TEMPORAL PATTERN ANALYSIS – A NEW ALGORITHM FOR DETECTING PATCH SIZE IN PLANT POPULATIONS

Á. Méri and L. Körmöczi

Méri, Á. and Körmöczi, L. (2010): Temporal Pattern Analysis – a new algorithm for detecting patch size in plant populations*. — Tiscia 38, 3-9.*

Abstract. Pattern analysis is one of the most important evaluation methods of population structure and of community assembly rules. Several algorithms have been developed to detect the deviation of spatial arrangement of organisms from random. These algorithms reveal in general the pattern intensity that is estimated by the average size of aggregation patches. It is often important to detect the number and actual size of patches and distinguish from the random appearance of individuals. We used the framework of python programming language to develop a new algorithm for computing patch size and position along transects. We used a long term data set originating from a dry sand grassland to test the program. The algorithm was designed to process very large data sets, and compute temporal variation of patch size distribution.

Key words: spatial pattern, PYTHON, time series, model

Á. Méri, L. Körmöczi, Department of Ecology, University of Szeged, H-6726 Szeged Közép fasor 52., Hungary

Introduction

Pattern — according to the Oxford English Dictionary — is a regular or discernible form or order in which a series of things occur. We talk about spatial patterns in ecology, if the distribution of the living organisms deviates from random, most frequently in the direction of aggregation. It means that the individuals form groups in space, and this pattern can be examined also in time. The importance of patterns manifests very well in variable fields of ecology. The 'patchiness' of individual species distributions contributes to the community organization and therefore it is of central importance in ecology (Anderson and Neuhauser 2002). To understand the natural processes and structures we need to define the scales at which the pattern occurs (Fortin and Dale 2005) and is the result of the local processes of population interaction and dispersal (Murrel *et al.* 2001). Scale appears in spatial structure, if the non-randomness exhibits a certain periodicity (Dale 1991). These examinations have crucial importance in applied use as well. In the investigation of endangered species or in the examination of the structures of habitats we use different kinds of pattern analysis. The information gained by these methods can be valuable in itself, however it can refer to some kind of background factors as well (Fortin and Dale 2005).

The aim of this study was to find an adequate method for analysing our data set about psammophile grassland populations. It was expected to find temporal changes in the spatial pattern of the certain plant populations.

Materials and methods

The database, sampling methods

The data, used for the analyses and testing derive from sandy grassland in Bugac, belonging to the area of Kiskunsági National Park. A transect with 55 meters length and 1 meter width was established (Körmöczi and Balogh 1990), and the monitoring was carried out three times per year (Körmöczi et al 2009). Seasonal data can be compared in the consecutive years, because the field work took place approximately on the same dates every year. In a line of adjacent quadrats, the presence/absence data (0 or 1 values) of the plant populations were recorded. The size of the quadrats was 25×25 centimeters, which is a widely used size in the case of dry grasslands. It

means that a grid of 220×4 cells represents each plant population in each monitoring date. The database can be considered as one dimensional data, because the transect is much longer than wide. The four rows of the grid were treated as four individual series and the mean of the values was counted at the analyses.

Quadrat variance methods

There are many ways of analysing one dimensional spatial data. One group of the widely used methods is based on variance analysis of the adjacent blocks of quadrats (Hill 1973). The sampling method must always be carried out with a contiguous grid of quadrats, where presence/absence data of species are recorded (Dale *et al.* 2002). Considering the properties of our database, in the following we will examine the methods only of the aspect of binary data.

All of the quadrat variance analyses apply the moving split window technique. The essence of this method can be summarized in the following main points (Körmöczi 2005). One dimensional data set is adequate for this technique. In the first step we select the first two adjacent cells, as the two halves of the window (1). Then we record the difference of the values in the cells (2). The next step is moving the window one cell further and calculating the difference. These steps are iterated through the whole transect (3). Reaching the end of the data series we calculate the variance (4). Then we jump back to the first cell, and increase the size of the window (5). Then the steps 1-4 are repeated with this new window size. We can increase the size of the window until the halves reach the half size of the transect.

The result is a variance data for each window size. Plotting these data against block size, we get the variance graphs of the data set. The peaks represent the scale of the pattern. However these methods are unable to distinguish patches and gaps (Guo and Kelly 2004), some of them can separate a smaller and a larger phase (Dale 1999). After analysing artificial and simulated data sets as well, we recognized that this latter feature of the methods is reflected only by the analysis of simulated periodical data. Although there are many detailed description about the quadrat variance methods, the literature of this topic is nearly not exhausted. Further investigations may explore additional features of these methods. For examining them, we transformed the mathematical formulas into executable programs in python programming language.

TemPA

Based on our experiences with quadrat variance methods our aim was to create a new algorithm for one dimensional pattern analysis. However we used a transformed version of moving window technique, and the mathematical background is much easier, this method is similar to the quadrat variance analyses.

TemPA (***tem****poral* ***p****attern* ***a****nalysis*) is a precise patch size determining algorithm. We can get information about the size and position of each patch. The result is the simplified picture of the transect, containing all the necessary information about the bigger patches, but in a reduced form. It means that the computer records only the minimal information determining the patches. So one patch can be determined with actually two values, the position of the first cell and the extent of the patch. This data coding is widely used in raster based geographical information systems and called run-length coding. Simplifying the information about the data set provides an easier treatment in the followings.

The procedure performed on the data may seem rather difficult, but the gained information is as much as available in the smallest possible extent. We transformed the moving window technique by changing the order of switching and enlargement. First the position of window is set, then the window is increased with one cell in each step. After using all the possible block sizes beginning with the first cell, we shift the position one cell further, and repeat the entire process. We used an undivided window, thus the first block size in each position could be only one cell large. For practical reason the starting window size was three cells. Detection of a plant in a single cell can not be considered as a patch, thus there is no meaning of investigating under three cells extent. The maximum window size is equal to the total size of the transect.

Identification of patches is carried out with a system of variable conditions. These are simple mathematical relations, determining the acceptable number of absences in the currently examined block. The following traits of a potential patch are considered:

- the whole extent of the patch – the actual size of the window
- the first and the last cell of the patch – mus be 1
- one cell before and after the patch – must be 0
- the beginning and the end of the patch (3-4 cells) – sum of presences must be over or equal to 3

the area surrounding the patch (3-4 cells) – sum of presences must be under or equal to 2

For each of these intervals we set a mathematical relation. For instance the first and the last cell of the patch must be 1, or the presences in the surrounding area must be under or equal to 2. The predetermined limits are freely alterable, so are the sizes of the investigated intervals. This alterable structure makes the method flexible, thus we can optimize the parameters to our data set.

At each step of the analysis the computer counts the number of presences in the investigated interval. If the value is out of the threshold value, the process stops, and the analysis of the next potential patch starts. If all the values are within the predetermined limits, the investigated area is considered as a patch, and the requisite information are recorded.

The criteria are previously given by the user and can be optimized for the currently analysed data set. When monitoring rare species with small spatial extent, we are searching for small patches of plants, or individuals. Decreasing the minimum detected patch size to one cell will refine the scale. Common species require investigations on a larger scale. The settings of criteria 4-5 determine the required density in the boundary zone between a patch and a gap. With these options we can set, how clear the border should be. Ignorance of the smaller patches is often useful, avoiding the processing of unnecessary amount of information. During the tests we used an average setting, which is appropriate for a large scale of aggregation types.

Results

The validation of the results is carried out visually. The comparison of the original data and the model, created by the computer is required. Bar charts are proper for this aim (Fig. 1). The diagram represents the original view of the transect and the simplified model, built up from the data recorded by **TemPA**. The representation is very much distorted, thus the entire transect can be plotted onto one single picture. One small column represents one quadrat.

The analysed database contained data about 112 plant species in 24 monitoring dates. Twenty five of these plant species were represented on every occasion, so detailed investigations were performed in the case of these species. Some of them were highly abundant, others were rarely occurring. The optimization of the criteria system was performed based on these results. Working with the final settings the analysis of the 25 plants was carried out.

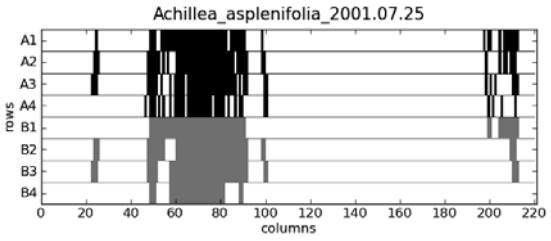

Fig1. The distribution of *Achillea asplenifolia* on the 25th of July 2005. The rows marked with A represent the real distribution, and the ones marked with B are the interpretation of the model built up by the computer. One column on the bar chart represents one quadrat.

After controlling the effectiveness of **TemPA** with the bar charts representing the distribution of each species, the temporal analyses were performed. This means a single line plot for each species, represented on Fig. 2. The lines follow the temporal trends very sensitively due to the punctual patch size detection. Whether counting the average patch size, or using the maximum patch size for the description of the distribution, depends on the user's demand. During the analyses we used the maximum patch size, and counted the mean of the 4 rows of data tables.

In the case of species with very big changes in seasonal distribution, we can represent the seasonal data with 3 different lines (Fig. 3).

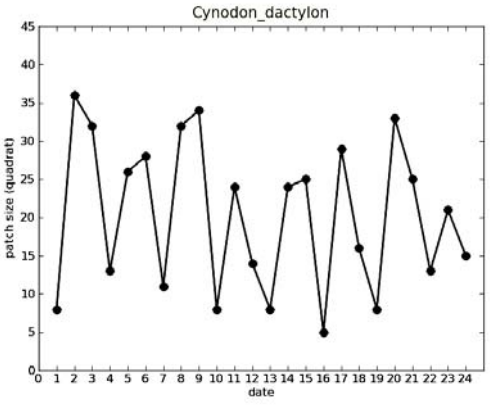

Fig 2. The temporal changes of the patch size of Cynodon dactylon. Each monitoring date got a number, simplifying the visualization. Patch size means in this case the size of the biggest patch.

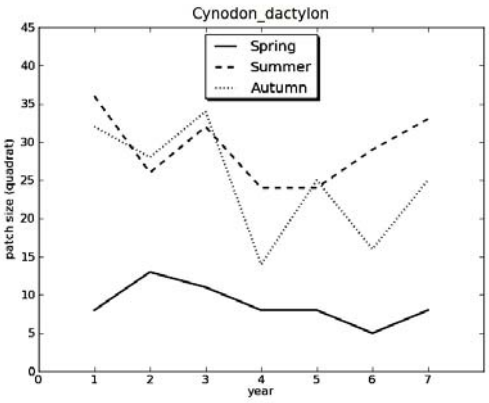

Fig 3. The annual changes of the patch size in the case of Cynodon dactylon. The same values are presented then in Fig 2. Here the seasonal data are plotted separately.

Discussion

Several kinds of methods, analysing one dimensional spatial data are currently available (Dale 2005). The literature, investigating the properties of these methods is comprehensive (Dale 2005, Hill 1973, Goodall 1974, Guo and Kelly 2004, Rosenberg 2001). However there are still undeveloped parts of the methodology of detecting spatial pattern. Our aim was the development of a concrete patch size detecting algorithm. One of the most important factors was the automation of the analysis, permitting of the processing of large amounts of data.

The algorithm, used in **TemPA** is a flexible application, written in python programming language. The properties of the patch size detection are highly modifiable, thus the user can easily form the method to the adequate criteria. The system of conditions provides the possibility of optimization for the user's needs. Without a graphical interface the method and the printed results are freely adjustable. Getting punctual information, we can perform a detailed investigation about the transect not only in space, but in time as well. The available information include the number and the size of the patches and the positioning of each.

Acknowledgement

This project was supported by the Hungary-Romania Cross-Border Co-operation Programme 2007-2013 (HURO/0801/194)

References

Anderson, K. and Neuhauser, C. (2002): Patterns in spatial simulations—are they real? — Ecological Modelling, 155, 19-30

Dale, M. R. T. and Blundon, D. J. (1991): Quadrat covariance analysis and the scales of interspecific association during primary succession – Journal of Vegetation Science 2: 103-112

Dale, M. R. T., Dixon, Ph., Fortin, M.-J., Legendre, P., Myers, D. E. and Rosenberg, M. S. (2002): Conceptual and mathematical relationships among methods for spatial analysis. — Ecography 25: 558-577

Fortin, M.-J. and Dale, M. R. T. (2005): Spatial Analysis, A guide for ecologists. — Cambridge University Press, New York

Goodall, D. W. (1974): A new method for the analysis of spatial pattern by the random pairing of quadrats. — Vegetatio 29: 135-146

Guo, Q. and Kelly, M. (2004): Interpretation of scale in paired quadrat variance methods. — Journal of Vegetation Science 15: 763-770

Hill, M. O. (1973): Diversity and Evenness: A Unifying Notation and Its Consequences. — Ecology 45: 427-432

Körmöczi L. (2005): On the sensitivity and significance test for vegetation boundary detection – Community Ecology 6(1): 75-81

Körmöczi, L. and Balogh, A. (1990): The analysis of pattern change in a Hungarian sandy grassland. — In: Krahulec, F., Agnew, A. D. Q., Agnew, S. and Willems, J. H. (eds): Spatial processes in plant communities. pp. 49-58.

Körmöczi, L., Margóczi, K. and Zalatnai, M. (2009): Kiskunsági homoki gyep hosszú távú állományszerkezet-változása. — In: Gallé, L. (ed.): Entomológia: kutatás, szemléletformálás, ismeretterjesztés. Szeged, 2009. pp. 91-106.

Murrell, D.J., Purves, D.W. and Law, R. (2001): Uniting pattern and process in plant ecology. — TREE 16, 529-530

Rosenberg, M. S. (2001): PASSAGE. Pattern Analysis, Spatial Statistics, and Geographic Exegesis. Version 1.0 — Beta documentation. Department of Biology, Arizona State University, Tempe, AZ.

Appendix
Source code of `TemPA:`

```
import cPickle
import pylab as plab
import numpy as np
from pylab import *

f1=open("data.dat", "r")
f2=open("dates.dat", "r")
f3=open("names.dat", "r")
adat=cPickle.load(f1)
dates=cPickle.load(f2)
names=cPickle.load(f3)
adatarray=np.array(adat)

year=8
year=year-1
plant=0
print dates.get(year)
print names.get(plant)

for k in range(0,4):
  a=adat[year][plant][k]
  n=220
  s=[]
  h=[]
  p=[]
  l=[]
  w=[]
  de=0
  woo=[]
  for i in range(0,n-3):   #position
   q=0
   for r in range(2,n-i+1): #window size
    v=[]
    x1=[]
    x2=[]
    x3=[]
    x4=[]
    t=0
    q=0
    g=[]
    no=0
    nu=0
    el=1
    veg=0
    for j in range(i,i+r):
     v.append(a[j])
     t=(sum(v))
    if (t>=0.85): #terms
     el=a[i]
     veg=a[i+r-1]
     if (i+r==n-1):
      no=0
```

```
      nu=0
     if (i+r<=n-2):     #cell before/after the patch
      nu=a[i+r]
     if (i>=1):
      no=a[i-1]
     if r<10:
      x1=[4]
      x2=[4]
      x3=[0]
      x4=[0]
     if r>=10:
      for y1 in range(i+r-3,i+r): #last 4 cels
       x1.append(a[y1])
      for y2 in range(i,i+3):  #first 4 cels
       x2.append(a[y2])
      if (i+r<=n-4):
       for y3 in range(i+r,i+r+3): #4 cels after the patch
        x3.append(a[y3])
      if (i>=4):
       for y4 in range(i-4,i-1):  #4 cels before the patch
        x4.append(a[y4])
      if (i+r>n-4) and (i+r<n-1):    #end of patch
       for y3 in range(i+r,n):
        x3.append(a[y3])
      if (i<4) and (i>1):       #beginning of patch
       for y4 in range(0,i+1):
        x4.append(a[y4])
      if (i+r>=n-1):
       x3=[0]
      if (i<=1):
       x4=[0]
    if ((sum(x1))>=2) and ((sum(x2))>=1)and ((sum(x3))<=1) and
((sum(x4))<=1)and (el==1) and (veg==1):
     q=t
     if (t>=(0.87*r)):
      g.append(q)
      g.insert(1,r)
      g.insert(2,i)
      g.insert(3,i+r-1)
      h.append(g)

  for c in h:
   p.append(c[2])
  for e in range(len(p)):
   if (p[e]>(p[e-1]+4)) or (e==(len(p)-1)):
    b=de
    de= p.index(p[e])
    w= h[b:de]
    s=[]
    l=[]
    we=[]
    fe=[]
    ba=0
    femax=0
    for f in w:
```

```
 s.append(f[0])
 if (s!=[]):
  ba=max(s)
for z in range(len(s)):
 if (s[z]==ba):
  we.append(w[z])
for ii in we:
 fe.append(ii[1])
 if (fe!=0):
  femax=min(fe)
for y in range(len(fe)):
 if (fe[y]==femax):
  print we[y]
```