi g r^2 \end{aligned} \qquad \text{(V2)}$$

and the L-function becomes:

$$L(r) = -r + \sqrt{\frac{K(r_1)}{\pi} - g r_1^2 + g r^2}\,, \qquad \text{(V3)}$$

which collapses back to equation V1 if there are no second-order effects [i.e., $K(r_1) = \pi g r_1^2$]. Note that equation V3 approximates the impact of virtual aggregation only for a limited range of scales r, and for large scales the assumption $g(r) = g$ does not hold because in this case the circles will overlap the gap.

Weak virtual aggregation increases the local density $O(r)$ at smaller scales r only slightly and it should therefore not seriously affect the outcome of second-order analysis. However, the problem is that the Monte Carlo test for Ripley's K will indicate highly significant aggregation because the K-function is a cumulative measure where aggregation at smaller scales influences the estimate at larger scales (equation M1). The Monte Carlo test for the non-accumulative O-ring statistic, however, will indicate the expected weak aggregation. Example CSR_1 illustrates this point.

2.5. Numerical implementation of second-order statistics

Numerical methods require division of the study region into a grid of cells (figure 1). Selection of an appropriate cell size is constrained by the sampling error of the coordinates of the points that defines a minimum cell size, and by computational time for larger grids. A resolution coarser than the sampling error can be selected; this will depend on the minimum resolution of distance classes necessary for responding to the scientific question.

2.5.1. Approximation of rings and circles in a grid

Circles and rings need to be approximated in a grid-based implementation. In a first step, the grid-based approximation transforms the original coordinates of points to coordinates of cells in a grid, and in a second step it uses the coordinates of the grid cells to define (integer) distance classes r for the distance between cells. For a ring width of one cell (i.e., $dr = 1$), *Programita* uses the intuitive definition $r = trunc(d)$ where d is the Euclidian distance between the coordinates (x_1, y_1) and (x_2, y_2) of two cells and the function *trunc* truncates all digits of the real number and transform it to an integer (figure I1).

If the ring width dr is greater than one cell, the range of distances d that fall into the scale r is broadened and two cells with distance d may belong to different rings. To determine the different scales r to which a given pair of cells with distance d belongs, a lower scale $r_- = trunc(d - (dr - 1)/2)$ and a upper scale $r_+ = trunc(d + (dr - 1)/2)$ is calculated. The pair of cells belongs to all rings with scales r that fall inside the interval $[r_- , r_+]$.

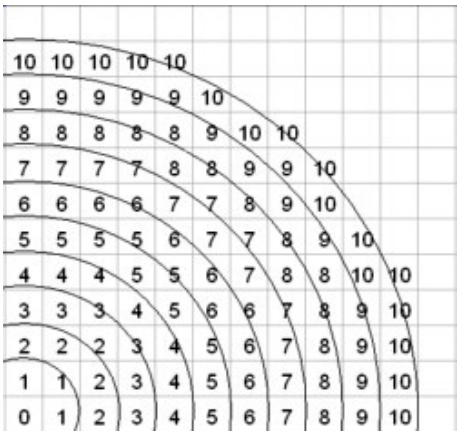


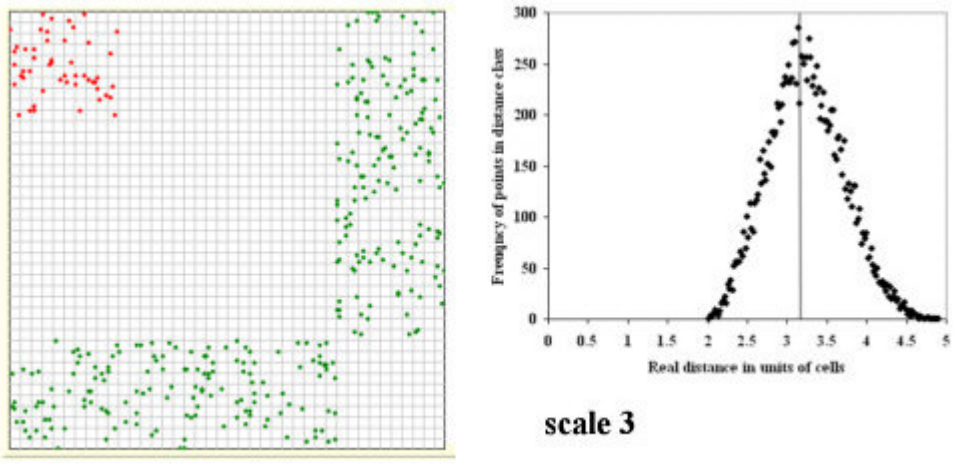
Figure I1. Approximation of rings with scale r and ring width of one cell ($dr = 1$) in the grid-based implementation. The integer number in a given cell corresponds to the scale r of a ring with centre in the cell "0".

For example, two cells with distance $d = 3.8$ belong to a ring with scale $r = 3$ if the ring width is one cell. For a ring width of two cells, they belongs to the rings with scales 3 and 4 since $r_- = trunc[3.8 - (2-1)/2] = trunc[3.3] = 3$ and $r_- = trunc[3.8 + (2-1)/2] = trunc[4.3] = 4$.

The definition of the scale r of circles is analogous to the definition of the rings: a pair of cells with distance d belongs to all circles with scale $r \geq trunc[d]$.

2.5.2. Test of ring approximation

To test our classification scheme for the scale r we calculated the probability that random points in cells separated by scale r have a real distance d. For a valid classification scheme we expect an average distance d close to the scale r, and that real distances have some normal-like distribution around scale r which is only little skewed. To this end, we distributed random points over a study region which had a x- and y-extension of 4 units (figure I2, left). Next we classified the points in cells with a cell size of one unit following our grid approximation and calculated the real distance of all points in the focal cell (scale $r = 0$, figure I1) to all points in cells which are scale $r = 1$ away. We then repeated this procedure for scales $r = 2$ and 3 (figure I2). We find that the scale r coincides well with the mean distances d, and the frequency distribution of the real distances of the real distances are only slightly skewed (figure I2, right). Thus, our grid-approximation of rings and circles is appropriate.



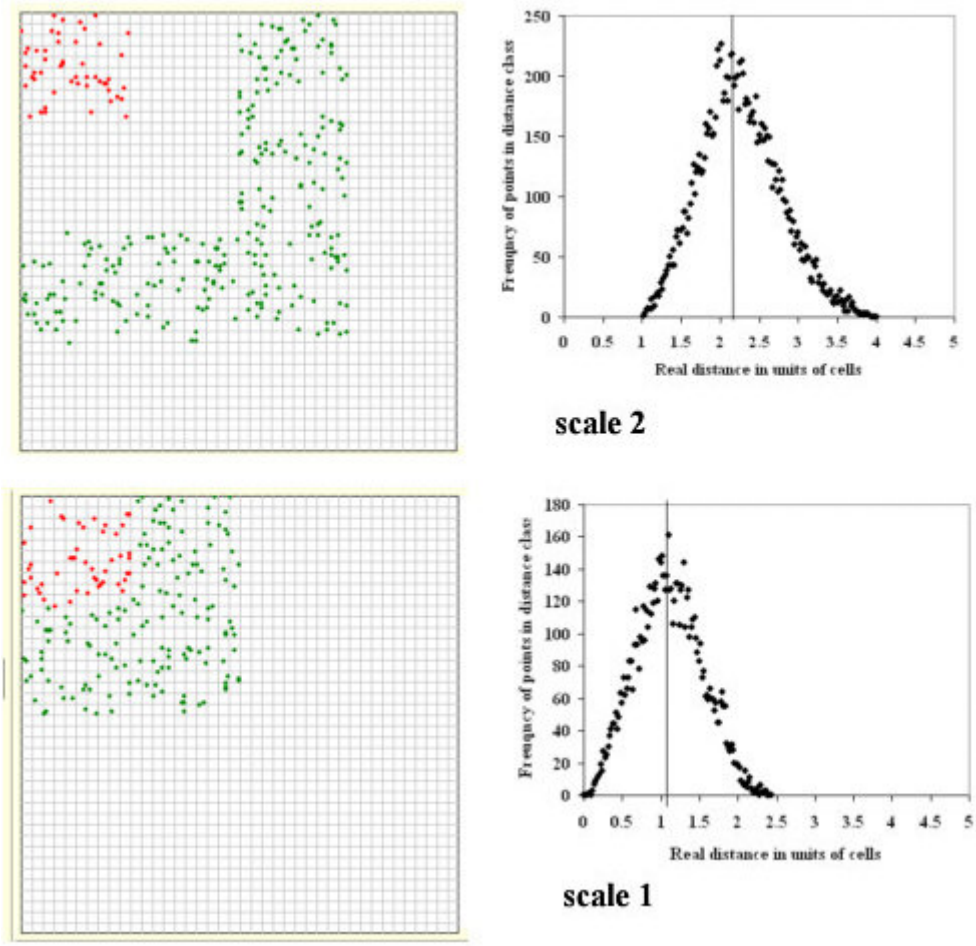


Figure I2. Test of our grid-approximation. Left figures: random points in the focal cell (red) and random points in cells which we classified to be at scale r from the focal cell (green dots). Right figures: frequency distribution of the real distance *r* of all points in the focal cell to all points in cells which are scale *r* away. The scale *r* coincides well with the mean distance *d*, and the frequency distributions of the real distances are only slightly skewed. Thus, our grid-approximation of rings is appropriate.

2.5.3. Numerical implementation of *L*(r) and *O*(r)

Ripley's *K*-function is defined via $\lambda K_{12}(r)$, which is "the expected number of points of pattern 2 at distances smaller or equal than *r* from an arbitrary point of pattern 1". The grid-based implementation of *Programita* considers only cells inside the study region and calculates

- $\frac{1}{n_1}\sum_{i=1}^{n_1}\mathbf{Points}_2[C_{1,i}(r)]$, (I1a)
  the mean number of points of pattern 2 in circles of radius *r* (centered in the points of pattern 1) inside the study region

- $\frac{1}{n_1}\sum_{i=1}^{n_1}\mathbf{Area}[C_{1,i}(r)]$, (I1b)
  the mean area of these circles inside the study region

where $C_{1,i}(r)$ is the circle with radius r centered on the ith point of pattern 1, n_1 the total number of points of pattern 1 in the study region, the operator $\mathbf{Points}_2[\mathrm{X}]$ counts the points of pattern 2 in a region X, and the operator $\mathbf{Area}[\mathrm{X}]$ determines the area of the region X. The full formula of the grid-based estimator of $\lambda K_{12}(r)$ used in *Programita* yields:

$$\lambda_2 \hat{K}_{12}(r) = \frac{\frac{1}{n_1}\sum_{i=1}^{n_1}\mathbf{Points}_2[C_{1,i}(r)]}{\frac{1}{n_1}\sum_{i=1}^{n_1}\mathbf{Area}[C_{1,i}(r)]}\pi r^2. \quad \text{(I3)}$$

To implement equation I3 we marked each cell (x, y) with an identifier $S(x, y)$ [$S(x, y) = 1$ if a cell with coordinates (x, y) is inside the boundaries of the study region, otherwise $S(x, y) = 0$] and with two additional marks $P_1(x, y)$ and $P_2(x, y)$ that give the number of points of pattern 1 and pattern 2 lying within the cell, respectively. Using these definitions, the numerator of equation I3 becomes:

$$\mathbf{Points}_2[C_{1,i}(r)] = \sum_{\text{all } x}\sum_{\text{all } y} S(x,y)P_2(x,y)I_r(x_i,y_i,x,y) \quad \text{(I4)}$$

where (x_i, y_i) are the coordinates of the ith point of pattern 1, and the counter variable I_r defines the circle with radius r that is centered at the ith point of pattern 1:

$$I_r(x_i,y_i,x,y) = \begin{cases} 1 & \text{if} \quad \sqrt{(x-x_i)^2+(y-y_i)^2} \le r \\ 0 & \text{otherwise} \end{cases}. \quad \text{(I5)}$$

The denominator of equation I3 is calculated analogously to equation I4, but it counts cells instead of points:

$$\mathbf{Area}[C_{1,i}(r)] = z^2\sum_{\text{all } x}\sum_{\text{all } y} S(x,y)I_r(x_i,y_i,x,y) \quad \text{(I6)}$$

where z^2 is the area of one cell. Because equation I4 and equation I6 include the identifier $S(x, y)$ of the study region, only points and cells are counted that are inside the boundaries of the study region. Therefore, the study region can be of any complex shape accommodated by the underlying grid. Using equation I3, our numerical estimator of the L-function is given by:

$$\begin{aligned}\hat{L}_{12}(r) &= \sqrt{\frac{K_{12}(r)}{\pi}} - r \\ &= r\left(\sqrt{\frac{1}{\lambda_2}\left(\frac{\lambda_2 K_{12}(r)}{\pi r^2}\right)} - 1\right) \\ &= r\left(\sqrt{\frac{A}{n_2}\left(\frac{\sum_{i=1}^{n_1}\mathbf{Points}_2[C_{1,i}(r)]}{\sum_{i=1}^{n_1}\mathbf{Area}[C_{1,i}(r)]}\right)} - 1\right) \qquad \text{using eq. I3}\end{aligned} \tag{I7}$$

where A is the area of the study region, and n_2 the number of points of pattern 2 inside the study region.

The analogous numerical estimate for the bivariate O-ring statistic is:

$$\hat{O}_{12}^{w}(r) = g_{12}(r)\lambda_2 = \frac{\frac{1}{n_1}\sum_{i=1}^{n_1}\mathbf{Points}_2[R_{1,i}^{w}(r)]}{\frac{1}{n_1}\sum_{i=1}^{n_1}\mathbf{Area}[R_{1,i}^{w}(r)]}. \tag{I8}$$

where $R_{1,i}^{w}(r)$ is the ring with radius r and width w centered in the ith point of pattern 1. The numerator and the denominator of equation I8 are the same as given in equation I4 and equation I6, respectively, but the counter variable I_r for circles has to be replaced by a counter variable I_i^w that defines a ring with radius r and width w around the ith point with coordinates (x_i, y_i):

$$I_r^w(x_i, y_i, x, y) = \begin{cases} 1 & \text{if} \quad r - \frac{w}{2} \le \sqrt{(x-x_i)^2 + (y-y_i)^2} \le r + \frac{w}{2} \\ 0 & \text{otherwise} \end{cases} \tag{I9}$$

3. Univariate Null Models

The key for successful application of *Programita* is the selection of an appropriate null model that responds to the specific biological question asked. The null model constitutes a point of reference against which the data are compared. The simplest null models assume no interaction between the points of the pattern and deviation from this null model provides evidence for interactions. The grid-based implementation of *Programita* facilitates simple implementation of a variety of null models that account e.g., for irregularly shaped study regions, first-order heterogeneity, or a cluster process. Because there are fundamental differences between the univariate and bivariate point pattern analysis, we present null models separately for the univariate and the bivariate case.

3.1. Complete Spatial Randomness (CSR)

3.1.1. Background

The simplest and most widely used null model for univariate point patterns is complete spatial randomness (CSR) that assumes no interactions between the points of the pattern. CSR can be implemented as a homogeneous Poisson process. Homogeneous means that the first-order intensity λ of the pattern is constant over the study region (there are no first-order effects), and Poisson means that the probability of finding k points in an area W follows a Poisson distribution with mean λW. Thus, any point of the pattern has an equal probability of occurring at any position in the study region, and the position of a point is independent of the position of any other point (i.e., points do not interact with each other).

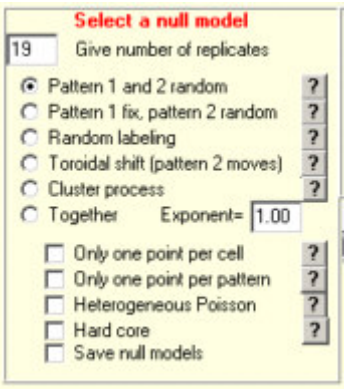
If a homogeneous pattern is spatially restricted by obstacles or environmental heterogeneity (e.g., differences in soil), the appropriate null model is CSR, but applied only within an irregularly shaped study region. Note that the numerical approach of *Programita* [equation I4 and equation I5] can deal with any irregularly shaped study region accommodated by the underlying grid.

3.1.2. Rectangular study region (CSR_1.res)

CSR is the basic null model for univariate patterns and most settings for CSR will apply equally for other univariate null models. Therefore, we explain all steps of the analysis for CSR in detail, but skip some of these details in the de-

scription of the other univariate null models.

1) highlight the data file "marcela.dat" in the window **Input data file**
2) select "List" in **How are your data organized**
3) select "Analyze all data in rectangle" **in Give modus of analysis**
4) select "Ring (Wiegand-Moloney)" in **Which method will you use?** if you like to use the *O*-ring statistic [and Circle (Ripley) if you like to use the *L*-function]
5) select an appropriate ring width *dr* in the box **ring width**. Usually a ring width of one cell is appropriate, however, if the intensity λ of points in the study region is too low, the graph of the *O*-ring statistic will be jagged and selection of a larger ring width *dr* is appropriate
6) click the button "change" in **set maximal radius rmax** to define the maximal scale *r* of the analysis and insert "40". A too large scale r_{max} will slow down *Programita*.
7) select "Data are given as list in grid" in **Select modus of data**
8) click button "Calculate index". You pattern appears on the left, and the *O*-ring function of your data appears on the right.
9) To determine Monte Carlo confidence intervals for CSR enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears:

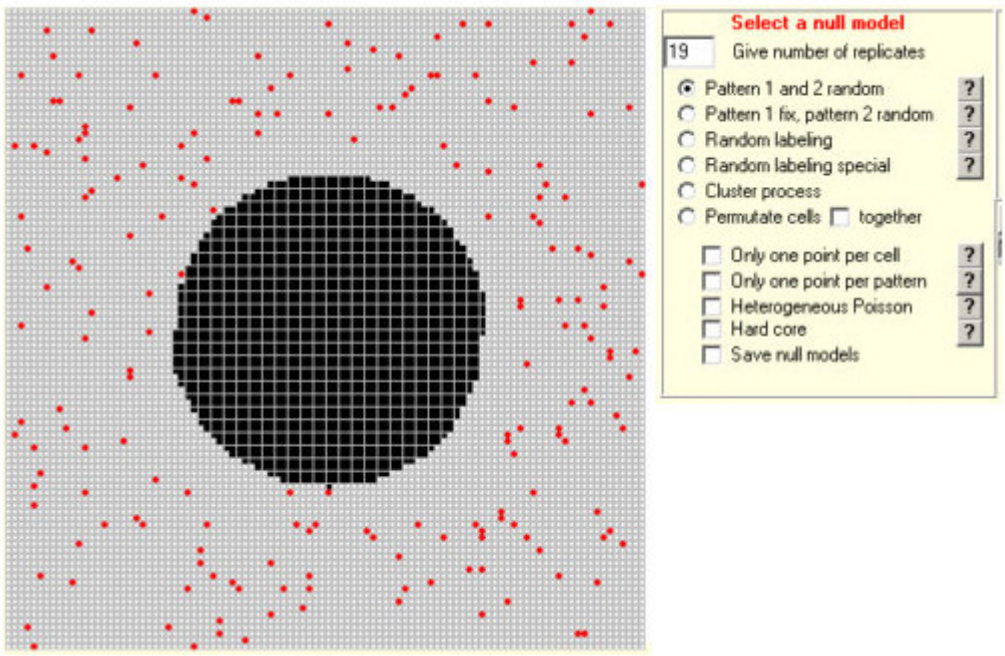


Select "Pattern 1 and 2 random".
10) You can change the number of replicate simulations of the null model in the box "Give number of replicates".
11) Press "Calculate index". *Programita* now performs the simulations of the CSR null model and shows you the pattern of the Monte Carlo null models. After termination of the simulations of the null model a graph appears, showing the *O*-ring function of your data and the confidence envelopes of your null model.
12) To save the results of the analysis press the button "Save results" that appears below the graph with the results of the univariate analysis and insert a name for the result file. The results file will be saved as ASCII file with a *.res extension in the same directory where programita.exe is located. It contains the settings of your analysis and the results of the univariate (and the bivariate) point-pattern analysis.
13) *Programita* uses the lowest and highest *O*(*r*) of the different simulations of the null model as confidence interval. However, it automatically produces two temporally

files (Uni_confidence.dat, Bi_confidence.dat) that contain the $O(r)$ for all simulations of the null model. The columns of these files are the scales r = 1, r_{max}, and the lines are the different simulations of the null model. The temporary files are overwritten if you start a new analysis.

3.1.3. Irregularly shaped study region, grid mode (CSR_2.res)

1) highlight the data file "CircularGap.dat" in window **Input data file.** This data file contains all cells with points of the pattern and all empty cells of the study region.
2) select "List" in **How are your data organized**
3) select "Data are given as list in grid" in **Select modus of data**
4) select "Irregularly shaped study region" **in Give modus of analysis**
5) select in box **ring width** a ring width dr = 2. For dr = 1 the O-ring statistic has a somewhat jagged plot at smaller scales r.
6) click the button "Calculate index". You pattern appears on the left, grey: the irregularly shaped study region, black: the area outside the study region, and red dots: the points of the pattern.



7) To determine Monte Carlo confidence intervals enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Pattern 1 and 2 random".
8) Press "Calculate index". *Programita* now performs the simulations of the null model and shows you the pattern of the Monte Carlo null models. After termination of the simulations a graph appears showing the O-ring function of your data and the confidence envelopes of your null model.

3.1.4. Irregularly shaped study region, point mode (CSR_3.res)

1) highlight the data file "Loch25.dat" in window **Input data file.** This data file was created by distributing points randomly over a 100 × 100 quadrate, but rejecting all points inside a circle with radius 25 located in the centre of the quadrate.
2) select "List" in **How are your data organized**
3) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
4) select "Irregularly shaped study region" **in Give modus of analysis.** A window opens and asks you (1) to confirm the cell size, and (2) to select a file that delineates the irregularly shaped study region. Click the button "Cell size" and the button "ok" in the window for defining the cell size (the cell size must be 1.00), next highlight the file "circle25.shp" and click "ok".
5) click button "Calculate index". You pattern appears on the left, grey: the irregularly shaped study region, black: the area outside the study region, and red dots: the points of the pattern.
6) To determine Monte Carlo confidence intervals for the CSR null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Pattern 1 and 2 random".
7) Press "Calculate index". *Programita* now performs the simulations of the null model and shows you the pattern of the Monte Carlo null models. After termination of the simulations a graph appears showing the *O*-ring function of your data and the confidence envelopes of your null model.

3.2. Heterogeneous Poisson process (HP)

3.2.1. Background

If a pattern is not homogeneous, the null model of CSR is not suitable for exploration of second-order characteristics, and a null model accounting for first-order effects (or for clustering) has to be used to reveal "true" second-order effects. The heterogeneous Poisson process is the simplest alternative to CSR if the pattern shows first-order effects. The constant intensity of the homogeneous Poisson process is replaced by a function $\lambda(x, y)$ that varies with location (x, y), but the occurrence of any point remains independent of that of any other. The intensity function $\lambda(x, y)$ determines the process completely, and numerical implementation of this null model is a matter of finding an appropriate estimate of the intensity function.

The grid-based implementation of *Programita* facilitates a simple method to implement the heterogeneous Poisson process using a moving-window estimate $\hat{\lambda}^R$ of the non-constant first-order intensity $\lambda(x, y)$:

$$\hat{\lambda}^R(x,y) = \frac{\textbf{Points}[C_{(x,y)}(R)]}{\textbf{Area}[C_{(x,y)}(R)]} \qquad \text{(HP1)}$$

where $C_{(x, y)}(R)$ is a circular moving window with radius R that is centered in cell (x, y), the operator **Points**$_2$[X] counts the points of pattern 2 in a region X, and the operator **Area**[X] determines the area of the region X. This is basically a kernel estimate with fixed bandwidth R (e.g., Bailey and Gatrell 1995). As edge correction, the number of points in an incomplete circle is divided by the proportion of the area of the circle that lies within the study region.

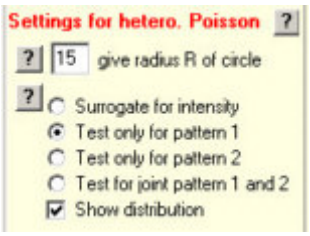
The algorithm for creating a pattern under a heterogeneous Poisson process is simple: a provisional point is placed at a random cell (x, y) in the study area, but this point is only retained with probability $\hat{\lambda}^R(x,y) / \max[\hat{\lambda}^R(x,y)]$ (the function max[X] determines the maximum of a variable X). This procedure is repeated until n points are distributed.

The moving window estimator $\hat{\lambda}^R(x,y)$ involves a decision on an appropriate radius R of the moving window. Because the bandwidth R is the scale of smoothing, possible departure from this null model may only occur for scales $r < R$, and for small moving windows it will closely mimic the original pattern,

whereas a large moving window approximates CSR.

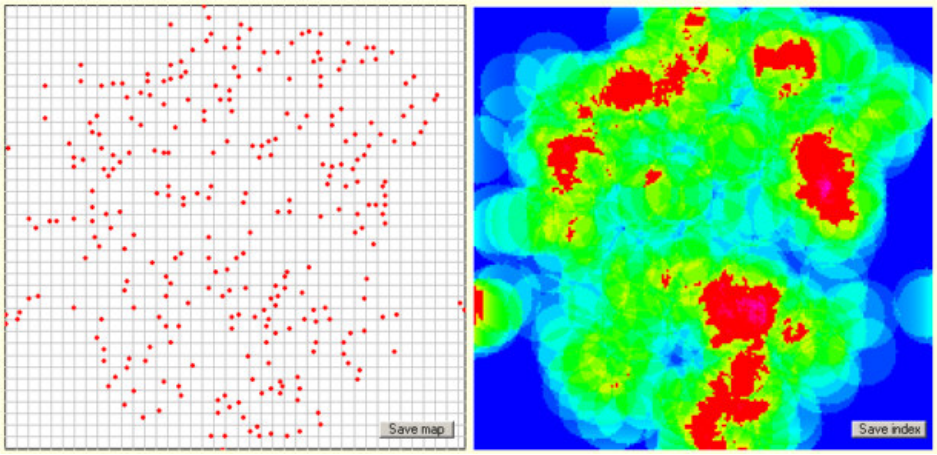
3.2.2. Example (HP_1.res)

1) highlight the data file "marcela.dat" in window **Input data file**
2) select "List" in **How are your data organized**
3) select "Analyze all data in rectangle" **in Give modus of analysis**
4) select "Data are given as list in grid" in **Select modus of data**
5) click button "Calculate index".
6) Enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Pattern 1 and 2 random" and enable the check box "Heterogeneous Poisson".
7) A window with settings for the moving window estimate of the heterogeneous Poisson appears:



Select "Test only for pattern 1" (pattern 2 does not exist in this example), and select a radius R=15 for the moving window.

8) click button "Calculate index". *Programita* now calculates the moving window estimate of the first-order intensity of the pattern (right graph):



9) Click "ok" at the message window. *Programita* now performs the simulations of the heterogeneous Poisson null model and shows the patterns of the simulated null models. After termination of the simulations a graph appears showing the O-ring function of your data and the confidence envelopes of the heterogeneous Poisson null model.

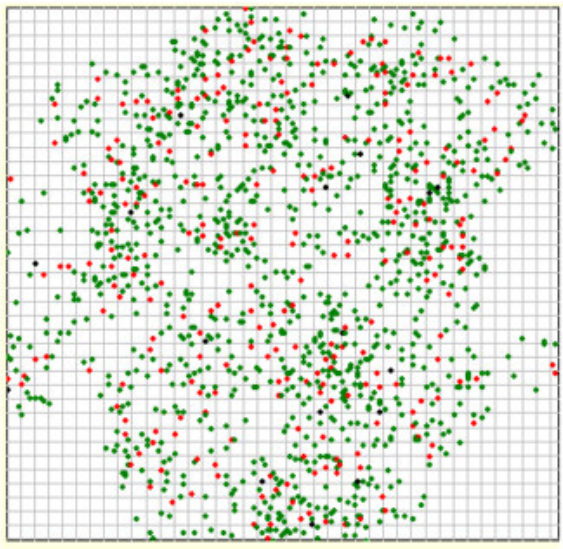
3.3. Random labeling (RL)

3.3.1. Background

Univariate random labeling is a somewhat different approach to correct for underlying environmental heterogeneity that can be used where a "control" pattern is available to act as surrogate for the varying environmental factor. The assumption of univariate random labeling is that the pattern of controls was created by the same stochastic process as the primary pattern ("cases"). Therefore, the n_1 cases represent a random sub-sample of the joined pattern of the n_2 control points and n_1 case points. The test is devised by computing the univariate *g*-function (or *L*-function) for the observed cases, then randomly re-sampling sets of n_1 points from the $(n_1 + n_2)$ points of the cases and controls to generate the confidence envelopes. Note that the univariate random labeling null model makes sense only if there are many more controls than cases. Univariate random labeling is closely related to bivariate random labeling.

3.3.2. Example (RL_1.res)

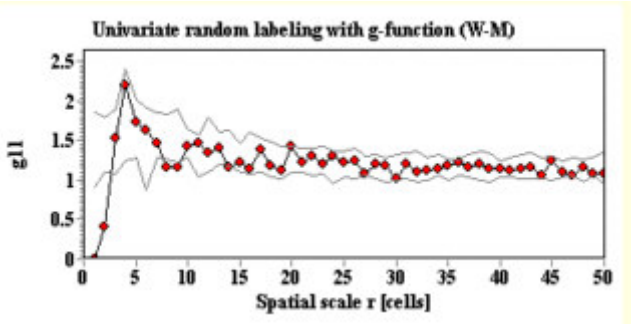
1) highlight the data file "marcela_RL.dat" in window **Input data file.** The data for pattern 1 (red dots) are identical to the data in example HP 1.res. The points of pattern 2 (green dots) were created by the heterogeneous Poisson null model using a moving window with radius $R = 15$ (as in example HP 1.res) and are thus a surrogate for the heterogeneous first-order intensity of the pattern:



2) select "List" in **How are your data organized**

3) select "Analyze all data in rectangle" in Give modus of analysis
4) select "Data are given as list in grid" in Select modus of data
5) click button "Calculate index".
6) enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Random labeling".
7) click "Calculate index". *Programita* now performs the simulations of the random labeling null model. After termination of the simulations a window appears:

8) enable O12 which is the appropriate setting for univariate random labeling. The other options are for the bivariate random labeling. *Programita* shows the univariate *g*11-function (instead of the *O*-ring statistic) together with the confidence envelopes for the univariate random labeling null model:



As expected, the confidence envelopes are non-symmetric to $g = 1$ because the null model corrected for the heterogeneity of the pattern.

3.4. Poisson cluster process (NS)

3.4.1. Background

The Poisson cluster process explicitly incorporates a clustering mechanism. Parent events form a CSR process and each parent produces a random number of offspring according to a probability distribution $f()$. Offspring are spatially distributed around their parent according to some bivariate probability density $h()$. The final pattern consists of the offspring only. To avoid edge effects, the parents must be simulated over a region larger than the study region but the offspring falling outside the study region are lost (Bailey and Gatrell 1995), or alternatively the simulation of the cluster process may be implemented on a torus. If the number of offspring follows a Poisson distribution and the location of the offspring, relative to the parent individual, have a bivariate, Gaussian distribution, the offspring follow a Neyman-Scott process (e.g., Diggle 1983). The K-function and the pair-correlation function $g(r)$ for the Neyman-Scott process are given by:

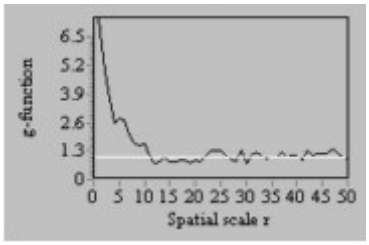
$$\begin{aligned} K(r,\sigma,\rho) &= \pi r^2 + \frac{1-\exp(-r^2/4\sigma^2)}{\rho} \\ g(r,\sigma,\rho) &= 1+\frac{\exp(-r^2/4\sigma^2)}{4\pi\sigma^2\rho} \end{aligned}, \qquad \text{(C1)}$$

where ρ is the intensity of the parent process, and σ^2 the variance of the Gaussian distribution that determines the locations of the offspring relative to the parent. The unknown parameters ρ and σ must be fit by comparing the empirical $\hat{K}(r)$ with the theoretical K-functions $K(r, \sigma, \rho)$ (see Diggle 1983).

Because σ is the standard deviation of the distance between each offspring and its parents, the cluster diameter yields ~ 2σ. For scales r below the cluster size (i.e., $r < \sigma$) the K-function can be approximated by $K(r) = r^2\pi + r^2/(4\rho\sigma^2)$ (Diggle 1983), and the L- and g-functions are approximated by

$$\begin{aligned} L(r,\sigma,\rho) &\approx r\left(\sqrt{1+\frac{1}{4\pi\rho\sigma^2}}-1\right) \\ g(r,\sigma,\rho) &\approx 1+\frac{1}{4\pi\rho\sigma^2} \end{aligned} \qquad \text{(C2)}$$

From equation C2 follows that the parameter ρ and σ cannot be determined independently if the *K*- and *g*-functions are fitted only at smaller scales *r*. Comparison of equation C2 with equation V1 shows that the compound parameter $\rho\sigma^2$ is directly related to the fraction *c* of the study region covered by the cluster: $c = \rho\sigma^2/(\rho\sigma^2 + 1/4\pi)$. Thus, the simplest property of clustering (the area of the study region covered by the cluster) is influenced in the same way by the intensity ρ of parents and by the variance σ^2 of the distance between each offspring and its parents. The proportion *c* does not change if the intensity ρ of parents increases but the variance σ^2 of the distance between each offspring and its parent decreases accordingly. Clearly, the shape of the *L*- and *g*-function at larger scales *r* may allow separating the two parameters ρ and σ^2.



The *g*-function is an important tool to visually estimate the cluster size. If $g(r)>1$ there are more points at distance *r* than expected for a random pattern, thus you have aggregation at scale *r*. For example, if $g(r) = 4$, you find 4 times more points at distance *r* from an arbitrary point of the pattern than you would expected under a random pattern.
In the small inlet figure above you see that $g>1$ for $r < 10$ cells and $g \approx 1$ for $r > 10$. Thus, the cluster size will be < 10 cells and you can restrict the interval of *r* for fitting the cluster model to, say $r_{max} = 20$. Deviations from the Neyman-Scott model at larger scales (which you may depict with the *L*-function) are caused by larger-scale effects.

3.4.2. Implementation of the fit of σ and ρ

Programita follows basically the approach of Diggle 1983 to fit equation C1 to your data, but uses the *L*-function instead of the *K*-function. *Programita* allows you to fit your data to a Neyman-Scott process by fitting the *g*-function, the *L*-function, or both, the *L*-function and the *g*-function simultaneously to the theoretical functions. The simultaneous fit of the *g*- and *L*-function usually works best since the *g*-function is more sensible at smaller scales, and the *L*-function at larger scales.

Programita minimizes three error functions with tuning constants r_0, r_{max}, and *c*:

$$
\begin{aligned}
error\_L &= \sum_{r=r0}^{r\max}[\hat{L}(r)^c - L(r,\sigma,\rho)^c]^2 \Big/ \sum_{r=r0}^{r\max}[\hat{L}(r)^c]^2 \\
error\_g &= \sum_{r=r0}^{r\max}[\hat{g}(r)^c - g(r,\sigma,\rho)^c]^2 \Big/ \sum_{r=r0}^{r\max}[\hat{g}(r)^c]^2 \qquad \text{(C3)} \\
error\_Lg &= \sqrt{error\_g * error\_L}
\end{aligned}
$$

that measure the discrepancy between model and data. The constant r_0 is the minimal scale of the fit, r_{max} the maximal scale of fit, and c a power transformation.

The error function *error_g* (or *error_L*) gives the fraction of the total sum of squares of the transformed empirical *g*-function (or *L*-function) which is not explained by the model. The error functions are normalized to make them comparable among fits with different adjustment intervals (r_0, r_{max}) and data sets and to facilitate construction of confidence intervals around the estimates of σ and ρ.

3.4.3. Selection of the tuning constants

An immediate question is how to choose appropriate values for the tuning constants. The first choice for the minimal scale is $r_0 = 1$, however, if strong small-scale effects (e.g., repulsion) overlay the clustering, one may select a minimal scale $r_0 > 1$ to omit interference of the small-scale deviation from clustering.

Note that *Programita* calculates the theoretical *K*-function using equation M2, thus accounting for the "memory" caused by a possible departure from the theoretical *K*-function at scale r_0.

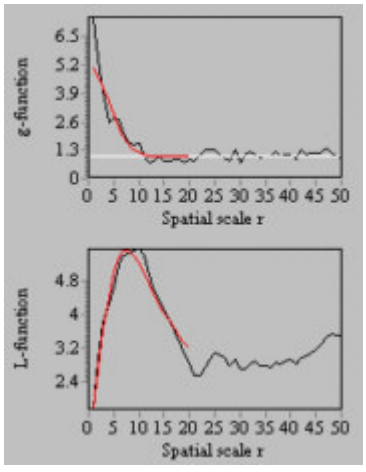


Figure C1. Selection of the

An appropriate choice for the maximal scale of the analysis for fitting a cluster process is the scale r at which the *g*-function well approximates the value 1. Remember, for $g(r) \approx 1$ you find points at distance r as frequent as under a random pattern. Thus for $g(r) \approx 1$ there is no aggregation at this scale. In the example on the left the choice would be $r_{max} = 20$. The *g*-function is not very sensible to larger-scale effects (i.e., effects larger than the cluster size, for example clustering of the clusters at larger scales), therefore selection of the maximal scale r_{max} is not very sensitive if you use only

maximal scale r_{max} of the fit. For fitting $g(r)$ a transformation c_g = 0.5 was used, and for fitting $L(r)$ a transformation c_L = 1.

the g-function for fitting the parameters σ and ρ. However, the L-function is sensible to such larger-scale effects (figure C2, left), and if you use the L-function for fitting the parameters σ and ρ you may restrict the range of r_{max} (as done in fig. C1). Alternatively, for depicting the peak of the L-function (which describes the smaller-scale clustering you are interested in), you may select a transformation with a large value of c (figure C2 right).

A power transformation with $c > 1$ weights larger values of $L(r)$ or $g(r)$ more than a transformation with $c = 1$ (figure C2), whereas a transformation with $c < 1$ weights larger differences less. Thus, to produce a fit that reproduces a peak in $L(r)$ or $g(r)$ well you may select a larger value of c (e.g., $c = 4$ as in figure C2

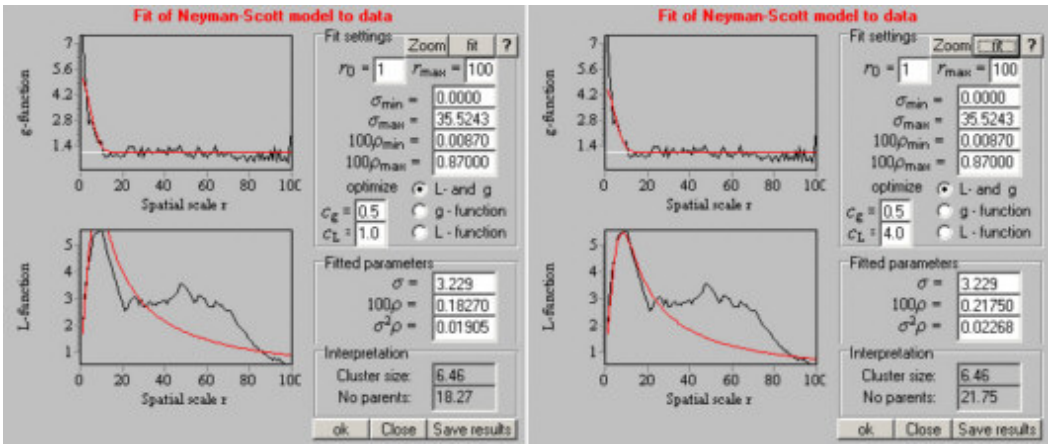


Figure C2. Influence of the tuning constant c (equation C3) on the fit of the L-function. A weight $c > 1$ weights larger values of $L(r)$ stronger than smaller values, and for $c = 4$ the fitting procedure fits basically the peak of the L-function (at scales $0 < r < 20$).

right). However, to fit intermediate values of $L(r)$ or $g(r)$ well, select $c < 1$ (e.g., $c = 0.5$ for fitting $g(r)$ in figure C2 left). A reasonable range for the power transformation c is $c \in (0.1, 4)$.

Programita allows you to repeat the fit with different selections of the tuning constants and to visually control the fit. There is no harm to try few different values of r_0, r_{max}, and c in order to assess the extent to which the results are sensitive to these choices.

3.4.4. The settings window for the fit

The settings window Fit of Neyman-Scott model contains the settings for the fitting procedure and shows the results and the interpretation of the best fit. The win-

dow "Fit settings" allows you to select:

- r_0, the minimal scale used to fit the data
- r_{max}, the maximal scale used to fit the data
- σ_{min} and σ_{max}: the minimal and maximal value of σ used in the fit.
- $100\rho_{min}$ and $100\rho_{max}$: the minimal and maximal value of ρ used in the fit.
- c_g and c_L: the power transformation for the g- and L-function, respectively
- whether you optimize only the *g*- or the *L*-function, or both simultaneously

After clicking the button "fit" *Programita* performs the fit with the settings specified in the window "Fit settings". If you click the button "Zoom" *Programita* determines the intervals (σ_{min}, σ_{max}) and ($100\rho_{min}$, $100\rho_{max}$) that enclose the area in parameter space with an acceptable fit (i.e., *error_g*, *error_L*, or *error_gL* < 0.025). Next click "fit" and *Programita* runs a parameter search in the optimized area in the parameter space. After termination of the parameter search, *Programita* shows the error-surface of the fit (figure C4). The black area is the region in parameter space with an unacceptable fit with error > 0.025. If the best fit has an error > 0.025, *Programita* gives you a warning and continues with an acceptable error of 0.05 instead of 0.025.

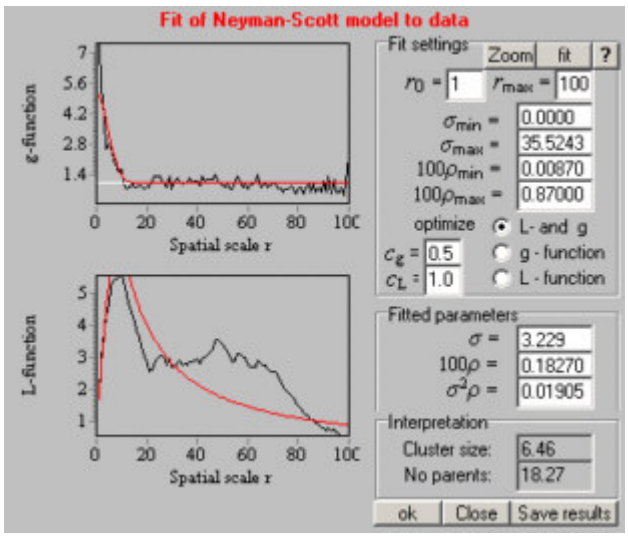
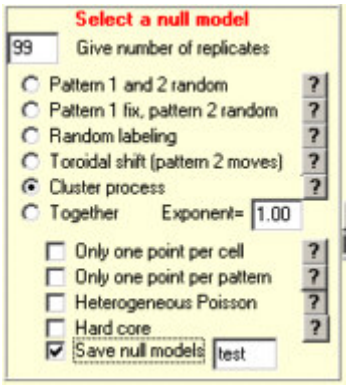


Figure C3. The window **Fit of Neyman-Scott model** to data that contains the settings for the fitting procedure (window "Fit settings") and shows the results (window "Fitted parameter") and the interpretation of the best fit (window "Interpretation").

If you are satisfied with the fit, you can safe the settings and results of your fit in an ASCII file by clicking the button "Save results". This file will have the extension *.fit. To apply the Neyman-Scott cluster null model with the parameters

determined during the fitting procedure click the button "ok". The window Fit of Neyman-Scott model disappears and after clicking "Calculate index "*Programita* starts with the simulations of the null model.

Note that *Programita* uses the parameters ρ and σ specified in the window “Fitted parameters” for the simulation of the null model. This offers you the possibility to create artificial data sets with any value of ρ and σ you find appropriate. To save the artificial data sets enable the check box “Save null models” in the window containing the settings of the null model.



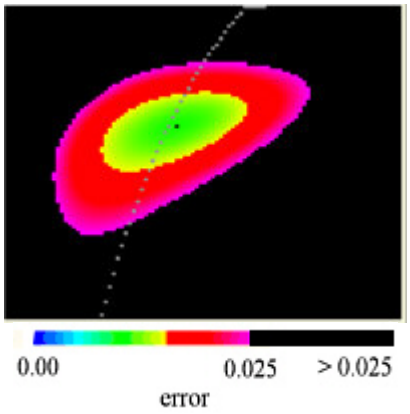


Figure C4. The error surface for the zoomed parameter area. The x-axis gives the parameter σ, the y-axis the parameter ρ. Note that the values of ρ (the y-axis) are shown with increasing values from top to bottom. Black: Non-acceptable fit with error > 0.025. Spectral colors from blue to magenta: increasingly poorer fit.

3.4.5. Constructing confidence intervals for σ and ρ

The error functions *error_g*, *error_L*, and *error_Lg* give the fraction of the sum of squares of the transformed L- or g-functions of the data not explained by the fit. Therefore, confidence intervals for the estimate of the parameter σ and ρ can be estimated by determining the intervals in σ and ρ for which the error is smaller than a certain level of say, 0.025 or 0.01.

The data files with the *.fit extension contain a list with the parameter values and the corresponding errors. You can use this data to produce a contour plot of the error and constructing the confidence intervals for the estimates of σ and ρ (figure C5).

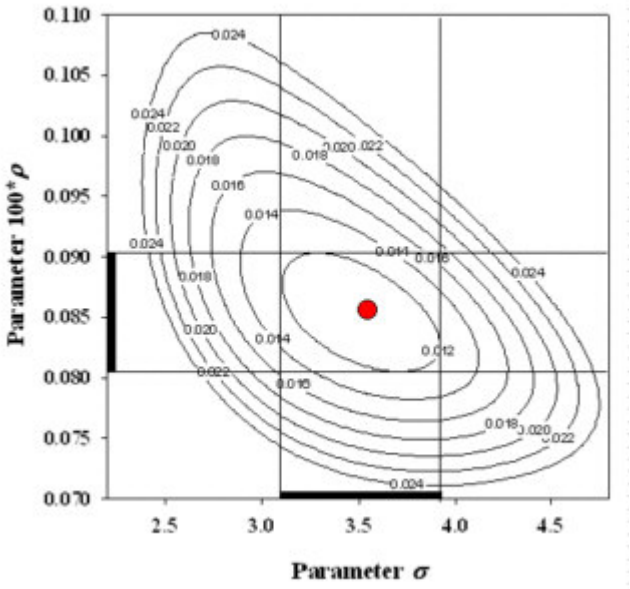
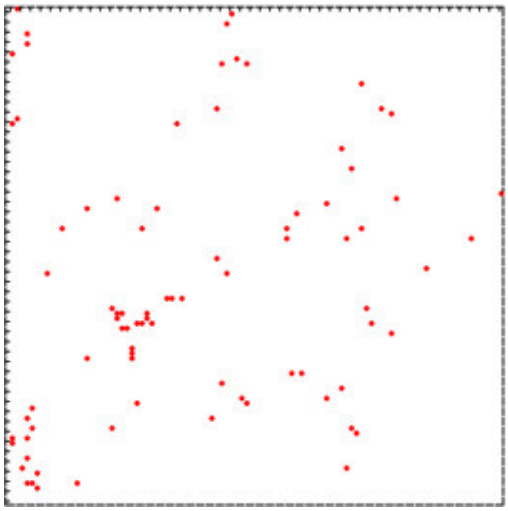


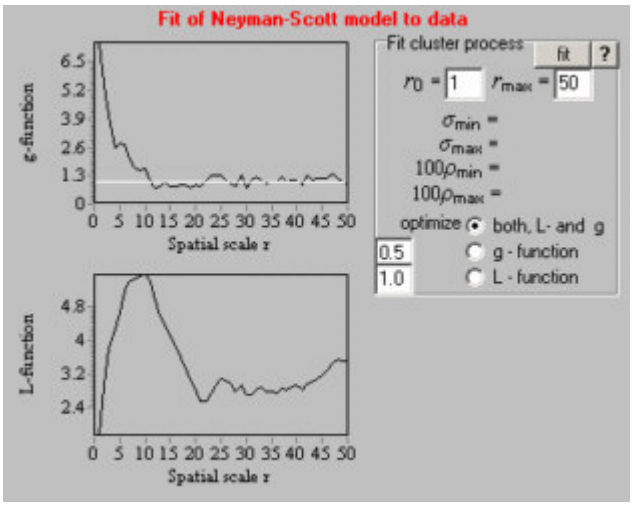
Figure C5. Contour plot of the error in dependence on the parameters σ and ρ of the fit for determination of confidence intervals for σ and ρ. The confidence intervals for an error < 0.012 are shown as bold intervals at the axes. The best fit is indicated as red dot.

3.4.6. Univariate cluster process (NS_1.res)

1) Highlight the data file "adults_real.dat" in window **Input data file.** This data set gives the location of adult trees at a meter scale, but has a resolution of 1 centimeter.
2) select "List" in **How are your data organized**
3) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "5.00". Thus, the cell size is 5m × 5m.
4) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to $r_{max} = 50$.
5) click button "Calculate index", *Programita* shows you the pattern

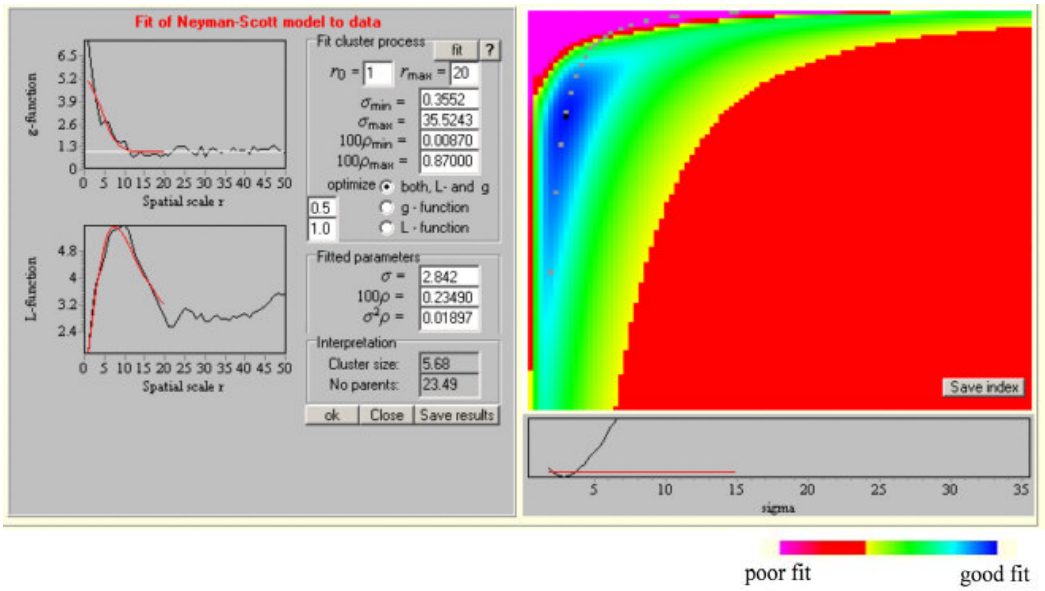


and calculates the *O*-ring function of the data.
6) To determine Monte Carlo confidence intervals for the Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "univariate Neyman-Scott" and press ok.
7) *Programita* now calculates the *g*- and the *L*-function for $r = 1$ to r_{max} and the window **Fit of Neyman-Scott models to data** appears.



8) You can specify the tuning constants r_0, r_{max}, and c for the fit in the window "Fit cluster process". Visualization of the g-function in the window w Fit of Neyman-Scott models to data helps you to find the appropriate range r_0 - r_{max}. The g-function approximates the theoretical value for a random pattern (g = 1) at scale r = 10. In practice, a good choice for r_0 is more or less the double of this scale, thus select r_{max} = 20. The default r_0 = 1 is appropriate since no repulsion occurs at small scales, and the default power transformations c = 0.5 for the g-function and c = 1 for the L-function are reasonable starting values. To optimize simultaneously the g- and the L-function enable "both, L- and g-function".

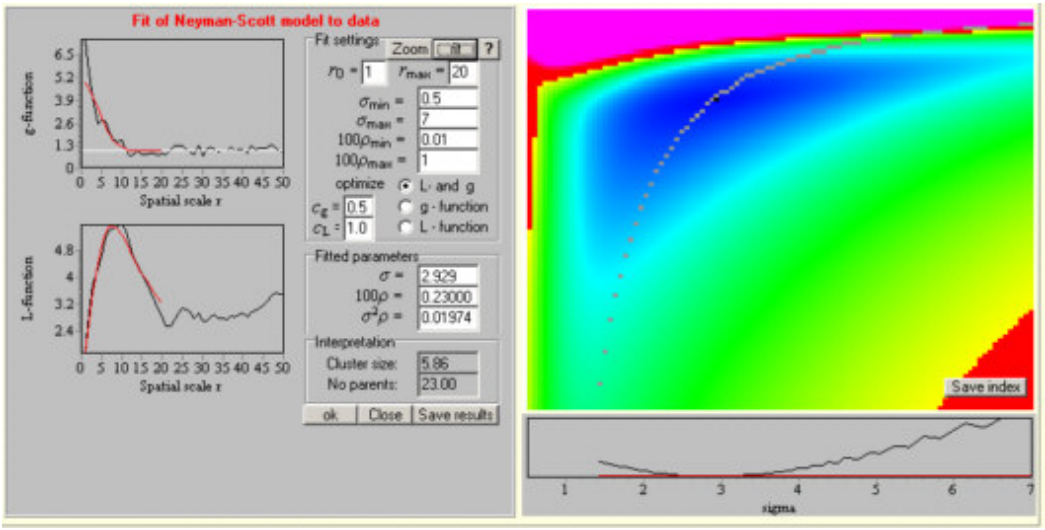
9) Click the button "fit" and *Programita* searches the parameters of the Neyman-Scott model that simultaneously fits the g- and L- function of your data best (red line: fit, black line: data):



10) *Programita* automatically calculates initial intervals for the two parameters σ and ρ of the Neyman-Scott process. The minimal and maximal coordinates for σ (σ_{min}, σ_{max}) and ρ

(ρ_{min}, ρ_{max}) are shown in the box "Fit cluster process". *Programita* uses 100 parameter values for ρ and σ [equidistantly distributed within the intervals (σ_{min}, σ_{max}) and (100*ρ_{min}, 100* ρ_{max})] to find the parameters of the Neyman-Scott model that simultaneously fit the *g*- and *L*- function of your data best (See "Implementation of the fit ...").

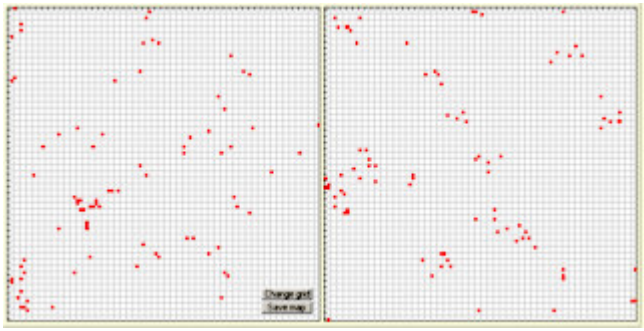
11) The estimates of σ_{best} ρ_{best} are shown in the window "Fitted parameters", and the best fits of the *g*-and the *L*-function are shown as red line in the two graphs on the left, your data are shown as black line. Additionally, *Programita* shows the deviation between data and fit (the error surface, right figure) plotted in the σ - ρ plane. The black cell indicates the estimated values of σ and ρ, dark blue color indicates a small deviation between data and fit, and colors with increasing spectral color indicate successively poorer fits.
12) The small graph below the deviation shows the values of the deviation for points that satisfy $\rho\sigma^2$ = ρ_{best} σ^2_{best}, plotted over the value of σ, and the red line is the an *error gL* < 0.025 for an acceptable fit.
13) The points that satisfy $\rho\sigma^2$ = ρ_{best} σ^2_{best} are shown as grey cells. Remember that equation C2 implies that the value of $\rho\sigma^2$ may be estimated with higher precision that the values of ρ and σ separately. This plot helps you to control for this effect.
14) To optimize the settings of the fit, you can manually change the minimal and maximal coordinates for σ (σ_{min}, σ_{max}) and 100*ρ (100*ρ_{min}, 100* ρ_{max}) in the window "Fit cluster process". As help you may compare the current settings of σ and ρ (window "Fit cluster process") with the estimated values σ_{best} ρ_{best} (window "Fitted parameters"), and adjusting the new intervals only inside the area of a good fit as indicated by the plot of the deviation. In the example, the interval for σ is (0.5, 7), and the interval for 100*ρ is (0.01, 1):



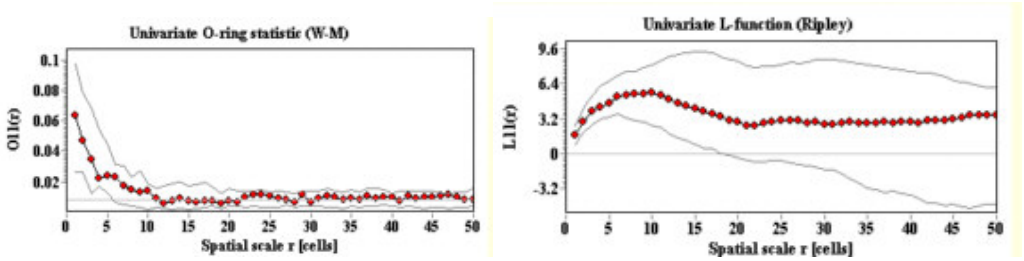
15) Alternatively, you can use the "Zoom" function of programita. If you click the button "Zoom", *Programita* determines the intervals (σ_{min}, σ_{max}) and (100ρ_{min}, 100ρ_{max}) that enclose the area in parameter space with an acceptable fit with error g, error L, or error Lg < 0.025. Next click

"fit" and *Programita* runs another parameter search in the optimised area of the parameter space (see figure C4).

16) You can save the results of the fit in an ASCII file (click button "Safe results" and provide a data name). The file will have the extension *.fit. The results file contains a list with the parameter values and the error. You can use this data for producing a contour plot of the error and constructing the confidence intervals for the estimates of σ and ρ (figure C5).
17) Once you are satisfied with the fit, click "ok". The window **Fit of Neyman-Scott models to data** disappears and after clicking "Calculate index" *Programita* continues with the simulations of the Neyman-Scott null model for estimation of confidence envelopes:



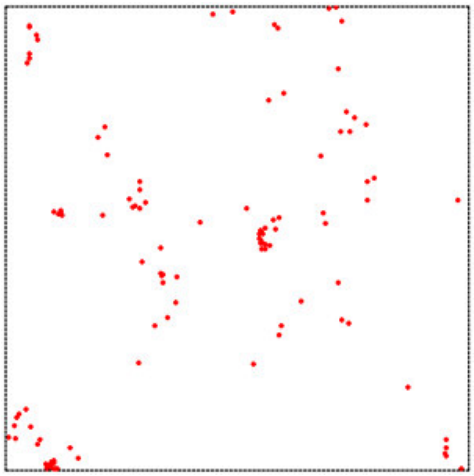
The pattern of the data is shown on the left, and the simulated patterns of the null models are show on the right. After termination of the simulations, *Programita* shows the results of the point-pattern analysis at the right instead of the simulated patterns:



18) The simulation of 99 replicates of the Neyman-Scot null model with σ_{best} = 2.888 and ρ_{best} = 0.0023303 show that the data are well within the confidence envelopes of the null model. The left figure above shows the univariate *O*-ring statistic and the figure on the right the univariate *L*-function.
19) Note that the confidence envelopes of the *L*-function are relatively wide for larger scales. This is because the *L*-function is more sensitive at larger scales.

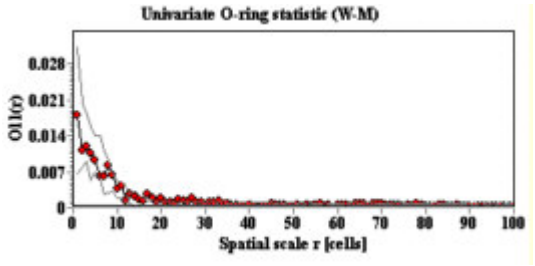
3.4.7. Univariate cluster process and recruits (NS_2.res)

1) This example analyzes the spatial pattern of recruits of example NS 4.res. We show that the recruits are clustered at two different scales.
2) Highlight the data file "recruits.dat" in window **Input data file.** This data set gives the location of recruits in cells with a cell size of 1 m^2.
3) select "List" in **How are your data organized**
4) select "Data are given as list in grid" in **Select modus of data**.
5) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to r_{max} = 30.
6) click button "Calculate index", *Programita* shows you the pattern:



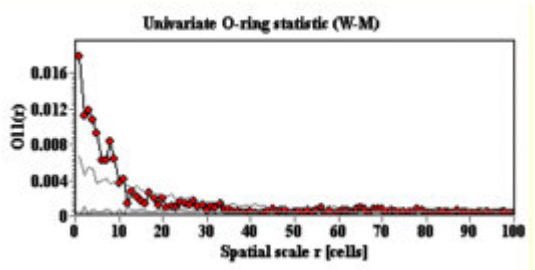
and calculates the *O*-ring function of the data. Visualization of the data show that the recruits are clearly clustered.
7) In a first step we investigate the small-scale clustering of the recruits. To determine Monte Carlo confidence intervals for a Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "univariate Neyman-Scott" and press ok.
8) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. You can specify the tuning constants r_{min}, r_{max}, and c for the fit in the window "Fit cluster process":
9) Select r_{max} = 1 and r_0 = 15 since we are interested in the aggregation at small scales. To optimize simultaneously the g- and the *L*-function enable "both, L- and g-function".
10) Click the button "fit" and *Programita* searches the parameters of the Neyman-Scott model that simultaneously fit the *g*- and *L*- function of your data best (red line: fit, black line: data). We find
    - σ_{best} = 4.8 and ρ_{best} = 0.000135 (some 33 parents)

11) The simulation of 19 replicates of the Neyman-Scot null model show that the data are at small scales well within the confidence envelopes of the null model (NS_2_small_scale.res), but the data are at larger scales partly above the confidence interval (i.e., r = 13, 17-20, and 24-27):



12) Next we investigate the clustering at larger scales (i.e., r = 15 - 100). Repeat the steps 1 - 8 and select r_{max} = 15 and r_0 = 100. The default power transformations c = 0.5 for the g-function and c = 1 for the L-function are reasonable starting values. To optimize the g- and the L-function simultaneously enable "both, L- and g-function".
13) Click the button "fit" and *Programita* searches the parameters of the Neyman-Scott model that simultaneously fits the g- and L- function of your data best (red line: fit, black line: data). To optimize the parameter fit, press the button "Zoom". *Programita* now determines the probable range of the parameters. We find
    - σ_{best} = 14.4 and ρ_{best} = 0.000095 (some 24 parents).

    Press the button "ok" and then "Calculate index".
14) The simulation of 19 replicates of the Neyman-Scot null model show that the data are for larger scales well within the confidence envelopes of the null model (NS_2_larger_scale.res):



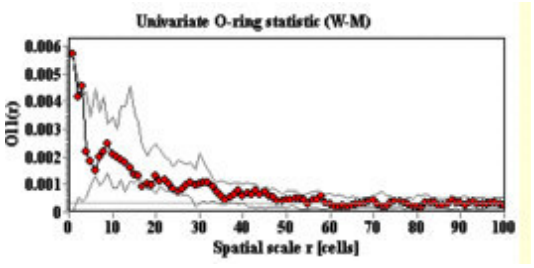
but as expected, recruits are significantly clustered at small scales r = 1 - 10. Overall we find that the recruits are clustered at two different spatial scales. To adequately describe such a situation we would need a double-cluster model where the parents events are not a random pattern, but follow itself a Neyman-Scott process.

3.4.8. Univariate cluster process and adult trees (NS_3.res)

1) This example analyzes the spatial pattern of adult trees (which we already analyzed in example NS 1.res), but now at the same spatial resolution as the pattern of recruits in the previous example (NS 2.res).
2) Highlight the data file "adults.dat" in window **Input data file.** This data set gives the location of adult trees in cells with a cell size of 1 m^2.
3) select "List" in **How are your data organized**
4) select "Data are given as list in grid" in **Select modus of data**.
5) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to r_{max} = 100.
6) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data. The visualization of the data shows that the adults are clearly clustered.
7) To determine Monte Carlo confidence intervals for the Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "univariate Neyman-Scott" and press ok.
8) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} = 100 and the window **Fit of Neyman-Scott models to data** appears. You can specify the tuning constants r_{min}, r_{max}, and c for the fit in the window "Fit cluster process":
9) Select r_{max} = 1 and r_0 = 100 since we are interested in the overall aggregation of the adults. The default power transformations c = 0.5 for the *g*-function and c = 1 for the *L*-function are reasonable starting values. To optimize the *g*- and the *L*-function simultaneously enable "both, L- and g-function".
10) Click the button "fit" and *Programita* searches the parameters of the Neyman-Scott model that simultaneously fits the *g*- and *L*- function of your data best (red line: fit, black line: data). To optimize the parameter fit, press the button "Zoom". *Programita* now determines the probable range of the parameters. We find
    - σ_{best} = 14.1 and ρ_{best} = 0.000083 (some 21 parent events).

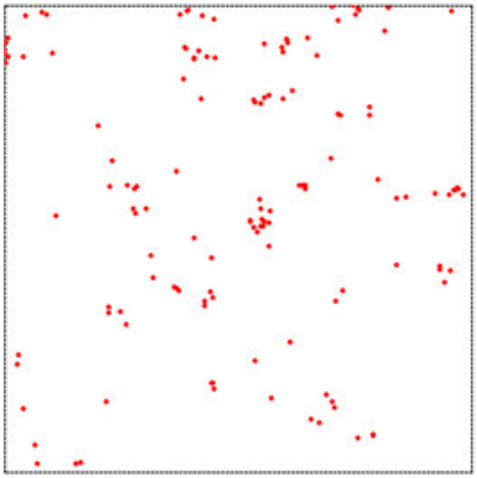
    This estimates accord well with the results from example NS 1.res (which used a cell size of 25m^2 instead of 1m^2) with σ_{best} = 5*2.88 = 14.4 and 23 parents. Note that these estimates are strikingly similar to the estimates for recruits at larger scales (i.e., a cluster size of some 29m, and some 24 parent events) obtained in the previous example.
11) Before continuing with the simulation of the null model select a ring width of 3 cells; otherwise the *O*-function will be slightly jagged. Press the button "ok" and then "Calculate index".
12) The simulation of 19 replicates of the Neyman-Scot null model show that the data are well within the confidence envelopes of the cluster null model (NS_3.res. However, at scales r = 1 - 3 there is a tendency to a stronger clustering than accommodated by the Neyman-Scott cluster

null model which, however, is not significant:



3.4.9. Univariate cluster process and dead trees (NS_3b.res)

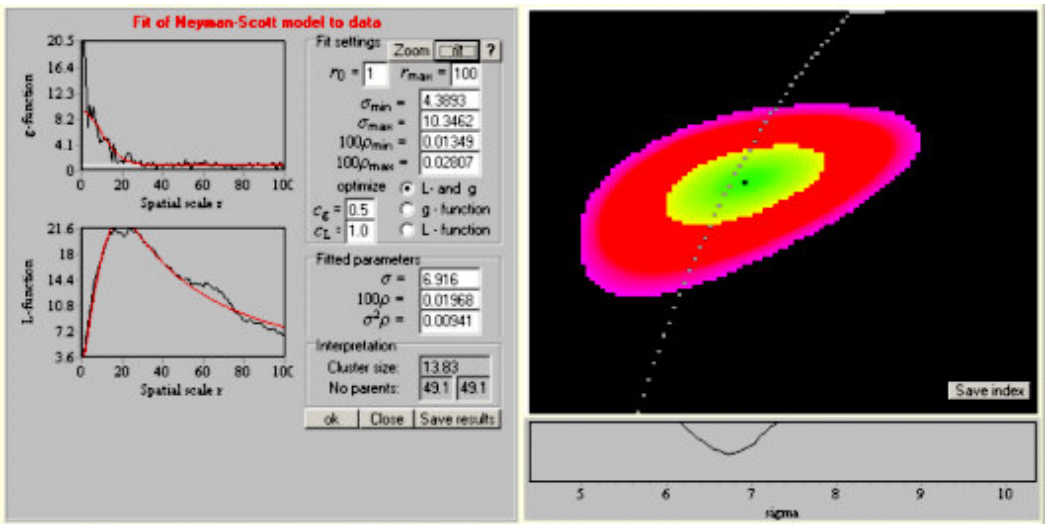
1) This example analyzes the spatial pattern of dead trees at the same spatial resolution as the pattern of adult trees (NS 2.res) and recruits (NS 3.res) in the previous examples.
2) Highlight the data file "dead.dat" in window **Input data file.** This data set gives the location of adult trees in a resolution of 1cm.
3) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00". Thus, the cell size is 1m × 1m.
4) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to r_{max} = 100.
5) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data. The visualization of the data shows that the adults are clearly clustered:



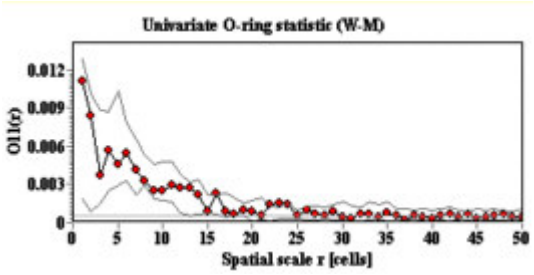
6) To determine Monte Carlo confidence intervals for the Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "univariate Neyman-Scott" and press ok.
7) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} = 100 and the window **Fit of Neyman-Scott models to data** appears. You can specify the tuning constants r_{min}, r_{max}, and

c for the fit in the window "Fit cluster process".

8) Select r_{max} = 1 and r_0 = 100 since we are interested in the overall aggregation of the adults. The default power transformations *c* = 0.5 for the *g*-function and *c* = 1 for the *L*-function are reasonable starting values. To optimize the *g*- and the *L*-function simultaneously enable "both, L- and g-function".
9) Click the button "fit" and *Programita* searches the parameters of the Neyman-Scott model that simultaneously fits the *g*- and *L*- function of your data best (red line: fit, black line: data). To optimize the parameter fit, press the button "Zoom". *Programita* now determines the probable range of the parameters:



10) We find σ_{best} = 6.92 and ρ_{best} = 0.0002 (some 49 parents). Interestingly, adult trees do not show the tendency to double-clustering of adult trees (NS 3.res) or recruits (NS 2.res).
11) We can compare the *g*- or *K*-functions of different patterns through the compound parameter $\sigma^2\rho$. This compound parameter determines the steepness of the *L*-function at small scales and the proportion *c* of the study region covered by the cluster (equation C2). We find that dead trees are most clustered (their cluster covers some 11% of the study region), followed by adult tress (their cluster covers some 17% of the study region) and the less clustered are recruits (their larger-scale cluster covers some 20% of the study region).
12) Click the button "Calculate index". The simulation of 19 replicates of the Neyman-Scot null model show that the data are well within the confidence envelopes of the cluster null model (NS_3b.res):


Univariate O-ring statistic (W-M)
O11(r)
0.012
0.009
0.006
0.003
0
0 5 10 15 20 25 30 35 40 45 50
Spatial scale r [cells]

3.5. Univariate double-cluster process (DC)

3.5.1. Background

In some cases a univariate pattern may show clustering at two different scales. Imagine a forest where the suitable habitat for a given species is heterogeneously distributed, perhaps due to different orientation on mountain slopes of local differences in soil. This may cause a patchy (or clustered) distribution of this species at a larger scale. However, a limited seed dispersal radius or much localized safe sites (created by a dead tree) may cause a small-scale clustering of the recruits, and as a result the overall pattern of the recruits follows a double-clustered structure which cannot be well described by the simple cluster process equation C1.

We will now introduce the double-clustered univariate Neyman-Scott process which is an extension of the univariate Neyman-Scott process equation C1:

$$\begin{aligned} K(r,\sigma_2,\rho_2) &= \pi r^2 + \frac{1-\exp(-r^2/4{\sigma_2}^2)}{\rho_2} \\ g(r,\sigma_2,\rho_2) &= 1 + \frac{\exp(-r^2/4{\sigma_2}^2)}{4\pi{\sigma_2}^2\rho_2} \end{aligned} \qquad \text{(DC1)}$$

where ρ_2 is the intensity of the parent process, and ${\sigma_2}^2$ the variance of the Gaussian distribution that determines the locations of the offspring relative to the parent. The simple cluster process equation DC1 assumes that the parents show a random pattern whereas the double-clustered process assumes that the parents follow itself the cluster process equation C1 with parameters ${\sigma_1}^2$ and ρ_1. To not mix up the two types of parents we define three types of points:

- parents
- type 1 points (= the offspring of the parents)
- type 2 point (= the offspring of type 1 points)

but analyze the univariate structure of type 2 points without explicit knowledge of the locations of the of parents and type 1 points. The bivariate case where the locations of the type 1 points are known, however, is discussed in a separate section for bivariate null models (Bivariate double-cluster process for antecedent condition).

The univariate *g*- and *K*-functions expected under a univariate double-clustered

Neyman-Scott process are:

$$g_{22}(r,\sigma_1,\rho_1,\sigma_2,\rho_2) = 1+\frac{1}{\rho_2}\frac{\exp(-r^2/4\sigma_2^2)}{4\pi\sigma_2^2}+\frac{1}{\rho_1}\frac{\exp(-r^2/4\sigma_{sum}^2)}{4\pi\sigma_{sum}^2} \quad \text{(DC2)}$$

with $\sigma_{sum}^2 = \sigma_1^2 + \sigma_2^2$

$$K_{22}(r,\sigma_1,\rho_1,\rho_2,\sigma_2) = \pi r^2+\frac{1-\exp(-r^2/4\sigma_2^2)}{\rho_2}+\frac{1-\exp(-r^2/4\sigma_{sum}^2)}{\rho_1} \quad \text{(DC3)}$$

with $\sigma_{sum}^2 = \sigma_1^2 + \sigma_2^2$

with the four parameters:

- σ^2_2, the parameter that gives the variance of the locations of type 2 points relative to their parents (= type 1 points).
- ρ_2, the intensity of the parents of the type 2 points.
- σ^2_1, the parameter that gives the variance of the locations of type 1 points relative to their parents.
- ρ_1 is the intensity of the parents of type 1 points.

The first term in equation DC2 (= 1) describes the situation where the points of the univariate pattern are independent from each other (i.e., type 2 points are not clustered around their type 1 parents), the second term describes the effect of clustering of type 2 points around their parents, and the third term describes the compound effect of the clustering of type 1 points and the clustering of type 2 points around type 1 points. The variance σ_{sum}^2 is the combined variance that describes the interaction of the clumping at the two scales σ^2_2 and σ^2_1.

If the parents are a random pattern (i.e., $\sigma^2_1 \to \infty$) then $\sigma^2_{\text{sum}} \to \infty$ and consequently, the third term disappears and equation DC2 collapses back to equation DC1. If type 2 points are independent from their parents (i.e., $\sigma^2_2 \to \infty$) it follows that $\sigma^2_{\text{sum}} \to \infty$ and equation DC2 collapses, as expected, back to a CSR process with $g(r) = 1$.

Fitting of the observed *g*- and *K*-functions to a double-cluster process as described by equation DC2 and equation DC3 could be done analogously to that of the simple univariate cluster process, but fitting all 4 parameters simultaneously instead of two parameters simultaneously as done for fitting the simple cluster process. This procedure is not yet implemented in *Programita*. However, we would expect that the scale of clustering of the parents is larger than the scale of clustering of type 2 points around their type 1 parents (i.e., $\sigma^2_1 << \sigma^2_2$ and consequently $\sigma^2_{\text{sum}} << \sigma^2_2$). Otherwise, a situation with $\sigma^2_1 > \sigma^2_2$ does not

really make sense because in this case the smaller-scale clusters of the parents would basically function like one parent and the pattern can not be distinguished from a simple clustered pattern.

If the parents show clustering at a larger scale and if the clustering of type 2 points around type 1 points occurs at a smaller scale, however, we can separate the scales because in this case the contribution of the clustering of the offspring to equation DC2 disappears for scales r above the cluster size $2\sigma_2$. Therefore, a fit of the data to the expected g- and K-function of a simple cluster process equation DC1 at larger scales $r > 2\sigma_2$ will reveal the unbiased parameters σ_1 and ρ_1 of the larger-scale clustering of the parents. To estimate the missing parameters σ_2 and ρ_2 of the small-scale clustering of type 2 points, we therefore first determine the (unbiased) parameters σ_1 and ρ_1 using a simple cluster process and use in a next step equations equation DC2 and equation DC3 to obtain an unbiased estimate of the smaller-scale clustering of the offspring.

We illustrate the analysis of double-clustered univariate patterns with three examples, three artificial data sets with

1. $\sigma^2_1 >> \sigma^2_2$
2. $\sigma^2_1 << \sigma^2_2$

and the data of the recruits we already analyzed in example NS_2 and which showed indications for double-clustering.

3.5.2. Double-cluster process $\sigma^2_1 >> \sigma^2_2$ (DC_1.res)

This pattern was created using a parents pattern with 136 points and parameters

- $\sigma_1 = 14.1$, $\rho_1 = 0.000169$ (some 42 parents)

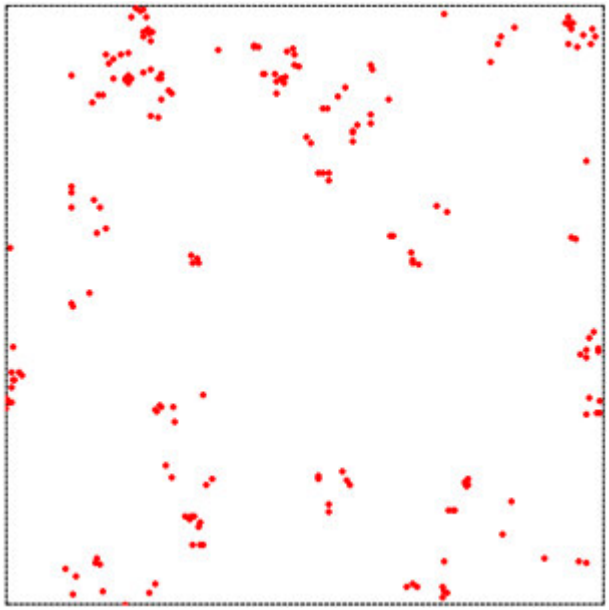
and the parameters of the offspring were

- $\sigma_2 = 4$, $\rho_2 = 0.00054$ (some 136 parents, i.e., all type 1 points are parents)

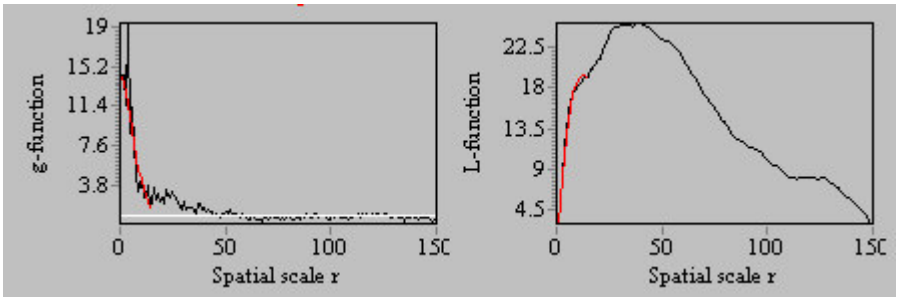
We analyze first the clustering of the pattern at larger scale using the simple cluster process equation DC1 and apply then the double-clustered model to determine the parameters of the small-scale clustering.

First step: univariate analysis at larger scales with simple cluster model

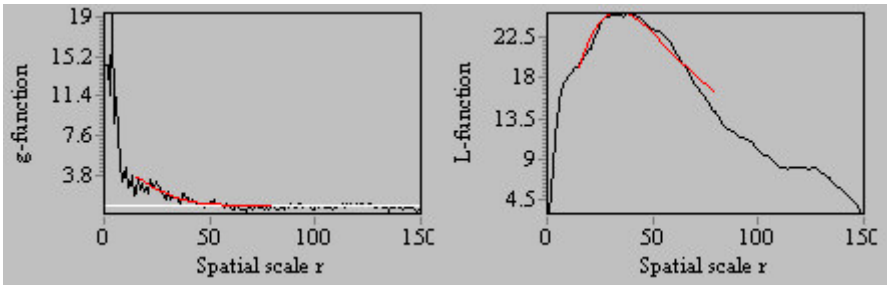
1) Highlight the data file "DC1_uni_c.dat" in window **Input data file**.
2) select "List" in **How are your data organized**
3) select "Data are given as list in grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
4) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data:



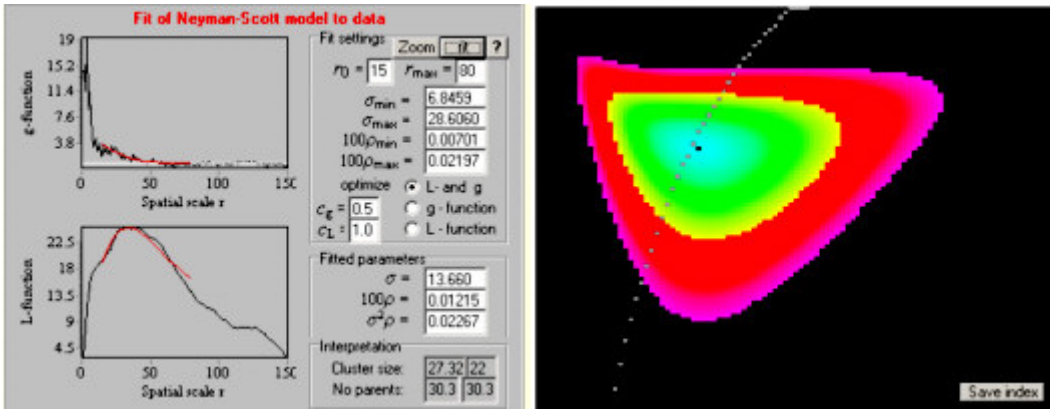
5) For univariate analysis of this pattern at larger scale enable enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "univariate Neyman-Scott" and press ok.
6) *Programita* calculates the *g*- and the *L*-function of the data and the window **Fit of Neyman-Scott models to data** appears. You can specify the tuning constants r_{min}, r_{max}, and *c* for the fit in the window "Fit cluster process".
7) To estimate the scale where the contribution of the small-scale clustering to the double-clustered g-function disappears, select r_{max} = 1 and r_0 = 15 and click the button "fit". *Programita* now shows the *L*- and *g*-function of the data and the fit at small scales:



8) The contribution of small-scale clustering to the *g*-function may disappear at scale *r* = 15 and the *g*-function approximates the value 1 roughly at scale r = 80. Therefore, select r_{max} = 15 and r_0 = 80 for assessment of the parameters of the large-scale clustering and click the button "fit". *Programita* now shows the *L*- and *g*-function of the data and the fit at small scales:

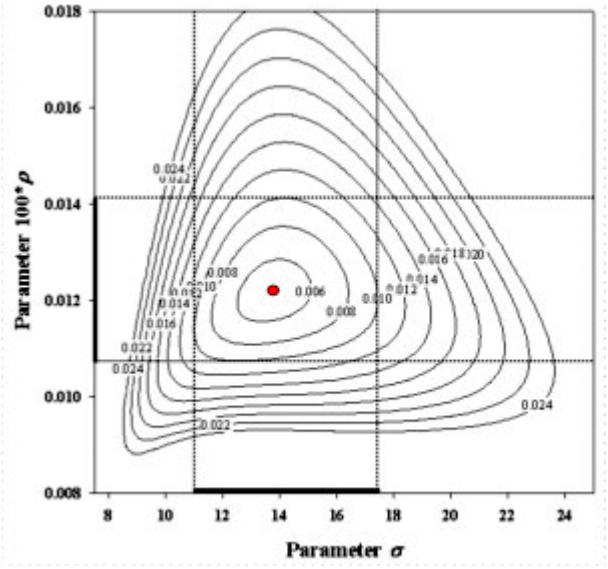


13) To optimize the parameter fit, press the button "Zoom" and ten "Fit". *Programita* now determines the probable range of the parameters:



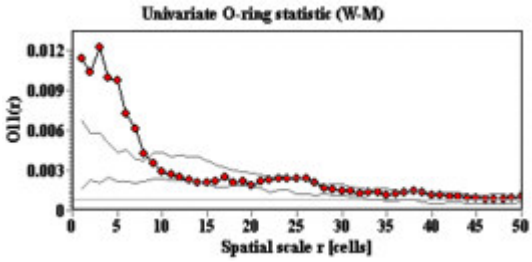
We find σ_{best} = 13.66 and ρ_{best} = 0.0001215 (some 30 parents).

Saving the results of the fit (DC_uni_c.fit) and plotting the error surface shows that the fit determined well the known parameter (σ = 14) under which the pattern was created. However, the number of parents is slightly underestimated (30 instead of 42). This is because some of the larger-scale clusters of the parents overlap and function as one single cluster.



Error surface for construction of confidence intervals for the fit. Contour plot of the error in dependence on the parameters σ and ρ of the fit The confidence intervals for an error < 0.010 are shown as bold intervals at the axes. The best fit is indicated as red dot.

9) To simulate the process click in the windows **Fit of Neyman-Scott models to data** "ok", select a ring width of dr = 3, and "Calculate Index". Simulation of the Neyman-Scott null model with larger scale clustering shows indeed good accordance at scales r > 15, however (as expected), the small-scale clustering is not captured (DC1_uni_c_largescale.res):



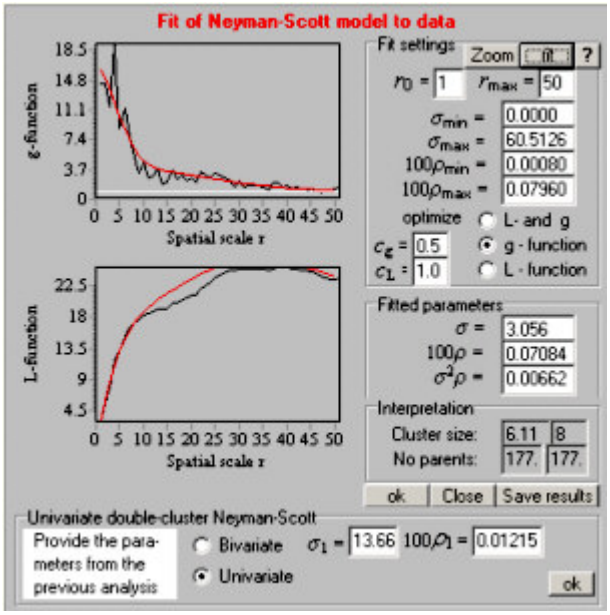
Second step: univariate analysis with double-cluster model

1) Highlight the data file "DC1_uni_c.dat" in window **Input data file**.
2) select "List" in **How are your data organized**
3) select "Data are given as list in grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
4) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data.
5) For univariate analysis of this pattern assuming a double-clustered process enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "Univariate double cluster".
6) The windows **Fit of Neyman-Scott models to data** and "Univariate double-cluster Neyman-Scott" appear. Select the option "Univariate" and provide the results of a univariate analysis of pattern 1 at larger scale (r = 15 - 80):

- 13.66 and $100\rho_{best}$ = 0.01215 (some 30 parents):

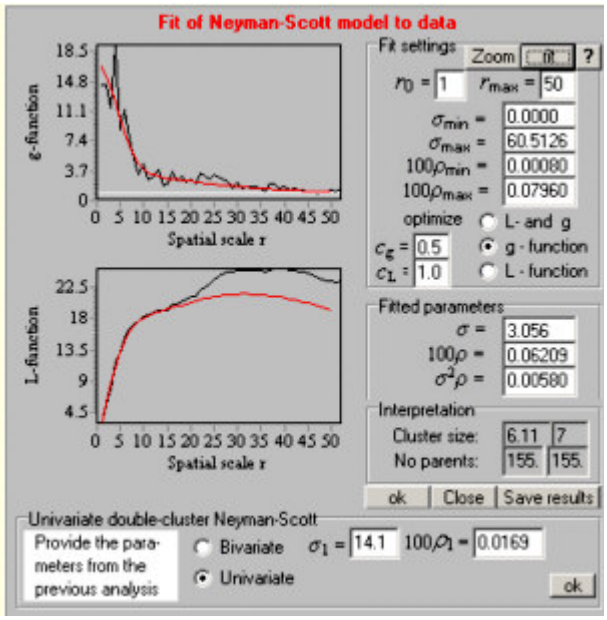
and click "ok" in the window "Univariate double-cluster Neyman-Scott" and again "ok" in the window "Null models".

7) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. Because we fit the small scale structure of the pattern, select r_{max} = 1 and r_0 = 50 and "g-function" (i.e., adjust only the g-function) and click the button "fit". (The *L*-function is more sensitive at larger scales).
8) *Programita* now searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the *g*- and *L*-function of your data best (red line: fit, black line: data).
9) *Programita* finds for the initial parameter intervals the best fit
   - σ = 3.056 and 100ρ = 0.07084 (some 177 parents):



The results are reasonable estimates of the parameters under which the pattern was created (σ_2 = 4, ρ_2 = 0.00054 (some 136 parents). *Programita* estimates the cluster size a bit too small (3.1 instead of 4) and the number of parents a bit too high (177 instead of 136).

10) To find out whether this differences are due to uncertainty in the estimates of the large scale clustering we repeat the fit with the known parameters of the large-scale clustering
    - 14.1, $100\rho_1$ = 0.0169 (some 42 parents):



The estimate for the cluster size (σ = 3.056) coincides with the result from the previous analysis, but the number of parents is closer to the known value (155 instead of 136).

11) These results indicate that the estimation of the cluster size is relatively insensitive to the uncertainty in the previous estimate of the parameters of the large-scale clustering, but that the estimate of the number of parents depends more sensitively on the correct estimate of the parameters of the large-scale clustering. Because of the stochastic nature of the simulation process and the finite number of points, there will be always some variation in the realized parameters in respect to the original parameters under which the process was simulated.

3.5.3. Double-cluster process $\sigma^2_1 << \sigma^2_2$ (DC_2.res)

This pattern was created using a parents pattern with 136 points and parameters

- $\sigma_1 = 3.687$, $\rho_1 = 0.0002416$ (some 60 parents)

and the parameters of the offspring were

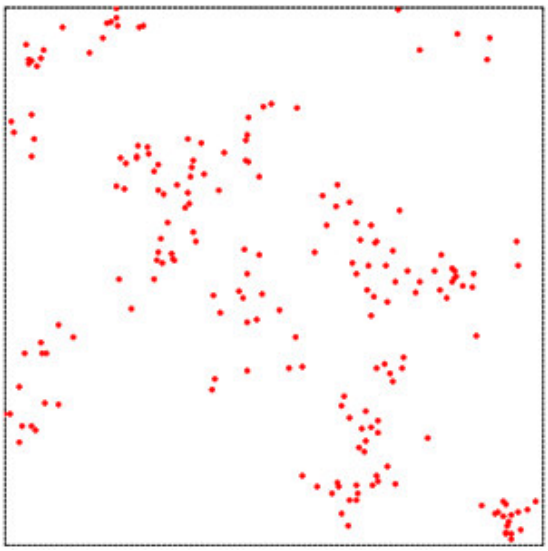
- $\sigma_2 = 14$, $\rho_2 = 0.00054$ (some 136 parents, i.e., all type 1 points are parents)

We analyze first the clustering of the pattern at larger scale using the simple cluster process equation DC1 and apply then the double-clustered model to determine the parameters of the small-scale clustering.

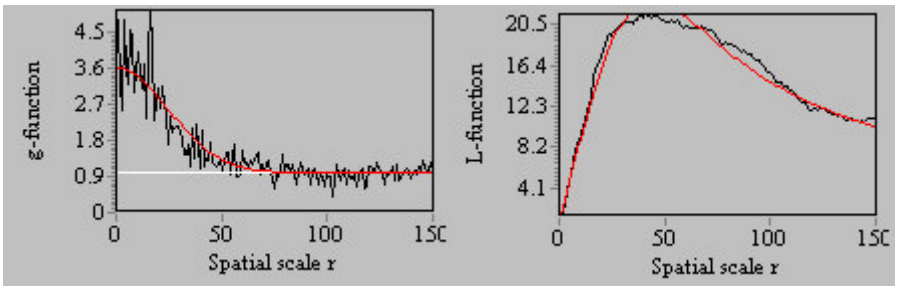
First step: univariate analysis at larger scales with simple cluster model

10) Highlight the data file "DC2_uni_c.dat" in window Input data file.
11) select "List" in How are your data organized

12) select "Data are given as list in grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
13) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data:



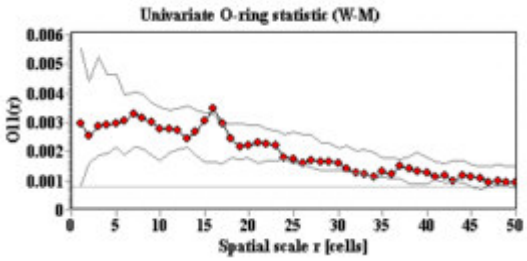
14) For univariate analysis of this pattern at larger scale enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "univariate Neyman-Scott" and press ok.
15) *Programita* calculates the *g*- and the *L*-function of the data and the window **Fit of Neyman-Scott models to data** appears. You can specify the tuning constants r_{min}, r_{max}, and c for the fit in the window "Fit cluster process".
16) To analyze the cluster structure of the patter select r_{max} = 1 and r_0 = 150 and click the buttons "fit", "Zoom" and "Fit". *Programita* now shows the *L*- and *g*-function of the data and the fit at small scales:



17) There is no indication that this pattern may be a double-clustered pattern, the theoretical *g*- and *L*-functions (red line) fit the *g*- and *L*-functions of the data (black lines) well. We find
    - σ_{best} = 17.4 and $100\rho_{best}$ = 0.01 (some 25 parents)

    which approximate the known parameters of the larger-scale clustering of the parents [σ_t = 14.1 and $100\rho_{best}$ = 0.0169 (some 42 parents)].
18) To simulate the process click in the windows **Fit of Neyman-**

Scott models to data "ok", select a ring width of 3 cells (*dr* = 3) and "Calculate Index". Simulation of the Neyman-Scott null model shows indeed good accordance at all scales (DC2_uni_c_largescale.res):

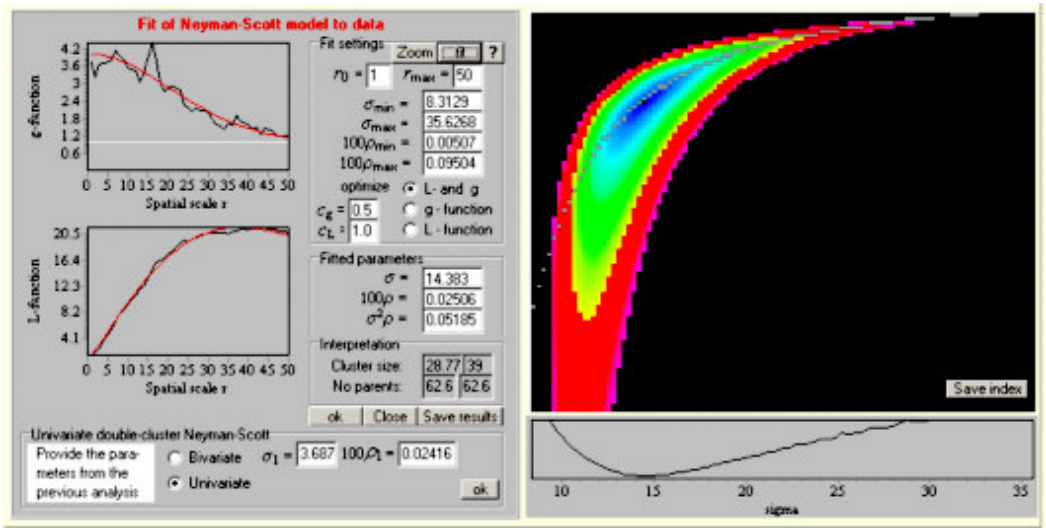


Second step: univariate analysis with double-cluster model

Because the cluster size of the parents is much smaller than the cluster size of the parents, the univariate analysis using the simple cluster model equation DC1 did not indicate the double-clustering under which the process was created. To find out if *Programita* reveals the known parameters of the large-scale clustering under knowledge of the parameters of the small-scale clustering of the parents we now analyze the pattern with the double-cluster model.

1) Highlight the data file "DC2_uni_c.dat" in window Input data file.
2) select "List" in How are your data organized
3) select "Data are given as list in grid" in Select modus of data. A window opens asking you to provide a cell size. Insert "1.00".
4) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data.
5) For univariate analysis of this pattern assuming a double-clustered process enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "Univariate double cluster".
6) The windows Fit of Neyman-Scott models to data and "Univariate double-cluster Neyman-Scott" appear. Select the option "Univariate" and provide the parameters of the small-scale clustering of the parents:
   - 3.687 and $100\rho_{best}$ = 0.02416 (some 60 parents):

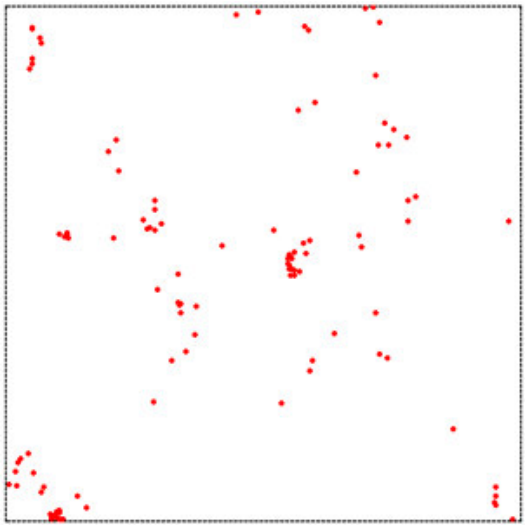
   and click "ok" in the window "Univariate double-cluster Neyman-Scott" and again "ok" in the window "Null models".
7) *Programita* calculates the *g*- and the *L*-function for $r = 1$ to r_{max} and the window Fit of Neyman-Scott models to data appears. Select $r_{max} = 1$ and $r_0 = 50$ and click the buttons "fit", "Zoom" and "fit".
8) *Programita* now searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the *g*- and *L*-function of your data best (red line: fit, black line: data). *Programita* finds for the initial parameter intervals the best fit
   - σ = 14.383 and 100ρ = 0.02506 (some 63 parents):



The results are in excellent accordance with the known parameters. This result indicates that the information on the small-scale clustering of the parents was hidden in the pattern of the offspring but may only be revealed with simultaneously fitting the four parameters of the double-clustered model.

3.5.4. Double-clustered recruits (DC_3.res)

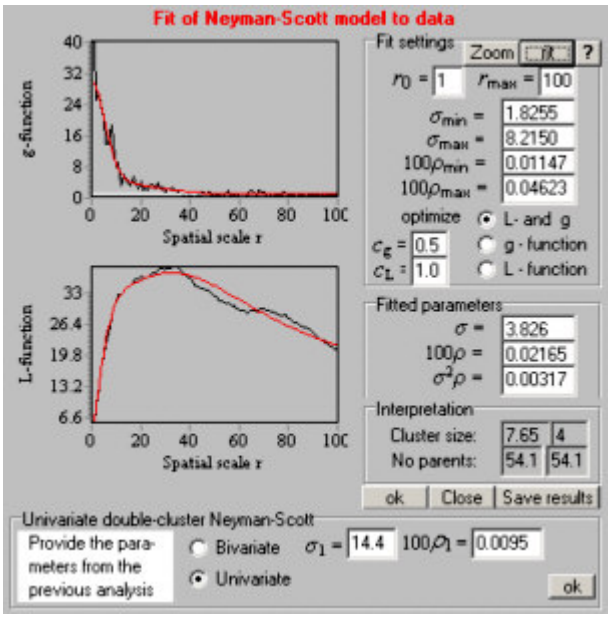
9) Highlight the data file "recruits.dat" in window **Input data file**. This data set gives the location of recruits at a meter scale, but has a resolution of 1 centimeter.
10) select "List" in **How are your data organized**
11) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
12) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data:



13) For univariate analysis of this pattern assuming a double-clustered process enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "Univariate double cluster".
14) The windows **Fit of Neyman-Scott models to data** and "Univariate double-cluster Neyman-Scott" appear. Select the option "Univariate" and provide the results of a univariate analysis of pattern 1 at larger scale (r = 15 - 100) (example NS_2.res):
    - σ = 14.4 and 100ρ = 0.0095 (some 24 parents).

    Click "ok" in the window "Univariate double-cluster Neyman-Scott" and again "ok" in the window "Null models".
15) *Programita* calculates the g- and the L-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. Select r_{max} = 1 and r_0 = 100 and click the button "fit".
16) *Programita* now searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the g- and L-function of your data best (red line: fit, black line: data).
17) *Programita* finds for the initial parameter intervals the best fit
    - σ = 4 and 100ρ = 0.02150 (some 54 parents)
18) To optimize the parameter fit, press the buttons "Zoom" and "Fit". *Programita* finds the best estimates:
    - σ = 3.83 and 100ρ = 0.02165 (some 54 parents)

    Thus, the scales of clustering a different: there is a large-scale clustering with σ = 14 which may correspond to a heterogeneity of the suitable habitat and a small-scale clustering with σ = 4 which corresponds to clustering of recruits inside the larger-scale clusters. The fitted g- and L- functions approximate the data very well:



3.6. Hard-core process (HC)

3.6.1. Background

A hard-core process is the simplest extension of CSR to describe small-scale regularity where points have a minimal distance δ (figure HC1). In this case the *g*-function yields:

$$g(r) = \begin{cases} 0 & \text{for } r \le \delta \\ 1 & \text{for } r > \delta \end{cases} \qquad \text{(HC1)}$$

and the *K*-function can easily be calculated using equation D4:

$$K(r) = \int_0^r 2\pi r' g(r')dr' = \pi r'^2 \Big|_\delta^r = \pi r^2 - \pi\delta^2 , \qquad \text{(HC2)}$$

which yields

$$L(r) = \begin{cases} -r & r \le \delta \\ r\left(\sqrt{1 - \left(\frac{\delta}{r}\right)^2} - 1\right) & r > \delta \end{cases} \qquad \text{(HC3)}$$

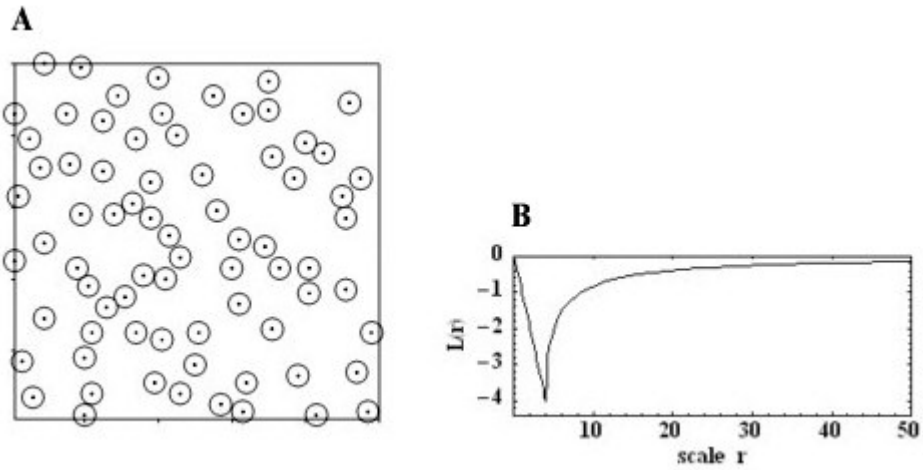


Figure HC1. (A) Hardcore pattern where the points have a minimal distance δ. This corresponds to the case where the points have a finite size and are represented non-overlapping disks of radius $\delta/2$. (B) The theoretical *L*-function for a hardcore process given through equation HC3.

Figure HC1A shows an example of a pattern created by a hardcore process with a hardcore radius $\delta = 4$ which corresponds to non-overlapping disks with radius 2. The resulting L-function (equation HC3) is shown in figure HC1B. Note that the entire departure from the expected L-function under CSR (i.e., $L = 0$) at scales $> \delta$ results from the hardcore and the "memory" of the L-function which arises because the K-function is accumulative (equation HC2). Dixon (2002) and Stoyan and Stoyan (1994) review further analytical formulas of the K- and g-function under different hard-core processes.

A hard core null model does not allow two points to have a distance smaller than the minimal distance δ. However, in real situation the probability that a point can be found at distance d from another point may not be a step-function as assumed under a hard-core null model (equation HC1), but rather a function that decreases with decreasing distance d, thus defining a soft-core null model.

3.6.2. Implementation of the hard-core null model

For numerical simulation of a univariate hard-core process provisional point are placed (following the specific null model selected) and the distance d to the nearest accepted point is determined. The provisional points are accepted if the distance $d > \delta$ (i.e., no overlap of the disk of the two points), otherwise it is rejected.

3.6.3. Implementation of the soft-core null model

For implementation of a soft-core null model *Programita* uses a probability p_{HC} of a provisional point to be accepted that varies between 0 and 1, depending on the distance d to the nearest (accepted) neighbor, and an exponent p that gives the degree of "softness" (figure HC2):

$$p_{HC}(d) = \begin{cases} d^{1/p} & \text{for } d \le \delta \\ 1 & \text{for } d > \delta \end{cases} \qquad \text{(HC4)}$$

For $p = 0$, we obtain the hardcore model, and for $p > 0$ a soft core model.

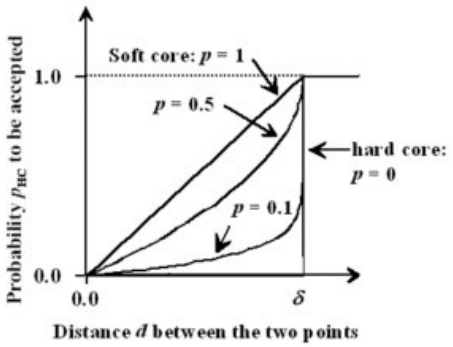


Figure HC2. The probability to accept a provisional point in dependence on the distance d to its nearest neighbor. For a hardcore null model the point is rejected if $d < \delta$ (i.e., the two hard-core disks overlap). For a softcore null model the probability p_{HC} of acceptance is $p_{HC} = d^{(1/p)}$.

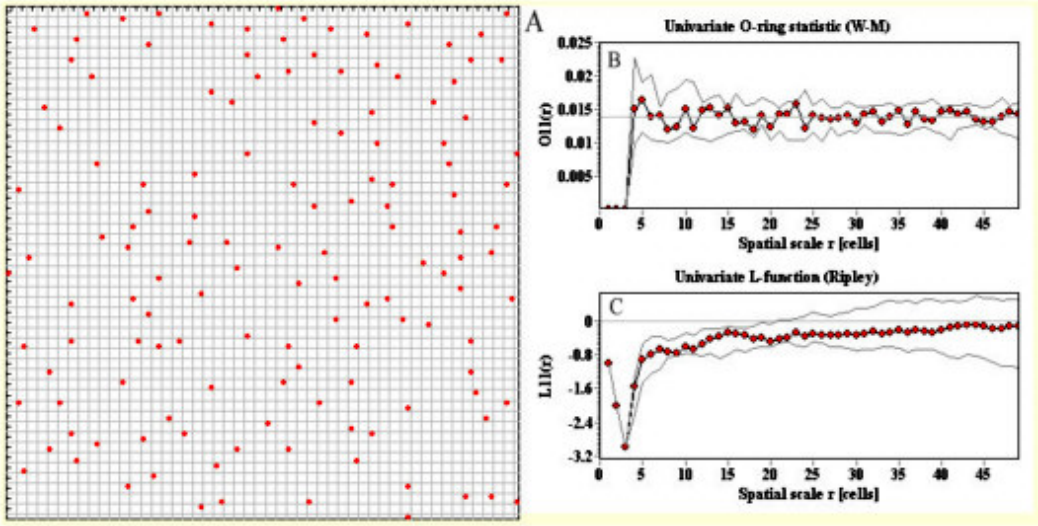


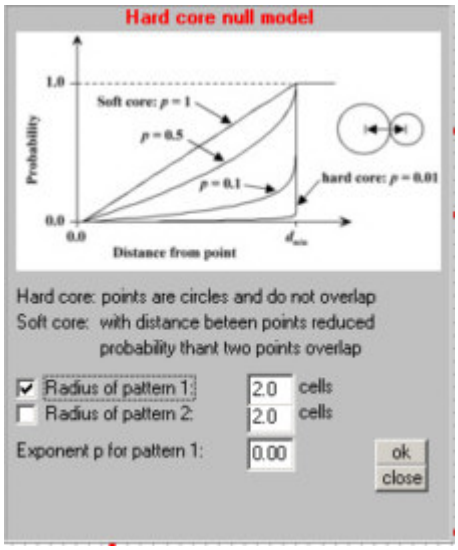
Figure HC3. Example HC_1.res. (A): pattern of the data file HC1.dat that was created with a hardcore process and $\delta = 4$ (this corresponds to a disk with radius 2 cells in the window Hard core null model). (B): The *O*-ring statistic for the pattern shown in (A) and confidence envelopes for 19 simulations of a hardcore null model with radius of pattern 1 of 2 cells. (C) The *L*-function for the pattern shown in (A) and confidence envelopes for 19 simulations of a hardcore null model with radius of pattern 1 of 2 cells. The analyses of (B) and (C) confirm that HC1.dat is a random pattern where each point has a hard core radius of 2. Note that (B) and (C) seems to suggest $\delta = 3$ and not $\delta = 4$ as expected. This is because of the definition of scales *r*. For example, cells with distance $d = 4.45$ belong to scale $r = 4$.

3.6.4. Hard core null model (HC_1.res)

1) highlight the data file "HC1.dat" in window Input data file. The data file was created with a hardcore null model and a minimal distance of 4 cells between points.
2) select "List" in How are your data organized
3) select "Analyze all data in rectangle" in Give modus of analysis
4) select "Data are given as list in grid" in Select modus of data
5) click button "Calculate index".
6) Enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models ap-

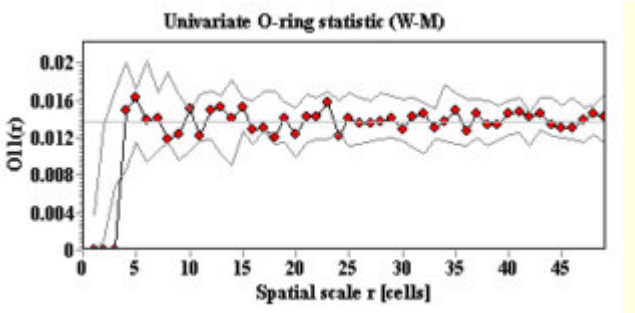
pears. Select "Pattern 1 and 2 random" and enable the checkbox "Hard core". The window **Hard core null model** appears:

7) enable the checkbox "Radius of pattern 1" (univariate analysis) and provide the hardcore radius of pattern 1. The minimal distance between two points is the double of the hardcore radius. Next provide the hardcore radius and the exponent. The example is for a hard-core null model, therefore select the exponent $p = 0$, click "ok", and click the button "Calculate index".
8) *Programita* applies now the hardcore null model for calculation of the confidence envelopes shown in figure HC3B.



3.6.5. Soft core null model (HC_2.res)

1) Same settings as in example HC_1, but select in the window **Hard core null model** an exponent $p > 0$ ($p = 0.5$ in the example).
2) Click the button "Calculate index". *Programita* applies now the softcore null model for calculation of the confidence envelopes shown below:



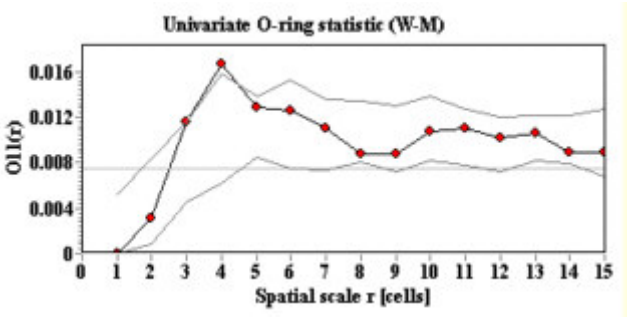
3) Note that the confidence envelopes show a soft core with a reduced (but not zero) probability of having points closer than $r = 3$. The data set "HC1.dat" that was created with a hard core model does not satisfy the softcore null model at scales r = 1, 2, and 3.

.

3.6.6. Heterogeneous Poisson and hardcore (HC_3.res)

This example shows the combination of the hardcore null model with the heterogeneous Poisson null model shown in example HP_1. Provisional points are only accepted if they satisfy the condition for heterogeneous Poisson and the condition for hard core simultaneously.

1) highlight the data file "marcela.dat" in window **Input data file**
2) select "List" in **How are your data organized**
3) select "Analyze all data in rectangle" **in Give modus of analysis**
4) select a maximal radius of 15 cells in **set maximal radius rmax**.
5) select "Data are given as list in grid" in **Select modus of data**
6) click button "Calculate index".
7) A window with settings for null models appears. Select "Pattern 1 and 2 random" and enable the checkbox "Hard core".
8) the window **Hard core null model** appears. Enable the checkbox "Radius of pattern 1" (univariate analysis) and provide the hardcore radius of pattern 1 (a value of 2). The minimal distance of two points is the double of the hardcore radius (i.e., $\delta = 4$). Provide the exponent p = 0 for hardcore and click "ok". The window **Hard core null model** disappears.
10) enable the check box check box "Heterogeneous Poisson" in the window **Select a null model**. The window **Settings for heter. Poisson** appears. Select "Test only for pattern 1" (pattern 2 does not exist in this example), and select a radius R=15 for the moving window.
9) click button "Calculate index". *Programita* now calculates the moving window estimate of the first-order intensity of the pattern. Click "ok" at the message window and *Programita* performs the simulations of the heterogeneous Poisson null model for calculation of the confidence envelopes shown below:

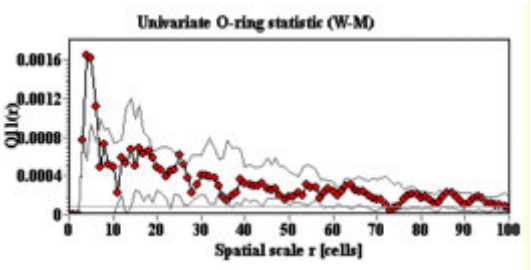


The confidence envelopes were constructed with 99 replicate simulations of the combined heterogeneous Poisson and hardcore null model. The data are almost within the confidence envelopes, with $O_{11}(4)$ being only slightly above the confidence interval.

3.6.7. Combined cluster and hardcore (HC_4.res)

This example shows the combination of the hardcore null model with a univariate Neyman-Scott cluster null model and continues the analysis of example NS_4.res. The pattern of adult trees showed at a fine resolution of 0.25 m a marked hard-core up to 1m and a peak at some 2.5 m. Here we investigate whether the hard core and the pear at 2m may be explained be a combined hard-core and cluster null model. We analyze the bivariate pattern of adult trees and recruits in example HC_7.res.

1) Highlight the data file "adults_real.dat" in window **Input data file.** This data set gives the location of adult trees and recruits at a meter scale, but has a resolution of 1 centimeter.
2) select "List" in **How are your data organized**
3) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "0.5", thus using a cell size of 50cm.
4) click the button "change" in **set maximal radius rmax**, set the maximal scale r of the analysis to r_{max} = 50, and select a ring width of dr = 3.
5) click button "Calculate index", *Programita* shows you the pattern and calculates the O-ring function of the data.
6) To determine Monte Carlo confidence intervals for the bivariate Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "univariate Neyman-Scott" and press ok.
7) *Programita* calculates the g- and the L-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears.
8) Select r_{max} = 15 and r_0 = 200. To optimize the g- and the L-function simultaneously enable "both, L- and g-function". Click the button "fit" and *Programita* searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the g- and L- function of your data best (red line: fit, black line: data).
9) To optimize the parameter fit, press the button "Zoom". *Programita* now determines the probable range of the parameters. Next, press "fit" and *Programita* now searches the best fit. We find σ_{best} = 29.5 and ρ_{best} = 0.0000205.
11) Enable the check box "Hard core" and the window **Hard core null model** appears. Enable the checkbox "Radius of pattern 1" (univariate analysis) and provide the hardcore radius of pattern 1 (a value of 2). The minimal distance of two points is the double of the hardcore radius (i.e., δ = 4). Provide the exponent p = 0 for hardcore and click "ok". The window **Hard core null model** disappears.
12) click button "Calculate index" and *Programita* performs the 19 simulations of the combined cluster and hard-core null model:



Indeed, the combined hard-core and cluster null model describes the data well, only the peak at 2m (4 cells) remains significant. Thus, adult trees show a strict repulsion and are at least 2 m apart.

3.7. Recommendations for selection of univariate null models

In the following we give recommendations for the selection of appropriate null models for univariate point-pattern analysis and we provide an exploratory step-by-step protocol. Note that point-pattern analysis is a descriptive analysis. Even if a particular null model describes your pattern well, it is not appropriate to conclude that the mechanism behind the null model is the mechanism responsible for your pattern. Other mechanisms may lead to exactly the same pattern. However, point-pattern analysis helps to characterize your pattern and to put forward hypotheses on the underlying mechanisms that should be tested in subsequent steps in the field.

1. Visualize the pattern, define a preliminary study region and plot the second-order statistics $\hat{L}(r)$ and $\hat{O}(r)$.
2. If the size of your biological objects cannot be neglected (i.e., they are large and do not overlap) you might combine a hard-core null model with the null models suggested in the next steps. You may apply a hard or soft-core null model if $\hat{O}(r) << \lambda$ for scales 1- r_0 (compare figure HC3).
3. If there is no indication for strong aggregation (clearly visible clusters in the pattern or a $\hat{O}(r)$ typical for virtual aggregation) use CSR as the null model for detecting aggregation or inhibition. Virtual aggregation (large scale clustering) is indicated by a constant $\hat{O}(r)$ over a range of scales, and at this range $\hat{O}(r)$ is well above the intensity λ of the pattern (e.g., Fig. 3B in Wiegand and Moloney 2004). Smaller-scale clustering is indicated by a steep linearly increasing $\hat{L}(r)$ at smaller scales (e.g., example NS_1). The cluster size is slightly below the value of r where $\hat{L}(r)$ is maximal.
4. If step (3) indicates virtual aggregation (i.e. large clusters) exclude the gaps (or use smaller rectangular sub-regions) and apply CSR only in the sub-region without gaps (or in the smaller plot). Think about a biological explanation for the heterogeneity encountered. Perhaps there are obstacles in the study region, or clear environmental heterogeneity that prevent points from occurring in the gap.
5. If there is a biological explanation for the heterogeneity encountered in step (3) (e.g., clear differences in soil), you might map the environmental factor and use this map to obtain an intensity function of a heterogeneous Poisson process. Otherwise, you can use the pattern itself to estimate the non-constant first-order intensity λ using the moving window estimator for simulation of a heterogeneous Poisson process null model. Alternatively, if there is a surrogate pattern for the environmental heterogeneity (e.g., the locations of a different, more common plant species that is hy-

pothesized to be subject to the same environmental factor), use univariate random labeling as the null model for testing whether your pattern is more (or less) clustered than the control.

6. If there is no obvious environmental heterogeneity, your pattern may be a realization of a cluster process. Use $\hat{L}(r)$ to obtain rough (initial) estimates of the parameter ρ and σ of a Neyman-Scott process and fit the parameters using the methods given in Diggle (1983). Use the estimated parameters ρ and σ to simulate confidence envelopes for the Neyman-Scott process null model. Clearly, there are a number of other point-processes you might fit to your data. However, because of small number of points and noisy data, you might not be able to statistically separate them.
7. If there is small-scale regularity and larger scale clustering, the expected L-function for the Neyman-Scott process needs to consider the small-scale regularity because the L-function is accumulative and conserves at larger scales some "memory" on the small-scale regularity. This can be done analogously to equation M1. Alternatively, one may use only the pair-correlation function g (which has no memory) to fit the unknown parameters ρ and σ, but omitting the smaller scales r.

4. Bivariate Null Models

Interpreting a bivariate *K*-function or *O*-ring statistic can be confusing because it differs from the univariate case. In the univariate case, visualization of the pattern usually provides an intuitive idea of the first and second-order properties of a pattern. However, in the bivariate case we analyze the spatial relation between two spatial patterns at different spatial scales where each pattern individually may have a complicated spatial structure. Confusion may also arise because there is not one simple and intuitive null model such as CSR, and because a null model based on CSR (i.e., randomization of both patterns) leads to an inadequate test for absence of interaction between the points of bivariate patterns.

For bivariate patterns, three conceptually different null models correspond to an absence of interaction between the two types of points:

- **Independence** assumes that the two patterns were generated by two independent processes (e.g., one process generated the locations of shrubs, and the other process generated the locations of grass tufts). Thus, the expected absence of interaction between the two types of points corresponds to an absence of interaction between the two patterns.
- **Random labeling** assumes that both patterns were created by the same stochastic process (or were subject to the same constraining factors), and that the labels (or "marks") are randomly distributed among the locations of the joined pattern. Thus, the absence of interaction between the two types of points corresponds to an absence of spatial correlation in the occurrence of the labels.
- **Antecedent condition** assumes that the two types of points were created in sequence (e.g., adult trees did not change during the development of recruits). Creation of pattern 1 occurred independently on pattern 2 (because it did not yet exist) but creation of pattern 2 may be influenced by presence of points of pattern 2. Therefore, under an antecedent condition the null model needs to conserve the locations of pattern 1 and a specific hypothesis on the null model for pattern 2 needs to be formulated. In the simplest case the null model for pattern 2 is a CSR process where the expected absence of interaction between the two types of points corresponds to an absence of facilitation or competition exerted by type 1 points over type 2 points.

Departure from independence indicates that the two processes display attraction or repulsion, regardless of the univariate pattern of either group by itself whereas the interpretation of departure from random labeling is more complicated. The

distinction between independence and random labeling requires some care and consideration (Dixon 2002). When there is no relationship between two processes, the two approaches lead to different expected values of $K_{12}(r)$ and $O_{12}(r)$, and to different procedures for generating null models (Goreaud and Pèlissier 2004). Assessment of departure from random labeling is conditional on the univariate structure of the joined pattern whereas assessment of departure from independence is conditional on the univariate structure of the component patterns.

Since bivariate point-pattern analysis investigates the relation of points of pattern 2 in respect to points of pattern 1 one may specify only the null model for the stochastic process that created pattern 2, but keep the locations of pattern 1 fixed (antecedent condition). This approach need to be used in cases where pattern 1 was unchanged during the creation of pattern 2. An example of such an antecedent condition is seedlings in relation to adult trees. Another case where one may keep the locations of pattern 1 fixed and specify only a null model for pattern 2 is the relation between shrubs (fixed) and grass tufts. In this case the null model distributes grass tufts at random in the area not occupied by shrubs. Departure from the null model (e.g., there are more tufts in the neighbourhood of shrubs than expected under this null model) may indicate facilitation. Because a null model with an antecedent condition specifies only the null model of the second pattern, all univariate null models (e.g., heterogeneous Poisson, hard core, Neyman-Scott cluster null model) may be used. In this antecedent condition is a hermaphrodite null model with characteristic of a univariate null model (i.e., only pattern 2 is simulated whereas pattern 1 remains unchanged) and with characteristic of a bivariate null model (the statistic of interest is the bivariate *L*-function or the bivariate *O*-ring statistic, i.e., the relation of type 2 points to type 1 points is analyzed).

The expected values of the bivariate *g*- and *L*-functions under independence are $g_{12}(r) = 1$ and $L_{12}(r) = 0$, whereas the expected values of the bivariate *g*- and *L* functions under random labeling are determined by the spatial structure of the univariate joined pattern, thus $L_{12}(r) = L_{1+2,1+2}(r)$ and $g_{12}(r) = g_{1+2,1+2}(r)$. Failure to distinguish between random labeling and independence may lead to the analysis of data by methods which are largely irrelevant to the problem at hand (Diggle 1983). Random labeling and independence are equivalent only if all the component processes are homogeneous Poisson processes.

Random labeling offers a number of ways for investigating the spatial structure of a bivariate pattern in detail. In the section "bivariate random labeling" we will present and interpret the different variants of random labeling.

If the null hypothesis to test is absence of interaction between the two types of

points, the nature of the data and phenomena under study (i.e., whether the processes that created pattern 1 and 2 acted simultaneously [independence] in sequence [random labeling] or antecedent condition) provide guidelines for the selection of independence or random labeling as appropriate null model (see e.g., Goreaud and Pèlissier 2004). In some cases, the nature of the data and the biology of the species involved may make the choice between random labeling and independence relatively straightforward. In other cases, however, this may be difficult and open to debate and interpretation and other more specific null models may be used instead.

Beside independence, random labeling and antecedent condition there are a number of more complex bivariate point processes (e.g., bivariate cluster processes, or bivariate hard- and soft-core processes) that may be used to respond to specific biological questions. For example, a bivariate Neyman-Scott cluster process may be used to describe a clustered environmental heterogeneity that affects both patterns in the same way.

4.1. Independence

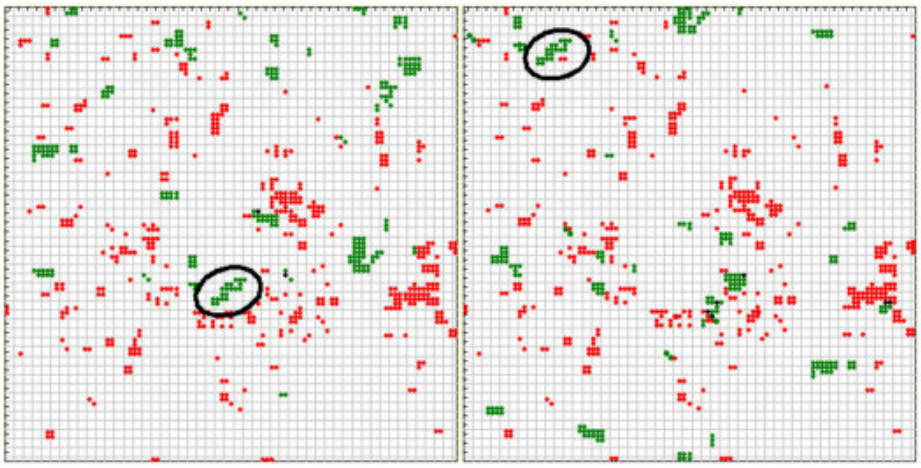
4.1.1. Background

Testing for independence is not that straight forward than testing for CSR in the univariate case because inferences are conditional on the second-order structure of each pattern (Dixon 2002). This is because the theoretical values of $K_{12}(r)$ and $O_{12}(r)$ do not depend on CSR of the component patterns and therefore no assumption can be made about models for either of the component patterns. Thus, the null model of CSR is not appropriate to test for independence; the separate second-order structures of the patterns need to be preserved in their observed form in any simulation of the null model, but one has to break the dependence between the two patterns. One way of achieving this is by simulations that involve random shifts of the whole of one component pattern relative to the other. In practice, a rectangular study region is treated as a torus where the upper and lower edges are connected and the right and left edges are connected.

We present three examples for the application of the toroidal shift null model, one based on real data, and additionally two artificially generated data sets to demonstrate the ability of this null model to detect known departure from independence.

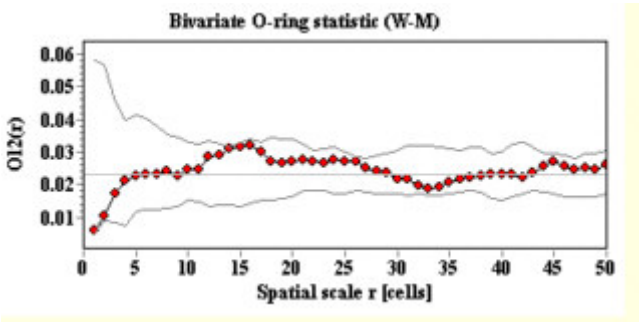
4.1.2. Example Indep_1.res

1) highlight the data file "A1.dat" in window **Input data file.** The two patterns are disturbances observed at a plot of southern German grassland, mapped is a 10m × 10 m area with a resolution of 100 × 100 cells. The points of pattern 1 (read) are cells with ant disturbances and the points of pattern 2 (green) are cells with rabbit disturbances:



Note that each of the two patterns, taken individually, shows marked clustering.

2) select "List" in **How are your data organized**
3) select "Analyze all data in rectangle" **in Give modus of analysis**
4) select "Data are given as list in grid" in **Select modus of data**
5) click button "Calculate index".
6) Enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Toroidal shift".
7) click "Calculate index". *Programita* now performs the simulations of the independence null model and determines the confidence envelopes.
8) The confidence envelopes of the bivariate O-ring statistic for the toroidal-shift null model reveal independence of the two types of disturbance, although the patterns show certain non-significant tendencies to repulsion at smaller scales r = 1, 2 and attraction at scales r = 13 - 17. The tendency for repulsion at small scale arise because it was difficult in the field to recognize cells with both, ant and rabbit disturbance and some cells with ant and rabbit disturbances my actually be mapped as cells with rabbit disturbance only. As a consequence, the bivariate *O*-ring function depicts a tendency to repulsion:



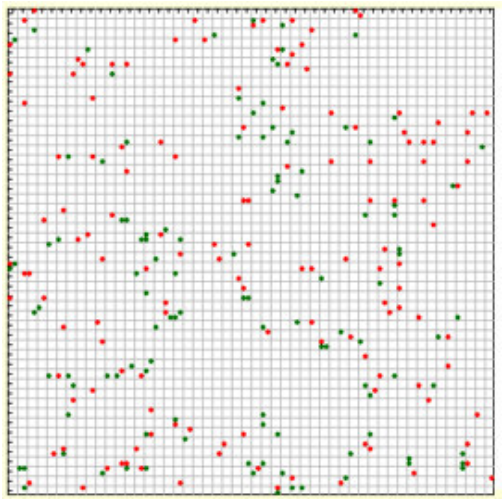
4.1.3. Example Indep_2.res

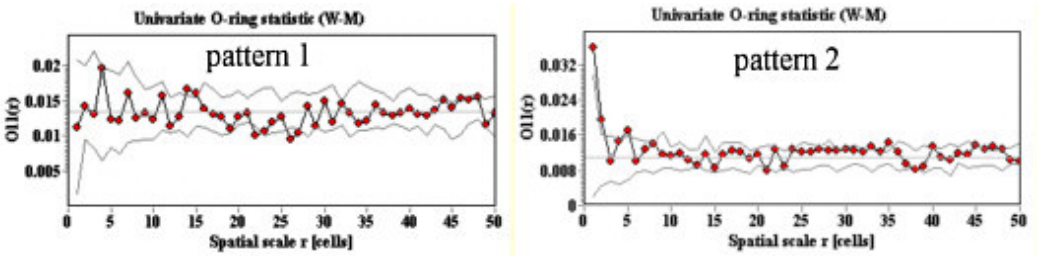
This example corresponds to a situation where creation of pattern 1 was not influenced by pattern 2, but points of pattern 2 experienced facilitation from point of pattern 1. We used a simple attraction process to simulate the points of pattern 2 in dependence on a given pattern 1 that was previously created by a CSR process. The probability $p_{att}(d)$ to accept a provisional point of pattern 2 with a nearest neighbour of pattern 1 at distance d is:

$$p_{att}(d) = \begin{cases} 1-(\frac{d}{\delta})^p & \text{for } d \le \delta \\ 0 & \text{for } d > \delta \end{cases} \qquad \text{(Indep1)}$$

where $\delta = 6$ and $p = 0.1$ in our example. This process is a sort of "inverse" soft-core process; compare equations (Indep1 and Indep2). Note that the points of pattern 2 are, as a result of the non-random creation process, aggregated at scale $r = 1$.

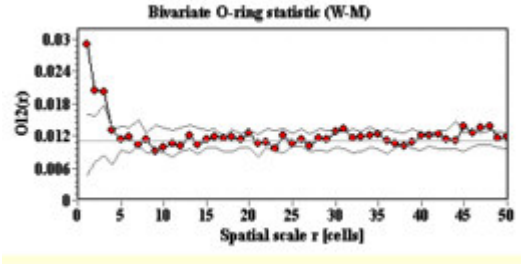
1) highlight the data file "attraction1.dat" in window **Input data file.** Pattern 1 is a random pattern and pattern 2 was created with an explicit aggregation mechanism in respect to pattern 1: random provisional points of pattern 2 were only accepted if they had a nearest neighbor of pattern 1 at distance $d \le 6$ (i.e., $\delta = 6$), and the probability of acceptance increased with decreasing distance d to a point of pattern 1 (equation Indep1):





The univariate analyses show that pattern 1 is indeed a random pattern, whereas the univariate structure of pattern 2 shows (a weak but) significant aggregation at scale $r = 1$.

2) select "List" in How are your data organized
3) select "Analyze all data in rectangle" in Give modus of analysis
4) select "Data are given as list in grid" in Select modus of data
5) click button "Calculate index".
6) Enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Toroidal shift".
7) click "Calculate index". *Programita* now performs the simulations of the independence null model and determines the confidence envelopes:



8) The toroidal shift null model that test for independence of the two patterns is rejected for spatial scales r = 1-3. There are more points of pattern 2 at distances $r \leq 3$ than expected under independence. Thus, application of the null model for independence reveals the a priori known attraction at smaller scales. The distance dependent prob-

ability of acceptance of an provisional point is $p_{att}(d)$ = 0.16 for d = 1, but for d = 4 it decreased to $p_{att}(d)$ = 0.04. Therefore the aggregation mechanism is weak and non-significant at nearest neighbour distances d >3.

9) This example illustrates the difficulty to predict the second-order characteristics of a bivariate pattern visually (pattern 1: random, pattern 2: random but weak aggregation at scale r = 1 and attraction of pattern 2 to pattern 1 at scales r = 1, 2, and 3).

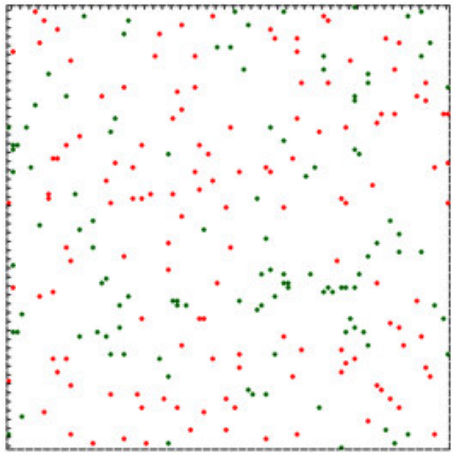
4.1.4. Example Indep_3.res

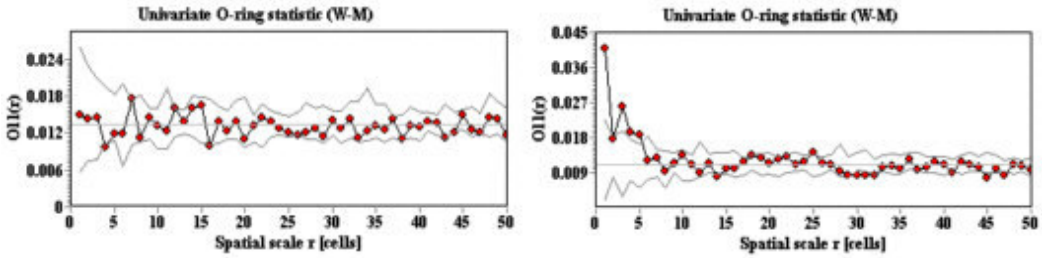
In this example we used a bivariate soft-core process to simulate repulsion of points of pattern 2 in relation to points of pattern 1 that were previously created by a CSR process. The parameters of the bivariate soft-core process are: radius = 3 and exponent p = 5 (for pattern 1), radius = 3 and exponent p = 5 (for pattern 2), and the exponent for repulsion of pattern 2 by pattern 1 was p = 0.1. Thus, points of pattern 2 are placed at random with respect to already accepted points of pattern 2, but the probability $p_{HC}(d)$ to accept a provisional point of pattern 2 with a nearest neighbour of pattern 1 at distance d is:

$$p_{HC}(d) = \begin{cases} d^{1/p} & \text{for } d \le \delta \\ 1 & \text{for } d > \delta \end{cases} \qquad \text{(Indep2)}$$

where δ = 3 +3. Note that the univariate analysis of pattern 2 reveals aggregation at scales r = 1 - 5 which is a result of is non-random creation process.

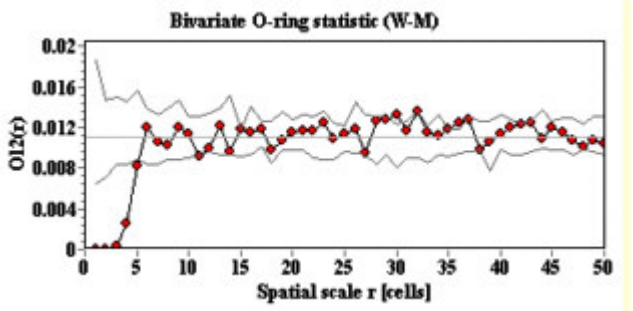
1) highlight the data file "repulsion1.dat" in window **Input data file.** Pattern 1 is a random pattern and pattern 2 was created with an explicit repulsion mechanism: random provisional points of pattern 2 were only accepted if they had a nearest neighbor distance of at least δ = 6 to a point of pattern 1 and the probability of acceptance decreased with increasing distance to a point of pattern 1 (equation Indep2):





The univariate analyses show that pattern 1 is indeed a random pattern, whereas the univariate structure of pattern 2 shows aggregation at scale r = 1 and 3. Note that the aggregation of pattern 2 is an indirect effect induced by the repulsion to pattern 1 (i.e., points of pattern 2 had are squeezed in gaps of pattern 1).

2) select "List" in How are your data organized
3) select "Analyze all data in rectangle" in Give modus of analysis
4) select "Data are given as list in grid" in Select modus of data
5) click button "Calculate index".
6) Enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Toroidal shift".
7) click "Calculate index". *Programita* now performs the simulations of the independence null model and determines the confidence envelopes:



8) The toroidal shift null model is rejected for spatial scales r = 1-4. There are less points of pattern 2 at distances r < 5 than expected under independence. Thus, application of the null model for independence reveals the a priori known repulsion at smaller scales.

4.2. Bivariate random labeling

In the case of random labeling we ask not about the interaction between two processes, but we investigate whether or not the labels "type 1" and "type 2" have a random structure within the given spatial structure of the joined pattern. Numerical implementation of the random labeling null model involves repeated simulations using the fixed $n_1 + n_2$ locations of pattern 1 and 2, but randomly assigning "case" labels to n_1 of these locations (Bailey and Gatrell 1995). Therefore, the expected bivariate *g*- or *L*-function under random labeling is the univariate *g*- or *L*-function of the joined pattern.

(Goreaud and Pèlissier 2004) discuss the differences between the null hypotheses independence and random labeling and derive rules for the appropriate use of these null models if the researcher wants to test for **absence of interaction between the two types of points**. If the two types of points correspond to two "populations" whose specific spatial pattern can *a priory* be the result of different processes (e.g., plants of different species), then the expected absence of interaction between the two types of points corresponds to an absence of interaction between the two populations. On the other hand, if the two types correspond to some events affecting *a posterory* the individuals of a single "population" (e.g., tree dead or disease spread), then the absence of interaction between the two types of points corresponds to an absence of interaction in the occurrence of these events.

There are several variants of random labeling which each valuate different biological effects. In the following we will provide interpretations for these variants.

4.2.1. Different possibilities to asses departures from random labeling

Under random labeling both component patterns taken separately represent "random thinning" of the joined pattern, and from their definition, *K*-functions and *g*-functions are invariant under random thinning. Therefore we would expect that

$$g_{12}(r) = g_{21}(r) = g_{11}(r) = g_{22}(r). \qquad \text{(RL1)}$$

Because the component patterns taken separately are "random thinning" of the joined pattern, we expect additionally the identities:

$$g_{12}(r) = g_{1+2,1+2}(r)$$

$$g_{12}(r) = g_{1,1+2}(r)$$
$$g_{21}(r) = g_{2,1+2}(r) \qquad \text{(RL2)}$$

where "1+2" symbolizes the joined locations of pattern 1 and 2. Equation RL1 suggests that a useful way of investigating departures from random labeling is to assess the significance of differences amongst estimates of $g_{12}(r)$, $g_{21}(r)$, $g_{11}(r)$, and $g_{22}(r)$ (Bailey and Gatrell 1995). Using differences instead of *g*- or *K*-functions has the advantage that the expected value under random labeling is always zero, whereas the univariate *g*- or *L*-function of the joined pattern (which is the expected value under random labeling) can have any shape.

The expectation $g_{12}(r) = g_{1,1+2}(r)$ (equation RL2) suggests assessment of the significance of the quotient $g_{12}(r)/g_{1,1+2}(r)$. The expected value of this quotient under random labeling is $g_{12}(r)/g_{1,1+2}(r) = 1$.

The advantage of using differences or quotients is that the results are easier to visualize and interpret because the expected values under random labeling are zero for differences and one for quotients. Additionally, the analysis of the differences and quotients allows deeper insight into the relation of the two patterns because each pairwise difference or quotient evaluates different biological effects. The difference $\hat{g}_{11}(r) - \hat{g}_{12}(r)$ for example evaluates whether points of type 1 tend to be surrounded by other points of type 1, while $\hat{g}_{11}(r) - \hat{g}_{22}(r)$ evaluates whether one pattern is more (or less) clustered than the other (Dixon 2002).

In order to properly interpret these differences and quotients, we go back to the grid-based definitions of the bivariate *g*-function (equation I8). Note that the same arguments presented here for the *g*-function are also valid for *K*-functions, the only difference is that the rings have to be replaced by circles.

The bivariate *O*-ring statistic and the *g*-function are estimated with:

$$\hat{O}_{12}^{w}(r) = g_{12}(r)\frac{A}{n_2} = \frac{\frac{1}{n_1}\sum_{i=1}^{n_1}\mathbf{Points}_2[R_{1,i}^{w}(r)]}{\frac{1}{n_1}\sum_{i=1}^{n_1}\mathbf{Area}[R_{1,i}^{w}(r)]} \qquad \text{(RL3)}$$

where $R^{w}_{i,k}(r)$ is the ring with radius r and width w centered on the kth point of type i, n_i is the total number of points of type i in the study region of area A, the operator $\mathbf{Points}_j[X]$ count the points of type j in a region X and the operator **Area**[X] counts the number of cells in the region X. The bivariate *O*-ring statis-

tic relates the average number of type 2 points in rings with radius r centered in type 1 points (nominator of equation RL3) to the average number of cells of these rings (denominator of equation RL3), thus it calculates the average number of points per area (if a cell is the unit area).

We introduce the following definitions:

$$P_{ij}(r) = \frac{1}{n_i} \sum_{k=1}^{n_i} \mathbf{Points}_j[R_{i,k}^w(r)] \qquad \text{(RL4)}$$

is the average number of type j points in rings with radius r centered in type i points, and

$$A_i(r) = \frac{1}{n_i} \sum_{k=1}^{n_i} \mathbf{Area}[R_{i,k}^w(r)] \qquad \text{(RL5)}$$

is the average number of cells in rings with radius r centered in type i points. With this definitions, equation RL3 simplifies to $\hat{O}_{12}^w(r) = P_{12}(r) / A_1(r)$.

In the bivariate case (i.e., $i \neq j$), the term $n_i P_{ij}$ gives the total number of ordered pairs of type i and type j points. The number of ordered pairs is symmetric, thus: $n_1 P_{12} = n_2 P_{21}$. We will use this relation when deriving interpretations of the differences and quotients of g-functions.

If the mean area of rings centered in points of pattern 1 is the same as the mean area of rings centered in points of pattern 2 (i.e., $A_1 = A_2$) edge effects are equilibrated and none of the two patterns has the tendency to occur closer (or further away) to the edge of the study region. Note that the edge correction used in our grid-based and numerical implementation (Wiegand and Moloney 2004) differs from the analytical edge correction usually used (e.g., Goreaud and Pèlissier 1999).

From the definition of A_i and P_{ij} we derive the relations for the joined pattern:

$$A_{1+2} = (n_1 A_1 + n_2 A_2)/(n_1 + n_2) \qquad \text{(RL6)}$$

$$P_{1+2,\,1+2} = [n_1 (P_{11} + P_{12}) + n_2 (P_{22} + P_{21})]/(n_1 + n_2) \qquad \text{(RL7)}$$

where $P_{i,1+2} = P_{i1} + P_{i2}$. Without loosing generality we can assume, when interpreting departure of a given quotient from the expected value under random labeling, that both patterns have the same number of points (i.e., $n_1 = n_2$). This is

because *g*- and *K*-functions are invariant under random thinning. However, the number of points of the pattern determines whether the confidence envelopes are wide (low number of points) or narrow (higher number of points), and the confidence envelopes of a randomly thinned pattern with $n_1 = n_2$ will be wider than that of the original pattern with $n_1 << n_2$.

4.2.2. Variant 1, the bivariate *g*-function g_{12}

Some authors compare the bivariate *g*- or *L*-function to confidence envelopes generated by randomization of the labels (e.g., Goreaud and Pèlissier 2004). A value of $\hat{g}_{12}(r)$ above the random labeling confidence envelopes indicates that type 2 points are more frequent at distance *r* around type 1 points than expected under the random labeling null hypothesis. This test does basically compare the bivariate *g*-function $\hat{g}_{12}(r)$ to the univariate *g*-function $\hat{g}_{1+2,1+2}(r)$ of the joined pattern (see example RL_2), but it uses only n_1 randomly selected points out of $n_1 + n_2$ points for construction of the confidence envelopes. For this reason the confidence envelopes tend to be wide if $n_1 << n_2$ and narrow if $n_1 >> n_2$. This is a disadvantage of this assessment method.

To interpret departure of $\hat{g}_{12}(r)$ from random labeling we analyze the conditions under which $\hat{g}_{12}(r) = \hat{g}_{1+2,1+2}(r)$, thus

$$\hat{g}_{12} = \frac{P_{12}}{A_1}\frac{A}{n_2} = \frac{[n_1(P_{11}+P_{12}) + n_2(P_{22}+P_{21})]}{n_1A_1 + n_2A_2}\frac{A}{n_1+n_2} = \hat{g}_{1+2,1+2} \quad \text{(RL8)}$$

If we assume that edge effects are equilibrated (i.e., $A_1 = A_2$) and considering identity he $n_1 P_{12} = n_2 P_{21}$, we find the solution

$$P_{12} = \frac{n_2}{(n_1^2 + n_2^2)}\left[n_1P_{11} + n_2P_{22}\right] \quad \text{(RL9)}$$

which yields

$$\hat{g}_{12} = \left[\left(\frac{n_1}{n_1+n_2}\right)^2 \hat{g}_{11} + \left(\frac{n_2}{n_1+n_2}\right)^2 \hat{g}_{22}\right] \quad \text{(RL10)}$$

Therefore, variant 1 assesses departure from random labeling by comparing the bivariate *g*-function with the density-corrected average of the two univariate *g*-functions of the two component patterns. In the simplest case when both patterns have the same number of points (i.e., $n_1 = n_2$), we find that variant 1 compares

the bivariate g_{12} directly to the average of the two univariate g functions [i.e., $g_{12} = (0.5g_{11} + 0.5g_{22})$], and for $n_1 \neq n_2$ the averaging is slightly more complex.

When using variant 1 we assess departure from random labeling relatively to the structure of the univariate component patterns. If $\hat{g}_{12}(r) > \hat{g}_{1+2,1+2}(r)$ type points 2 are at scale r stronger correlated to points 1 than expected by the average aggregation of the two component patterns.

4.2.3. Variant 2, the difference $g_{12} - g_{11}$

The difference $\hat{g}_{11}(r) - \hat{g}_{12}(r)$ evaluates whether type 1 points tend to be surrounded by other points of type 1 (i.e., points of type 1 are correlated at scale r). With the definitions of equations RL4 and RL5 we find

$$\hat{g}_{11}(r) - \hat{g}_{12}(r) = \frac{P_{11}}{A_1}\frac{A}{n_1} - \frac{P_{12}}{A_1}\frac{A}{n2} = \frac{A}{A_1}\left[\frac{P_{11}}{n_1} - \frac{P_{12}}{n_2}\right] \qquad \text{(RL11)}$$

Thus, a positive difference $\hat{g}_{11}(r) - \hat{g}_{12}(r) > 0$ indicates that rings with radius r around type 1 points contain relatively more type 1 than type 2 points. The term "relatively" refers to the correction that considers the different intensities of pattern 1 and 2 (i.e., dividing P_{12} by the total number n_2 of type 2 points in the study region, and dividing P_{11} by the total number n_1 of type 1 points in the study region). In other words, type 1 points are relatively more frequent at distance r around type 1 points than type 2 points. Thus, a positive difference $\hat{g}_{11}(r) - \hat{g}_{12}(r)$ indicates that type 1 points are at distance r positively correlated with other type 1 points.

4.2.4. Variant 3, the difference $g_{21} - g_{11}$

A negative difference $\hat{g}_{21}(r) - \hat{g}_{11}(r)$ indicates that type 1 points are more frequent in rings around other type 1 points than in rings around type 2 points. With the definitions of equations RL4 and RL5, the identity $n_1 P_{12} = n_2 P_{21}$ and the assumption that edge effects are equilibrated (i.e., $A_1 = A_2$) we find:

$$\hat{g}_{21}(r) - \hat{g}_{11}(r) = \frac{P_{21}}{A_2}\frac{A}{n_1} - \frac{P_{11}}{A_1}\frac{A}{n_1} = \frac{A}{A_1}\left[\frac{P_{12}}{n_2} - \frac{P_{11}}{n_1}\right] \qquad \text{(RL12)}$$

Thus, the difference $\hat{g}_{21}(r) - \hat{g}_{11}(r)$ is equivalent to the difference $\hat{g}_{12}(r) - \hat{g}_{11}(r)$ if edge effects are equilibrated.

4.2.5. Variant 4, the difference g_{22} - g_{11}

The difference $\hat{g}_{22}(r) - \hat{g}_{11}(r)$ does not evaluate the correlation of type 1 or type 2 points directly, but evaluates whether pattern 2 is more clustered than pattern 1, conditional on the structure of the joined patterns. Thus, it evaluates whether the given difference in the univariate clustering of pattern 1 and 2 is probable under the overall clustering of the joined pattern.

Note that a given bivariate pattern may show significant departure from random labeling, but the difference $\hat{g}_{22}(r) - \hat{g}_{11}(r)$ may not depict this departure.

4.2.6. Variant 5, the difference g_{12} - g_{21}

The difference $\hat{g}_{12}(r) - \hat{g}_{21}(r)$ evaluates the symmetry of the bivariate *g*-functions, i.e., whether or not type 2 points surround type 1 points in the same way as type 2 points are surround by type 1 points. The difference $\hat{g}_{12}(r) - \hat{g}_{21}(r)$ is positive if:

$$\frac{P_{12}}{A_1}\frac{A}{n_2} > \frac{P_{21}}{A_2}\frac{A}{n_1}, \qquad \text{(RL13)}$$

and considering the identity $n_1P_{12} = n_2P_{21}$ this is equal to $A_2 > A_1$. Thus, the difference $\hat{g}_{12}(r) - \hat{g}_{21}(r)$ evaluates directly whether the mean number of cells in rings around type 1 points (= A_1) is the same as the mean number of cells in rings around type 3 points (= A_2). The mean number of cells in rings around type *i* points (= A_i) will be smaller than the number of cells in a ring with the same radius because some points are located close to the boarder of the study region and have incomplete rings (figure D1).

4.2.7. Variant 6, the quotient $g_{12}/g_{1,1+2}$

A quotient $\hat{g}_{12}(r) / \hat{g}_{1,1+2}(r) < 1$ indicates that type 2 points are less frequent in rings around type 1 points than type 1 and 2 points in rings around type 1 points. With the definitions of equations RL4 and RL5 we find:

$$\frac{\hat{g}_{12}(r)}{\hat{g}_{1,1+2}(r)} = \frac{\frac{A}{n_2}\frac{P_{12}}{A_1}}{\frac{A}{n_1+n_2}\frac{P_{11}+P_{12}}{A_1}} = \frac{P_{12}}{P_{11}+P_{12}}\frac{n_1+n_2}{n_2}. \qquad \text{(RL14)}$$

Equation RL14 yields one if $n_1P_{12} = n_2P_{11}$. Therefore testing $g_{12}/g_{1,1+2} < 1$ is equivalent to testing $g_{11} - g_{12} > 0$. If $g_{12}/g_{1,1+2} < 1$, type 1 points are at distance r positively correlated with other type 1 points. If $g_{12}/g_{1,1+2} > 1$, the two types of points are positively correlated at distance r.

Equation RL14 indicates that estimation of $g_{12}/g_{1,1+2}$ requires only counting the number of type 1 and type 2 neighbors of type 1 points, but it does not require to count empty cells as necessary for estimation of the bivariate *g*-functions. Therefore, the value of the quotient does not depend on the univariate structure of the joined pattern, but only on the number of pairs of points at different distances.

Note that the quotient in equation RL14 is equivalent to the definition of the bivariate *g*-function for an irregularly shaped study region that comprises only cells with points. This provides an elegant and simple method for estimation of this quotient with *Programita*; the only change compared to calculation of $\hat{g}_{12}(r)$ is the selection of the modus "Irregularly shaped study region" instead of "All points in rectangle". Clearly, the input data need to be a list in grid ("data are given as list in grid" in window **select modus of data**) and include only cells with points.

4.2.8. Variant 7, comparing the quotients $g_{12}/g_{1,1+2}$ and $g_{21}/g_{2,1+2}$

We showed in the last section that the quotient $g_{12}/g_{1,1+2}$ provides information on departure from random labeling from the viewpoint of type 1 points (i.e., correlation of type 1 points). Conversely the quotient $g_{21}/g_{2,1+2}$ provides information on departure from random labeling from the viewpoint of type 2 points. It is interesting to compare both viewpoints, thus to assess whether of not the correlation of type 1 to type 1 points and type 2 to type 2 points is symmetric. This question can be answered by testing the difference

$$\frac{\hat{g}_{12}(r)}{\hat{g}_{1,1+2}(r)} - \frac{\hat{g}_{21}(r)}{\hat{g}_{2,1+2}(r)} \qquad \text{(RL15)}$$

Simple arithmetic manipulations, including the identity $n_1 P_{12} = n_2 P_{21}$, show that the difference of equation RL15 is zero if

$$P_{21}(r) + P_{22}(r) = P_{12}(r) + P_{11}(r)\,. \qquad \text{(RL16)}$$

Equation RL15 yields a positive difference if type 2 points have at distance r more neighbors than type 1 points have neighbors. This indicates that type 2 points are mainly located in areas of higher point density whereas type 1 points

are located in areas of lower point density. Thus, variant 7 evaluates intervention of a first-order effect.

Similarly to the quotient $g_{12}/g_{1,1+2}$, *Programita* estimates equation RL15 if you select g_{12} - g_{21} and the modus "Irregularly shaped study region" instead of the modus "All points in rectangle" in the in window select modus of data.

4.2.9. Evaluation of the different variants of random labeling

We now summarize the biological effects and their interpretation which are depicted by the different variants of random labeling. Table 1 shows that there are basically 5 different features of bivariate patterns that can be tested with bivariate random labeling:

- **Correlation:** Variants 6, 2, and 3 investigate whether type 1 points are correlated with each other (conditional on the given structure of the joined pattern). These variants of random labeling depict positive or negative correlation of one type of points (or conversely negative or positive between the two types of points) in a direct way and have a straight-forward interpretation. Note that the correlation structure of the two patterns might or might not be symmetric (e.g., type 1 points may show correlation but type 2 points not). *Programita* computes variant 6 significantly quicker than the equivalent variants 2 or 3 because variant 6 does not require to count empty cells.
- **Interaction with heterogeneity of joined pattern:** Variant 7 reveals information about the symmetry of the correlation of type 1 and type 2 points. A positive difference $g_{12}/g_{1,1+2}$ - $g_{21}/g_{2,1+2}$ indicates that type 2 points have at distance r more neighbours (= type 1 and type 2 points) than type 1 points. Thus, type 2 points are mainly located in areas with higher intensity of the joined pattern whereas type 1 points are mainly located in areas of lower intensity. Thus, departure from random labeling depicted with variant 7 indicates that the process that assigns the labels to the points interacts with the heterogeneity of the joined pattern. This is an interesting feature of random labeling that is of special interest e.g., in case of fire (with a spreading mechanism that depends on the density of plants) or spread of a tree disease (that depends on distance between trees).
- **Correlation between patterns vs aggregation of component patterns**. Variant 1 compares the bivariate *g*-function to the average of the *g*-functions of the univariate component patterns (which is the expected *g*-function under random labeling). This test of random labeling may have

wide confidence envelopes if $n_2 >> n_1$.

- **Univariate structure of component patterns:** Variant 4 compares the aggregation (or regularity) of the univariate component patterns and reveal whether one pattern is more clustered (or less regular) than the other, conditional on the structure of the joined pattern. This test does not necessarily detect departure from random labeling.
- **Equilibrated edge correction:** Variant 5 investigates if one pattern tends to be closer to the boarder of the study region than the other (i.e., the edge correction is not equilibrated). Equilibrated edge correction is an assumption for the interpretation of variants 0, 1, 3, and 4.

Basic relations tested by the different variants of random labeling and interpretation. $P_{ij}(r)$ is the average number of type j points in rings with radius r centered in type I points, $A_i(r)$ is the average number of cells in rings with radius r centered in type i points, n_i is the number of points of pattern I in the study region comprising A cells. With this definitions, the grid-based estimate of the bivariate g function yields $g_{12}(r) = (A/n_2)\, P_{12}(r)/A_i(r)$.

Variant	Test	Assumption for interpretation	Basic relation tested with the variant	Interpretation of test for scale r
1	$g_{12}(r) < g_{1+2,1+2}(r)$	$A_1 = A_2$	$g_{12} > b^2\, g_{11} + (1 - b^2)g_{22}$ with $b = n_1/(n_1 + n_2)$	**Type 2 points are stronger correlated than expected by the average aggregation of the two component patterns**.
	"	$A_1 = A_2$ $n_1 = n_2$	$g_{12} > (g_{11} + g_{22})/2$	
2	$g_{12}(r) - g_{11}(r) < 0$	—	$P_{12}/n_2 < P_{11}/n_1$	Type 1 points are relatively more frequent at distance r around type 1 points than type 2 points around type 1 points. **Type 1 points are positively correlated with other type 1 points.**
2a	$g_{21}(r) - g_{22}(r) < 0$	—	$P_{21}/n_1 < P_{22}/n_2$	**Type 2 points are positively correlated with other type 2 points.**
3	$g_{21}(r) - g_{11}(r) < 0$	$A_1 = A_2$	$P_{12}/n_2 < P_{11}/n_1$	Equivalent to variant 2 if $A_1 = A_2$
3a	$g_{12}(r) - g_{22}(r) < 0$	$A_1 = A_2$	$P_{21}/n_1 < P_{22}/n_2$	Equivalent to variant 2a if $A_1 = A_2$
4	$g_{22}(r) - g_{11}(r) > 0$	$A_1 = A_2$	$P_{22}/n_2 > P_{11}/n_1$	**Pattern 2 is more clustered than pattern 1**, conditional on the structure of the joined patterns.
5	$g_{12}(r) - g_{21}(r) > 0$	—	$A_2 > A_1$	**Tests for equilibrated edge correction**. Mean number of cells at distance r from type 2 points is larger than that of type 1 points.
6	$g_{12}(r)/g_{1,1+2}(r) < 1$	—	$P_{12}/n_2 > P_{11}/n_1$	Equivalent to variant 2. **Type 1 points are positively correlated with other type 1 points**.
6a	$g_{21}(r)/g_{2,1+2}(r) < 1$	—	$P_{21}/n_1 > P_{22}/n_2$	Equivalent to variant 2a. **Type 2 points are positively correlated with other type 1 points**.
7	$g_{12}(r)/g_{1,1+2}(r) - g_{21}(r)/g_{2,1+2}(r) > 0$	—	$P_{21} + P_{22} > P_{12} + P_{11}$	Heterogeneity of the joined pattern interacts with the process that assigns the labels. **Type 2 points are mainly located in areas with high intensity of the joined pattern**.

4.2.10. Random labeling and grass tufts (RL_2.res)

This example analyzes the spatial pattern of grass tufts in a semiarid grass-shrub steppe in Patagonia, Argentina. We analyze the pattern of *Stipa speciosa* (pattern 1) in relation to the joined pattern of all other grass species present at the study plot (pattern 2, *Poa ligularis*, *Stipa humilis*, *Stipa ibari*, and *Carex sp*). The study plot comprises a 133 × 91 cell rectangle with 20cm × 20cm cells, covering approximately a 27m × 18m area of the shrub-grass steppe.

We use the random labeling null model to investigate whether *S. speciosa* tufts are randomly distributed among all grass tufts. We use random labeling to investigate the spatial structure of tufts because we hypothesize that the processes and constrains that determine the locations of the grass tufts are the same for all species and that the labels may depend on factors which are independent on those which determine the location of the tufts.

1) highlight the data file "RL2.dat" in window **Input data file.** The locations of tufts of *S. speciosa* are pattern 1 (red dots), and the tufts of the other grass species (*P. ligularis*, *S. humilis*, *S. ibari*, and *Carex* sp) are pattern 2 (green dots):

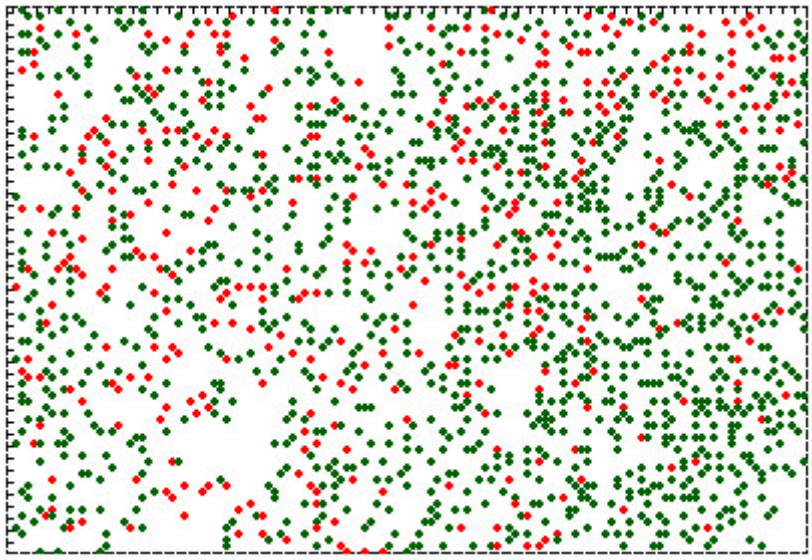
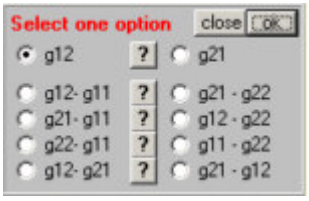


Figure RL1. Grass tufts in a Patagonian shrub-grass steppe. Red: tufts of *S. speciosa*, green: all other gras tufts.

2) select "List" in **How are your data organized**
3) select "Analyze all data in rectangle" **in Give modus of analysis**
4) select "Data are given as list in grid" in **Select modus of data**
5) select in box **ring width** a ring width *dr* = 3. For *dr* = 1 the *O*-ring statistic for random labeling has a somewhat jagged plot at smaller scales *r*.

6) click button "Calculate index".
7) Enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Random labeling".
8) Provide "99" for the number of replicate simulations of the random labeling null model
9) click "Calculate index". *Programita* now performs the simulations of the random labeling null model. After termination of the simulations a window appears:



where can select the different pairwise differences between *g*- (or *L*-) functions which evaluate different biological effects.
10) enable g12 which corresponds to univariate random labeling and to the test for bivariate random labeling proposed by Goreaud and Pèlissier (2004):

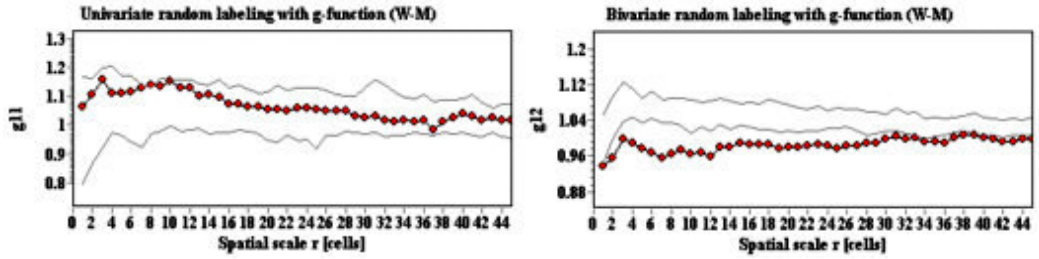


Figure RL2. Univariate (left) and bivariate random labeling (right). The confidence envelopes were constructed using 99 replicate simulations of the null model.

Univariate random labeling corrects for an underlying environmental heterogeneity by using a control pattern (in our example the locations of tufts of all other grass species) which is more abundant than the pattern of cases (the locations of the tufts of *S. speciosa*). Under the assumption that environmental heterogeneity conditioned the locations of pattern 1 and 2 in the same way, univariate random labeling investigates whether there is aggregation among tufts of *S. speciosa*.The environmental heterogeneity in this system is given by shrubs, which cover in our plot some 11% of the area (figure at the right with red: grass tufts, black: dead and living shrubs). The analysis with univariate random labeling shows that the aggregation of *S. speciosa* is not significantly different from the overall degree of aggregation of all grass tufts.

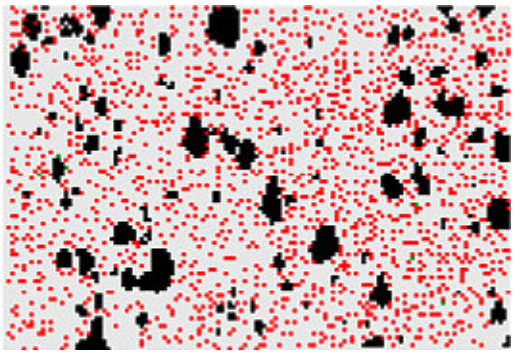


Figure RL3. Shrubs (black) and grass tufts (red) in the Patagonian shrub-grass steppe.

11) Bivariate random labeling with variant 1 shows that estimates of $g_{12}(r)$ are below the confidence envelopes for random labeling, therefore type 2 points are at scale r stronger correlated than expected by the average aggregation of the two component patterns.
12) To show that random labeling compares the g_{11}- and g_{12}-function to the univariate $g_{1+2,1+2}$ of the joined pattern, we plot the g-functions (black dots), the confidence envelopes (lines) and the univariate $g_{1+2,1+2}$-function of the joined pattern (white dots) in the same graphic. We find that the confidence envelopes for g_{11}- and g_{12} are both perfectly symmetric to $g_{1+2,1+2}$ which confirms our interpretation of this test:

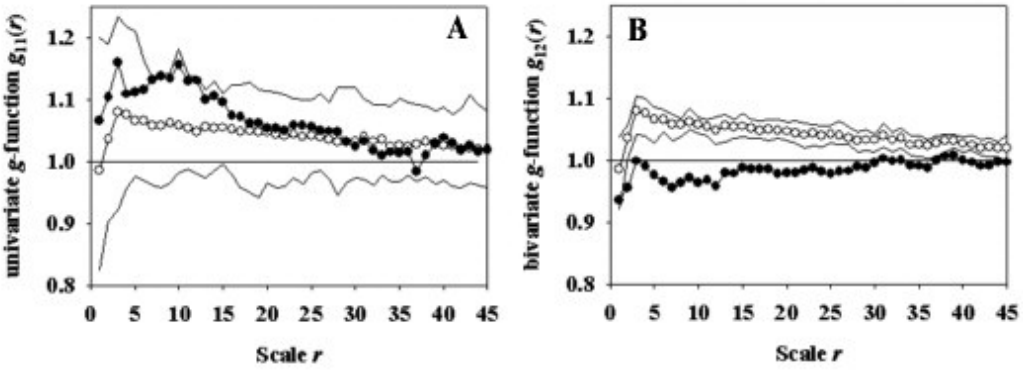


Figure RL4. Univariate random labeling (left) and bivariate random labeling with variant 1 (right) of the pattern shown in figure RL1. Comparison of the confidence envelopes (lines) and the the univariate $g_{1+2,1+2}$ of the joined pattern (open circles).

Note that the confidence envelopes of g_{11} are much wider than the confidence envelopes of g_{12}. This is because the number of points varies greatly between patterns (n_1 = 395 and n_2 = 1285). Each randomization of pattern 1 occupies only 24% of the total number of points, which leaves room for many different spatial configurations that include the actual configuration of pattern 1. On the other hand, the confidence interval of g_{12} is quite narrow since each randomization of pattern 2 occupies 76% of the total number of points. This leaves little room for differing spatial configurations and the actual configuration of pattern 2 is not probable. This results point to a weak point of variant 1: the number of points does greatly influence the

confidence envelopes.

13) Figure RL4 right shows that the correlation of type 2 points is significant for almost all scales studied. This suggests a first-order effect: the intensity of *S. speciosa* (pattern 1) tends to be higher in areas where the density of type 2 points (all other tufts) is lower. To illustrate this result, we plot the (density-corrected) intensity of type 2 points (i.e., the moving window estimate with a moving window of $R = 10$) together with the points of pattern 1. This figure depicts, in a spatially-explicit way, the bivariate K-function, i.e., the number of points of pattern 2 in circles with radius $R = 10$ (the intensity) in relation to the points of pattern 1 (the white dots). Points of pattern 1 are more frequently located in areas with low intensity of pattern 2 (blue area):

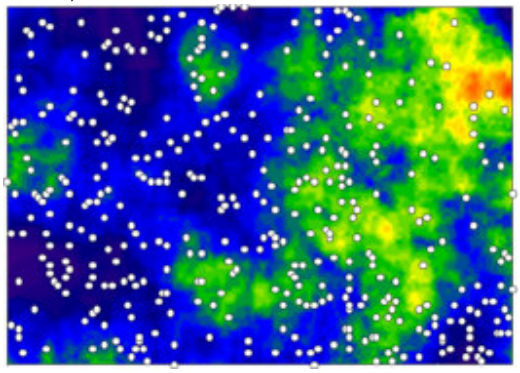


Figure RL5. Intensity of pattern 2 (lowest intensity: blue, highest intensity red, with incrementing spectral colors) and points of pattern 1 (white dots).

14) Next we analyze the differences g12-g11 (variant 2) to investigate whether type 1 points are at distance r relatively more frequent around type 1 points than type 2 points around type 1 points. The difference g12-g11 shows significant negative correlation between pattern 1 and 2 at spatial scales $r = 5$ to 16 (which is equivalent to a significant positive correlation among type 1 points). Thus, points of pattern 1 tend to be more frequent in the neighborhood ($r = 5$ to 16) of pattern 1 than points of pattern 2. The inverse relation g21-g22 (variant 2a) shows at spatial scales $r = 1$ to 27 significant positive correlation among type 2 points. Thus, points of pattern 2 tend to be more frequent in the neighborhood of type 2 points than in the neighborhood of points of pattern 1. This result indicates that the spatial distribution of type 1 and type 2 points shows a tendency to segregation: type 1 points are correlated to other type 1 points, and type 2 points are correlated to other type 2 points.

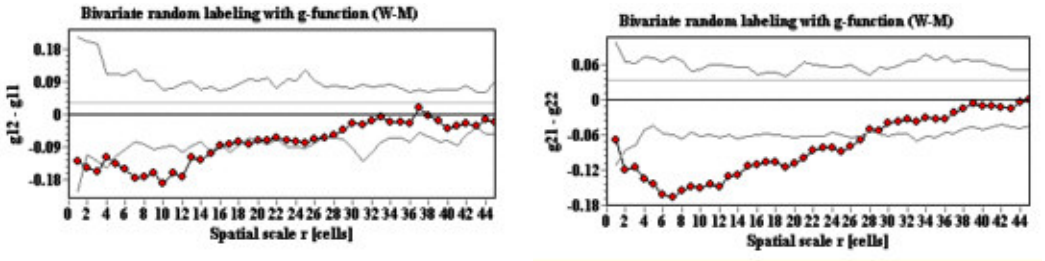


Figure RL6. Variant 2 of random labeling that investigates correlation among type 1 points (left) and type 2 points (right).

Analogously to figure RL5 we plot the difference in the number of type 2 and type 1 points in moving windows of radius $R = 10$ together with the locations of points of pattern 1. This plot is a spatially-explicit visualization of K_{12}-K_{11}. It shows clearly that points of pattern 1 are more frequently located in areas which have in their neighborhood more type 1 than type 2 points within a distance $R = 10$ (blue area).As a consequence, the difference g12-g11 depicts at scale $R = 10$ a correlation among type 1 points.

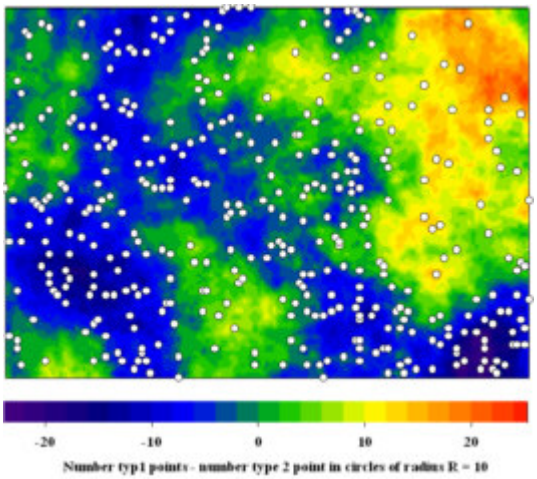
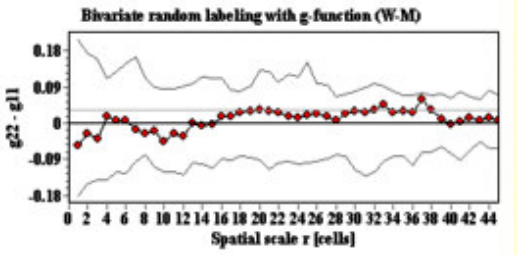
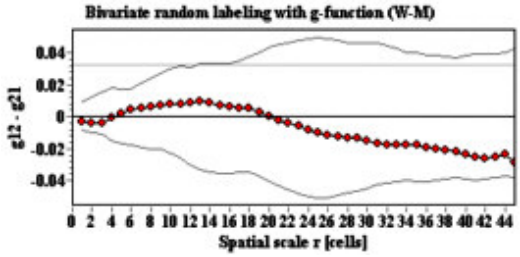


Figure RL7. Moving window estimate showing the difference in the number of type 2 and type 1 points in circles with radius $R = 10$ (lowest difference: blue, highest difference: red, with incrementing spectral colors). Pints of pattern 1 are shown as white dots.

15) Testing the difference g22-g11 under random labeling (variant 4) reveals that both patterns show, taken separately, the same degree of aggregation. This test does not depict the existing departure from random labeling.



16) Testing the difference g12-g21 under random labeling (variant 5) reveals that the bivariate *g*-functions are symmetric and that rings around type 2 points have on average the same area within the study region than rings around type 1 points.



17) To test variants 6 and 6a (which are equivalent to variants 2 and 2a) we repeat simulations of random labeling, but with the option "Irregularly shaped study" region instead of "analyze all data in rectangle", thus excluding cells without points. To obtain variant 6 we select "g12", and to obtain variant 6a we select "g21".

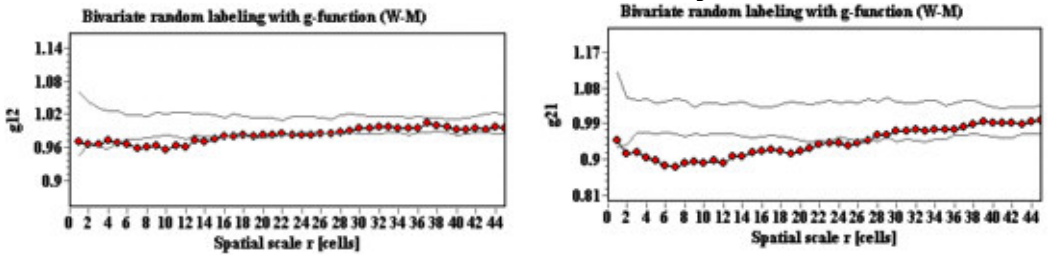
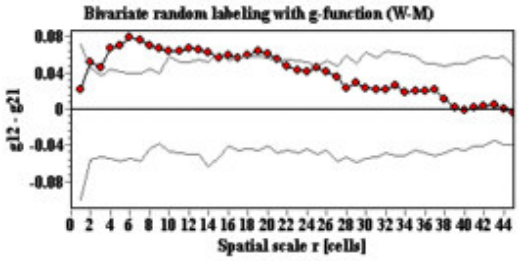


Figure RL8. Variant 6 of random labeling that investigates correlation among type 1 points (left) and type 2 points (right). As expected, the results are identical to the results of variant 2 (figure RL6).

18) To investigate departure from random labeling in relation to a possible first-order effect of the joined pattern (which was already indicated by variants 2, figures RL5 and RL7) we test variant 7. To obtain variant 7 we select "g12-g21" together with the option "Irregularly shaped study" region; thus excluding cells without points. The results of variant 7 reveal that the mean number of neighbors of type 2 points exceeds at distances r = 3 - 11 the mean number of neighbors of type 1 points. Thus, type 2 points are mainly located in areas with high intensity of the joined pattern:



19) Summarizing the results of random labeling for this data set we find that
- edge correction is equilibrated (variant 5), none of the patterns show a tendency to occur closer to the border of the study region),
- the univariate structures of both patterns show the same degree of aggregation (variant 4),

- type 1 points are correlated and type 2 points are correlated (figures RL6 and RL8), and
- points of pattern 2 occur mainly in areas of high intensity of the joined pattern (variant 7).

20) Interpretation of these results include:

- tufts of type 1 and type 2 show similar biological characteristics, but a limited seed dispersal radius leads to violation of random labeling because the probability of occurrence of one type depends on the neighbors. A limited seed dispersal radius may promotes positive correlation of type 1 and type 2 tufts.
- Univariate clumping of both patterns is not different. This suggests that the seed dispersal mechanism is the same for both types of tufts.
- Points of pattern 1 occur mainly in areas of lower overall tuft intensity. This result may indicate that higher densities of type 2 tufts (which are >3 times more frequent than tufts of pattern 1) in combination with a limited seed dispersal radius leave few safe sites for establishment of tufts of *S. speciosa* in the surrounding of type 2 tufts and promotes additional the tendency to spatial segregation.

4.2.11. Random labeling of adult and dead trees (RL_3.res)

This example analyzes the spatial pattern of dead and adult trees of one species in a tropical rainforest. This example extends the analyses of examples NS_1, NS_2, NS_3, and NS_4. We use the random labeling null model to investigate whether dead trees are randomly distributed among the joined pattern of dead and adult trees. This example is a classical example for application of the random labeling null model.

1) highlight the data file "RL3_1m" in window **Input data file.** This data file gives the locations of adult trees and dead trees of one species in cells of $1m^2$ within a 500m × 500m study region:

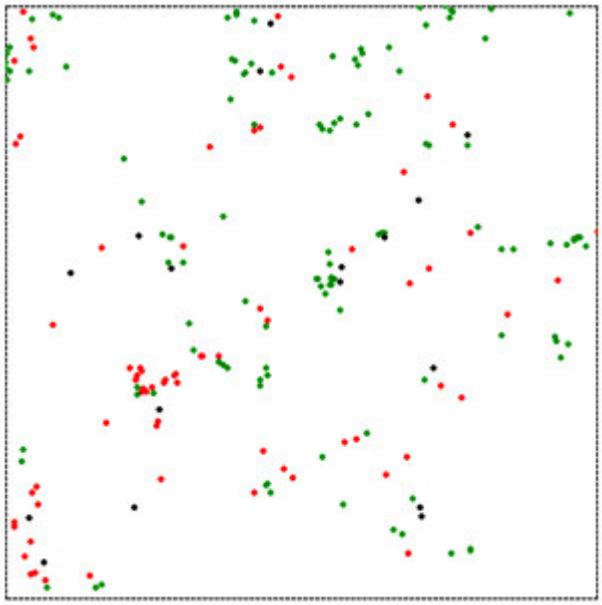
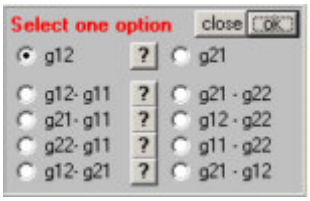


Figure RL3. Adult trees and dead trees in cells of $1m_2$. Adult trees (red), dead trees (green), and cells with dead and adult tree (black).

2) select "List" in **How are your data organized**
3) select "Data are given as list in grid" in **Select modus of data**.
4) select "Data are given as list in grid" in **Select modus of data**
5) select in box **ring width** a ring width *dr* = 3. For *dr* = 1 the *O*-ring statistic for random labeling has a somewhat jagged plot at smaller scales *r*.
6) click the button "change" in **set maximal radius rmax** and set the maximal scale *r* of the analysis to r_{max} = 50.
7) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data. The visualization of the data (figure RL3) shows that the adults are clearly clustered.
8) select "Analyze all data in rectangle" **in Give modus of analysis**
9) Enable the check box "Calculate confidence interval" on

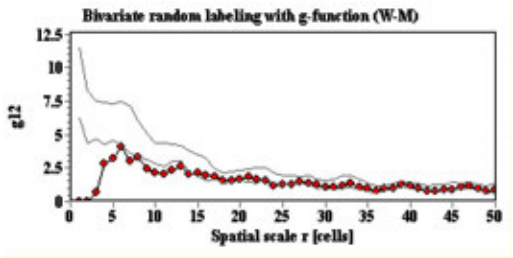
the upper left. A window with settings for null models appears. Select "Random labeling".

10) Provide "19" for the number of replicate simulations of the random labeling null model
11) click "Calculate index". *Programita* now performs the simulations of the random labeling null model. After termination of the simulations a window appears:

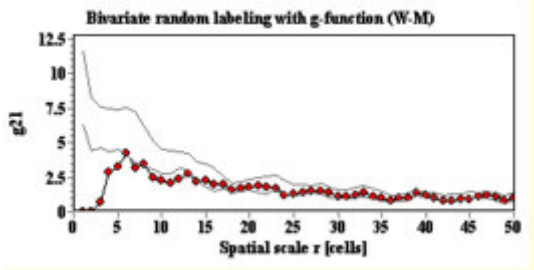


You can select the different pairwise differences between *g*- (or *L*-) functions which evaluate different biological effects.

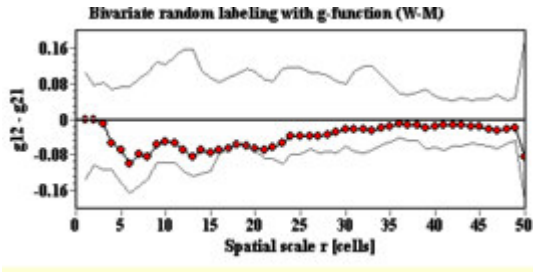
12) enable g12 (variant 1) which corresponds to the test for bivariate random labeling proposed by Goreaud and Pèlissier (2004):



The bivariate g12 is for scales r = 1 - 14 below the confidence interval of random labeling which indicates that the locations of dead trees are significantly correlated, conditional on the joined locations of adult and dead trees. Especially, there is a minimal distance between adult trees and dead trees of some 4 m which could be the caused by non-overlapping canopies of adult trees. The inverse relation g21 is perfectly symmetric:

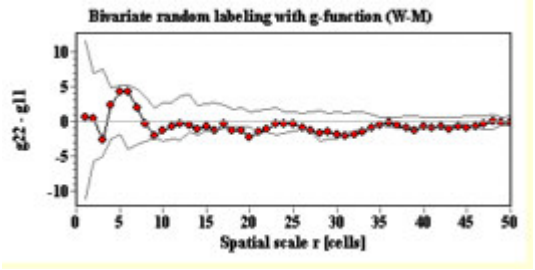


which indicates that adult trees are as well correlated at scales r = 1 - 14. Consequently, the edge correction is perfectly symmetric as indicated by the difference g12-g21 (variant 5):



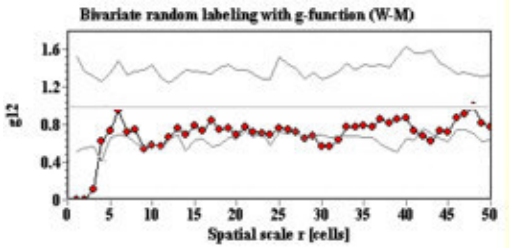
Note the difference in the scaling of the y-axis. The maximal value for e.g., g21 is 12.5 whereas the maximal value for the difference is about 0.2.

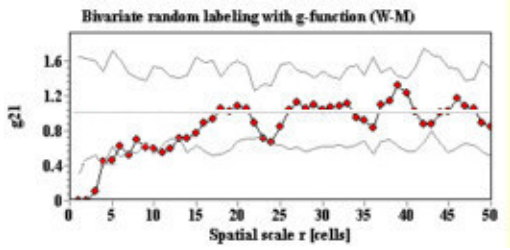
13) Analyzing the difference g22-g11 (variant 4) indicates that the degree of aggregation of both patterns is not significantly different:



This result strengthens the previous results of examples NS_2 and NS_3 that indicated that the overall clustering of adults and recruits at a scale of some 30m is a result of environmental heterogeneity that affects all life-stages of the species in the same way.

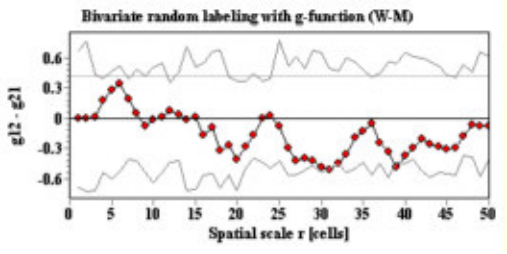
14) To test variants 6 and 6a (which are equivalent to variants 2 and 2a) we repeat simulations of random labeling, but with the option "Irregularly shaped study" region instead of "analyze all data in rectangle", thus excluding cells without points. To obtain variant 6 we select "g12", and to obtain variant 6a we select "g21":





We find that adult trees are positively correlated at scales r = 1 - 3. This correlation is caused by the non-overlapping canopies. Additionally, they show at all scales a non-significant tendency to correlation and at scales around r = 30, adult trees are weakly positively correlated. This is the correlation to the next cluster. The inverse relation g21 indicates that dead trees show the same correlation at small scales than adult trees which is caused by non-overlapping canopies, but they show a weak positive correlation up to scales r = 12. Note that variant 1 and variant 6 have a different interpretation and different confidence intervals.

15) To investigate departure from random labeling in relation to a possible first-order effect of the joined pattern we test variant 7. To obtain variant 7 we select "g12-g21" together with the option "Irregularly shaped study" region; thus excluding cells without points. The results of variant 7 show that the correlation among dead trees and the correlation among adult trees is symmetric and thus do not provide indications of a significant first-order effect:



4.3. Antecedent condition

If the two types of points were not created at the same time, but in sequence, pattern 2 did not influence the development of pattern 1, but pattern 1 may influence the development of pattern 2. An appropriate null model for this biological situation needs to consider the antecedent condition. For example, for investigating the relationship between adult trees (pattern 1) and seedlings (pattern 2) an appropriate null model to test for competition (repulsion) or attraction (facilitation) would be to randomize the locations of the seedlings (because they could potentially be found at the entire study region) and to keep the locations of the trees fixed. Randomizing the locations of the trees would be inappropriate because they did not change their position during the development of the seedlings. Moreover, possible repulsion or attraction between seedlings and trees might be obscured by randomizing the locations of the trees. Another example where one may keep the locations of pattern 1 fixed and specify only a null model for pattern 2 is the relation between shrubs (fixed) and grass tufts. In this case the null model distributes grass tufts at random over the area not occupied by shrubs. Departure from the null model (e.g., there are more tufts in the neighbourhood of shrubs than expected under this null model) may indicate facilitation.

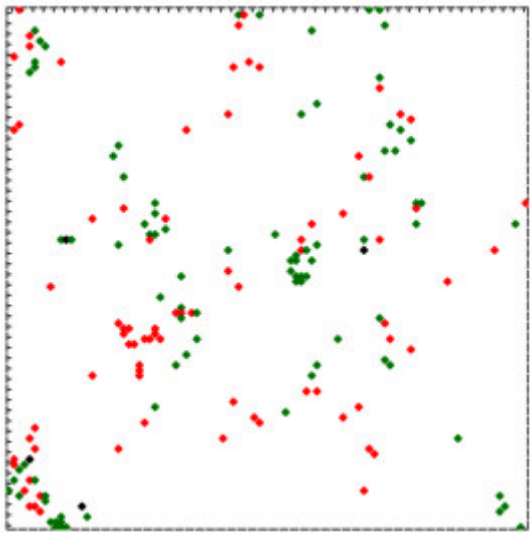
Because a null model with an antecedent condition specifies only the null model of the second pattern, all univariate null models (e.g., heterogeneous Poisson, hard core, Neyman-Scott cluster null model) may be used.

4.3.1. Trees and recruits (A_1.res and derivates)

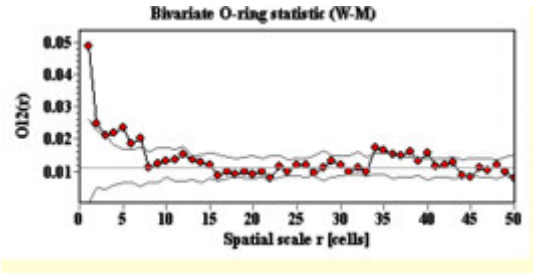
The data for this example are adult trees and recruits of one species in a 500m × 500m plot of tropical forest. The spatial distribution of this tree species appears clumped, which may be caused by environmental heterogeneity. To investigate the relation between recruits and adult trees we proceed in several steps by contrasting the data to different null models. In the **first step** we use a null model that fixes the location of the adult trees (i.e., an antecedent condition) and randomize the location of the trees (thus ignoring a possible environmental heterogeneity). The assumption of this null model is that recruits could potentially be found all over the study region. In a **second step** we consider the environmental heterogeneity and use a Poisson null model for the distribution of the recruits.

```
1)  highlight the data file "A_1.dat" in window Input data file. This
    data set gives the location of adult trees and recruits at
    a meter scale, but has a resolution of 1 centimeter.
```

2) select "List" in **How are your data organized**
3) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "5.00".
4) click the button "change" in **set maximal radius rmax** and set the maximal scale *r* of the analysis to r_{max} = 50.
5) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data:

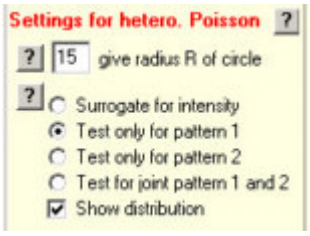


6) enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Pattern 1 fix, pattern 2 random".
7) provide "99" for the number of replicate simulations of the random labeling null model
8) click "Calculate index". *Programita* now performs the simulations of the random labeling null model. The results show that there are significantly more recruits in the r = 1 - 7 (1 - 35m) neighborhood than expected by a random distribution of recruits (i.e., recruits are attracted by adults). Interestingly, there is also a departure from the null model at scales r = 34 - 40 (170m - 200m) with more recruits than expected. This attraction is due to the patchy distribution of adults and describes the attraction to the next cluster of adults.



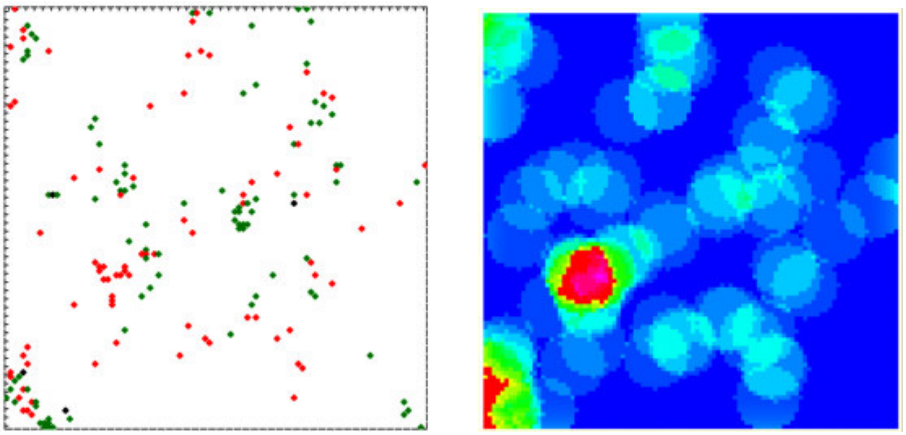
13) To investigate the hypothesis that the attraction may be partly a result from environmental heterogeneity that restricts the tree species to occur in clusters, we repeat the analysis of example A_1.res but use a heterogeneous Poisson null model for the recruits.
14) Select in the null model window "Pattern 1 fix, pattern 2

random" and "heterogeneous Poisson". A window with settings for the moving window estimate of the heterogeneous Poisson appears:

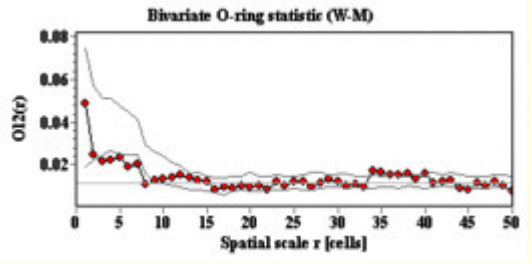


Select "Test only for pattern 1" (i.e., the intensity of pattern 1 will be used to distribute points of pattern 2), and select a radius *R* = 8 for the moving window. This radius is the radius with attraction.

15) click button "Calculate index". *Programita* now calculates the moving window estimate of the first-order intensity of the adults (right graph):



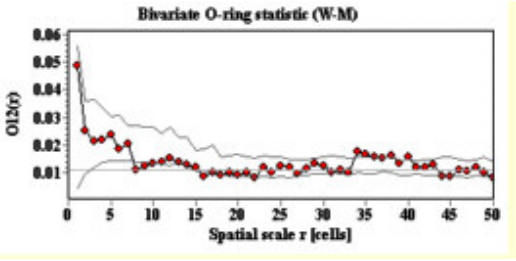
16) Click "ok" at the message window. *Programita* now performs the simulations of the heterogeneous Poisson null model and shows the patterns of the simulated null models. After termination of the simulations a graph appears showing the *O*-ring function of your data and the confidence envelopes of the heterogeneous Poisson null model:



The results show that the heterogeneous Poisson null model with a moving window radius *R* = 8 yields a too strong aggregation at scales r = 3 -9 (15m - 45m) and does thus not describe the data well.

17) In a next step we repeated the analysis with a radius of the moving window of R = 15, which is the scale at which

the bivariate *O*-ring statistic dropped to the expected value (i.e., the overall density of recruits). This null model describes attraction at small scales well, but leads to repulsion at intermediate scales:



We are thus not satisfied with these null models and search for a better one. An alternative to describe the common clustering of trees and recruits due to an heterogeneous environment is a bivariate Neyman Scott null model where parent events (which represent the clusters of the environmental heterogeneity) are randomly distributed and pattern 1 and pattern 2 are the offspring from these parents. The analysis of this null model is given in example NS_4.res.

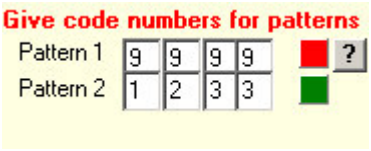
4.3.2. Shrubs and grass tufts (A_2.res)

In this example we extend the analysis of the grass-shrub steppe already started in example R_2.res and investigate the relation between grass tufts and shrubs. Because there is a hypothesized facilitation effect exerted by shrubs on grass tufts we use a null model with antecedent condition.

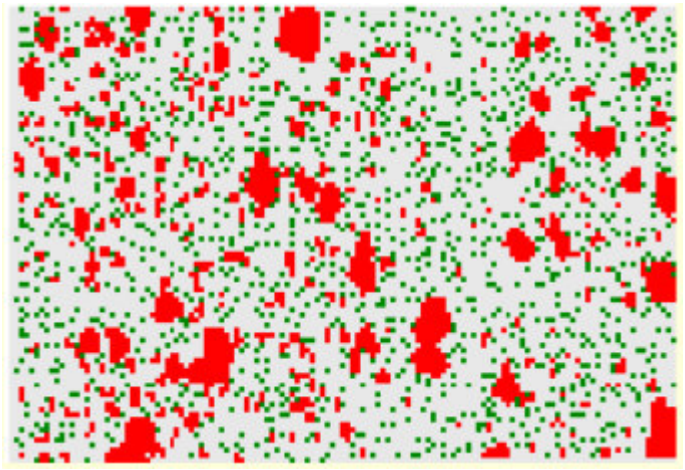
In this example we need to perform the analysis in the matrix mode for several reasons. First, the size of shrubs considerably exceeds the size of the grass tufts and an approximating of shrubs with points (as usual in point-pattern analysis) we would loose all information on the immediate neighbourhood relations between shrubs and grass tufts. Second, grass tufts do in general not grow inside shrubs and therefore we can only accept one category per cell (i.e., a cell is either empty, or covered by a shrub or by a grass tuft). Analysis in the matrix model under antecedent condition allows us to distribute the grass tufts (which each occupy exactly the area of one cell) randomly over the area of the study area not occupied by shrubs.

1) highlight the data file "A_2.dat" in window **Input data file.** This data is a categorical map that contains cells occupied by shrubs of all species (category 9) and cells occupied by grass tufts of all species (categories 1, 2, and 3).
2) select "Matrix" in **How are your data organized** and select "Matrix map" in **Select modus of data**. Provide the code numbers for the two patterns in the window **code numbers for patterns**: 9 for pat-

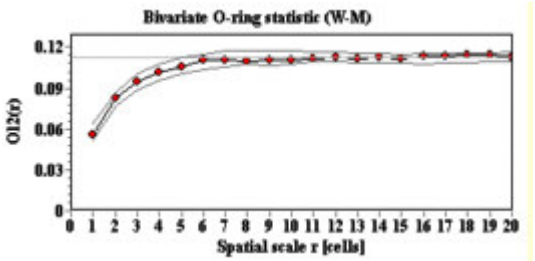
tern 1 and 1, 2, and 3 for pattern 2:



3) click the button "change" in set maximal radius rmax and set the maximal scale r of the analysis to r_{max} = 20. We are only interested in smaller scales where facilitation may occur. Click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data. Shrubs are red and grass tufts are green:



9) enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Pattern 1 fix, pattern 2 random".
10) provide "99" for the number of replicate simulations of the random labeling null model and click "Calculate index". *Programita* now performs the simulations of the null model where grass tufts are randomly distributed over the cells not occupied by shrubs and only one grass tuft is allowed per cell.



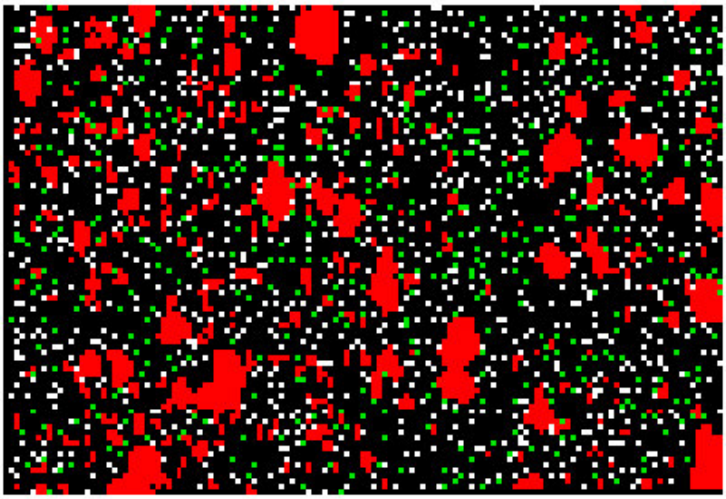
11) The results of the analysis show that the confidence intervals are perfectly symmetric to the bivariate *g*-function of the data. Thus, grass tufts are not closer to shrubs than expected by the antecedent condition CSR null model and we have therefore to reject the hypothesis of facilitation in the overall relation between shrubs of all species and grass tufts of all species.

4.4. Random labeling under antecedent condition

Random labeling is in some respect a hermaphrodite null model with characteristics of both, a bivariate null model and a univariate null model. This is because the relation between pattern 1 and pattern 2 is assessed conditionally on the location of all points. The process that distributes the labels (e.g., whether or not a tree is dead or burned or infected) determines only the location of one type of points within all points (e.g., the occurrence of the event dead, burned or infected). The locations of the second type of points follow automatically from the locations of type 1 points (i.e., all points which are not type 1).

Because random labeling shares characteristics of univariate null models we can extend the idea of random labeling to a situation which combines features of an antecedent condition with bivariate random labeling. Such a null model is appropriate for situations where the process that assigns labels may depend on a third pattern. Biological examples of such situations are burned and non-burned (non-serotinuos) shrubs in relation to a second (serotinuos) shrub species. In this case the question is whether burned shrubs are closer to the (serotinuos) shrub species than non-burned shrubs. This would correspond to a "kill my neighbour" strategy of the serotinuos shrub species. Another example is to extend the analyses of the shrub-grass steppe (examples RL_2.res and A_2.res) and investigate whether or not the grass tufts *S. speciosa* (pattern 1) are closer to shrubs than the tufts of all other grass species (pattern 2).

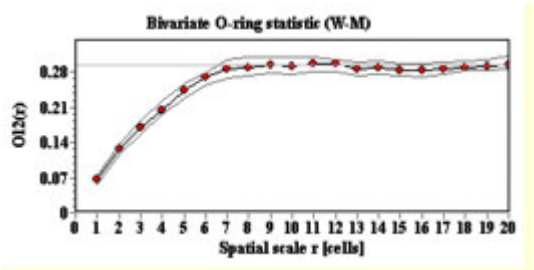
Because classical point pattern analysis with Ripley's *K*-function and the Wiegand-Moloney *O*-ring statistic allows only analysis of 2 patterns, but not three patterns as necessary for random labeling under antecedent condition, we use a dirty trick that takes advantage of the feature of *Programita* to calculate second-order statistics in any irregularly shaped study region supported by the underlying grid. We use the modus list with coordinates in a grid and code cells occupied by shrubs as patter 1, cells occupied by *S. speciosa* as pattern 2 and cells occupied by all other grass tufts are coded as empty cells. If we now apply the option "*irregularly shaped study region*" and the null model "*random labeling special*" *Programita* distributes the points of pattern 2 randomly over the locations of the study region not occupied by pattern 1. The bivariate *O*-ring statistic therefore investigates whether or not *S. speciosa* tufts are more frequently in the neighborhood of cells occupied by shrubs than tufts of all other grass tufts.



Map used in the example RL_A_1.res. Red: shrubs, green: grass tufts of the species *S. speciosa*, white: grass tufts of all other species, and black: bare ground.

4.4.1. Shrubs and grass tufts (RL_A_1.res)

1) highlight the data file "RL_A1.dat" in window **Input data file.** This data set contains the location of shrubs (pattern 1), grass tufts of *S. speciosa* (pattern 2), and grass tufts of all other species (empty cells).
2) select "List" in **How are your data organized**
3) select "Analyze all data in rectangle" **in Give modus of analysis**
4) select "Data are given as list in grid" in **Select modus of data**
5) select "irregularly shaped study region".
6) click button "Calculate index".
7) Enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Random labeling special".
8) Provide "99" for the number of replicate simulations of the random labeling null model
9) click "Calculate index". *Programita* now performs the simulations of the random labeling null model. The results



show that grass tufts of the species *S. speciosa* are not more frequently in the neighborhood of cells occupied by shrubs than tufts of all other grass tufts. Therefore we reject the hypothesis that *S. speciosa* differs in its relation to shrubs from grass tufts of all other species.

4.5. Bivariate cluster processes

4.5.1. Background

There are several possibilities to construct bivariate cluster processes. We show here and in the next section only two simple cases which are straight forward generalizations of the univariate Poisson cluster process.

The bivariate cluster process is the analogue to the univariate cluster process, but two types of points are generated (instead of one type) using a common set of parents.

Parent events form a CSR process. Parents may produce a random number of offspring of two different types (however, not each parent needs to produce offspring of both types) and offspring of both types are spatially distributed around their parent according to two bivariate probability density functions. The final bivariate pattern consists of the offspring of the two types only. Thus, both types of points are clustered around shared parents. This null model e.g., describes an environmental heterogeneity that affects both patterns in the same way.

If the number of offspring follows for both types of points a Poisson distribution and the location of the both offspring types, relative to the parent individual, have each a bivariate, Gaussian distribution, the offspring of each type follow a univariate Neyman-Scott process (e.g., Diggle 1983) and the bivariate pattern follows a bivariate Neyman-Scott processes. The *K*-function and the pair-correlation function *g*(*r*) for this bivariate Neyman-Scott process are given by:

$$\begin{aligned} K(r,\sigma_{12},\rho_{12}) &= \pi r^2 + \frac{1-\exp(-r^2/4\sigma_{12}^2)}{\rho_{12}} \\ g(r,\sigma_{12},\rho_{12}) &= 1+\frac{\exp(-r^2/4\sigma_{12}^2)}{4\pi\sigma^2\rho_{12}} \quad , \\ \text{with} \quad \sigma_{12}^2 &= (\sigma_1^2+\sigma_2^2)/2 \end{aligned} \tag{C4}$$

where ρ is the intensity of the parent process, and σ_{12}^2 is the resulting "bivariate variance" of the distance between type 1 and type 2 points. Note that the theoretical expectation of σ_{12}^2 is the average of the two univariate Gaussian distribution σ_1^2 and σ_2^2 that determine the locations of the type 1 and type 2 offspring relative to the parent. The unknown parameter ρ_{12} must be fit by comparing the

empirical $\hat{K}(r)$ with the theoretical *K*-functions $K(r, \sigma, \rho)$ (see Diggle 1983).

Comparison of the resulting parameters σ_{12}^2 and ρ_{12} of bivariate cluster process with the parameters of the two univariate component processes (σ_1^2, ρ_1, σ_2^2, and ρ_2) reveal which proportion of the parents are shared parents and whether there are some inconsistencies which may indicate that this null model does not well describe the data.

If all parents are shared we obtain $\rho_{12} = \rho_1 = \rho_2$. If only a certain proportion of the parents are shared (i.e., some of them are only cluster centres of type 1 or type 2 points) we may expect that $min(\rho_1, \rho_2) < \rho_{12} < \rho_1 + \rho_2$. In this case the total number of shared parents is $A(\rho_1 + \rho_2 - \rho_{12})$, the proportion of shared parents among all parents is $(\rho_1 + \rho_2 - \rho_{12})/\rho_{12}$, and the proportion of parents of pattern *i* is ρ_i/ρ_{12}. *A* is the total number of cells in the study region.

However, if we find $\rho_{12} >> \rho_1 + \rho_2$ we have an argument that the data are not well described by a bivariate Neyman-Scott process. In this case the fitted parameter ρ_{12} indicates that there are more shared parents than parents of the two individual component patterns. Similarly, this null model is only likely if $\sigma_{12}^2 \approx (\sigma_1^2 + \sigma_2^2)/2$. However, when evaluating the plausibility of the null model, we need to consider the uncertainty in the fit and construct confidence intervals of σ_{12} and ρ_{12}.

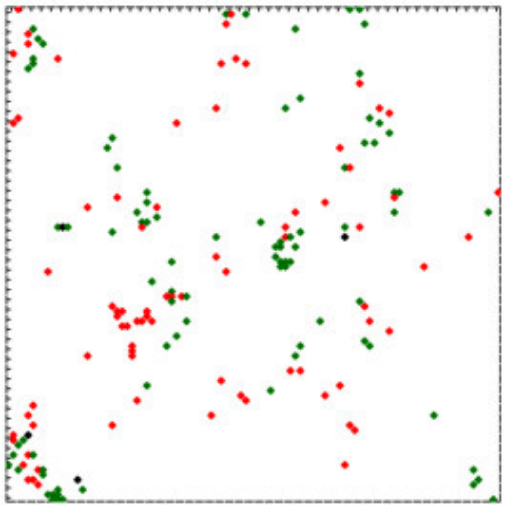
4.5.2. Implementation of bivariate Neyman-Scott process

The procedures for fitting a bivariate Neyman-Scott process to a bivariate pattern are analogous to the univariate case. The only difference is that it requires previous analysis of the univariate patterns.

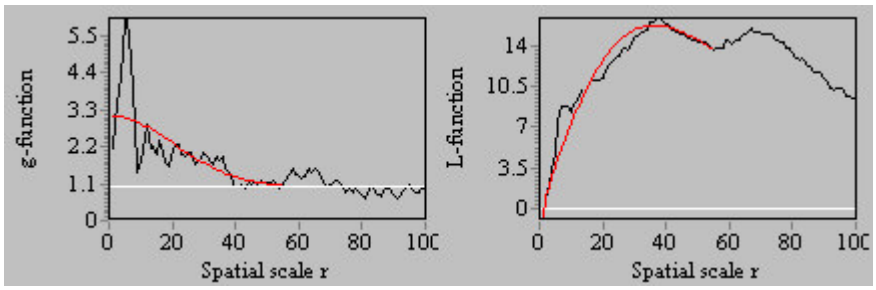
The implementation of the cluster null model based on the bivariate Neyman-Scott process equation C4 is analogously to the implementation of the univariate process, however, we consider the possibility that not all parents are shared. At the beginning of each simulation of the null model, *Programita* determines the random locations for the parents (= trunk[$A\rho_{12}$], *A* = number of cells in study region, and the function trunk[x] truncates x to the nearest integer), and the number of parents of pattern 1 (= trunk[$A\rho_1$) and of pattern 2 (= trunk[$A\rho_2$]). It uses only the first trunk[$A\rho_1$] parents for simulation of type 1 points, and the last trunk[$A\rho_2$] parents for simulation of type 2 points. In this way *Programita* uses three types of parents: parents only for type 1 points, parents for type 1 and type 2 points, and parents only for type 2 points.

4.5.3. Adult trees and recruits (NS_4.res)

1) This example analyzes the spatial pattern of recruits and adult trees and continues the analysis of example A 1.res. The univariate analyses showed that the pattern of the recruits is the superposition of clusters at a small and a larger scale (example NS 2.res) whereas the adults are clustered at only one scale (example NS 3.res). The larger scale of recruit clustering coincides with the scale of adult clustering which suggests that the clustering describes basically the environmental heterogeneity.
2) Highlight the data file "A_1.dat" in window **Input data file.** This data set gives the location of adult trees and recruits at a meter scale, but has a resolution of 1 centimeter.
3) select "List" in **How are your data organized**
4) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
5) click the button "change" in **set maximal radius rmax** and to set the maximal scale r of the analysis to r_{max} = 100, and select a ring width of dr = 3.
6) click button "Calculate index", *Programita* shows you the pattern and calculates the O-ring function of the data:

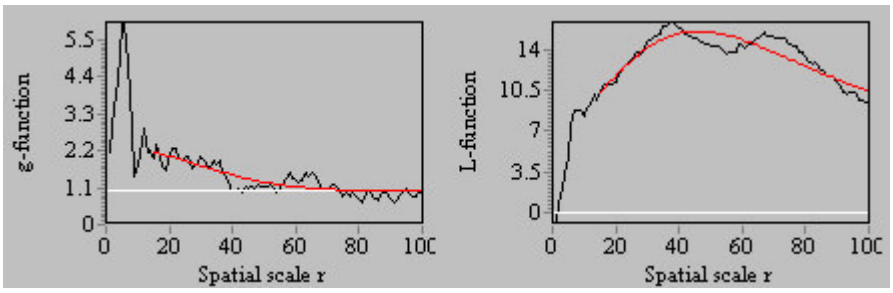


7) To determine Monte Carlo confidence intervals for the bivariate Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "bivariate Neyman-Scott" and press ok.
8) *Programita* calculates the g- and the L-function for r = 1 to r_{max}:

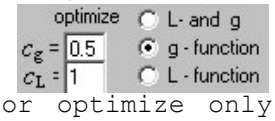


At this point we are confronted with a problem that may frequently appear if the data we want to fit do probably not follow the bivariate Neyman Scott process perfectly. Remember that the recruits show a double-clustered pattern (see example NS 2): we fit two different solutions for the fit.

- If we adjust equation C4 for scales r = 1 - 55 the best fit yields σ_{best} = 14.24 and ρ_{best} = 0.000185 which corresponds to some 46 parents (graphic above).
- If we adjust equation C4 for scales r = 15 - 100 the best fit yields σ_{best} = 19.77 and ρ_{best} = 0.000154: which corresponds to some 38 parents (graphic below).



We can also switch between the two solutions if we select r = 6 - 100 and optimize only the g-function (first solution):



or optimize only the L-function (second solution). However, both solutions look reasonable. To assess which of the two solutions is more probable we first check the plausibility (or biological interpretation) of the two solutions and next simulate the two corresponding processes to find confidence envelopes.

9) For the biological interpretation of the two solutions we compare them to the results of the univariate analyses (NS 2.res and NS 3.res). The two solutions of the bivariate analysis were:

- σ_{best} = 14.28 and ρ_{best} = 0.000185 (46 parents)
- σ_{best} = 19.77 and ρ_{best} = 0.000154 (38 parents)

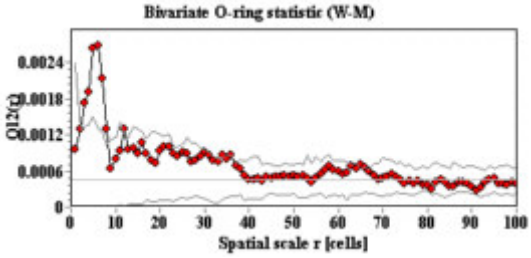
and the results of the univariate analyses were:

- σ_{1best} = 14.08, $100\rho_{1best}$ = 0.0083 (20 parents)
- σ_{2best} = 14.40, $100\rho_{2best}$ = 0.0095 (23 parents)

Thus, the first solution yields an excellent accordance with the expected theoretical variance $\sigma_t^2 = (\sigma_{1best}^2 + \sigma_{2best}^2)/2$, and the number of parents of the component processes (20 + 23 =43) is approximately the number of parents of the bivariate case (46). The interpretation of this is that the cluster centers of the two patterns are basically disjunct.

The second solution yields a poorer accordance with the expected theoretical variance σ_t^2, but there are shared cluster centers: 39% of all parents are only parents of pattern 1, 13% of all parents are parents for patterns 1 and 2, and 47% of all parents are only parents for pattern 2.

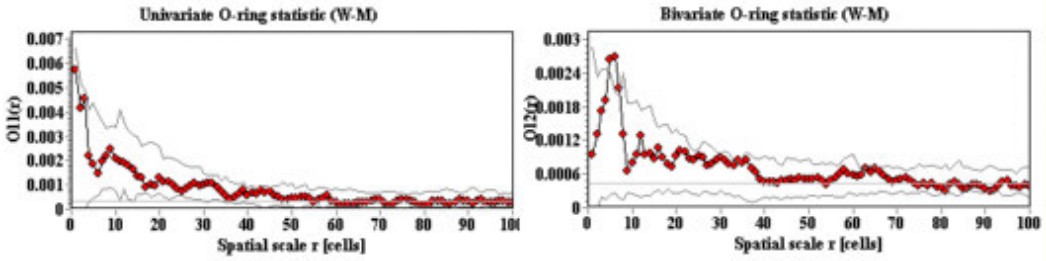
10) For construction of confidence envelopes for the first solution click "Calculate index". *Programita* now performs the simulations of the bivariate Neyman-Scott null model (NS_4_solution1.res):



This process describes the data well at spatial scales r > 10, but does not well describe the peak at scales r = 5-7.

11) For construction of confidence envelopes for the second solution click "Calculate index". *Programita* now performs the simulations of the bivariate Neyman-Scott null model (NS_4_solution2.res).

12) The simulation of 99 replicates of the Neyman-Scot null model shows that the data are well within the confidence envelopes of the null model (right figure), except for the scales *r* = 5 - 7:

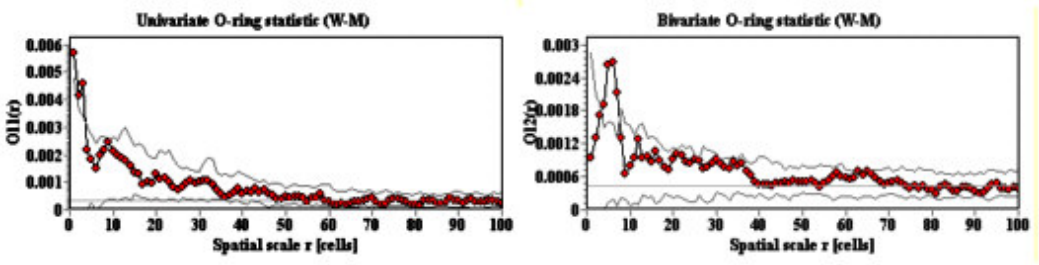


The confidence envelopes are much wider at smaller scales than those of the first solution.

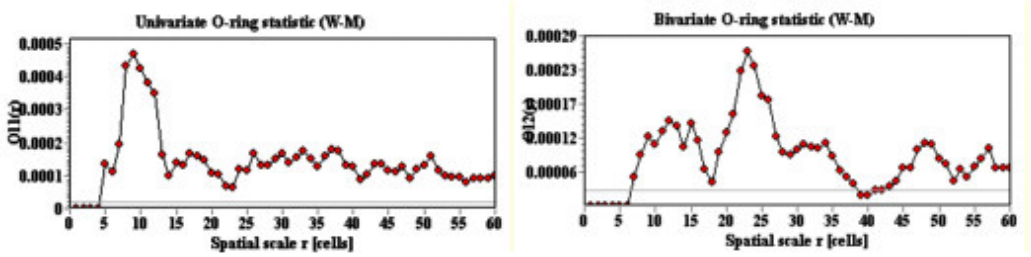
In summary, our results indicate that the locations of the recruits may for larger scales (i.e., r > 10) not directly be correlated with the locations of adult trees, but indirectly via an environmental heterogeneity that constrains both patterns in the same way (note that the univariate estimates for σ_{best} were the same for recruits and adult and also used for the bivariate Neyman Scott null model).

However, to exclude the alternative hypothesis that the locations of the recruits are directly linked to the locations of adult trees, we need to confront the data to the null model that corresponds to this hypothesis (i.e., the linked process using a bivariate Neyman-Scott cluster process with antecedent condition, example A C 4.res)

Analogous simulations of the null model, but using the estimated value of σ_{12}^2 (i.e., $\sigma_1 = \sigma_2 = \sigma_{12}$) instead of σ_t^2 (NS_4_sigma12.res) yields narrower confidence envelopes for the bivariate analysis but does not change the results at larger scales (i.e., r >15):



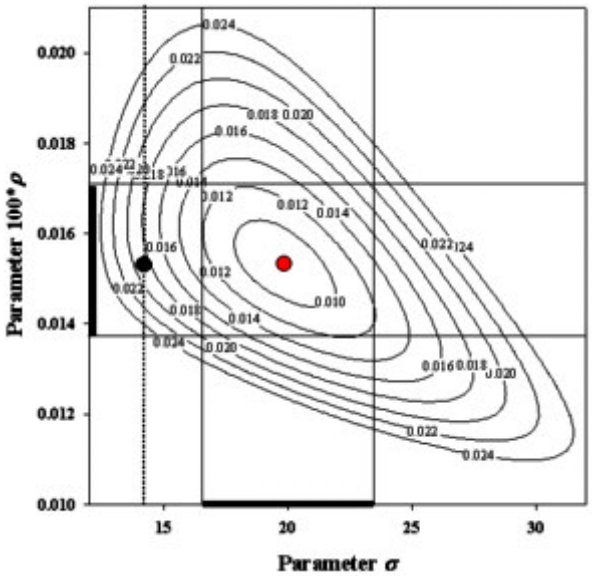
The bivariate *O*-ring function shows a significant peak at scales r = 5 - 7 (some 6m). In order to properly interpret this peak, we show the univariate and bivariate g-function with a resolution of 25cm and a ring width of 5 cells, thus "zooming" into the clusters of adult trees:



The left figure (univariate g-function of adult trees) shows that adult trees have a minimal distance of some 6 cells = 1.5m. There is a sharp peak at a scale of some 10 cells (= 4 m) which indicates aggregation of adults at 4m and the rest of the *g*-function shows clear indications for "virtual aggregation" which of course is the large scale clustering already discovered in example NS 3.res.

This results indicate that some adult trees occur within the larger clusters in randomly distributed clumps of some 2 or 3 trees with stems 4m away. If recruits and adults share a cluster, then recruits appear more frequently some 6m away from the adults (right figure above). This attraction could be an effect of avoidance of direct competition to adult trees in combination with a limited seed dispersal radius from direct seed rain.

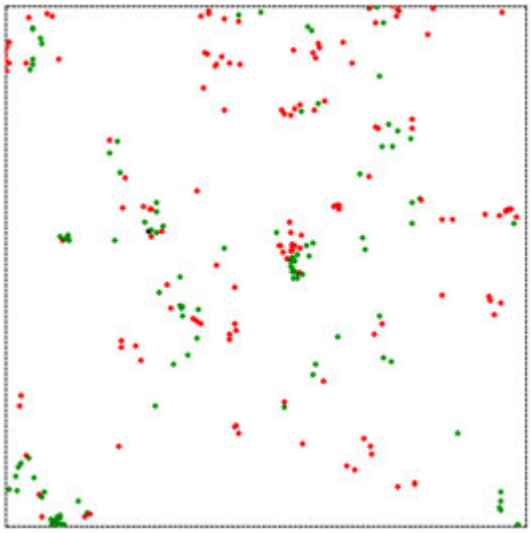
13) We saved the settings and results of the fit of the bivariate Neyman-Scott null model to the data (NS_4.fit) and use them now to estimate confidence intervals for the estimates of the parameters σ_{best} and ρ_{best} (see section "Constructing confidence intervals for σ and ρ"):



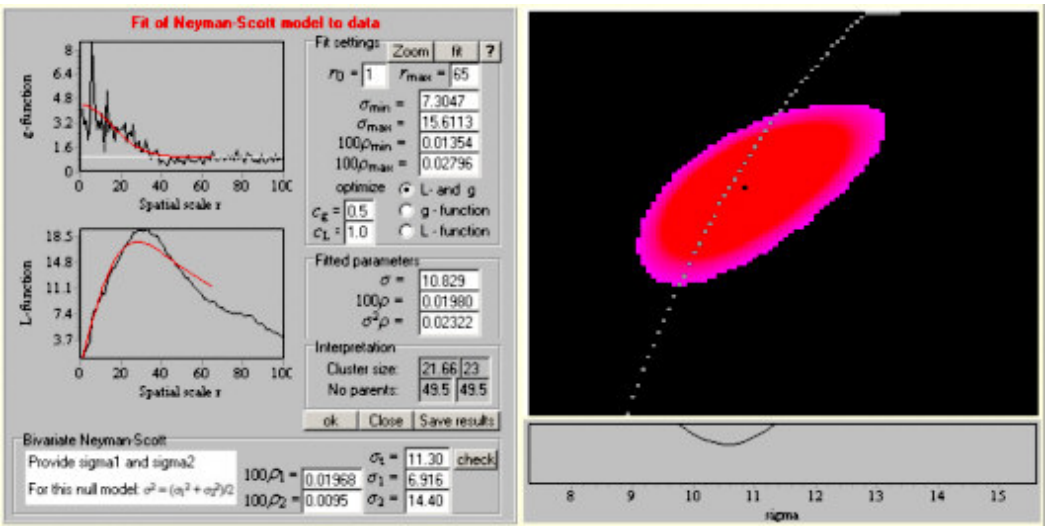
The confidence intervals for an error < 0.012 are shown as bold intervals at the axes. We find $\sigma \in (16.7, 23.3)$ and $100*\rho \in (0.014, 0.017)$. The theoretical value of sigma, $\sigma_t^2 = (\sigma_{1best}^2+\sigma_{2best}^2)/2 = 14.24$ is indicated by a dashed vertical line and the parameters used for simulations of the null model are shown as a black dot. σ_t^2 is outside the confidence interval for an error < 0.012, but inside the confidence intervals for error < 0.02, which is still small.

4.5.4. Dead trees and recruits (NS_5.res)

1) This example analyzes the spatial pattern of recruits and dead trees and complements the analysis of example NS 4.res. The univariate analyses showed that the pattern of the recruits is the superposition of clusters at a small and a larger scale (example NS 2.res) whereas the dead trees are clustered at only one scale (example NS 3b.res).
2) Highlight the data file "Dead_recruits.dat" in window **Input data file.** This data set gives the location of dead trees and recruits at a meter scale, but has a resolution of 1 centimeter.
3) select "List" in **How are your data organized**
4) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
5) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to $r_{max} = 50$.
6) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data:



7) To determine Monte Carlo confidence intervals for the bivariate Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "bivariate Neyman-Scott" and press ok.
8) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. You can specify the tuning constants r_{min}, r_{max}, and c for the fit in the window "Fit cluster process".
9) Select r_{max} = 1 and r_0 = 65. The default power transformations c = 0.5 for the *g*-function and c = 1 for the *L*-function are reasonable starting values. To optimize the *g*- and the *L*-function simultaneously enable "both, L- and g-function".
10) Click the button "fit" and *Programita* searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the *g*- and *L*- function of your data best (red line: fit, black line: data).
11) To optimize the parameter fit, press the button "Zoom". *Programita* now determines the probable range of the parameters. Next, press "fit" and *Programita* searches the best fit. We find σ_{best} = 10.8 and ρ_{best} = 0.000198:



This corresponds to a cluster size of some 22m, and some 49 parents events. The estimated 49 parents coincide with the estimated 49 parents found in the univariate analyses of the dead trees (example NS_3b.res)and is larger than the 24 parents of the recruits (NS_2.res). This indicates that all cluster centers of the recruits are also cluster centers for dead trees. Note that in the previous analysis of adult trees and recruits some cluster centers were exclusively for adults or recruits.

12) To save the settings and results of the fit click button "Save results" and provide "NS_5".

13) Provide the values of σ_{1best}, σ_{2best}, ρ_{1best}, and ρ_{2best} from the analysis of the univariate patterns. *Programita* uses the values of ρ_{best}, ρ_{1best}, and ρ_{2best} to determine which parent serves as cluster centre for pattern 1, pattern 2, or both patterns at the same time. Note that *Programita* does not use the fitted estimate of σ_{12}^2 for simulation of the null model, but your input $\sigma_t^2 = (\sigma_{1best}^2+\sigma_{2best}^2)/2$ which uses the estimates from the univariate analysis. Therefore it is important that you perform previous univariate analyses. The values from the bivariate analyses were:

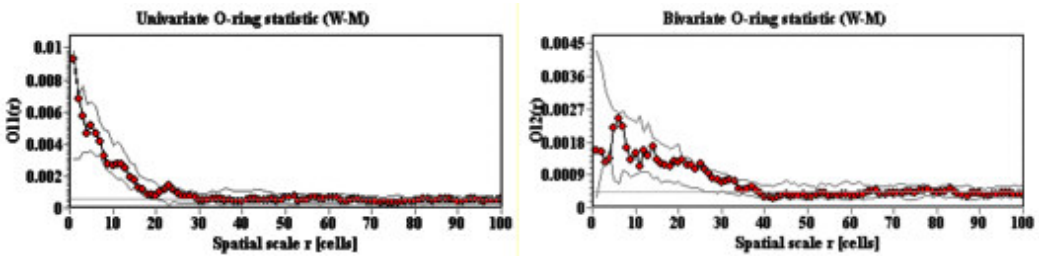
- σ_{best} = 10.8 and $100\rho_{best}$ = 0.0198 (some 49 parents)

The values from the univariate analyses were:

- σ_{1best} = 6.916, $100\rho_{1best}$ = 0.0197 (49 parents)
- σ_{2best} = 14.40, $100\rho_{2best}$ = 0.0095 (23 parents)

Thus, half of the cluster centers are cluster centers of the recruits, and all are centers of the dead trees.

14) Next click "ok", select a ring width of *dr* = 3, and click "Calculate index". *Programita* now performs the simulations of the bivariate Neyman-Scott null model. The simulation of 19 replicates of the Neyman-Scot null model shows that the data are well within the confidence envelopes of the null model (right figure):



These results indicate that the locations of the recruits may indirectly be correlated with the locations of dead trees. One hypothesis to explain this result is that the correlation is induced, similarly to the case of adult trees and recruits (example NS 4) via an environmental heterogeneity that constrains both patterns in the same way. However, this result leaves still room for the alternative hypothesis that the locations of the recruits are directly linked with the locations of the dead trees (i.e., a competition release effect combined with an environmental heterogeneity). To reject or accept the alternative hypothesis we need to confront the data to the null model that corresponds to this hypothesis (i.e., the linked process using a bivariate Neyman-Scott cluster process with antecedent condition, example A C 3.res)

4.6. Bivariate cluster processes under antecedent condition

4.6.1. Background

The bivariate cluster process under antecedent condition assumes that points of pattern 1 are parents of type 2 points and that pattern 1 is a random pattern. Thus, the process that creates pattern 2 is linked to pattern 1, e.g., a clustered distribution of seedlings around randomly distributed adult trees. The locations pattern 1 have to be preserved (i.e., an antecedent condition), and the type 2 points are randomized following a Neyman-Scott process null model where their parents are selected randomly among the type 1 points. Thus, only a certain proportion of type 1 points need to serve as centre for a cluster of type 2 points. The assumption for calculating the theoretical expectation for the g and L-function of this null model is that the points of pattern 1 are randomly distributed. The intensity of type 1 points which are cluster centers of pattern 2 is ρ_2. The K-function and the pair-correlation function $g(r)$ for this bivariate Neyman-Scott process are given by:

$$\begin{aligned} K(r,\sigma_{12},\rho_{12}) &= \pi r^2 + \frac{1-\exp(-r^2/2\sigma_{12}^2)}{\rho_{12}} \\ g(r,\sigma_{12},\rho_{12}) &= 1+\frac{1}{\rho_{12}}\frac{\exp(-r^2/2\sigma_{12}^2)}{2\pi\sigma_{12}^2} \end{aligned}, \qquad \text{(C5)}$$

with parameters ${\sigma_{12}}^2$ and ρ_{12}. The parameter ${\sigma_{12}}^2$ is the variance of the Gaussian distribution that determines the locations of type 2 points relative to their (type 1) parents, and the parameter ρ_{12} is the intensity of pattern 1 (i.e., $\rho_{12} = \lambda_1$). Note that equation C5 does not allow to determine the intensity ρ_2 of parents of type 2 points. This must be done by previous univariate analysis of pattern 2 (see equation C6).

If the cluster size is large (i.e., σ_{12} is large), equation C5 approximates $K(r) = \pi r^2$ and $g(r) = 1$ which corresponds to independence of the offspring from their parents.

The univariate g- and K-functions of pattern 2 are given through equation C1:

$$K_{22}(r,\sigma_2,\rho_2) = \pi r^2 + \frac{1-\exp(-r^2/4\sigma_2^2)}{\rho_2}$$
$$g_{22}(r,\sigma_2,\rho_2) = 1 + \frac{\exp(-r^2/4\sigma_2^2)}{4\pi\sigma_2^2\rho_2} \quad . \qquad \text{(C6)}$$

The parameter σ_2 determines the locations of type 2 points relative to their (type 1) parents and should coincide with the parameter σ_{12} of equation C5. The parameter ρ_2 is the intensity of type 1 points which are parents of type 2 points and should be smaller or equal than the intensity λ_1 of type 1 points. The proportion of points of pattern 1 which serve as parents for points of pattern 2 can be calculated by comparing the estimates of ρ_2 from the univariate analysis of pattern 2 with the intensity λ_1 of pattern 1.

4.6.2. Implementation of the antecedent condition cluster process

The procedures for fitting the bivariate Neyman-Scott process under antecedent condition to a bivariate pattern are analogous to the univariate case. The only difference is that it requires previous analysis of the univariate patterns for checking the assumption that pattern 1 is a random pattern and for determining the proportion of points of pattern 1 which serve as parents.

4.6.3. Antecedent cluster process (C_A_1.res)

The data for this example was created with an antecedent cluster process. A total of 25 type 1 points were randomly distributed over a 200 × 200 cell grid and all of them served as cluster centre for 100 points of pattern 2. The (bivariate) parameters of pattern 2 were:

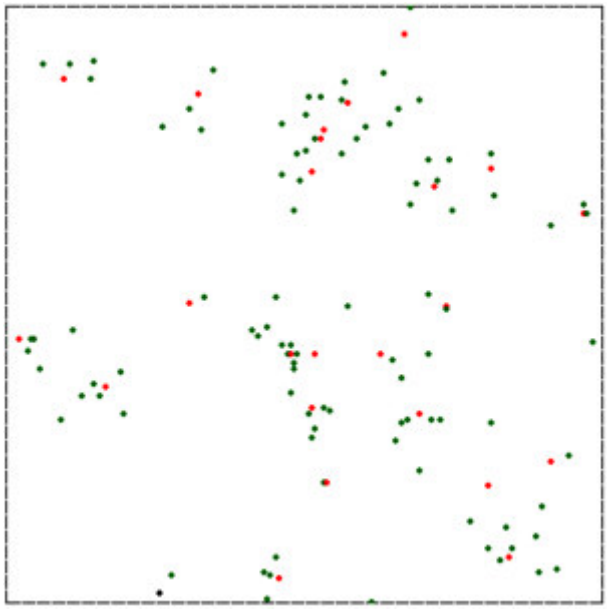
- $\sigma_{12} = 8$, and
- $\rho_{12} = 0.000625$.

In a first step we analyzed the two univariate component patterns and in a second step the bivariate pattern.

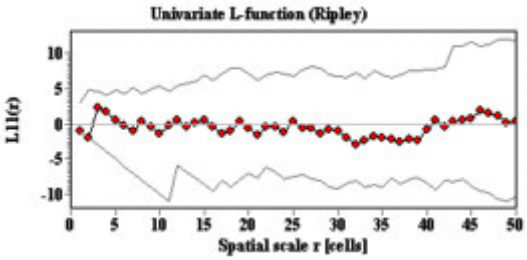
1) Select the data file "C_A1.dat" in window **Input data file.** This artificial example was created with an antecedent cluster process with parameters $\sigma_{12} = 8$ and $\rho_{12} = 0.000625$.
2) select "List" in **How are your data organized** and "Data are given as a list in grid" in **Select modus of data**.
3) click the button "change" in **set maximal radius rmax** and set the

maximal scale r of the analysis to r_{max} = 50.

4) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data:



5) The univariate analysis of pattern 1 (C_A_1uniPat1.res) shows that this pattern is indeed an random pattern:



6) For the univariate analysis of pattern 2 select the data file "C_A1_21.dat" in window **Input data file.** This is the data from "C_A1.dat" but pattern 1 and 2 are exchanged. Fitting a univariate Neyman-Scott cluster yields
   - σ_2 = 7.54 and ρ_2 = 0.00077 (31 parents)

   which are in good agreement with the parameters under which the pattern was created:
   - σ_{12} = 8 and ρ_{12} = 0.000625 (25 parents).

7) For the bivariate analysis highlight the data file "C_A1.dat" in window **Input data file** for the bivariate analysis.

8) select "List" in **How are your data organized** and "Data are given as a list in grid" in **Select modus of data**.

9) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to r_{max} = 50.

10) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data.

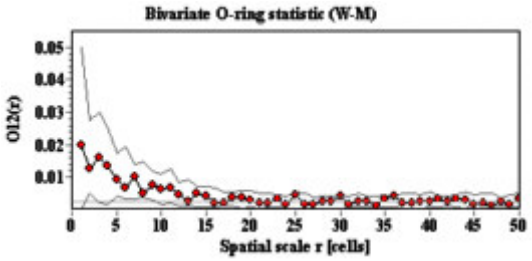
11) To determine Monte Carlo confidence intervals for a bivariate Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "Bivariate linked double-

cluster".

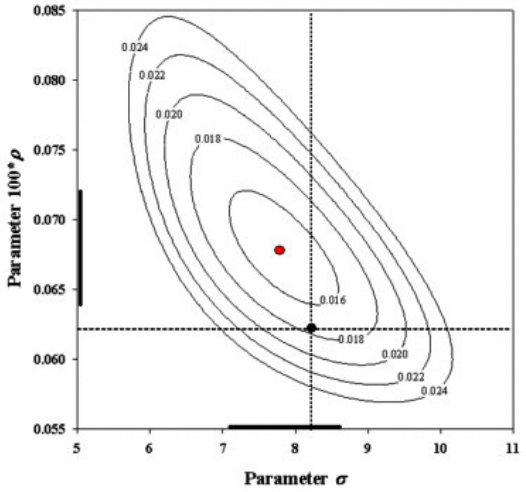
12) The window **Fit of Neyman-Scott models to data** appears and asks you to provide the results of the univariate analysis of pattern 1 and 2. The results of the univariate analysis are needed for performing the fit and for simulation the null model. Since your pattern is a random pattern, insert a high value for σ_1 and a value for ρ_1 that is below the intensity of pattern 1, and provide the results from univariate analysis of pattern 2:
    - σ_1 = 1111 and $100\rho_1$ = 0.1 (40 parents)
    - σ_2 = 7.54 and $100\rho_2$ = 0.077 (31 parents)
13) Click "ok" in the window for inserting the results of the univariate analyses and again "ok" in the window for null models.
14) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. You can specify the tuning constants r_{min}, r_{max}, and c for the fit in the window "Fit cluster process":
15) Select r_{max} = 1 and r_0 = 50. We are only interested in the structure of the pattern at smaller scales. The default power transformations c = 0.5 for the *g*-function and c = 1 for the *L*-function are reasonable starting values. To optimize the *g*- and the *L*-function simultaneously enable "both, L- and g-function".
16) Click the button "fit" and *Programita* searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the *g*- and *L*- function of your data best (red line: fit, black line: data). For the initial parameter intervals *Programita* finds the best fit for
    - σ_{best} = 7.403 and ρ_{best} = 0.0007 (some 28 parents)

    Since pattern 1 comprises only 25 points *Programita* gives you a warning. Click "ok and *Programita* continues calculating the best estimate of σ_{best} for 25 parents under constant $\sigma^2_{best}\rho_{best}$ (see discussion of equation C2) which is σ_{best} = 7.835. Thus, *Programita* finds a best estimate for σ_{12} = 7.8 which is in good agreement with the parameter value under which the pattern was created (σ_{12} = 8).
17) To optimize the parameter fit, press the button "Zoom". *Programita* now determines the probable range of the parameters. *Programita* finds
    - σ_{best} = 7.842 and ρ_{best} = 0.0006764 (some 27 parents).

    Correction of σ for 25 parents yields a best estimate of σ_{12} = 8.158 which is in excellent agreement with the parameter value under which the pattern was created (σ_{12} = 8).
18) Insert the results of the univariate analysis of pattern 2 and confirm in the window for inserting the results of the univariate analyses by clicking "ok" and again "ok" in the window for null models. The window **Fit of Neyman-Scott models to data** disappears.
19) The simulation of 99 replicates of the Neyman-Scot null model shows that the data are well within the confidence envelopes of the null model:



20) We saved the settings and results of the fit of the bivariate Neyman-Scott null model to the data (C_N_1.fit) and use them now to estimate confidence intervals for the estimates of the parameters σ_{best} and ρ_{best} (see section "Constructing confidence intervals for σ and ρ"):



The confidence intervals for an error < 0.016 are shown as bold intervals at the axes. We find $\sigma \in (7.14, 8.55)$ and $100*\rho \in (0.064, 0.072)$. The best estimate for 25 parents (black dots) is within the confidence interval.

4.6.4. Antecedent cluster process (C_A_2.res)

This artificial example is a variant of the previous example (example C_A_1.res) where pattern 1 is the same as in the previous example, but pattern 2 was simulated only with 10 instead of 25 parents which were randomly selected out of the 25 type 1 points. The (bivariate) parameters of pattern 2 were:

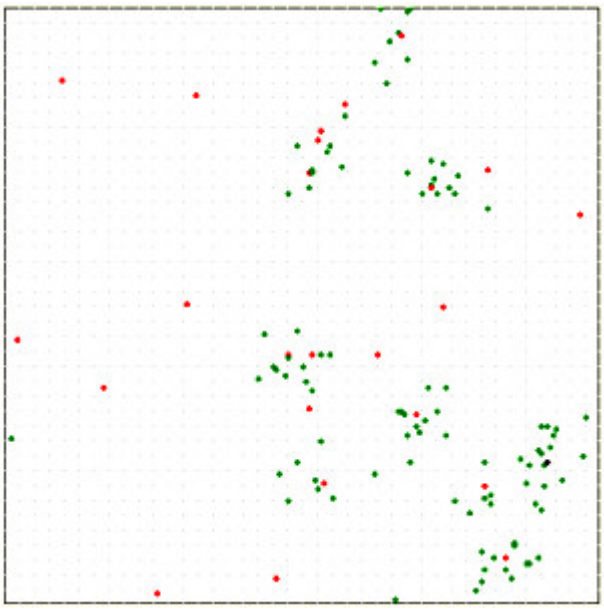
- $\sigma_{12} = 8$, and
- $\rho_{12} = 0.00025$.

In a first step we analyzed the two univariate component patterns and in a second step the bivariate pattern.

1) Select the data file "C_A2.dat" in window **Input data file.** This artificial example was created with an antecedent cluster

process with parameters σ_{12} = 8 and ρ_{12} = 0.00025.

1) select "List" in **How are your data organized** and "Data are given as a list in grid" in **Select modus of data**.
2) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to r_{max} = 50.
3) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data. The visualization of the data shows that type 2 points are clearly clustered around type 1 points, but not around all type 1 points:



4) For the univariate analysis of pattern 2 select the data file "C_A2_21.dat" in window **Input data file.** This is the data from "C_A2.dat" but pattern 1 and 2 are exchanged. The analysis fitting a univariate Neyman-Scott cluster yields
   - σ_2 = 7.3 and ρ_2 = 0.00027 (11 parents)

   which are in good agreement with the know parameters
   - σ_{12} = 8 and ρ_{12} = 0.00025 (10 parents).
5) For the bivariate analysis highlight the data file "C_A2.dat" in window **Input data file** for the bivariate analysis.
6) select "List" in **How are your data organized** and "Data are given as a list in grid" in **Select modus of data**.
7) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to r_{max} = 50.
8) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data.
9) To determine Monte Carlo confidence intervals for a bivariate Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "Bivariate linked double-cluster".
10) The window **Fit of Neyman-Scott models to data** appears and asks you to provide the results of the univariate analysis of pattern 1 and 2. The results of the univariate analysis are needed for performing the fit and for simulation the null model. Since your pattern is a random pattern, insert a high value for σ_1 and a value for ρ_1 that is below the in-

tensity of pattern 1, and provide the results from univariate analysis of pattern 2:

- σ_1 = 1111 and $100\rho_1$ = 0.1 (40 parents)
- σ_2 = 7.3 and $100\rho_2$ = 0.027 (11 parents)

11) Click "ok" in the window for inserting the results of the univariate analyses and again "ok" in the window for null models.

12) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. You can specify the tuning constants r_{min}, r_{max}, and c for the fit in the window "Fit cluster process":

13) Select r_{max} = 1 and r_0 = 50. The default power transformations c = 0.5 for the *g*-function and c = 1 for the *L*-function are reasonable starting values. To optimize the *g*- and the *L*-function simultaneously enable "both, L- and g-function".

14) Click the button "fit" and *Programita* searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the *g*- and *L*- function of your data best (red line: fit, black line: data). For the initial parameter intervals *Programita* finds the best fit for

- σ_{best} = 7.779 and ρ_{best} = 0.00065 (some 26 parents).

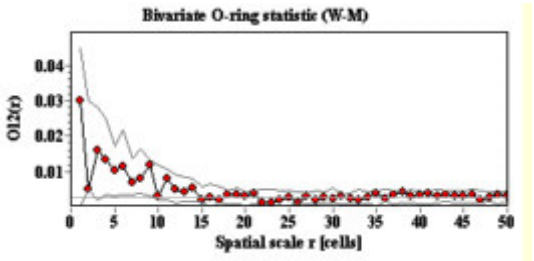
Since pattern 1 comprises only 25 points *Programita* gives you a warning. Click "ok and *Programita* continues calculating the best estimate of σ_{best} for 25 parents under constant $\sigma^2_{best}\rho_{best}$ (see discussion of equation C2) which is σ_{best} = 7.933 which is in excellent agreement with the parameter value under which the pattern was created (σ_{12} = 8).

15) To optimize the parameter fit, press the buttons "Zoom" and "Fit". *Programita* now determines the probable range of the parameters. *Programita* finds

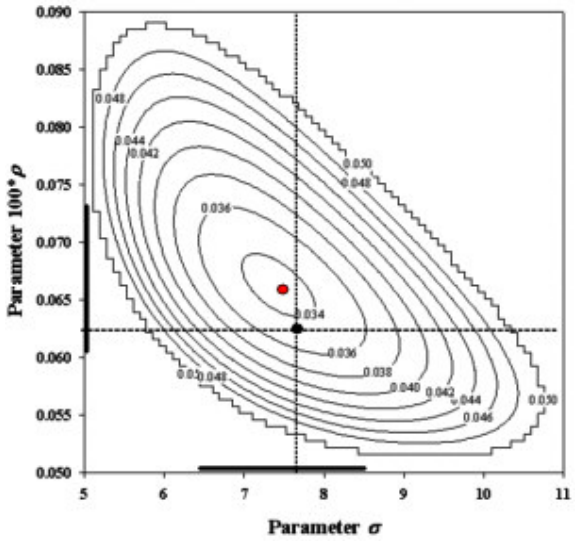
- σ_{best} = 7.436 and ρ_{best} = 0.00066 (some 27 parents).

Correction of σ for 25 parents yields a best estimate of σ_{12} = 7.656 which is in good agreement with the parameter value under which the pattern was created (σ_{12} = 8).

16) The simulation of 99 replicates of the Neyman-Scot null model shows that the data are well within the confidence envelopes of the null model:



17) We saved the settings and results of the fit of the bivariate Neyman-Scott null model to the data (C_N_2.fit) and use them now to estimate confidence intervals for the estimates of the parameters σ_{best} and ρ_{best} (see section "Constructing confidence intervals for σ and ρ"):



The confidence intervals for an error < 0.036 are shown as bold intervals at the axes. We find $\sigma \in (6.44, 8.52)$ and $100*\rho \in (0.060, 0.073)$. The best estimate for 25 parents (black dots) is within the confidence interval.

4.7. Bivariate double-cluster process for antecedent condition

4.7.1. Background

The bivariate cluster process under antecedent condition assumes that points of pattern 1 are parents of pattern 2 and that pattern 1 is a random pattern. In real situations, however, this assumption may be violated and pattern 1 may itself show a clustered univariate structure. In this case we cannot determine the parameter ρ_2 that determines the number of parents of pattern 2 (and which is used for simulation of the null model) with equation C6 because the univariate structure of pattern 2 does not follow a simple Neyman-Scott cluster process, but a double-clustered Neyman-Scott process. We thus generalize equation C5 and equation C6 for a double-clustered process where pattern 1 follows a Neyman-Scott cluster process (equation C1), and some (or all) type 1 points are cluster centers for type 2 points:

$$g_{12}(r,\sigma_1,\rho_1,\rho_{12},\sigma_{12}) = 1+\frac{1}{\rho_{12}}\frac{\exp(-r^2/2\sigma_{12}^2)}{2\pi\sigma_{12}^2}+\frac{1}{\rho_1}\frac{\exp(-r^2/2\sigma_{sum}^2)}{2\pi\sigma_{sum}^2} \quad \text{(C7)}$$

$$\text{with} \quad \sigma_{sum}^2 = 2\sigma_1^2+\sigma_{12}^2$$

$$K_{12}(r,\sigma_1,\rho_1,\rho_{12},\sigma_{12}) = \pi r^2+\frac{1-\exp(-r^2/2\sigma_{12}^2)}{\rho_{12}}+\frac{1-\exp(-r^2/2\sigma_{sum}^2)}{\rho_1} \quad \text{(C8)}$$

$$\text{with} \quad \sigma_{sum}^2 = 2\sigma_1^2+\sigma_{12}^2$$

with the four parameters:

- σ^2_1, the variance of the Gaussian distribution that determines the locations of type 1 points relative to their parents. The parameter σ^2_1 needs to be determined previously through an univariate analysis of pattern 1
- ρ_1, the intensity of parents of pattern 1 and needs to be determined previously through an univariate analysis of pattern 1
- ρ_{12}, a fitted parameter that theoretically yields $\rho_{12} = \lambda_1$,
- σ^2_{12}, a fitted parameter that gives the variance of the Gaussian distribution that determines the locations of type 2 points relative to their (type 1) parents. The value of this parameter should coincide with the parameter σ^2_2 determined through univariate analysis of pattern 2

Note that equation C7 and equation C8 are the analogues to equation DC2 and

equation DC3, respectively, which describe a univariate double-cluster process.

The first term of equation C7 (= 1) describes the situation where the two patterns are independent (i.e., type 2 points are not clustered around type 1 parents), the second term describes the additional effect of clustering of type 2 points around type 1 parents with parameters σ^2_{12} and ρ_{12}, and the third term describes the compound effect of clustering of the parents (= pattern 1) and the offspring (= pattern 2) around the parents (= pattern 1). The variance σ_{sum}^2 is the combined variance that describes the interaction of clumping at the two scales σ^2_{12} and σ^2_1.

If pattern 1 is a random pattern, the third term disappears and equation C7 collapses back to equation C5. If type 2 points are independent from type 1 points (i.e., $\sigma^2_{12} \to \infty$) it follows that $\sigma^2_{sum} \to \infty$ and equation C7 and equation C8 collapse, as expected, back to a CSR process with $g(r) = 1$.

A realization of the process described by equation C7 and equation C8 results in double clustering of pattern 2.

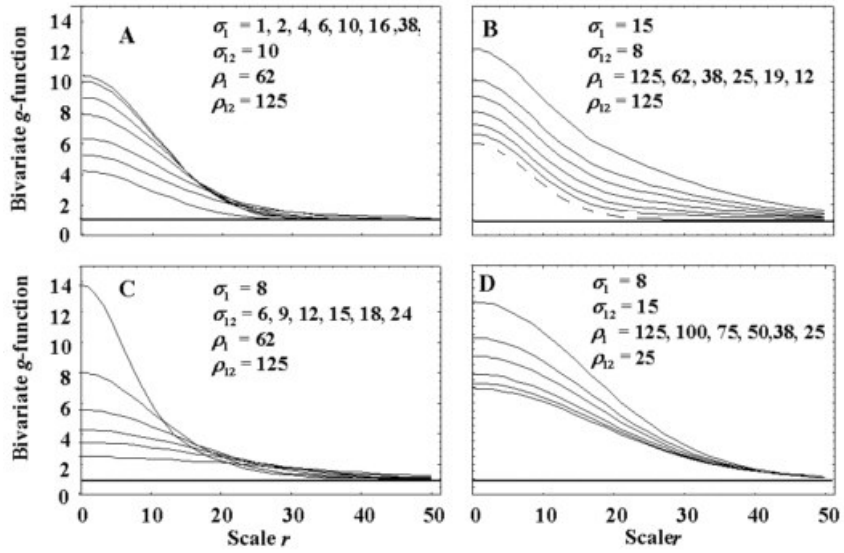


Figure C6. Effect of variation in the parameters σ_1^2, ρ_1, σ_{12}^2, and ρ_{12} on the shape of the bivariate g-function under antecedent condition and double clustering (equation C7). (A) Influence of variance σ_1^2 of the locations of type 1 points relative to their parents. Curves from top to bottom: σ_1 = 1, 2, 4, 6, 10, 16, 38. (B) Influence of the intensity ρ_1 of the parents of type 1 points. Curves from top to bottom: ρ_1 = 12, 19, 25, 38, 62, 125. The dashed line is the contribution of the first two terms only. (C) Influence of variance σ_{12}^2 of the locations of type 2 points relative to type 1 points. Curves from top to bottom: σ_1 = 6, 9, 12, 15, 18, 24. (D) Influence of the intensity ρ_{12} of the parents of type 1 points. Curves from top to bottom: ρ_{12} =25, 38, 50, 75, 100, 125.

Performance of an analysis using a double-clustered Neyman-Scott process under antecedent condition requires three steps:

1. univariate analysis of patterns 1
2. analysis of the univariate pattern of type 2 points within the framework of bivariate double-cluster processes
3. the final bivariate analysis

The parameters σ_1 and ρ_1 of the univariate analysis of pattern 1 are needed to fit equations C7 and equation C8 to the data (first step). The parameter ρ_2 of the univariate analysis of pattern 2 (the intensity of type 1 points which are parents of type 2 points) is not needed for the fit of the bivariate process equation C7, but for simulation of the process. If pattern 1 is clustered (i.e., equation C7), the parameter ρ_2 needs to be estimated by fitting the univariate *g*- and *K*-function to the expected *g*- and *K*-function of a univariate double-clustered Neyman-Scott process equation DC2 and equation DC3 (second step):

$$g_{22}(r,\sigma_1,\rho_1,\sigma_2,\rho_2) = 1+\frac{1}{\rho_2}\frac{\exp(-r^2/4\sigma_2^2)}{4\pi\sigma_2^2}+\frac{1}{\rho_1}\frac{\exp(-r^2/4\sigma_{sum}^2)}{4\pi\sigma_{sum}^2} \qquad \text{(C9)}$$

with $\sigma_{sum}^2 = \sigma_1^2+\sigma_2^2$

$$K_{22}(r,\sigma_1,\rho_1,\rho_2,\sigma_2) = \pi r^2+\frac{1-\exp(-r^2/4\sigma_2^2)}{\rho_2}+\frac{1-\exp(-r^2/4\sigma_{sum}^2)}{\rho_1} \qquad \text{(C10)}$$

with $\sigma_{sum}^2 = \sigma_1^2+\sigma_2^2$

with the four parameters:

- σ^2_2, the parameter that gives the variance of the locations of type 2 points relative to their parents (= type 1 points).
- ρ_2, the intensity of the parents of the type 2 points.
- σ^2_1, the parameter that gives the variance of the locations of type 1 points relative to their parents.
- ρ_1 is the intensity of the parents of type 1 points.

Comparison of the resulting parameters $\sigma_{12}{}^2$ and ρ_{12} of bivariate double-cluster process under antecedent condition with the parameters of the two univariate component processes ($\sigma_1{}^2$, ρ_1, $\sigma_2{}^2$, and ρ_2) reveal how much type 1 points served actually as parent for type 2 points (the parameter ρ_2, this information is needed for simulation of the null model) and whether there are some inconsistencies which may indicate that this null model does not well describe the data.

If all type 1 points are parents we expect $\rho_{12} = \lambda_1 = \rho_2$ and $\rho_1 < \rho_2$. Thus, the fitted intensity of type 1 points should coincide with the known intensity of type 1 points and there should be less parents of pattern 1 than parents of pattern 2. If only a certain proportion of type 1 points are parents of type 2 points we expect $\rho_{12} = \lambda_1$, $\rho_1 < \lambda_1$and $\rho_2 < \lambda_1$, thus the number of parents of type 1 points cannot exceed the number of type 1 points and there should be less parents of pattern 1 than type 1 points. Additionally, we expect $\sigma^2_{12} = \sigma_2^2$, thus the parameter σ^2_{12} (that determines locations of type 1 points relative to their parents) which is fitted with the bivariate model (third step) should be the same as the corresponding parameter σ^2_2 fitted with the univariate model (second step).

If $\rho_2 < \lambda_1$ *Programita* assigns in every simulation of the null model a different, randomly chosen, set of parents among all type 1 points for simulation of the locations of type 2 points.

4.7.2. Bivariate double-cluster process (C_A_3.res)

This artificial example uses the pattern of the dead trees (example NS_3b.res) as parents and creates offspring (= pattern 2) with parameters:

- $\sigma_2 = 10$ and $100\rho_2 = 0.0542$ (all 136 type 1 points are parents)

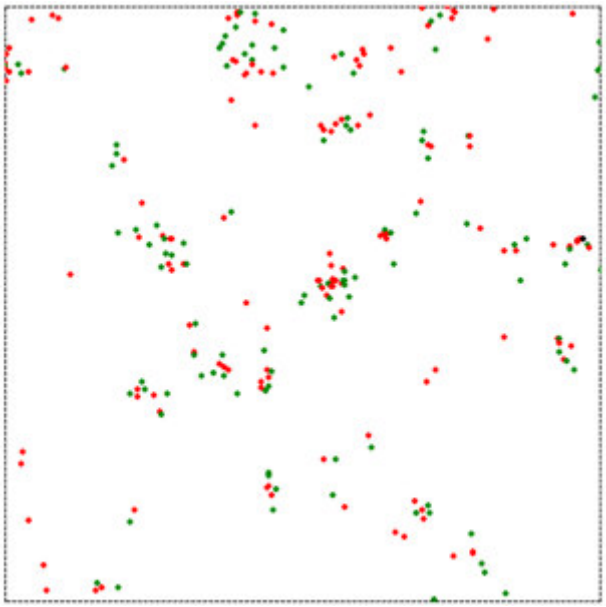
We proceed in the three steps: (1) univariate analysis of patterns 1, (2) analysis of the univariate pattern of type 2 points within the framework of bivariate double-cluster processes, and (3) the final bivariate analysis.

First step: univariate analysis of pattern 1 using a simple cluster model:

see example NS_3b.res. We found $\sigma_1 = 6.92$ and $100\rho_1 = 0.02$. This corresponds to a cluster size of some 14m, and some 50 parent events.

Second step: univariate analysis of pattern 1 using a double-cluster model (C_A_3a.res):

1) Select the data file "C_A3.dat" in window **Input data file.** This artificial example was created with an antecedent double-cluster process.
2) select "List" in **How are your data organized** and "Data are given as a list in grid" in **Select modus of data**.
3) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to $r_{max} = 50$. Select a ring width of $dr = 3$.
4) click button "Calculate index", *Programita* shows you the pattern and calculates the O-ring function of the data. The visualization of the data shows that type 2 points are clearly clustered around type 1 points:



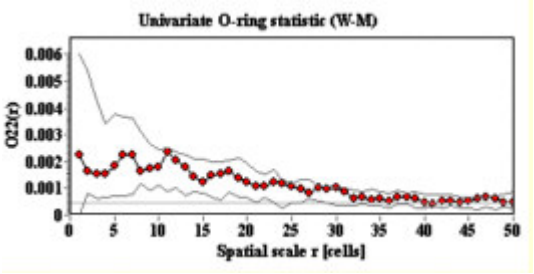
5) For the univariate analysis of pattern 2 enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears; enable "Univariate double-cluster".
6) The window "Univariate double-cluster Neyman-Scott" appears and asks you to provide the results of the univariate analysis of pattern 1. The results of the univariate analysis are needed for performing the fit and for simulation the null model. Provide the results from univariate analysis of pattern 1:
   - σ_1 = 6.92 and $100\rho_1$ = 0.02 (50 parents)

7) Click "ok" in the window "Univariate double-cluster Neyman-Scott" and again "ok" in the window "Null models".
8) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. Select r_{max} = 1 and r_0 = 80 an click the button "fit".
9) *Programita* searches the parameters of equations 9 and 10 that simultaneously fits the *g*- and *L*- function of your data best (red line: fit, black line: data). For the initial parameter intervals *Programita* finds the best fit for
   - σ_{2best} = 10.053 and $100\rho_{2best}$ = 0.0544 (136 parents) and $\sigma^2{}_2\rho_2$ = 0.055
10) Note that *Programita* defines the default value of ρ_{max} in a way that it yields the number of type 2 points (= 136). To increase the range of the parameter ρ set ρ_{max} = 0.1 and click "fit". *Programita* now performs a new fit and finds
   - σ_{2best} = 10.053 and $100\rho_{2best}$ = 0.06182 (164 parents) and $\sigma^2{}_2\rho_2$ = 0.06248

Because the estimate for ρ was larger than possible, *Programita* calculates the best estimate of σ_{2best} for 136 parents under constant $\sigma^2{}_{2best}\rho_{2best}$ (see discussion of equation C2) which is σ_{2best} = 10.717. This parameter estimates coincide well with the parameters under which the process was

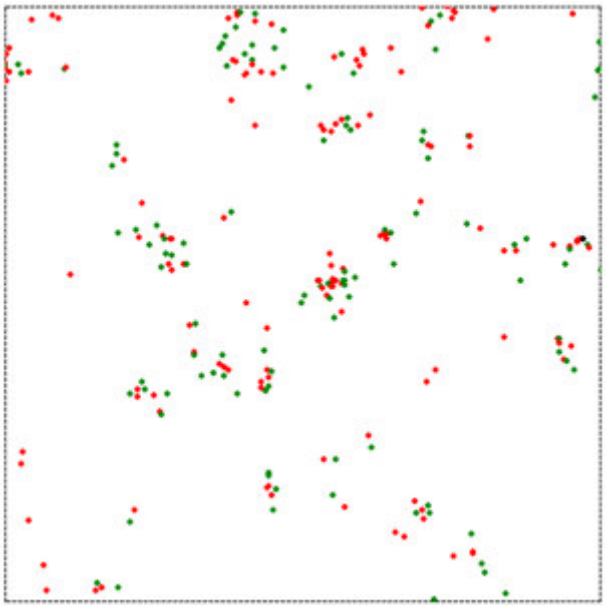
created.

11) To optimize the parameter fit, press the buttons "Zoom" and "Fit". *Programita* now determines the probable range of the parameters in the parameter space and performs a new fit. *Programita* finds
   - σ_{2best} = 10.5 and $100\rho_{2best}$ = 0.05772 (144 parents) and $\sigma^2_2\rho_2$ = 0.06372
12) Because the estimated number of parents is higher than the number of type 1 points, *Programita* gives you a warning. Click "ok and *Programita* continues calculating the best estimate of σ_{2best} for 136 parents under constant $\sigma^2_{2best}\rho_{2best}$ (see discussion of equation C2) which is σ_{2best} = 10.822.
13) To proceed with simulation of the null model, click the button ok in the window **Fit of Neyman-Scott models to data** and then "Calculate index".
14) The simulation of 19 replicates of the Neyman-Scot null model shows that the data are well within the confidence envelopes of the null model:



Third step: bivariate analysis of using a double-cluster model (C_A_3b.res).

1) Select the data file "C_A3.dat" in window **Input data file.** This artificial example was created with an antecedent double-cluster process.
2) select "List" in **How are your data organized** and "Data are given as a list in grid" in **Select modus of data**.
3) click the button "change" in **set maximal radius rmax** and set the maximal scale *r* of the analysis to r_{max} = 50. Select a ring width of *dr* = 3.
4) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data. The visualization of the data shows that type 2 points are clearly clustered around type 1 points:



5) For the bivariate analysis of pattern 2 enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears; enable "Bivariate double-cluster".
6) The window "Linked double-cluster Neyman-Scott" appears and asks you to provide the results of the univariate analysis of patterns 1 and 2. Provide the results from univariate analyses:
   - σ_1 = 6.92 and $100\rho_1$ = 0.02 (50 parents)
   - σ_2 = 10.822 and $100\rho_2$ = 0.0544 (136 parents)

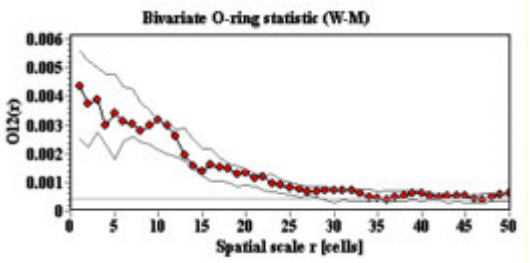
7) Click "ok" in the window "Univariate double-cluster Neyman-Scott" and again "ok" in the window "Null models".
8) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. You can specify the tuning constants r_{min}, r_{max}, and c for the fit in the window "Fit cluster process":
9) Select r_{max} = 1 and r_0 = 50. and click the button "fit" and *Programita* searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the *g*- and *L*-function of your data best (red line: fit, black line: data). For the initial parameter intervals *Programita* finds the best fit for
   - σ_{12best} = 10.127 and $100\rho_{12best}$ = 0.0546 (125 parents) and $\sigma^2_{12}\rho_{12}$ = 0.05156

   This parameter estimates coincide well with the parameters under which the process was created.
10) To optimize the parameter fit, press the buttons "Zoom" and "Fit". *Programita* now determines the probable range of the parameters in the parameter space and performs a new fit. *Programita* finds
   - σ_{12best} = 10.329 and $100\rho_{2best}$ = 0.0496 (123 parents) and $\sigma^2_2\rho_2$ = 0.05292

   which coincides very well with the parameters under which the process was created.
11) To proceed with simulation of the null model, click the button ok in the window **Fit of Neyman-Scott models to data** and then

"Calculate index".

12) The simulation of 19 replicates of the Neyman-Scot null model shows that the data are well within the confidence envelopes of the null model:



13) To show the need to use a double-clustered process for fitting the parameter ρ_2 of the univariate analysis of pattern 2 we repeat the analysis, but under the assumption of a random structure of pattern 1 (i.e., σ_{12best} =1111). Now *Programita* finds best fits of

- σ_{12best} = 12.6 and $100\rho_{12best}$ = 0.015 (37 parents) and $\sigma^2_{12}\rho_{12}$ = 0.0233

which do not coincide at all with the original parameters under which the process was created (σ_2 = 10 and $100\rho_2$ = 0.0542 [all 136 type 1 points are parents].

4.7.3. Analysis dead trees and recruits (C_A_4.res)

This example uses the univariate double-cluster process equation C9 and equation C10 to analyze the bivariate pattern of dead trees and recruits. As in the previous example, we proceed in three steps: (1) univariate analysis of patterns 1, (2) analysis of the univariate pattern of type 2 points within the framework of bivariate double-cluster processes, and (3) the final bivariate analysis.

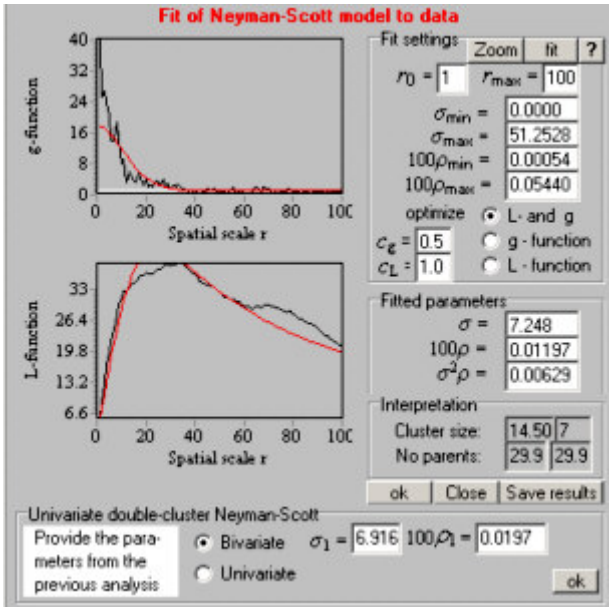
First step: univariate analysis of pattern 1 using a simple cluster model:

see example NS 3b.res. We found σ_{best} = 14.1 and ρ_{best} = 0.000083. This corresponds to a cluster size of some 29m, and some 21 parent events.

Second step: univariate analysis of pattern 1 using a double-cluster model (C_A_4a.res):

1) Highlight the data file "Dead_recruits.dat" in window **Input data file.** This data set gives the location of dead trees and recruits at a meter scale, but has a resolution of 1 centimeter.
2) select "List" in **How are your data organized**

3) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
4) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to r_{max} = 50.
5) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data.
6) For univariate analysis of pattern 2 assuming a double-clustered process enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "Univariate double cluster".
7) The windows **Fit of Neyman-Scott models to data** and "Univariate double-cluster Neyman-Scott" appears. Select the option "bivariate" in this window and provide the results of the univariate analysis of pattern 1:
   - σ_1 = 6.916 and $100\rho_1$ = 0.0189 (some 49 parents).
8) Click "ok" in the window "Univariate double-cluster Neyman-Scott" and again "ok" in the window "Null models". *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. Select r_{max} = 1 and r_0 = 100 and click the button "fit".
9) *Programita* now searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the *g*- and *L*-function of your data best (red line: fit, black line: data). As expected, the best fit is not very good:

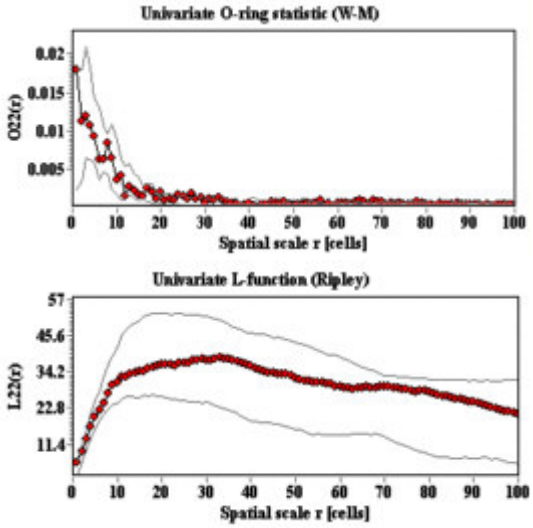


10) *Programita* finds for the initial parameter intervals the best fit
    - σ_{2best} = 7.248 and $100\rho_{2best}$ = 0.01197 (some 30 parents)
11) Because we adjust here parameters of the smaller-scale clustering we repeat the fit for a smaller maximum scale r_0 = 35. For these scales, *Programita* finds for the initial parameter intervals the best fit
    - σ_{2best} = 5.177 and $100\rho_{2best}$ = 0.01741 (some 43 parents)
12) To optimize the parameter fit, press the buttons "Zoom"

and "Fit". *Programita* finds as best estimates

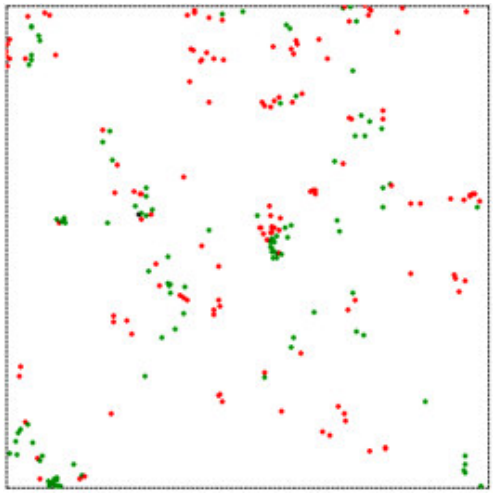
- σ_{2best} = 5.246 and $100\rho_{2best}$ = 0.01702 (some 43 parents).

13) For simulation of the null model click "ok" at the window **Fit of Neyman-Scott models to data** and click "Calculate index". *Programita* simulates the confidence intervals of the univariate analysis of pattern 2 using the best parameter estimates for σ_2 and ρ_2 and 43 type 1 points as parents. The simulation of 19 replicates of the Neyman-Scot null model shows that the data are well within the confidence envelopes of the null model (although the confidence envelopes are not symmetric around the data):



Third step: bivariate analysis of using a double-cluster model (C_A_4b.res).

1) Highlight the data file "Dead_recruits.dat" in window **Input data file.** This data set gives the location of dead trees and recruits at a meter scale, but has a resolution of 1 centimeter.
2) select "List" in **How are your data organized**
3) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
4) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to r_{max} = 50.
5) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data:

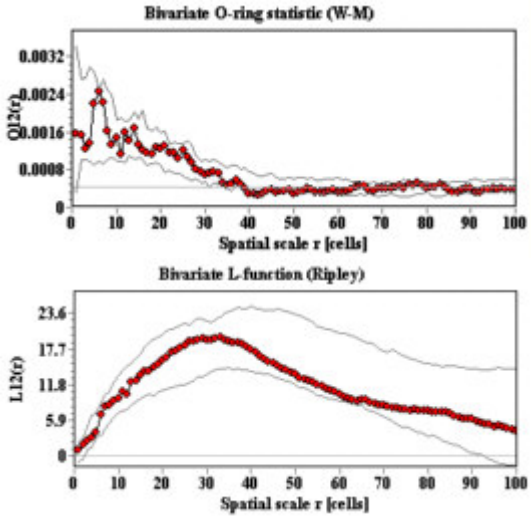


6) For bivariate analysis assuming a double-clustered process enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "Bivariate linked double cluster".
7) The windows **Fit of Neyman-Scott models to data** and "Univariate double-cluster Neyman-Scott" appear and ask you to provide the results of the univariate analysis of pattern 1 and pattern 2. This data are important for performing the fit and simulating the null model. Insert
   - σ_1 = 6.916 and $100\rho_1$ = 0.0197 (some 49 parents).
   - σ_2 = 5.246 and $100\rho_2$ = 0.01702 (some 43 parents)

   Click "ok" in the window "Univariate double-cluster Neyman-Scott" and again "ok" in the window "Null models".
8) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. Select r_{max} = 1 and r_0 = 35 and click the button "fit".
9) *Programita* now searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the *g*- and *L*-function of your data best (red line: fit, black line: data).
10) *Programita* finds for the initial parameter intervals the best fit
    - σ_{2best} = 17.209 and $100\rho_{2best}$ = 0.06 (some 150 parents).
11) Because the estimated number of parents (=150) is higher than the number of type 1 points (136), *Programita* gives you a warning. Click "ok and *Programita* continues calculating the best estimate of σ_{12best} for 136 parents under constant $\sigma^2_{12best}\rho_{12best}$ (see discussion of equation C2) which is σ_{12best} = 18.078.
12) To optimize the parameter fit, press the buttons "Zoom" and "Fit". *Programita* finds as best estimates
    - σ_{12best} = 17.192 and $100\rho_{12best}$ = 0.06023 (some 150 parents).
13) Because the estimated number of parents (=150) is slightly higher than the total number of type 1 points (remember that the theoretical expectation is $\rho_{12} = \lambda_1$), *Programita* gives you a warning. Click "ok and *Programita* continues calculating the best estimate of σ_{12best} for 136 parents under constant $\sigma^2_{12best}\rho_{12best}$ (see discussion of equation C2) which is σ_{12best} = 18.089.
14) Note that the estimate of σ_{12best} (= 18.089) is much higher

than the estimate of σ_{12best} (= 5.246) from the univariate analysis of pattern 2. This result indicates that there may be something wrong with assumption that the locations of the dead trees function as parents for the recruits.

15) For simulation of the null model select a ring width of *dr* = 3 and a maximal scale rmax= 100 and click "ok" at the window **Fit of Neyman-Scott models to data** and click "Calculate index". *Programita* simulates the confidence intervals of the bivariate analysis using the best parameter estimates for σ_{12} and ρ_{12} and 43 type 1 points as parents. The simulation of 19 replicates of the Neyman-Scot null model shows that the data are well within the confidence envelopes of the null model:



We obtain the same result when using the *L*-function Thus, the simulation of the double-clustered bivariate cluster null model does not reject the hypothesis that the locations of the recruits are clustered around the locations of the dead trees.

19) Univariate analysis of the recruits using a double-cluster model (example DC 3.res) yielded the unbiased parameters of the large scale clumping:

- σ_1 = 14.4 and $100\rho_1$ = 0.0095 (some 24 parents).

and the estimated parameters for the small-scale clustering are:

- σ_2 = 3.83 and $100\rho_2$ = 0.02165 (some 54 parents)

The results of the analysis under the antecedent condition that the dead trees are the parents are:

- first step: σ_1 = 6.916 and $100\rho_1$ = 0.0197 (some 49 parents).
- second step: σ_2 = 5.246 and $100\rho_2$ = 0.0170 (some 43 parents)
- third step: σ_{12} = 18.089 and $100\rho_{12}$ = 0.0544 (some 136 parents).

Thus, the clustering of the dead trees does not accord with the large-scale clustering of the recruits determined through univariate analysis using a double-cluster model (example DC 3.res) and consequently the estimate for σ_2

and σ_{12} are biased. This example shows that testing the null hypothesis of antecedent double-clustering requires extensive analyses of the univariate patterns and checking for internal consistency.

4.7.4. Adult trees and recruits (C_A_5a.res)

This example continues the analysis of example NS_4.res where we applied a bivariate Neyman-Scott cluster model to the data. We found that data may be described by a model that assumes random cluster centers (which may correspond to patches with favorable conditions for the tree species) and clustered distribution of recruits and adult trees around the cluster centers. Here we assume the alternative null model that the locations of the recruits are directly linked to the adult trees which correspond to an effect of a limited seed dispersal radius.

We proceed in three steps: (1) univariate analysis of patterns 1, (2) analysis of the univariate pattern of type 2 points within the framework of bivariate double-cluster processes, and (3) the final bivariate analysis.

First step: univariate analysis of pattern 1 using a simple cluster model:

See example NS_3.res, we found

- σ_1 = 14.1 and $100\rho_1$ = 0.0083 (some 21 parent events).

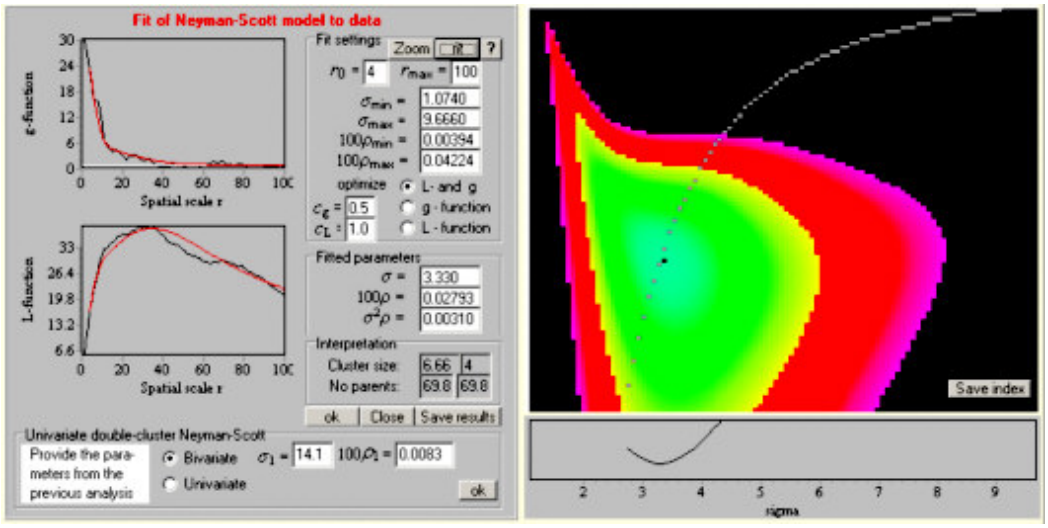
Second step: univariate analysis of pattern 1 using a double-cluster model (C_A_5a.res):

1) Highlight the data file "A_1.dat" in window **Input data file.** This data set gives the location of dead trees and recruits at a meter scale, but has a resolution of 1 centimeter.
2) select "List" in **How are your data organized**
3) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
4) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to r_{max} = 50.
5) click button "Calculate index", *Programita* shows you the pattern and calculates the O-ring function of the data.
6) For univariate analysis of pattern 2 assuming a double-clustered process enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "Univariate double cluster".
7) The windows **Fit of Neyman-Scott models to data** and "Univariate double-cluster Neyman-Scott" appears. Select the option "bivariate" in this window and provide the results of the univariate analysis of pattern 1:

- σ_1 = 14.1 and $100\rho_1$ = 0.0083 (some 21 parent events).

8) Click "ok" in the window "Univariate double-cluster Neyman-Scott" and again "ok" in the window "Null models". *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears. Select r_{max} = 1 and r_0 = 100 and click the button "fit", "Zoom" and "fit".

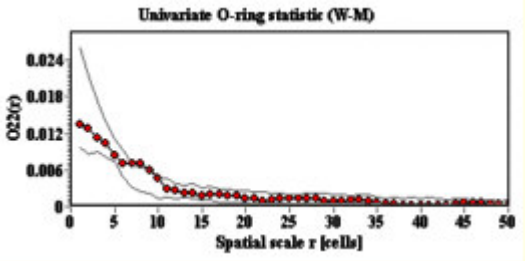
9) *Programita* now searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the *g*- and *L*-function of your data best (red line: fit, black line: data):



Programita finds the best fit

- σ_1 = 3.3 and $100\rho_1$ = 0.02793 (some 70 parent events).

10) For simulation of the null model click "ok" at the window **Fit of Neyman-Scott models to data** and click "Calculate index". *Programita* simulates the confidence intervals of the univariate analysis of pattern 2 using the best parameter estimates for σ_2 and ρ_2 and 70 type 1 points as parents. The simulation of 19 replicates of the Neyman-Scot null model shows that the data are well within the confidence envelopes of the null model:



Third step: bivariate analysis of using a double-cluster model (C_A_5b.res).

1) Highlight the data file "A_1.dat" in window **Input data file.** This data set gives the location of dead trees and recruits at a meter scale, but has a resolution of 1 centimeter.

2) select "List" in **How are your data organized**
3) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "1.00".
4) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to $r_{max} = 50$.
5) click button "Calculate index", *Programita* shows you the pattern and calculates the O-ring function of the data.
6) To determine Monte Carlo confidence intervals for a bivariate Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "Pattern 2 cluster process with parents = pattern 1".
7) The window **Fit of Neyman-Scott models to data** appears and asks you to provide the results of the univariate analysis of pattern 1. This data are important for performing the fit. Insert
   - $\sigma_1 = 14.1$ and $100\rho_1 = 0.0083$
   - $\sigma_2 = 3.3$ and $100\rho_2 = 0.02793$
8) Click "ok" in the window for inserting the results of the univariate analyses and again "ok" in the window for null models.
9) *Programita* calculates the g- and the L-function for $r = 1$ to r_{max} and the window **Fit of Neyman-Scott models to data** appears. Select $r_{max} = 1$ and $r_0 = 35$ and click the button "fit".
10) *Programita* now searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the g- and L-function of your data best (red line: fit, black line: data).
11) *Programita* does not find an appropriate fit for this data since the fitted number of parents (some 224) does always far exceed the number of adult trees (= 88). Playing around with different tuning constants does not change this result.
12) We conclude that we have to reject the hypothesis that the recruits are directly linked to adult trees.

4.8. Bivariate hard- and soft-core processes

The bivariate soft-core null model is a straight forward generalization of the univariate null model and follows basically equation HC4:

$$p_{HC}(d) = \begin{cases} d^{1/p} & \text{for } d \le \delta \\ 1 & \text{for } d > \delta \end{cases} \qquad \text{(HC5)}$$

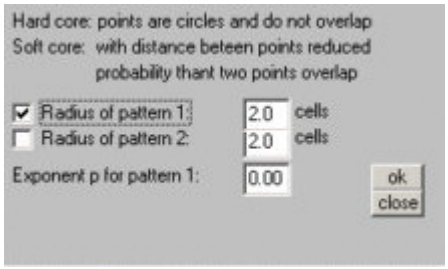
where d is the distance between the provisional point an its nearest accepted neighbour and p is an exponent that describes the "softness" of the process between the two points. If $p \to 0$, equation HC5 yields a hard-core, and for large values of p (e.g., $p = 11$) equation HC5 yields a very soft core with $p_{HC}(d) \approx 1$ for $d < \delta$.

Programita distributes in a first step all points of pattern 2 and in a second step all points of pattern 2. Therefore we have to consider, in contrast to the univariate case, three different situations:

1. both points are type 1 points
2. the provisional point is of type 2 and the nearest neighbour is of type 1
3. both points are type 2 points

Type 1 points are described by a disk with radius $\delta_1/2$, and type 2 points by a disk with radius $\delta_2/2$. The softness of the relation between points is described by three different exponents, describing the softness (1) between type 1 and type 1 points (p_1), (2) between type 2 and type 1 points (p_{12}), and (3) between type 2 points (p_2). The parameters for the three cases of the bivariate soft-core process that follows equation HC5 are:

1. $p = p_1$ and $\delta = 2\delta_1$
2. $p = p_2$ and $\delta = 2\delta_2$
3. $p = p_{12}$ and $\delta = \delta_1 + \delta_2$



4.8.1. Implementation of the hard-core null model

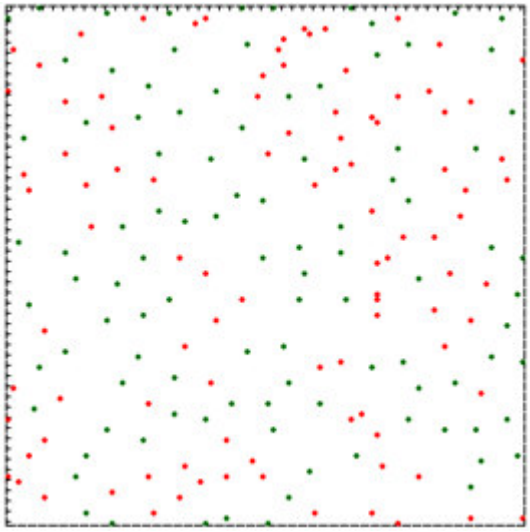
Programita allows you to specify whether pattern 1 or/and pattern 2 has a soft-core:

- **Only pattern 1 has a soft-core**:
  You need to provide only the parameters p_1 and δ_1 for pattern 1. Pattern 1 follows a soft-core null model, but the acceptance of a provisional type 2 point does not depend on the distance d to its nearest accepted neighbor. Thus, the soft-core null model does not incorporate an interaction between the two patterns.
- **Only pattern 2 has a soft-core, but no interaction between patterns**:
  In this case you need to provide the parameters p_1, p_{12}, and δ_1 for pattern 2. If the exponent p_{12} that describes the repulsion of type 2 points by type 1 points is large (e.g., $p_{12} = 11$), acceptance of a provisional type 2 points does not depend on the distance d to its nearest accepted type 1 neighbor. Thus, the soft-core null model does not incorporate an interaction between the two patterns.
- **Only pattern 2 has a soft-core, and type 1 - type 2 interaction occurs**:
  In this case you need to provide the parameters p_1, p_{12}, and δ_1 for pattern 2. If the exponent p_{12} that describes the repulsion of type 2 points by type 1 points is small (e.g., $p_{12} < 0.5$), acceptance of a provisional type 2 points does depend on the distance d to its nearest accepted type 1 neighbor. Thus, this soft-core null model incorporates an explicit interaction mechanism between the two patterns.
- **Both patterns have a soft core, but no interaction between patterns:**
  In this case you need to provide all parameters p_1, p_2, p_{12}, δ_1, and δ_2 for patterns 1 and 2, but the exponent p_{12} that describes the repulsion of type 2 points by type 1 points is large (e.g., $p_{12} = 11$). In this case both patterns show separately repulsion, but no interaction between the two patterns is incorporated.
- **Both patterns have a soft core, and type 1 - type 2 interaction occurs**:
  In this case you need to provide all parameters p_1, p_2, p_{12}, δ_1, and δ_2 for patterns 1 and 2, and the exponent p_{12} that describes the repulsion of type 2 points by type 1 points is small (e.g., $p_{12} < 0.5$). In this case pattern 1 shows repulsion (it was created independently on pattern 2). Pattern 2 is repulsed by points of pattern 1 and points of pattern 2 and may show an univariate structure with smaller scale repulsion (repulsion to type 1 points) and larger scale aggregation (because type 2 points have to be squeezed into the gaps of the existing pattern 1).

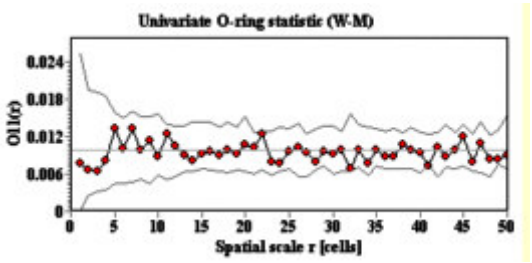
4.8.2. Example for soft core with interaction (HC_5.res)

This artificial data set was created with a soft core process with parameters p_1 =11, δ_1 = 2 cells, p_2 = 0.1, δ_2 = 3 cells, and p_{12} = 0. Thus, pattern 1 is basically a random pattern, type 2 points perceive type 1 points as having a hard core (there is always a minimal distance of $\delta_{12} = \delta_1 + \delta_2 = 5$ between type 2 and type 1 points), and type 2 points have a soft-core in relation to other type 2 points. We will first explore the univariate structures of two component patterns before performing the bivariate analysis.

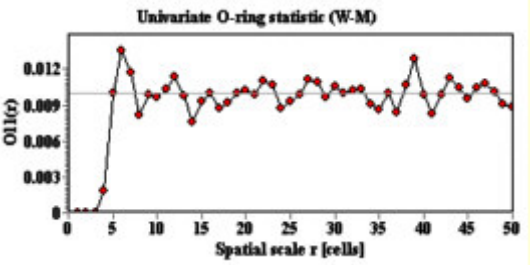
1) For univariate analysis of pattern 1 highlight the data file "HC5.dat" in window **Input data file.** The data file was created with a bivariate hardcore null model with parameters p_1 =11, δ_1 = 2 cells, p_2 = 0.1, δ_2 = 3 cells, and p_{12} = 0.
2) select "List" in **How are your data organized**
3) select "Analyze all data in rectangle" **in Give modus of analysis**
4) select "Data are given as list in grid" in **Select modus of data**
5) click button "Calculate index" and *Programita* shows you the pattern:



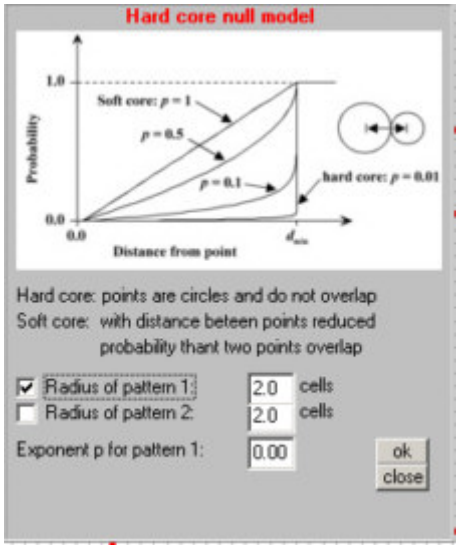
6) Enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears. Select "Pattern 1 and 2 random", 99 simulation of the null model and click the button "Calculate index". *Programita* now performs the univariate analysis for pattern 1 using a CSR null model. Pattern 1 is indeed a random pattern (HC_5_unipat1.res):



7) For univariate analysis of pattern 2 highlight the data file "HC5_21.dat" in window **Input data file**.
8) select "List" in **How are your data organized**
9) select "Analyze all data in rectangle" **in Give modus of analysis**
10) select "Data are given as list in grid" in **Select modus of data**
11) click button "Calculate index" and *Programita* shows you the pattern and the univariate analysis without confidence envelopes:

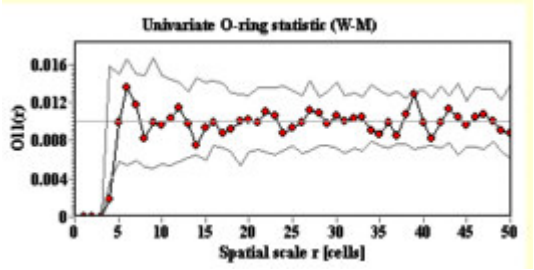


12) The univariate O-ring statistic suggests that pattern 2 has a hard core with δ_2 = 4 cells. To test this hypothesis Select "Pattern 1 and 2 random", 99 simulation, and enable the checkbox "Hard core". The window **Hard core null model opens**:

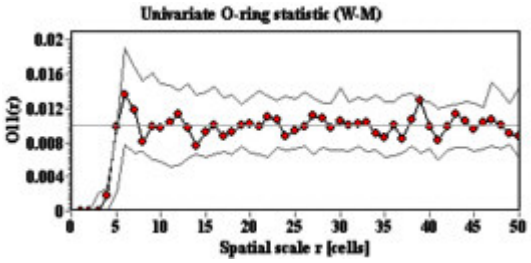


13) Enable the check box "Radius of pattern 1" (univariate analysis) and provide the hardcore radius of pattern 1. The minimal distance between two points is the double of the hardcore radius. Next provide the hardcore radius and the exponent. The example is for a hard-core null model,

therefore select the exponent p = 0, click "ok", and click the button "Calculate index" "Radius of pattern 1" (univariate analysis) and provide the hardcore radius of pattern 1. The minimal distance between two points is the double of the hardcore radius, thus insert a radius of 2 cells. Because the hypothesis is a hard core, select the exponent p = 0, click "ok", and click the button "Calculate index". *Programita* now performs the simulations of the univariate hard-core null model (HC_5_unipat2a.res):

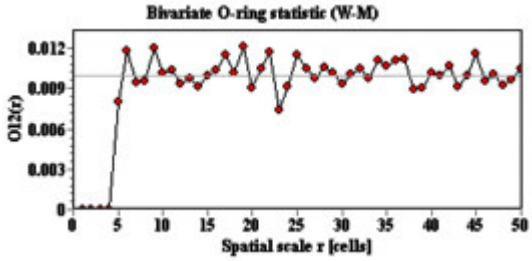


14) The null model does not well describe the data at scale r = 4. This result suggests that the process is perhaps a (almost hard-core) soft core process with a larger radius. Thus, we repeat the analysis with p = 0.1 and a radius of 3 cells (HC_5_unipat2b.res):



This parameter now describe the pattern at scales r = 1 - 4 well, but there is a small departure at scale r = 5.

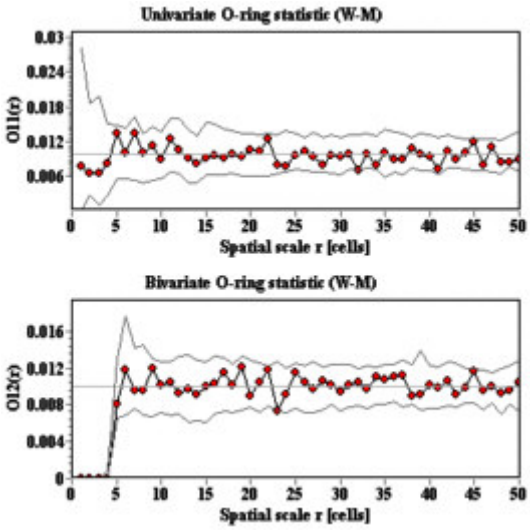
15) For the bivariate analysis highlight the data file "HC5.dat" in window Input data file.
16) select "List" in How are your data organized
17) select "Analyze all data in rectangle" in Give modus of analysis
18) select "Data are given as list in grid" in Select modus of data
19) click button "Calculate index" and *Programita* shows you the pattern and the univariate analysis without confidence envelopes:



20) The bivariate O-ring statistic suggests a hard-core of pattern 1 with a distance δ_{12} = 4 or 5 cells. Therefore enable the check box "Radius of pattern 1" and "Radius of pattern 2". Since pattern 2 may have a disc with radius 3

cells select for the random pattern 1 a radius of 2 cells p_1 = 11, and p_{12} = 0. Select for pattern 2, as in the previous example (HC_5_unipat2b.res)), a radius of 3 cells and p_2 = 0.1.

21) click "ok", and then the button "Calculate index". *Programita* now performs the simulations of the bivariate hard-core null model (HC_5.res):



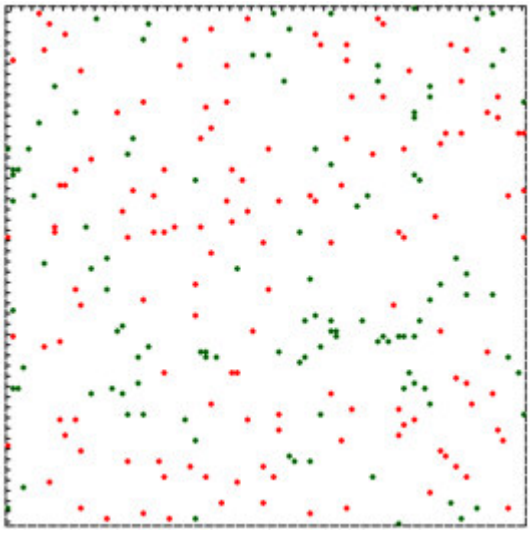
The bivariate hard-core null model describes the data well.

4.8.3. Reanalysis of example Indep3 (HC_6.res)

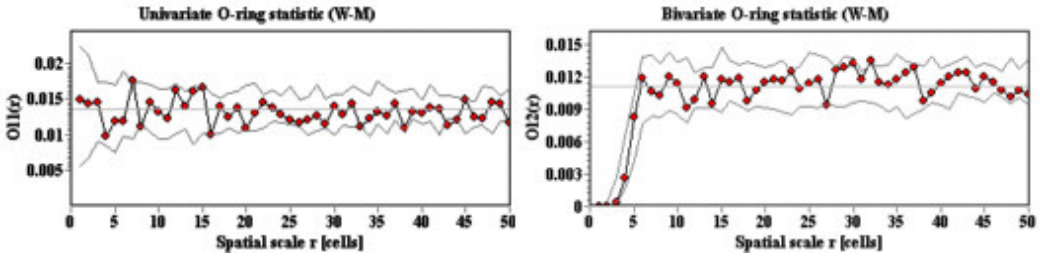
The data set of example Indep_3.res was created using used a bivariate soft-core process to simulate repulsion of points of pattern 2 in relation to points of pattern 1 that previously was created by a CSR process. The parameters of the bivariate soft-core process were: radius = 3 and exponent p = 5 (for pattern 1), radius = 3 and exponent p = 5 (for pattern 2), and the exponent for repulsion of pattern 2 by pattern 1 was p = 0.1. Thus, points of pattern 2 are placed at random with respect to already accepted points of pattern 2, but the probability $p_{HC}(d)$ to accept a provisional point of pattern 2 with a nearest neighbour of pattern 1 at distance d is given through equation HC5 with δ = 3 +3. Note that the univariate analysis of pattern 2 reveals aggregation at scales r = 1 - 5 which is a result of is non-random creation process.

1) highlight the data file "repulsion1.dat" in window **Input data file.** Pattern 1 is a random pattern and pattern 2 was created with an explicit repulsion mechanism: random provisional points of pattern 2 were only accepted if they had a nearest neighbor distance of at least δ = 6 to a point of pattern 1 and the probability of acceptance decreased with increasing distance to a point of pattern 1.
2) select "List" in **How are your data organized.**

3) select "Analyze all data in rectangle" in Give modus of analysis
4) select "Data are given as list in grid" in Select modus of data
5) click button "Calculate index" and *Programita* shows you the pattern and the univariate analysis without confidence envelopes:



6) The univariate O-ring statistic of pattern 1 indicates a random pattern, and the bivariate O-ring statistic indicates repulsion of type 2 points by type 1 points.
7) To perform the univariate analysis of pattern 2 without confidence envelopes highlight the file "repulsion1_21.dat" in window Input data file and click button "Calculate index". The univariate O-ring statistic of pattern 2 indicates aggregation, and the bivariate O-ring statistic indicates repulsion between type 1 and type 2 points.
8) With this diagnosis we parameterize the bivariate soft-core model. The univariate analysis of pattern 1 suggests a random pattern, thus select p_1 = 11. The univariate analysis of pattern 2 suggests no repulsion of type 2 points (it shows aggregation), therefore select p_2 = 11. Because there is interaction between type 1 and type 2 points with a relatively sharp soft-core enable the check box "Radius of pattern 1" and "Radius of pattern 2". Because the interaction between type 1 and type 2 points shows a sharp soft-core with a minimal distance $\delta \approx 6$ select p_1 = 0.1 and for both patterns a radius of 3 cells.
22) click "ok", and then the button "Calculate index". *Programita* now performs the simulations of the bivariate hard-core null model (HC_6.res):



The bivariate hard-core null model describes the data well. The interesting feature of this process is that the

aggregation of pattern 2 was not caused by an explicit aggregation mechanism but through repulsion by type 1 points which forced type 2 points into the random gaps left by of type 1 points.

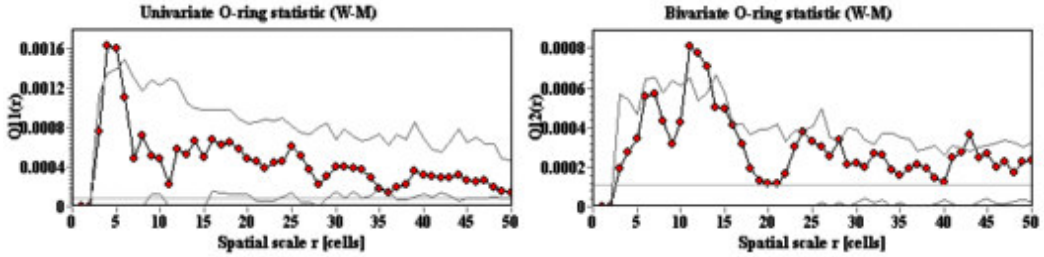
4.8.4. Example for cluster and hardcore (HC_7_res)

This is the bivariate variant of example HC_4.res and shows the combination of the hardcore null model with a bivariate Neyman-Scott cluster null model, continuing the analysis of example NS_4.res. The pattern of adult trees showed at a fine resolution of 0.25 m a marked hard-core up to 1m and a peak at some 2.5 m, and the bivariate O-ring statistics at a fine resolution of 0.25 m showed a marked hard-core up to 1m and two peaks at some 3m and 6 m.

1) Highlight the data file "A_1.dat" in window **Input data file.** This data set gives the location of adult trees and recruits at a meter scale, but has a resolution of 1 centimeter.
2) select "List" in **How are your data organized**
3) select "List with coordinates, no grid" in **Select modus of data**. A window opens asking you to provide a cell size. Insert "0.5", thus using a cell size of 50cm.
4) click the button "change" in **set maximal radius rmax** and set the maximal scale r of the analysis to r_{max} = 50.
5) click button "Calculate index", *Programita* shows you the pattern and calculates the *O*-ring function of the data.
6) To determine Monte Carlo confidence intervals for the bivariate Neyman-Scott null model enable the check box "Calculate confidence interval" on the upper left. A window with settings for null models appears, select "cluster process". A window with a selection of cluster process null models appears, enable "bivariate Neyman-Scott" and press ok.
7) *Programita* calculates the *g*- and the *L*-function for r = 1 to r_{max} and the window **Fit of Neyman-Scott models to data** appears.
8) Select r_{max} = 15 and r_0 = 100. To optimize the *g*- and the *L*-function simultaneously enable "both, L- and g-function". Click the button "fit" and *Programita* searches the parameters of the bivariate Neyman-Scott model that simultaneously fits the *g*- and *L*- function of your data best (red line: fit, black line: data).
9) To optimize the parameter fit, press the button "Zoom". *Programita* now determines the probable range of the parameters. Next, press "fit" and *Programita* now searches the best fit. We find σ_{best} = 36.6 and ρ_{best} = 0.0000424. Insert the results of the previous univariate analysis of adult trees (HC_4.res): σ_1 = 29.5 and ρ_1 = 0.0000205., and recruits: σ_2 = 25.47, ρ_2 = 0.0000227 and press "check". *Programita* calculates a theoretical σ_t = 27.6. Insert σ = 25 in the window "Fitted parameters" and click "ok".
18) Enable the check box "Hard core" and the window **Hard core null model** appears. Enable the checkboxes "Radius of pattern 1" and "Radius of pattern 2" and provide the hardcore radius of pattern 1 and 2(in both cases a value of 2). The minimal distance of two points is the double of the hardcore radius (i.e., δ = 4). Provide the exponent p = 0 for hard-

core for p_1, p_2, and p_{12} and click "ok". The window **Hard core null model** disappears.

19) click button "Calculate index" and *Programita* performs the 99 simulations of the combined cluster and hard-core null model:



Indeed, the combined hard-core and cluster null model describes the data well, even the first peak at 3m of the bivariate O-ring statistic, but recruits still show attraction at scale 6m.

4.9. Recommendations for selection of bivariate null models

The bivariate analysis is more complicated than the univariate analysis because there are several basic null models (independence, random labeling, and antecedent condition) and because null models from the univariate case can be combined in several ways to obtain specific bivariate null models. Therefore, it is especially important to define the biological question, the hypothesis, and the biological circumstances carefully to be able to find an adequate null model.

1. Visualize the patterns and perform univariate analysis of both patterns. Define the **basic null hypothesis**. If the univariate analysis indicated that both patterns were random, they are also independent. Otherwise, there are three conceptually different possibilities (i.e., **independence**, **random labeling**, **antecedent condition**) that lead to different procedures for null models and different values for the expected *g*- or *K*- function under absence of interaction between the two types of points:
    a. Two different processes may have created the two patterns and interactions between both types of points may have occurred. In this case the null hypothesis of independence may be appropriate.
    b. Two different processes created the two patterns, but pattern 1 already existed when pattern 2 was created. In this case the antecedent condition needs to be considered by selection of the appropriate null model. The locations of pattern 1 remain fixed and the null model distributes only pattern 2 in accordance to a specific univariate null model.
    c. The locations of both patterns were probably created by the same stochastic process and the labels (or marks) correspond to some events that acted independent from the process that created the locations of the points (e.g., tree dead or disease spread). In this case the null model of random labeling is appropriate.
2. A common environmental factor affected both patterns in the same way: In this case, the two patterns are heterogeneous and are merged in joint clusters. Under this circumstance, a random labeling null model may be appropriate if the environmental heterogeneity constrained the locations of both patterns in the same way. A patchy distribution of resources can also be modeled with a bivariate Neyman-Scott cluster process. This null model includes the case of partly overlapping clusters (i.e., some clusters are only occupied by type 1or type 2 points), and each pattern may have a different cluster size. However, there is also heterogeneous Poisson process null model with a similar effect: keep the locations of pattern 1 fixed

and randomize pattern 2 according to a heterogeneous Poisson process. An appropriate intensity function can be constructed using a moving window estimate of the joined intensity of pattern 1 and pattern 2, but with a relatively small radius *R*.

3. The two patterns were created by different processes: In this case, you might use the toroidal shift null model to test for independence, i.e., keeping pattern 1 fixed and shifting the whole of pattern 2 by treating the study region as a torus. Of course, this works only if you have a rectangular study region.
4. The two patterns were created by different processes related to different heterogeneous environmental factors: The appropriate null model for this hypothesis is to keep one pattern fixed and preserve the larger-scale heterogeneity of the other pattern, i.e., use a heterogeneous Poisson process to simulate pattern 2, and vice versa. An appropriate intensity function can be constructed using a moving window estimate of the intensity of pattern 2. The radius *R* of the moving window decides how closely you mimic the heterogeneity of pattern 2.
5. The two processes were linked: An example for this possibility is a clustered distribution of seedlings around adult trees e.g., due to a limited range of seed dispersal. In this case, the locations of trees have to be preserved, and the seedlings can be randomized following a Neyman-Scott process null model under antecedent condition where the parents are given through the pattern of adult trees. In this case, only one parameter of the cluster process has to be fit since the intensity of the parents is given through the density of pattern 1. Note that a similar effect of clustering of seedlings around trees may arise if both patterns are strongly impacted by the same environmental factor.

5. References

Bailey, T. C. and Gatrell, A. C. 1995. Interactive spatial data analysis. - Longman Scientific & Technical

Besag, J. 1977. Contribution to the discussion of Dr. Ripley's paper. - Journal of the Royal Statistical Society B 39: 193-195.

Diggle, P. J. 1983. The statistical analysis of spatial point patterns. - Academic Press.

Dixon, P. M. 2002. Ripley's *K* function. - Encyclopedia of Environmetrics 3: 1796-1803.

Goreaud, F. and Pelissier, R. 1999. On explicit formulas of edge effect correction for Ripley's *K*- function. - Journal of Vegetation Science 10: 433-438

Goreaud F., and R. Pèlissier. 2004. Avoiding misinterpretation of biotic interactions with the intertype K_{12}-function: population independence vs. random labeling hypotheses. *Journal of* Vegetation Science: 14: 681-692.

Naves, J., T. Wiegand, E. Revilla, and M. Delibes. 2003. Endangered species balancing between natural and human constrains: the case of brown bears (*Ursus arctos*) in northern Spain. *Conservation Biology* 17:1276-1289.

Ripley, B. D. 1981. Spatial statistics. - Wiley

Schadt, S., E. Revilla, T. Wiegand, F. Knauer, P. Kaczensky, U. Breitenmoser, L. Bufka, J. Cerveny, P. Koubek, T. Huber, C. Stanisa, and L. Trepl 2002. Assessing the suitability of central European landscapes for the reintroduction of Eurasian lynx. *Journal of Applied Ecology* 39:189-203.

Stoyan, D. and Stoyan, H. 1994. Fractals, Random Shapes and Point Fields. Methods of geometrical statistics. - John Wiley & Sons

Upton, G. and Fingleton, B. 1985. Spatial data analysis by example: volume 1: point pattern and quantitative data. - John Wiley & Sons.

Wiegand T., and K. A. Moloney. 2004 Rings, circles and null-models for point pattern analysis in ecology. Oikos 104: 209-229.