Acta Biologica Debrecina (2002) **24**: 235–249

Local and Global Indirect Spatial Series Analyses: Basic Procedures in Pascal

Erdei, Zsolt and Tóthmérész, Béla

Ecological Institute, Debrecen University
Debrecen, P. O. Box 71, H-4010 Hungary
zserdei@terra.ecol.klte.hu

Abstract Computer procedures are presented to analyze multispecies point patterns. Basic procedures of local and global indirect spatial series analyses are presented in Pascal. Four procedures are presented. The first two are based on bitmasking, to increase the number of sample plots. The third one is to count the frequency of species combinations in a series of sample plots in the case of a local indirect spatial series analysis. The fourth one is to identify and count the frequency of species combinations for a global spatial series analysis with special emphasis on the speed and efficiency of the procedure. The performance and application of the procedures are also discussed.

Keywords Multispecies spatial point patterns, Local and global indirect spatial series analyses, Frequency of species combinations, Bitmasking, Computer procedures in Pascal.

Introduction

The analysis of multispecies spatial point patterns is a rather computer-intensive area of science; especially the indirect spatial series analyses (Bartha and Horváth 1987, Czárán and Bartha 1989, Podani 1982, 1984a, 1984b). It is based on the analysis of species combinations of a community of a given plot size and a series of increasing plots are used to calculate a curve which characterize the spatial structure of the community (Juhász-Nagy 1993, Juhász-Nagy and Podani 1983, Podani et al. 1993). In the case of an S–species community there are 2^S species combinations. This fact reveals the first difficulty of the indirect spatial series analyses. When we would like to identify correctly the frequency distribution of species combinations, we need a sample which is larger than 2^S; more exactly much larger. This means, that even a pure sampling of a communty should be based on a couple of thousand sample plots. However, a lot of different sized plots are used. This is a rather difficult situation and there are intensive researches to develop special methods to decrease it (Tóthmérész 1994a). Depending on the parameter to be estimated it is possible to decrease the sample size (Camiz 1993, Camiz and Gergely 1990, Tóthmérész 1994c, Tóthmérész and Erdei 1994). Instead of that, the computer procedures are quite vital during the analyses. This was one of the basic motivation to publish the procedures presented in the paper. Despite the fairly intensive research in pattern analysis, the indirect spatial series analysis is not widely used enough to incorporate into generally used packages. Therefore many of the researchers develop and use their own computer programs. The another motivation was the new and highly recommended tendency to make public the most important procedures (Hand and Krzanowski 1992) and not to hide into expensive packages where the source code is not free.

General description of computer procedures

Four procedures are presented in the paper. Each of them is closely related. The procedures **PutSpecies** and **GetSpecies** are based on bitmasking and they are used to increase the number of sample plots during the analysis. The procedure **CountSpeciesCombinations** is to count the frequency of species combinations in a step of a local indirect spatial series analysis. This involves that the procedure just count the frequency of the species combinations which are really present in the sample but do not bother about the identification; i.e. the species combination encountered first is the "1", the second is the "2", etc. and there is no special bookkeping to identify the species list of the sample plots. This is done, however, in the **GlobalCountSpeciesCombinations** procedure. The procedure identify and count the frequency of the species combinations in the sample plots. The procedure produces two files containing the result. The file with *.**FRQ** extension contains the frequency of species combinations including the missing species combinations (zero frequency) and the second one with *.**SPC** extension supplies the species lists for all the 2^S species combinations in the from of a string which contains 1's and 0's to indicate whether the species is present or absent. The first three procedurs are included into a program called **LocalSpatialSeriesAnanlysis_OneStep**. The last one is included into the program **GlobalSpatialSeriesAnanlysis_OneStep**.

The running time of procedure **CountSpeciesCombinations** increases linearly with the number of species of the studied community also linearly with the number of sample plots. The running time of procedure

GlobalCountSpeciesCombinations also increases linearly with the number of sample plots; moreover it increases just linearly with the number of species. And the increase of the number of species combinations is really the main limit here. In the case of a procedure which directly identify and counts all the species combinations, the running time increases with 2^S as S increases. This is a rather severe limitations. Erdei et al. (1994) pointed out, that the computing time may be as high as a couple of years even for a moderately species rich community. In the presented procedure it is done in a tricky way using bitmasking and this makes the procedure very efficient and extremely fast.

The procedure **GlobalCountSpeciesCombinations** use special procedures for bitmasking: **PutSpecies2** and **GetSpecies2**. They are essentially identical with **PutSpecies** and **GetSpecies**. There are two important differences. The first one is that these procedures are faster and the second one is that the their bitmasking capacity is limited up to 30 species while **PutSpecies** and **GetSpecies** can be used for communinities as rich as 128 species. However, this is not a real restriction in the case of a global indirect spatial series analysis, because for a 30 species community the number of species combinations is $1'073'741'824$ which is rather high to cope with.

How to use the procedures

In this section we mention a few applications of the procedures presented in the paper. The general usefulesness of **PutSpecies** and **GetSpecies** is evident. For a species rich community where the number of species is 128 the storage capacity of the species lists of 1000 sample plots (i.e. a 0/1 matrix of 128 by 1000) is 125 **Kbyte**. Using the presented procedures it is 15.625 **Kbyte**; i.e. using the same storage capacity we can use eight-times more plots. The spatial series analyses frequently needs rather high sample size, therefore this is a crucial aspects of any computer program related to spatial series analysis.

The procedure **CountSpeciesCombinations** can be used in a standard local indirect spatial series analysis when the identification of a characteristic areas, usually the maximum area, are the main objective (see e.g. Bartha 1990, Juhász-Nagy 1967, Juhász-Nagy and Podani 1983). It supplies the number of species combinations and a vector which contains the frequency of those species combinations which were really present in the studied step of the spatial series analysis. The spatial statistics can be calculated using this vector. The diversity of species combinations (more exactly the *species list*

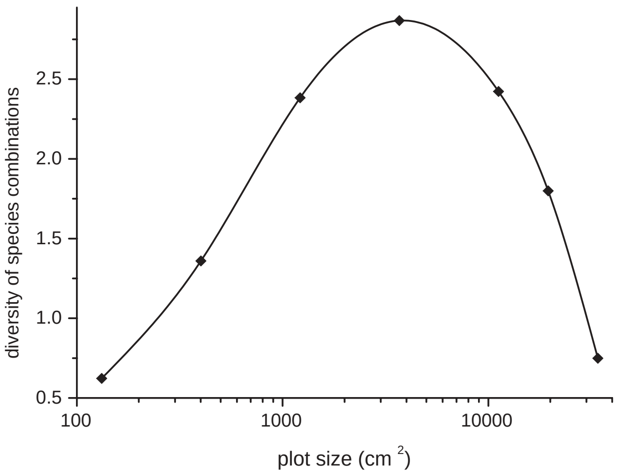

Fig. 1: A local indirect spatial series analysis with 7 steps (plot sizes); 1000 sample plots were used for each step of the spatial series analysis. The analysis is based on a completely spatially random alternative of the shrub community of a beech forest clearing at the Rejtek Project Research Area in Hungary.

- *number of plots* diversity) is used as a spatial statistics on the Figure 1, which presents the analysis of a completely spatially random alternative of the community of the four dominant species of a beech forest clearing in Hungary (Katona and Tóthmérész 1985, Tóthmérész 1989, 1994.).

Global spatial series analysis is crucial to study the scaling of the indirect spatial series analyses. The scaling behaviour of a global indirect spatial series analysis was studied for a completely spatially random alternative of the shrub community of a beech forest clearing at the Rejtek Project Research Area in Hungary. Nonmetric multidimensional scaling was used to display the change of the species combination vectors during the spatial series analysis (see also Tóthmérész 1994a). It is evident from the Figure 2 that the changes between the steps of the spatial series analysis are roughly linear; the increment of the plot size was logarithmic. At the beginning and at the very end the differences between the steps are a little bit smaller. It also can be seen that the first axis is strongly correlated with the plot size and also can be proved (Tóthmérész 1994a) that the second one correlates with the diversity of species combinations. Therefore, the step "4" is a crucial step of the spatial series analysis; this is the maximum area for the diversity of species combinations. This is even more clearly stressed by other multivariate techniques, such like correspondence analysis and PCA. In some respect similar idea was published by Camiz and Gergely to identify the maximum area (Camiz and Gergely 1990). There are tremendous lot of future perspectives in global spatial series analysis; only the difficulties to cope with are even larger.

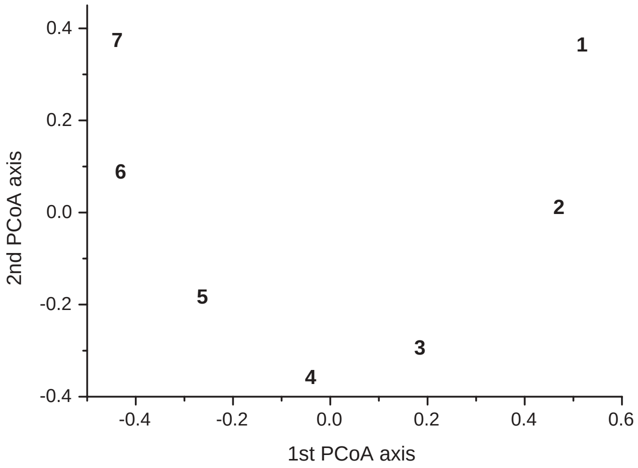

Fig. 2: Nonmetric multidimensional scaling of the steps of a global indirect spatial series analysis. A completely spatially random alternative of the shrub community of a beech forest clearing at the Rejtek Project Research Area in Hungary was studied.

Acknowledgements The research was partly supported by the the Hungarian Research Fund (OTKA) No. F6082 and T32130 research grants.

References

Bartha, S. 1990: Spatial processes in developing plant communities: pattern formation detected using information theory. In: Krahulec, F., Agnew, A.D.Q., Agnew, S. and Willems, J. (eds): Spatial Processes in Plant Communities, Academia, Praha, Czechoslovakia, pp. 31–47.

Batha, S. and Horváth, F. 1987: Long transects and information theoretical functions to pattern detection I. Transects versus isodiametric sampling units. *Abstracta Botanica* **11**: 9–26.

Camiz, S. 1993: STATIS ordination vs. the Juhász-Nagy models: The predictability of an exploratory tool. *Abstracta Botanica* **17**: 29–30.

Camiz, S. and Gergely, A. 1990: An exploratory method for determining optimal plot size in plant communities. *Abstracta Botanica* **14**: 83–108.

Czárán, T. and Bartha, S. 1989: The effect of spatial pattern on community dynamics: a comparison of simulated and field data. *Vegetatio* **83**: 229–239

Erdei Zs. and Tóthmérész B. 1993: MULTI-PATTERN 1.00. A program package to analyze and simulate community-wide patterns. *Tiscia* **27**: 45–48.

Erdei Zs. , Tóthmérész B. and Erdei, A. 1994. Linear algorithm to calculate indirect spatial statistics for completely random multi-species communities. *Tiscia* **28**: 67–72.

Hand, D. and Krzanowski, W. 1992. General Interest Section. *Applied Statistician* **41**: 561

Juhász-Nagy, P. 1967. On some 'characteristic area' of plant community stands. Proc. Colloq. Inf. Theory, Bolyai Math. Soc., Debrecen pp. 269-282.

Juhász-Nagy, P. 1993. Notes on compositional diversity. *Hydrobiologia* **249**: 173-182.

Juhász-Nagy, P. and Podani J. 1983. Information theory methods for the study of spatial processes and succession. *Vegetatio* **51**: 129–140.

Katona É., Tóthmérész B. 1985. Changes in the herb layer of submontane forest after deforestation. *Botanikai Közlemények* **72**: 17–25. (Hungarian)

Podani, J. 1982. Spatial Processes in the Analysis of Vegetation. PhD thesis, University of Western Ontario. National Library of Canada Microfiche No. TC 56124.

Podani, J. 1984a. Spatial processes in the analysis of vegetation. Theory and review. *Acta Bot. Hung.* **30**: 75-118.

Podani, J. 1984b. Analysis of mapped and simulated vegetation pattern by means of computerized sampling techniques. *Acta Botanica Hungarica* **30**: 403–425.

Podani, J., Bartha, S. and Czárán, T. 1993. Pattern, area and diversity: the importance of spatial scale in species assamblages. *Abstracta Botanica* **17**: 37–51.

Tóthmérész, B. 1989. Progress of succession: a study based on cluster analysis. *Botanikai Közlemények* **76**: 35–49. (Hungarian)

Tóthmérész, B. 1994a. Diversity ordering and spatial series analyses. DSc dissertation, pp. 196, Debrecen. (Hungarian)

Tóthmérész, B. 1994b. Statistical analysis of spatial pattern in plant communities. *Coenoses* **9**: 33–41.

Tóthmérész, B. and Erdei, Zs. 1992. The effect of species dominance on information theory characteristics of plant communities. *Abstracta Botanica* **17**: 62–68.

Tóthmérész, B. and Erdei, Zs. 1995. New Features of MULTI-PATTERN 1.10: Robust Nonlinear Smoothing. *Tiscia* **29**: 33–36.

Appendix

```
Program LocalSpatialSeriesAnalysis_OneStep;

Const
  MaxPlotN = 2500; {maximum number of plots}
  MaxSpeciesN = 100; {maximum number of species}
  CompressedSpN = (MaxSpeciesN+7) Div 8;
    {number of bytes to store MaxSpeciesN bits}

Type
  IntRowVector    = Array[1..MaxPlotN] Of Integer;
  IntColumnVector = Array[1..MaxSpeciesN] Of Integer;
  MatrixType      = Array[1..MaxPlotN,0..CompressedSpN-1] Of Byte;

Var
  DataFile             : Text;
  dumStr               : String;
  Information          : String;
  nSpeciesCombinations : Integer;
  nSpecies             : Integer;
  nPlots               : Integer;
  I ,J, Plot, Sp       : Integer;
  Data                 : Real;
  SpecN                : IntRowVector;
  PlotN                : IntColumnVector;
  SpeciesCombination   : IntRowVector;
  DataMatrix           : MatrixType;

(*
nSpeciesCombinations: number of species combinations (species
                      lists) found in the sample
nSpecies : number of species in the sample = number of
           columns of the uncompressed data matrix
nPlots : number of plots in the sample = number of
         rows of the uncompressed data matrix
SpecN : a vector containing the number of species
        for the plots of the sample
PlotN : a vector containing the number of occurences
        of the species in the sample plots
```

```
SpeciesCombination: a vector containing the frequency of
                    species combinations (species lists)
                    found in the sample
DataMatrix: a matrix containing the coded values of the
            raw data matrix coded by bitmasking
*)

Procedure PutSpecies(Plot, Species : Integer);
 Const Mask : Array[0..7] Of Byte = (128, 64, 32, 16, 8, 4, 2, 1);
 Var  Where : Byte;
Begin
  Dec(Species);
  Where:=(Species Div 8); { actual column of datamatrix }
  { setting bit to 1 }
  DataMatrix[Plot,Where] :=
            DataMatrix[Plot,Where] Or Mask[ Species Mod 8 ];
End;

Function GetSpecies(Plot, Species : Integer) : Boolean;
  { returns TRUE when the species is present and
            FALSE when it is absent }
  Const Mask : Array[0..7] Of Byte = (128, 64, 32, 16, 8, 4, 2, 1);
  Var  Where, CurrentMask : Byte;
Begin
  Dec(Species);
  Where:=(Species Div 8);
  CurrentMask:=Mask[ Species Mod 8 ];
  GetSpecies:=((DataMatrix[Plot,Where] And CurrentMask)=CurrentMask);
End;

Procedure CountSpeciesCombinations(
            Row, Column: Integer;
            Var SpecN : IntRowVector;
            Var nSpeciesCombinations: Integer;
               { number of species combinations }
            Var SpeciesCombination : IntRowVector;
            Var PlotN : IntColumnVector );
Var
  Counter1, Counter2, Sp : Integer;
  Free : Array[1..MaxPlotN] Of Boolean;

Function Identical(N1, N2: Integer): Boolean;
  {compare two species-combinations } Var Species : Integer;
```

```
Begin
  IF SpecN[N1]<>SpecN[N2] Then Begin
    identical:=False;
    Exit;
  End;
  For Species:=1 To Column Do
    IF GetSpecies(N1,Species)<>GetSpecies(N2,Species) Then Begin
     identical:=False;
     Exit;
   End;
   identical:=True;
 End;
 Begin
   For Sp:=1 To Column Do PlotN[Sp]:=0;
   For Counter1:=1 To Row Do Begin
     Free[Counter1]:=True;
     SpecN[Counter1]:= 0;
     SpeciesCombination[Counter1]:=0;
   For Sp:=1 To Column Do
     IF GetSpecies(Counter1, Sp) Then Begin
       Inc(SpecN[Counter1]);
       Inc(PlotN[Sp]);
     End;
   End;
   For Counter1:=1 To Row Do Begin
     IF Free[Counter1] Then Begin
       Inc(nSpeciesCombinations);
       Free[Counter1]:=False;
       Inc(SpeciesCombination[nSpeciesCombinations]);
       For Counter2:=Counter1 + 1 To Row Do Begin
         IF identical(Counter1,Counter2) Then Begin
           Inc(SpeciesCombination[nSpeciesCombinations]);
           Free[Counter2]:=False;
         End;
       End;
     End;
   End;
 End;
 Begin  {initialize data matrix }
   For I:=1 To MaxPlotN Do
     For J:=0 To CompressedSpN-1 Do DataMatrix[I,J]:=0;
     {open datafile and generate the coded species list matrix}
     {with bitmasking while loading the data }
```

```
      Assign(DataFile, 'local.dat');
      Reset(DataFile);
      ReadLn(DataFile, Information);
      ReadLn(DataFile, nPlots, nSpecies);
      For Plot:=1 To nPlots Do Begin
        For Sp:=1 To nSpecies Do Begin
           Read(DataFile, Data);
     If (Data > 0.0) Then putspecies(Plot,Sp);
       {species is present in this plot }
   End;
   ReadLn(DataFile, dumStr); { read until end of line }
   End;
   Close(DataFile);
   { calculate the frequency of species lists }
   CountSpeciesCombinations(
     nPlots, nSpecies, SpecN, nSpeciesCombinations,
     SpeciesCombination, PlotN);
   { generating the output: frequency of the species combinations }
   { which were really present in the sample }
   Assign(DataFile, 'local.out');
   ReWrite(DataFile);
   WriteLn(DataFile,
    'Number of species combinations found: ',
     nSpeciesCombinations:5);
   For I:=1 To nSpeciesCombinations Do
     WriteLn(DataFile, I:5, '.', SpeciesCombination[I]:7);
   Close(DataFile)
 End.
```

```
LOCAL.DAT to test the LOCAL.PAS program
 20 11
 22 14 10  5  6 6 3 1 2 3 3
 23  9 11  9  7 5 2 1 4 1 0
 17  0 11 10  8 4 4 4 2 1 0
 18 18  9  9  6 3 6 5 3 0 1
 20 15  6  8  7 5 6 0 2 1 2
 23 19  9  7  4 2 3 9 1 3 1
 23 14 10  8  9 6 6 3 3 1 0
 21  9 11 10  8 3 2 4 2 2 1
 22 14 14  9  7 7 3 1 4 1 0
  0  7 12 18  7 7 3 0 4 3 0
 21 14  8 10  5 6 2 0 3 2 3
  0 12 13  3  7 5 3 0 1 1 0
```

```
 22 22 16  9  3 5 3 4 1 1 1
 19 11  9  9  7 7 3 2 4 2 2
 22 15  4  8  7 7 2 4 2 1 1
 26 19  7  6  9 9 4 2 4 2 0
 21 16 10  9  7 1 5 7 0 2 1
 22  9  6 11 10 5 2 3 4 2 0
 18 18  9  5  3 6 5 1 1 1 1
 20 14  9 11  3 9 4 3 0 2 1

Number of species combinations found: 7
 1. 7
 2. 5
 3. 1
 4. 1
 5. 2
 6. 2
 7. 2
```

```
program GlobalSpatialSeriesAnalysis_OneStep;

 Const
   MaxSpecies = 10;
   max_NSC = 1024; { 2^MaxSpecies=1024 }
   MaxNumberofFiles = 50;
   StringLength = 79;
 (*
  MaxSpecies : maximal number of species for the analysed community
  max_NSC : maximal number of species combinations;
  max_NSC := 2^MaxSpecies
  MaxNumberofFiles:
     the species composition of the sample plots are stored
     in a file for each step of the spatial series analysis.
     Therefore, MaxNumberofFiles is the maximal number of
     the steps of the spatial series analysis.
 *)

 type
   StringS      = String[StringLength];
   VectorType   = array[0..max_NSC-1] of longint;
   StringVector = array[1..MaxNumberofFiles] of StringS;

 var
   inpfile, outfile : text;
```

```
  nFiles           : integer;
  nSpecies         : integer;
  dataVector       : VectorType;
  FileNames        : StringVector;

procedure putSpecies2(var target: longint; what: integer);
begin
 {set bit representing species 'what' to '1'}
 target:=target or ($00000001 shl (what-1));
end;

function GetSpecies2(var source: longint; what: integer):boolean;
begin
  {test bit representing species 'what' to '1'}
  GetSpecies2:=(source and ($00000001 shl (what-1))) > 0;
end;

function power2(exponent: integer): longint;
begin
  power2:={\$}00000001 shl exponent; { 2^exponent }
end;

Procedure GlobalCountSpeciesCombinations(
             nSpecies, nFiles : Integer;
             var FileNames    : StringVector );
var
  dumStr           : string;
  Information      : string;
  i, species, plot : Integer;
  nPlot            : Integer;
  speciesList      : LongInt;
  NSC              : LongInt;
  Data             : Real;

Begin
  for i:=0 to max_NSC-1 do dataVector[i]:=0;
    { initialize dataVector }
    NSC:=power2(nSpecies);
    assign(outfile, 'global.frqt');
    rewrite(outfile);
    writeLn(outfile,
      'Frequency of Species Combinations; GLOBAL.FRQ');
    writeLn(outfile, nFiles :5, NSC :10);
```

```
    for i:=1 to nFiles do begin
      assign(inpFile, filenames[i]);
      reset(inpFile);
      readln(inpFile, information);
      readln(inpFile, nPlot, nSpecies);
      for plot:=1 to nPlot do begin
        SpeciesList:=0;
        for species:=1 to nSpecies do begin
          read(inpFile, Data);
          if ( Data > 0.0 ) then putSpecies2(SpeciesList, species);
        end;
        readln(inpFile, dumStr);
        inc(datavector[SpeciesList]);
      end;
      close(inpFile);
      for species:=0 to NSC-1 do
        write(outFile, dataVector[species] :4);
      writeLn(outFile);
    end;
    close(outFile);
    assign(outfile,'global.spc');
    rewrite(outfile);
    writeln(outfile, 'Species Combinations'; GLOBAL.SPC);
    for SpeciesList:=0 to NSC-1 do begin
      for i:=1 to nSpecies do
        if getspecies2(specieslist,i) then write(outfile,'1') else
                                           write(outfile,'0');
      writeln(outfile);
    end;
  close(outfile);
End;

BEGIN
  { initialise the fileNames array }
  fileNames[1] := 'demo1.dat';
  fileNames[2] := 'demo2.dat';
  fileNames[3] := 'demo3.dat';
  nFiles := 3;
  nSpecies := 3;
  GlobalCountSpeciesCombinations(nSpecies, nFiles, FileNames );
END.
```

```
Demo data for GLOBAL.PAS; DEMO1.DAT
```

```
 10 3
 4 0 0
 0 0 0
 0 5 0
 0 0 3
 0 0 0
 0 2 0
 0 0 0
 5 0 0
 0 2 0
 0 0 0

Demo data for GLOBAL.PAS; DEMO2.DAT
  10 3
 5 0 3
 2 0 9
 0 7 0
 6 0 3
 0 7 3
 0 2 0
 0 0 0
 5 0 4
 4 2 0
 0 5 0

Demo data for GLOBAL.PAS; DEMO3.DAT
  10 3
 4 3 9
 2 3 2
 9 2 3
 0 3 7
 0 8 2
 0 2 2
 0 2 9
 3 2 7
 2 3 0
 6 4 0

Frequency of Species Combinations; GLOBAL.FRQ
 3 8
 4 2 3 0 1 0 0 0
 5 2 6 1 1 4 1 0
 5 2 6 3 1 4 5 4
```

```
Species Combinations; GLOBAL.SPC
 000
 100
 010
 110
 001
 101
 011
 111
```