

Cálculo de L de Ripley e O-Ring

Dentre os padrões que uma determinada população vegetal pode assumir, destacam-se três: Aleatório, Uniforme e Agregado. Uma maneira de verificar qual padrão espacial ocorre em diversas escalas espaciais é utilizando o L de Ripley ($K_{(r)}$) ou o O-Ring ($O_{(r)}$) que são, basicamente, medidas de densidade em torno de um ponto.

Estes índices são calculados da seguinte forma:¹⁾

O $L_{(r)}$ é baseado na função K de Ripley, que é a densidade média de pontos a uma dada distância r de cada ponto, dividida pela intensidade (λ) dos pontos na área de estudo²⁾:

$$K_{(r)} = \frac{\sum_{i \neq j} I(d_{ij} < r)}{n} \frac{1}{\lambda}$$

$$L_{(r)} = \left(\frac{K_{(r)}}{\pi} \right)^{1/2} - r$$

Onde:

* d_{ij} é a distância do ponto i ao ponto j ;

* $I(d_{ij} < r)$ função indicadora, sendo 1 se o ponto está a uma distância menor que r de i , fora desse raio o ponto tem valor 0; e

* n é o número de pontos total.

Sendo que $L(r) > 0$ indica agregação, enquanto $L(r) < 0$ indica padrão homogêneo.

Logo, definimos $O(r)$ como:

$$O_{(r)} = L_{(r)} - L_{(r-l)}$$

Onde:

* $r-l$: é o raio menos a largura do anel³⁾

Na completa aleatoriedade espacial $O(r) = \lambda$ (intensidade do padrão), quando o padrão é agregado $O(r) > \lambda$ e quando é homogêneo $O(r) < \lambda$

Para saber mais, clique [aqui](#).

Entrada:

`pad.esp(x, y, lim, rmax, r, ic, nsim)`

- x = posições x de cada árvore (*classe numeric*)
- y = posições y de cada árvore (*classe numeric*)
- lim = limite de máximo e mínimo de x e y, respectivamente, em m^2 .
- $rmax$ = raio máximo (*classe integer*)
- r = tamanho do anel ou diferença entre dois raios consecutivos. (*classe integer*)
- ic = Se TRUE, faz intervalo de confiança através de simulações, se FALSE, não. (*classe logical*)
- $nsim$ = numero de simulações utilizadas para gerar IC. (*classe integer*)

Verificação de parâmetros:

- x é numérico e > 0 ? Se não, escreve “x deve ser numérico e > 0 ”.
- y é numérico e > 0 ? Se não, escreve “y deve ser numérico e > 0 ”.
- O comprimento de x, y são iguais? Se não, retorna “x, y devem ter o mesmo comprimento”.
- lim é numérico e > 0 ? Se não, escreve “lim deve ser numérico e > 0 ”
- lim tem comprimento 4? Se não, escreve “lim deve conter c(xmax, xmin, ymax, ymin)”

- r_{max} = é inteiro e de comprimento 1? Se não, retorna “ r_{max} deve ser inteiro e de comprimento 1”.
- r é inteiro e de comprimento 1? Se não, retorna “ r deve ser inteiro e de comprimento 1”.
- ic é lógico? Se não, retorna “ ic deve ser lógico”.
- $nsim$ é inteiro de comprimento 1? Se não, retorna “ $nsim$ deve ser inteiro e de comprimento 1”.

Pseudo-Código

- Cria matriz *distancia*, com número de linhas e colunas igual ao comprimento de x e y contendo NAs.
- Cria vetor *ntot* guardando o comprimento de x .
- Cria vetor *dens* dividindo *ntot* por $(x_{max} - x_{min}) * (y_{max} - y_{min})$.⁴⁾
- Cria sequencia *sequ* de 1 a r_{max} , com $by = r$.
- Cria vetor *ind*, com comprimento igual ao comprimento de *sequ*, contendo NAs.
- Ciclo for de 1 a *ntot* (repete o próximo for para todos os pontos).
 - Ciclo for de 1 a *ntot*.
 - Calcula distância de um ponto a todos os pontos.
 - Guarda resultado em uma linha de *distancia*.
 - Ciclo for de *sequ*.
 - Conta quantas quantos elementos de *distancia* são menores que 1: r_{max} e guarda em *ind*.
- Multiplica *ind* por $1/(ntot * dens)$ e guarda em *kr*.
- Calcula o L de Ripley pela fórmula $(kr/\pi)^{1/2} - r$ e guarda em *rip*.
- Extrai as posições *sequ* de *ind* e guarda em *tempring*
- Calcula diferença entre 2 posições consecutivas de *tempring* e guarda em *ring*.

Se $ic = TRUE$

- Cria objeto *indsim*, matriz com $nsim$ colunas e *ntot* linhas, contendo NAs.
- Cria objeto *ripsim*, matriz com $nsim$ colunas e *ntot* linhas, contendo NAs.
- Cria objeto *ringsim*, matriz com $nsim$ colunas e *ntot*-1 linhas, contendo NAs.
- Ciclo for de 1 a $nsim$
 - Sorteio de *ntot* números uma distribuição uniforme entre x_{min} , x_{max} , guarda em *xsim*
 - Sorteio de *ntot* números uma distribuição uniforme entre y_{min} , y_{max} , guarda em *ysim*
 - Cria objeto *distsim*, com número de linhas e colunas igual ao comprimento de *xsim* e *ysim* contendo NAs.
 - Ciclo for de 1 a *ntot* (repete os próximos for para todos os pontos)
 - Ciclo for de 1 a *ntot*
 - Calcula distância de um ponto a todos os pontos
 - Guarda resultado em *distsim*
 - Ciclo for de *sequ*
 - Conta quantas quantos elementos de *distsim* são menores que 1: r_{max} e guarda em *indsim*.
 - Multiplica *indsim* por $1/(ntot * dens)$ e guarda em *krsim*.
 - Calcula o L de Ripley pela fórmula $(krsim/\pi)^{1/2} - r$ e guarda em *ripsim*[*i*].
 - Extrai as posições *sequ* de *ind* e guarda em *tempring*

- Calcula diferença entre 2 posições consecutivas de `tempring` e guarda em `ringsim[i]`.
- Calcula quantis para cada raio de `ripsim`
- Calcula quantis para cada raio de `ringsim`
- Cria vetor `saida`, um dataframe com mesmo número de colunas que `sequ` contendo na
 - primeira linha: “agregado” se `rip >` quantil superior de `ripsim` e “homogêneo” se `rip <` quantil inferior de `ripsim`
 - segunda linha: “aleatório” se `ring` está entre os quantis de `ringsim`, “agregado” se `ring >` quantil superior de `ringsim` e “homogeneo” se `ring <` quantil inferior de `ringsim`.

Saída

Plota gráfico(s) com o(s) valor(es) de L-Ripley e/ou O-Ring (com intervalos de confiança se `ic = TRUE`)

Retorna `saida`.

- 1)
Extraído da página da disciplina Ecologia de Populações e comunidades vegetais
- 2)
intensidade, nesse caso, é a densidade total; número de pontos médio por unidade de área
- 3)
raio menor do anel
- 4)
ou seja, a área total de estudo

From:

<http://ecor.ib.usp.br/> - **ecoR**

Permanent link:

http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2018:alunos:trabalho_final:jennifer.auler:fun2



Last update: **2020/08/12 06:04**