Environmental and Ecological Statistics
https://doi.org/10.1007/s10651-026-00750-5

Déjà vu: point pattern analysis and point-in-polygon methods in ecological coding adventures

Duccio Rocchini[1]

Received: 16 February 2026 / Revised: 19 June 2026 / Accepted: 5 July 2026

Abstract

Point pattern analysis (PPA) is a key spatial approach in ecology, used to investigate how individual organisms or events are distributed across a landscape and what ecological processes drive these patterns. Among the different point pattern analysis types, point-in-polygon (PIP) is a fundamental spatial technique in ecology, used to examine the relationship between discrete point-based observations and broader landscape or habitat units. By overlaying point features—such as species occurrences, nests, or sampling locations—onto polygonal regions like habitat types, land use zones, or conservation areas, ecologists can investigate spatial patterns, assess species–habitat associations, and quantify ecological processes at multiple scales. This method facilitates the aggregation of ecological data and supports the integration of fine-scale biological observations with coarse-scale environmental or management variables. Point-in-polygon analysis plays a critical role in conservation planning, biodiversity monitoring, and ecological modeling, especially as spatial datasets become increasingly high-resolution and accessible. This paper disentangles the fundamental mathematical principles behind point-in-polygon analysis, with emphasis on the most widely used algorithms, namely Ray Casting and the Sunday's methods. To bridge computational geometry and applied spatial ecology, I introduce the `insideR` package, a lightweight R implementation that operates directly on coordinate matrices and provides transparent, reproducible point-in-polygon testing. Both theoretical derivations and ecology-based simulation examples are presented in R, a widely used open-source software environment among ecologists.

Keywords Geographic Information Systems (GIS) · Habitat association · Landscape analysis · Point-in-polygon · Point pattern analysis · Spatial data visualization · Spatial ecology · Species distributions

Handling Editor: Luiz Duczmal.

Extended author information available on the last page of the article

Published online: 30 July 2026

Abbreviations
PIP Point-in-polygon
PPA Point pattern analysis

1 Point patterns in ecology

The spatial arrangement of organisms across landscapes is a central focus in ecology, as spatial structure often reflects underlying ecological processes (Bacaro et al. 2015). These spatial patterns are not only shaped by environmental heterogeneity and species interactions but also influence ecological dynamics such as competition, reproduction, dispersal, and survival (Clobert et al. 2009). One powerful tool for studying such spatial arrangements is *point pattern analysis* (PPA, (Gatrell et al. 1996), Box 1), which focuses on the spatial distribution of discrete events or individuals in a defined study area. In ecological contexts, these points could represent individual trees, animal sightings, nests, disease outbreaks, or any biologically meaningful event localized in space.

Point pattern analysis enables researchers to determine whether the observed distribution of individuals follows a random pattern or deviates significantly due to biological or environmental processes (Korner-Nievergelt et al. 2010). Typically, three major types of spatial patterns are considered: *random*, *clustered* (aggregated), and *regular* (uniform). A random distribution implies the absence of interaction or preference, whereas clustered patterns may suggest social behavior, localized resources, or limited dispersal ability. In contrast, regular patterns are often indicative of competition or territoriality, where individuals actively maintain a minimum distance from one another.

Several statistical techniques have been developed to quantify and analyze point patterns. Among the most widely used are Ripley's K-function, the pair correlation function $g(r)$, and nearest-neighbor distance analyses (Self et al. 2022). These methods allow ecologists to explore patterns at multiple spatial scales and test hypotheses about the processes generating observed distributions. Quadrat analysis and **kernel density estimation**[1] (Box 1 at the end of the paper) are also commonly used, particularly when visualizing the intensity and distribution of spatial points (Borruso 2005). Recent advances in computational ecology and geographic information systems (GIS) have further enhanced the accessibility and precision of point pattern analysis.

Importantly, the analysis of spatial point patterns has far-reaching implications in applied ecology. For instance, in conservation biology, understanding the clustering of rare or endangered species can inform the design of protected areas or corridors (Dobson et al. 1997). In forest ecology, spatial analysis of tree distribution helps in detecting patterns of succession, disturbance, or species coexistence (Hardy and Sonké 2004). Similarly, in epidemiology, point patterns are used to track the spread of wildlife diseases or invasive species (Staubach et al. 2002). Furthermore, the

[1] The terms in bold in this paper are part of the glossary of Box 1.

increasing availability of high-resolution spatial data from remote sensing, drones, and automated sensors has opened new frontiers for integrating point pattern analysis with other ecological modeling frameworks (Torresani et al. 2024).

As ecological systems face growing pressures, spatially explicit analyses such as point pattern analysis are becoming essential for monitoring, modeling, and managing biodiversity. These tools not only reveal how species and ecosystems are organized in space but also provide a quantitative basis for ecological inference, decision-making, and long-term ecological forecasting.

2 Point-in-polygon analysis in ecology

Among the point pattern analysis types, the **point-in-polygon** (PIP, Box 1) problem involves determining whether a given point lies inside, outside, or on the boundary of a polygon (Hormann and Agathos 2001). Solving this accurately is essential in fields such as computer graphics, GIS, path planning, and physical simulations.

Spatial relationships are a core component of ecological analysis, particularly when attempting to understand how individual organisms or events relate to broader landscape structures (Nelson and Boots 2008). The *point-in-polygon* analysis represents a powerful method to study these relationships (Haines 1994). This approach involves assessing the spatial occurrence of point-based ecological data (e.g., locations of species, nests, or sampling plots) relative to predefined polygonal regions such as habitat types, management zones, land cover classes.

By linking point data to the attributes of the polygons in which they fall, researchers can evaluate ecological processes at multiple spatial scales and assess the influence of landscape composition and configuration on species distributions and behaviors (Johnson 1990).

Before performing point-in-polygon analysis, spatial datasets often undergo **geometric partitioning** (see Box 1 and Box 2) to delineate distinct zones such as habitat patches, land use types, or administrative boundaries (Gilbert et al. 1998). This partitioning ensures that each point can be meaningfully assigned to a well-defined spatial unit.

In practice, point-in-polygon analysis involves overlaying a layer of spatial points onto a polygonal layer using Geographic Information Systems (GIS) or spatial analysis libraries in R or Python. Each point is then assigned to a specific polygon, and its spatial attributes can be aggregated, summarized, or statistically modeled based on the associated polygon characteristics. This enables ecologists to link fine-scale observations (e.g., GPS-tagged animal movements) with broader-scale environmental or management data.

Ray Casting is among the most widely used point-in-polygon approaches because it is conceptually simple, computationally efficient, and readily applicable to both convex and concave polygons. For these reasons, many point-in-polygon implementations rely on this principle. In algorithmic terms, the Ray Casting method (see Sect. 3 and Algorithm 1) is based on intersection counts of a horizontal ray from the point to the polygon edges. If odd, the point is inside the polygon, if even it is outside. In ecological data analysis, this operation is frequently required when

assigning species occurrences, vegetation plots, animal tracking locations, or sensor observations to habitat patches, management units, or protected areas. Efficient point-in-polygon algorithms therefore constitute a fundamental component of many spatial ecological workflows.

Algorithm 1 Ray Casting algorithm (general)

```
1: function IsInside(polygon, point)
2:     count ← 0
3:     for all edge ∈ polygon do
4:         if ray from point intersects edge then
5:             count ← count + 1
6:     return count mod 2 == 1
```

Taking into account a list of ordered vertices that form the polygon and a single point which is tested for inclusion, it initializes a counter to store the number of ray-edge intersections (*count* ← 0). Then, it iterates through each edge of the polygon (**for all** *edge* ∈ *polygon*). For each edge, it checks if the horizontal ray from the point intersects that edge and, if yes, it increments the count (*count* ← *count* + 1). Finally, it will return true if the number of intersections is odd (i.e., the point is inside) and false if even (point is outside). This is evaluated using the modulo operator (`mod`), which returns the remainder after integer division. Therefore, `count mod 2 = 1` indicates an odd number of intersections, whereas `count mod 2 = 0` indicates an even number.

Applications of point-in-polygon analysis in ecology are numerous. In conservation biology, it can help assessing which protected areas harbor the highest species richness or detect biases in biodiversity monitoring (Gaston et al. 2008). In landscape ecology, it can quantify species-habitat associations by comparing point densities across habitat types (Marini et al. 2019). In disease ecology, it is instrumental for mapping incidence rates across epidemiological zones and identifying high-risk regions (Enticott et al. 2021). Additionally, combining point-in-polygon with statistical models (e.g., generalized linear models or spatial regressions) allows researchers to test hypotheses about the drivers of spatial patterns in ecological phenomena (Warton et al. 2016).

Point-in-polygon analysis remains a foundational tool in ecological research, offering scalable, interpretable, and integrative ways to link organism-level observations to landscape-level patterns and processes. While traditional approaches have been implemented in various GIS software packages, many of these tools lack code-based, efficient, and integrated solutions specifically designed for ecological research. To address these gaps, I developed the `insideR` package in R https://github.com/ducciorocchini/insideR/, which provides a streamlined, open-source solution for performing point-in-polygon analysis. The package implements two algorithms—Ray Casting and Sunday's algorithms—providing flexibility and computational efficiency for large-scale ecological datasets. Unlike existing alternatives, `insideR` is tailored for ecological applications, ensuring accurate, transparent, and reproducible results.

This paper introduces `insideR` and links it with point pattern analysis, a rapidly expanding field as spatial datasets become increasingly detailed and with high-resolution. Using `insideR`, ecologists can now easily determine spatial relationships between point data—such as species occurrences—and predefined polygons, such as habitat boundaries or conservation zones. This seamless integration offers new opportunities for testing ecological theories and improving conservation planning.

3 Mathematical foundations behind point-in-polygon algorithms

The point-in-polygon (PIP) problem lies at the intersection of computational geometry and topology. This section formally defines polygons and explores the mathematical principles underlying two main algorithms used to determine point inclusion.

3.1 Polygon definitions

A polygon is a closed plane figure bounded by a finite chain of straight line segments. A polygon is said to be

- **Simple** Edges do not intersect, except at vertices.
- **Convex** All interior angles are less than 180°.
- **Concave** At least one interior angle is greater than 180°.
- **Self-intersecting (complex)** Edges cross over each other.

3.2 Polygonal representation

Let a polygon $\mathcal{P} \subset \mathbb{R}^2$ be defined as a closed sequence of n vertices:

$$\mathcal{P} = \{V_1, V_2, \ldots, V_n\}, \quad \text{with } V_i = (x_i, y_i).$$

The polygon is closed by defining $V_{n+1} = V_1$. Each edge is the directed segment:

$$E_i = \overrightarrow{V_i V_{i+1}} = (x_{i+1} - x_i,\ y_{i+1} - y_i).$$

A polygon is

- **Simple** Edges only intersect at vertices.
- **Non-simple** Self-intersecting, requiring winding-based methods.

3.3 Point-in-polygon as a decision problem

Formally, the PIP problem is:

$$\text{given } P = (x_0, y_0) \in \mathbb{R}^2, \text{ determine whether } P \in \text{Interior}(\mathcal{P}).$$

3.4 Ray Casting via edge intersections

This section presents the mathematical derivation underlying the Ray Casting algorithm summarized in Algorithm 1. While Algorithm 1 provides the computational procedure (Fig. 1), the following equations explain the geometric principles used to determine ray-edge intersections. To determine the number of ray crossings, consider a horizontal ray extending to the right from point $P = (x_0, y_0)$. An edge (V_i, V_{i+1}) intersects the ray if:

$$(y_i > y_0) \neq (y_{i+1} > y_0)$$

and the x-coordinate of the intersection point $x_{\text{intersect}}$ satisfies:

$$x_{\text{intersect}} = x_i + \frac{(y_0 - y_i)(x_{i+1} - x_i)}{y_{i+1} - y_i}.$$

$$\text{If } x_0 < x_{\text{intersect}} \Rightarrow \text{intersection occurs.}$$

The number of valid intersections (modulo 2) determines the point's status:

$$\text{Inside if \#intersections mod } 2 = 1.$$

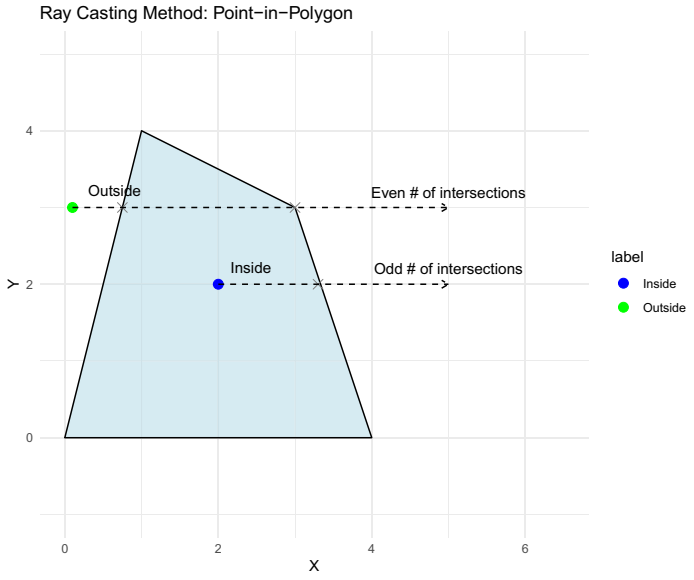

Fig. 1 Ray casting example: one ray intersects edges odd times (inside), the other even (outside). The code to reproduce this figure is freely available at: https://github.com/ducciorocchini/insideR/blob/main/examples/

3.5 Sunday's algorithm

Dan Sunday's point-in-polygon algorithm relies on the concept of the **winding number**, a topological invariant that measures how many times a closed polygon winds around a point $P = (x_0, y_0)$. For a polygon defined by vertices $V_0, V_1, \ldots, V_n$, where $V_i = (x_i, y_i)$, the algorithm considers each edge $e_i = (V_i, V_{i+1})$ and tests whether a horizontal ray emanating from P in the positive x-direction intersects the edge. The key test is whether the point P lies between the vertical span of the edge:

$$(y_i \leq y_0 < y_{i+1}) \quad \text{or} \quad (y_{i+1} \leq y_0 < y_i),$$

and whether the point lies to the left of the edge, determined by computing the intersection point's x-coordinate:

$$x_{\text{intersect}} = x_i + \frac{(y_0 - y_i)(x_{i+1} - x_i)}{(y_{i+1} - y_i)}.$$

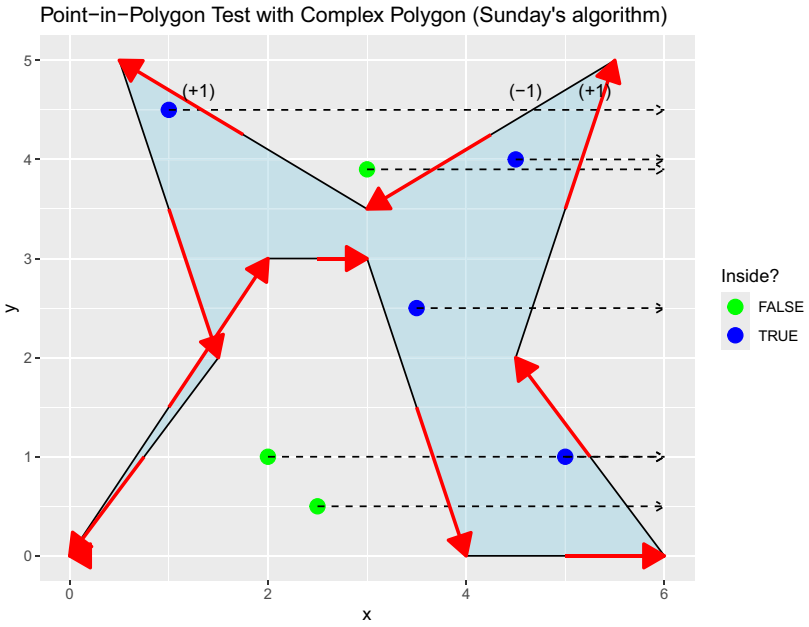

Fig. 2 Sunday's algorithm example: points are tested using a variation of the Ray Casting algorithm, making the sum of the intersections (winding number), considering the direction of edges of a polygon, i.e., crossing edges going upwards or downwards. As an example, for the first point on top which is inside the polygon, the straight line crosses an upward edge (+1), a downward edge (−1) and a upward edge again. the final winding number W equals 1. Being W non-zero the points lies inside the polygon. Winding number for the other points, from top to the bottom are: W=1, W=0, W=1, $W = 0$, $W = 1$, $W = 0$. The code to build this figure is available at: https://github.com/ducciorocchini/insideR/blob/main/examples/

If $x_0 < x_{\text{intersect}}$, the ray intersects the edge. The **winding number** W is then incremented or decremented based on the edge's direction:

$$W \leftarrow W + 1 \quad \text{if } y_i \leq y_0 < y_{i+1}, \quad \text{or}$$

$$W \leftarrow W - 1 \quad \text{if } y_{i+1} \leq y_0 < y_i.$$

Finally, the point P is considered inside the polygon if and only if $W \neq 0$. This avoids complex trigonometric operations and makes the algorithm efficient and numerically stable for both convex and non-convex polygons. When an edge crosses the ray in an upward direction, the winding number is increased; if it crosses in a downward direction, the winding number is decreased (Fig. 2).

4 Implementation in R

The `insideR` package hosted on GitHub (https://github.com/ducciorocchini/insideR/) provides streamlined and computationally efficient implementations of point-in-polygon algorithms, based on both the aforementioned Ray Casting and Sunday's algorithms. It can be installed via the `remotes` package as follows:

```
remotes::install_github("ducciorocchini/insideR")
```

This installation method allows users to quickly integrate the `insideR` package into their R workflows.

4.1 Ray Casting by the `point_in_polygon()` function

In order to perform the Ray Casting method in `insideR` the function `point_in_polygon()` can be used, based on Algorithm 2.

Algorithm 2 Ray Casting algorithm for point-in-polygon test as implemented in `insideR`

Input: Point (x, y), Polygon vertices $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$
Output: `TRUE` if the point is inside the polygon, `FALSE` if outside
`inside` $\leftarrow$ `FALSE`
$j \leftarrow n$ ▷ Last vertex index
for each vertex i from 1 to n **do**
    retrieve (x_i, y_i) and (x_j, y_j) ▷ Current edge of the polygon
    if $(y_i > y) \neq (y_j > y)$ **then**
        $x_{\text{intersect}} \leftarrow x_i + \frac{(y - y_i) \cdot (x_j - x_i)}{y_j - y_i}$
        if $x_{\text{intersect}} > x$ **then**
            `inside` $\leftarrow \neg$`inside`
    $j \leftarrow i$ ▷ Move to next edge
return `inside`

In practice, the function `point_in_polygon()` takes two inputs:

- `point` A vector representing the coordinates of the point to be tested (e.g., `point = c(x, y)`),
- `polygon` A matrix of size $n \times 2$ representing the coordinates of the polygon's vertices, where n is the number of vertices.

The function iterates through each edge of the polygon and tests whether a horizontal ray starting from the point intersects the edge. The main logic works as follows.

- It checks if the ray from the point intersects with the edge by evaluating whether the point lies between the vertical spans of the edge's endpoints.
- If the ray intersects the edge, the `inside` flag is toggled.
- At the end of the iteration, if the number of intersections is odd, the point is inside the polygon, otherwise, it is outside.

Making use of the `point_in_polygon()` function, one can test whether a point lies inside a polygon, based on the following example, using a polygon with four vertices and a point with coordinates (3, 3). The procedure involves creating the polygon, checking if the point is inside the polygon, and then plotting the results using `ggplot2`. Below is the R code implementation:

```
# Example polygon and point
polygon <- matrix(c(1,7, 5,4, 4,3, 2,6), ncol=2, byrow=TRUE)

point <- c(3, 3)

# Check if point is inside polygon
inside <- point_in_polygon(point, polygon)

# Prepare data for ggplot
polygon_df <- as.data.frame(polygon)
colnames(polygon_df) <- c("x", "y")
polygon_df <- rbind(polygon_df, polygon_df[1, ])  # Close the polygon

point_df <- data.frame(x = point[1], y = point[2],
                       label = ifelse(inside, "Inside", "Outside"))

# Plot
ggplot() +
  geom_polygon(data = polygon_df, aes(x = x, y = y),
               fill = "lightblue", color = "black", alpha = 0.5) +
  geom_point(data = point_df, aes(x = x, y = y, color = label), size = 4) +
  scale_color_manual(values = c("Inside" = "blue", "Outside" = "green")) +
  labs(title = "Point in Polygon Test",
       subtitle = paste("Point is", ifelse(inside, "INSIDE", "OUTSIDE"), "the polygon"),
       x = "X", y = "Y") +
  theme_minimal()
```

The resulting plot shows the polygon, the point, and the result of the point-in-polygon test in the plot title and subtitle (Fig. 3).

4.2 Sunday's algorithm using by the `point_in_polygon_sunday()` function

In order to perform the point-in-polygon test using Sunday's Winding Number Algorithm in `insideR`, the function `point_in_polygon_sunday()` can be used, based on Algorithm 3.

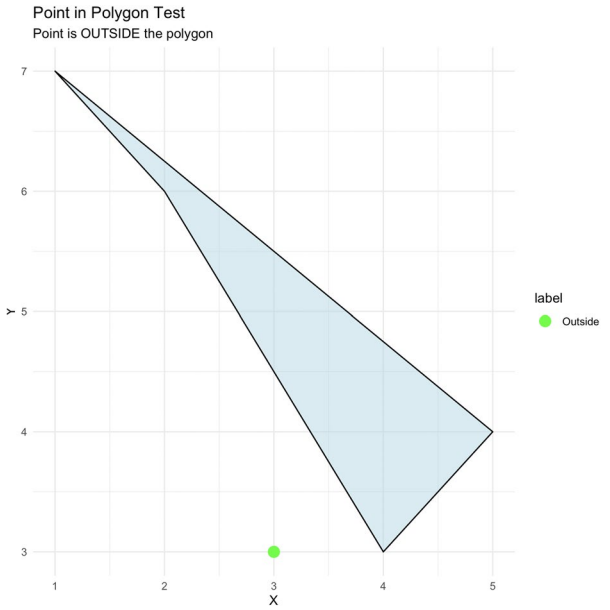

Fig. 3 Point-in-polygon test by Ray Casting method: the plot shows the polygon with the point marked. The point is outside (green) the polygon. The subtitle indicates the result of the point-in-polygon test

Algorithm 3 Sunday's winding number algorithm for point-in-polygon test

Input: Point (x, y), Polygon vertices $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$
Output: `TRUE` if the point is inside the polygon, `FALSE` if outside
`winding_number` $\leftarrow 0$
for each vertex i from 1 to n **do**
    retrieve (x_i, y_i) and (x_j, y_j) ▷ Current edge of the polygon
    if point is between y_i and y_j **then**
        $x_{\text{intersect}} \leftarrow x_i + \frac{(y-y_i)\cdot(x_j-x_i)}{y_j-y_i}$
        if the ray crosses the edge and $x_{\text{intersect}} > x$ **then**
            if $y_j > y_i$ **then**
                `winding_number` $\leftarrow$ `winding_number` $+ 1$ ▷ Upward crossing
            else
                `winding_number` $\leftarrow$ `winding_number` $- 1$ ▷ Downward crossing
return `winding_number` $\neq 0$ ▷ True if the point is inside

As for the `point_in_polygon()` function, `point_in_polygon_sunday()` takes two inputs:

- `point` A vector representing the coordinates of the point to be tested (e.g., `point = c(x, y)`),

- `polygon` A matrix of size $n \times 2$ representing the coordinates of the polygon's vertices, where n is the number of vertices.

The function iterates through each edge of the polygon and checks if a horizontal ray starting from the point intersects the edge. The winding number is updated based on whether the ray crosses an edge in an upward or downward direction. At the end of the iteration, if the winding number is non-zero, the point is inside the polygon.

Below is an example of the usage of the `point_in_polygon_sunday()` function, using a polygon with four vertices and a point with coordinates (3, 3). The procedure involves creating the polygon, checking if the point is inside the polygon, and then plotting the results using `ggplot2`:

```
# Example polygon and point
polygon <- matrix(c(2,2, 3,2, 6,1, 6,6, 5,6), ncol=2, byrow=TRUE)
point <- c(3, 3)

# Use Sunday's Algorithm to check if the point is inside the polygon
inside_sunday <- point_in_polygon_sunday(point, polygon)

# Print the result
print(inside_sunday)  # TRUE if inside, FALSE if outside

# Test with another point outside the polygon
outside_sunday <- point_in_polygon_sunday(c(6, 3), polygon)

# Print the result
print(outside_sunday)  # Should print FALSE

# Prepare data for ggplot
polygon_df <- as.data.frame(polygon)
colnames(polygon_df) <- c("x", "y")
polygon_df <- rbind(polygon_df, polygon_df[1, ])  # Close the polygon


 point_df <- data.frame(x = 3, y = 3,
                        label = ifelse(inside_sunday, "Inside", "Outside"))

 # Plot using ggplot2
 ggplot() +
   geom_polygon(data = polygon_df, aes(x = x, y = y),
                fill = "lightblue", color = "black", alpha = 0.5) +
   geom_point(data = point_df, aes(x = x, y = y, color = label), size = 4) +
   scale_color_manual(values = c("Inside" = "blue", "Outside" = "green")) +
   labs(title = "Point in Polygon Test",
        subtitle = paste("Point is", ifelse(inside_sunday, "INSIDE", "OUTSIDE"), "the polygon"),
        x = "X", y = "Y") +
   theme_minimal()
```

In the above example, the point with coordinates $(3, 3)$ is tested against a polygon with five vertices at $(2, 2)$, $(3, 2)$, $(6, 1)$, $(6, 6)$, and $(5, 6)$. The function `point_in_polygon_sunday()` checks if the point lies inside the polygon. The result is stored in `inside_sunday`, which is `TRUE` if the point is inside and `FALSE` if outside. The polygon vertices are converted into a data frame `polygon_df`, and the point is converted into `point_df` with a label indicating its status. Finally, the plot is created using `ggplot2`, where the polygon is filled with light blue, and the

point is marked in blue if inside or green if outside, with the result displayed in the plot title and subtitle (Fig. 4).

5 Simulated ecological example: species sightings and protected areas

Spatial analysis plays a fundamental role in ecology and conservation science. A common task is determining whether field observations—such as animal sightings or ecological samples—fall within designated conservation units. To illustrate this process in a transparent and reproducible way, I present a simulation based on the Ray Casting algorithm, using the `point_in_polygon()` function in `R`. Since both the Ray Casting and Sunday's algorithms lead to the same result, I chose to use Ray Casting because it is more intuitive compared to Sunday's method.

5.1 Simulation

The process begins by simulating a set of protected areas and species sightings, using as an example a tree species in a mixed forest (Fig. 5). Two simple square polygons represent protected areas, while 100 species sightings are randomly distributed across a 10-by-10 Cartesian grid. This controlled setup provides a convenient sandbox for testing and teaching spatial workflows.

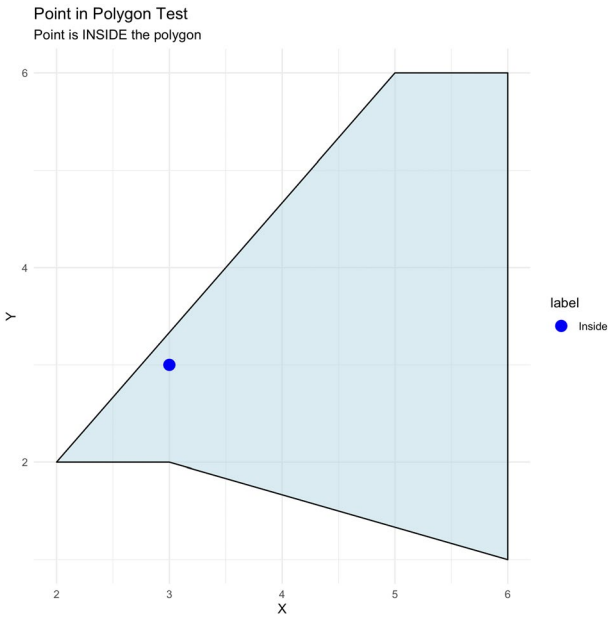

Fig. 4 Point-in-polygon test using Sunday's algorithm: the plot shows the polygon with the point marked. The point is inside (blue) the polygon. The subtitle indicates the result of the point-in-polygon test

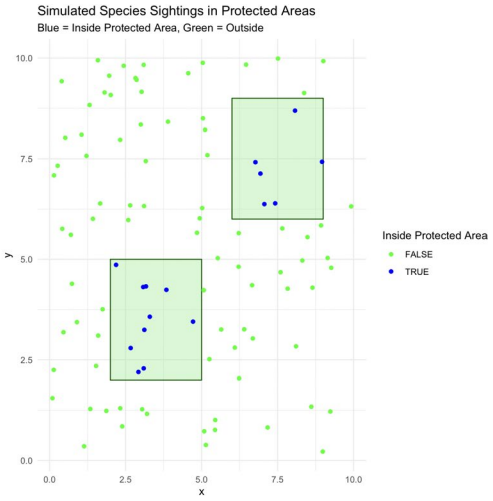

Fig. 5 Point-in-polygon analysis in ecological applications. Left: example UAV image supporting fine-scale spatial observation in a mixed forest. Right: simulated species sightings and protected areas used to demonstrate the Ray Casting point-in-polygon test (blue = inside protected areas; green = outside)

```
# Load required libraries
library(insideR)
library(ggplot2)
library(dplyr)

# Set seed for reproducibility
set.seed(1234)
```

```
# Define protected area polygons (as matrices)
protected_areas <- list(
  Reserve_A = matrix(c(2,2, 5,2, 5,5, 2,5), ncol = 2, byrow = TRUE),
  Reserve_B = matrix(c(6,6, 9,6, 9,9, 6,9), ncol = 2, byrow = TRUE)
)

# Simulate random species sightings
num_points <- 100
sightings <- data.frame(
  id = 1:num_points,
  x = runif(num_points, 0, 10),
  y = runif(num_points, 0, 10)
)
```

5.2 Method: point-in-polygon analysis

To determine whether a species sighting falls inside a protected area, I employed an explicit point-in-polygon test based on the Ray Casting algorithm. Unlike high-level spatial joins, this approach directly evaluates the geometric relationship between a point and a polygon, making the underlying logic fully transparent. Each species

sighting was tested against all protected area polygons and classified as occurring either inside or outside protected areas, as:

```
# Determine whether each sighting falls inside any protected area
sightings <- sightings %>%
  rowwise() %>%
  mutate(
    inside_protected_area = any(
      sapply(protected_areas, function(poly)
        point_in_polygon(c(x, y), poly)
      )
    )
  ) %>%
  ungroup()
```

This explicit implementation mirrors the logic used by standard GIS software while offering full control over the computational steps, making it particularly suitable for didactic purposes and methodological illustration.

5.3 Visualization

The final step concerns visualization of the results. Protected areas and species sightings were plotted using `ggplot2`, with sightings color-coded according to whether they fall inside or outside protected areas.

```
# Prepare polygon data for plotting
polygon_df <- bind_rows(
  lapply(names(protected_areas), function(name) {
    poly <- protected_areas[[name]]
    df <- as.data.frame(poly)
    colnames(df) <- c("x", "y")
    df$reserve <- name
    rbind(df, df[1, ])  # Close polygon
  })
)

# Create the plot
ggplot() +
  geom_polygon(
    data = polygon_df,
    aes(x = x, y = y, group = reserve),
    fill = "lightgreen",
    color = "darkgreen",
    alpha = 0.4
  ) +
  geom_point(
    data = sightings,
    aes(x = x, y = y, color = inside_protected_area),
    size = 1.5
  ) +
  scale_color_manual(values = c("FALSE" = "green", "TRUE" = "blue")) +
  theme_minimal() +
  labs(
    title = "Simulated Species Sightings in Protected Areas",
    subtitle = "Blue = Inside Protected Area, Green = Outside",
    color = "Inside Protected Area"
  )
```

The resulting map (Fig. 5) clearly distinguishes between sightings that fall within protected areas and those that do not. This type of visual summary is invaluable in conservation planning, allowing for rapid identification of gaps between species distributions and existing protected areas. Although the example presented here is simulated, it mirrors a common ecological workflow in which species occurrence records are assigned to habitat units, ecoregions, or protected areas through point-in-polygon operations. The same approach can be directly applied to real ecological datasets, such as species occurrences obtained from the Global Biodiversity Information Facility (GBIF) and protected area polygons obtained from the World Database on Protected Areas (WDPA). To facilitate these applications, a fully reproducible example showing how to combine GBIF occurrence records with protected area polygons is provided in the package GitHub repository (https://github.com/ducciorocchini/insideR/blob/main/examples/real_ecological_data_example.R).

5.4 Discussion

By performing point-in-polygon analysis with real-world data, ecologists can quickly determine which species are occurring within conservation zones, thus aiding conservation efforts (Dias et al. 2016). This approach can be expanded to examine environmental threats, assess biodiversity in protected areas, or guide conservation policy.

In ecology, understanding how species are distributed across landscapes is key to unraveling the complex dynamics of ecosystems. Spatial analysis tools such as point pattern analysis and point-in-polygon approaches are invaluable for ecologists, helping them identifying patterns in species distributions and environmental factors. Whether one is tracking rare species, studying habitat preferences, or managing landscapes for conservation, these methods provide the data-driven insights that guide decision-making.

Point pattern analysis, with its ability to classify distributions as random, clustered, or regular, offers a deeper understanding of how organisms interact with each other and their environment. This method allows ecologists to observe and quantify the organization of species in space, and its integration with other techniques such as Ripley's K-function or nearest-neighbor analysis enables researchers to test hypotheses about ecological processes (Goreaud and Pélissier 1999). By analyzing patterns at various spatial scales, point pattern analysis gives ecologists the flexibility to explore various hierarchical scales, enhancing the possibility to study systems starting from individual species' movements to large-scale habitat shifts (Zobel et al. 2023).

On the other hand, the point-in-polygon approach extends these insights by associating point-based ecological data with predefined landscape units (Roy and Behera 2002). Whether these units represent habitat types, land use classes, or conservation areas, the ability to link fine-scale observations (such as animal sightings or plant growth) with larger-scale polygons offers a wealth of opportunities for testing ecological theories and improving management strategies. This spatially explicit approach allows researchers to answer critical questions: What habitat types do species prefer? How are environmental conditions influencing the spread of an invasive species? What role do management zones play in biodiversity conservation?

By merging point pattern analysis and point-in-polygon techniques, ecologists can not only evaluate species-environment interactions (Liu and Gaines 2022) but also predict how changes in habitat configuration or environmental conditions may affect species distributions (Kolb and Diekmann 2004). This integrated approach allows for a more holistic understanding of ecosystems and a stronger foundation for ecological forecasting.

Moreover, the integration of modern tools like Geographic Information Systems (GIS), remote sensing, and high-resolution spatial data further enhances the power of these analyses (Neteler et al. 2012). As ecological data becomes increasingly available and detailed, the ability to incorporate point pattern analysis and point-in-polygon methods into larger modeling frameworks opens up new possibilities for research, conservation, and land management.

In future, as climate change, habitat fragmentation, and human-induced disturbances continue to alter ecosystems, these spatial analysis methods will remain crucial in guiding conservation efforts, policy decisions, and biodiversity protection strategies. The ongoing development of new statistical models and computational tools will only expand the potential applications of these approaches, making them even more central to the future of ecological research (Petrovskii and Petrovskaya 2012).

6 Conclusion

Point pattern and point-in-polygon analyses are more than just tools—they are the keys to unlock nature's intricate spatial puzzles. By weaving together these methods with ecological theory, we gain a deeper understanding of the patterns that shape life on Earth. Whether one is mapping the mysteries of wildlife movement or designing effective protected areas, these techniques help us explore, interpret, and conserve the world around us. As highlighted in Box 3, point-in-polygon methods underpin a wide range of practical ecological workflows—from assigning species occurrences to habitat units to evaluating pressures inside protected areas—making them essential for reproducible, spatially explicit inference.

In R, point-in-polygon operations are commonly performed using modern spatial frameworks such as the `sf` package (e.g., `st_within()`, `st_intersects()`), which rely on GEOS-based geometry engines and require spatial object construction and coordinate reference system management (Pebesma 2018; Pebesma and Bivand 2023). While these tools are powerful and robust for large-scale GIS workflows, they often involve substantial infrastructure and abstraction layers that can obscure the underlying geometric logic.

As seen at the end of Sect. 5.3, beyond the illustrative examples presented here, the proposed algorithms can be readily applied to species occurrence databases, vegetation-plot archives, UAV-derived observations, ecological monitoring networks, and protected area assessments (also refer to the code at: https://github.com/ducciorocchini/insideR/blob/main/examples/real_ecological_data_example.R). As ecological datasets continue to increase in volume and spatial resolution, transparent and reproducible point-in-polygon operations will remain an essential component of ecological data analysis.

The `insideR` package complements these approaches by providing a lightweight, coordinate-based implementation of both Ray Casting and Sunday's algorithms that operate directly on numeric coordinate matrices. This design allows users to perform point-in-polygon tests without importing shapefiles or constructing complex spatial objects—an advantage in educational settings, simulation studies, algorithmic research, or ecological workflows where GPS coordinates and polygon vertices are already known. By exposing the algorithmic foundations explicitly, `insideR` enhances transparency and reproducibility while remaining computationally efficient. Rather than replacing full GIS frameworks, `insideR` offers a focused, ecologically oriented solution that bridges computational geometry and

applied spatial ecology, reinforcing the integration of mathematical rigor with ecological practice.

Box 1. Glossary

- **Geometric partitioning** A method of dividing space into discrete regions based on a set of points or spatial features. In ecology, it can help delineate territories, habitats, or zones of influence among organisms or features.
- **Kernel density estimation** A non-parametric technique to estimate the intensity of point occurrences over space. It smooths point data into a continuous surface to highlight areas of high or low concentration, often used in species distribution and hotspot analysis.
- **Point pattern analysis** A suite of statistical methods for analyzing the spatial arrangement of points (e.g., trees, nests, sightings) to determine patterns such as clustering, randomness, or regular spacing.
- **Point-in-polygon** A computational method for determining whether a given point lies within a specific polygon. This is often used to assign spatial observations (e.g., animal sightings) to ecological zones or habitat types.
- **Ripley's *K*-function** A spatial statistic that quantifies point pattern clustering or dispersion at multiple spatial scales by measuring how many other points lie within a given distance of each point, compared to a random distribution.
- **Quadrat analysis** A method that divides the study area into equal-sized cells (quadrats) and counts the number of points within each. It helps assess the spatial distribution and identify deviations from randomness.
- **Ray Casting** An algorithm used in point-in-polygon testing that counts the number of times a ray projected from a point intersects the edges of a polygon. An odd count indicates the point is inside the polygon; even means it is outside.
- **Winding number** Another point-in-polygon method that computes how many times the polygon winds around the point. A non-zero winding number indicates the point is inside the polygon, while zero means it is outside.

Box 2. Geometric partitioning example: the 7-point, 3-line puzzle One famous example of geometric partitioning is the 7-point, 3-line puzzle, which challenges the solver to divide a set of 7 points into three distinct groups using exactly three straight lines. The puzzle is both a simple yet profound exercise in spatial reasoning, providing insight into the potential for straight lines to partition a plane into distinct regions. This section explores the puzzle and its underlying geometric principles, offering a foundation for understanding more complex spatial partitions used in ecological and conservation studies.

Box 2.1. The puzzle The puzzle involves 7 points, which must be divided into seven distinct groups, each containing exactly one point, by drawing exactly three straight lines. The key conditions are:

- Each line must divide the plane into distinct regions, with the goal being to partition the points into seven separate groups.
- No two lines can be parallel, and the lines may intersect.
- Each of the seven resulting regions must contain exactly one point.

The solution to the puzzle is achieved by carefully positioning the points and lines in such a way that each of the 7 points ends up in its own distinct region, with no two points sharing the same region.

Box 2.2. Mathematical insights The problem is rooted in Euclidean geometry, particularly the concept of dividing a plane with straight lines. A classic result in geometry tells us that the maximum number of regions R that can be formed by n straight lines in a plane is given by the formula:

$$R = \frac{n(n+1)}{2} + 1$$

For three lines ($n = 3$), this formula predicts that the maximum number of regions is:

$$R = \frac{3(3+1)}{2} + 1 = 7$$

Thus, three lines are sufficient to divide a plane into exactly 7 regions, aligning perfectly with the requirement of dividing 7 points into seven distinct groups.

This insight highlights the connection between geometric division and the number of points or regions involved. In ecological studies, similar principles can be applied when partitioning landscapes into different zones based on specific criteria such as land use, habitat type, or species range.

Box 2.3. Ecological applications of geometric partitioning While the 7-point, 3-line puzzle is primarily a mathematical curiosity, it mirrors several real-world applications in spatial ecology. For example:

- Habitat Zoning: Ecologists use geometric partitioning to divide ecosystems into distinct zones based on environmental features, species distribution, or conservation priorities. Just as the 7 points are divided into seven groups, a landscape may be segmented into regions with different conservation or management goals.
- Species Range Mapping: In ecology, species ranges are often delineated by polygonal boundaries (such as protected areas or habitats). The mathematical principles underlying the 7-point puzzle help ecologists understand how multiple zones or regions can be defined and analyzed.
- Land Use Planning: Planners often divide land into different use categories, such as residential, commercial, and agricultural. The ability to divide a geographic area into distinct regions using straight lines is a useful model for these tasks.

Just as the puzzle provides insight into how three lines can divide a plane into seven regions, ecologists can use similar techniques to define boundaries and assess spatial relationships within ecosystems.

Box 2.4. Visualization of the puzzle Figure 6 illustrates the solution to the 7-point, 3-line puzzle. In this diagram, 7 points are positioned on a plane, and three straight lines are drawn to partition the points into seven separate groups, with each group containing exactly one point.

In this diagram, the three lines successfully divide the plane into seven regions, each containing exactly one point. Each point is assigned to a separate group, illustrating the solution to the puzzle.

The 7-point, 3-line puzzle serves as an engaging and accessible example of how geometric principles can be used to partition a space. In the context of ecology, such geometric insights are invaluable for designing and analyzing habitat boundaries, species ranges, and land use zones. By leveraging the mathematical foundation of geometric partitioning, ecologists can improve their ability to manage and protect biodiversity through effective spatial planning and analysis.

Box 3. Ecological applications of the point-in-polygon algorithm Many ecological datasets involve coordinates of observations, species sightings, or sensor readings that need to be associated with habitat zones, protected areas, or biomes defined as polygons.

Box 3.1. Species distributions Ecologists often overlay species observation points onto polygonal habitat maps to determine presence within specific ecoregions. Using point-in-polygon tests, presence-absence matrices can be constructed, forming the basis of statistical species distribution models (SDMs).

Box 3.1.1. Protected area assessment To assess conservation effectiveness, researchers analyze whether deforestation, poaching events, or wildfire hotspots occur inside or outside protected areas (often defined by polygon shapefiles). This informs ecological integrity assessments and conservation planning.

Box 3.1.2. Marine and terrestrial sampling In both terrestrial and marine ecosystems, ecological surveys define polygonal plots or zones. PIP analysis helps associate sample points (e.g., water quality, biodiversity) with survey regions, ensuring spatial consistency and facilitating downstream analysis.

Box 3.1.3. Watershed and land cover analysis Watersheds, delineated as polygons, help attribute pollutant sources or hydrological events to specific basins. Similarly, land cover zones (e.g., forest, grassland) allow ecologists to quantify environmental metrics based on point locations collected from field surveys or remote sensing.

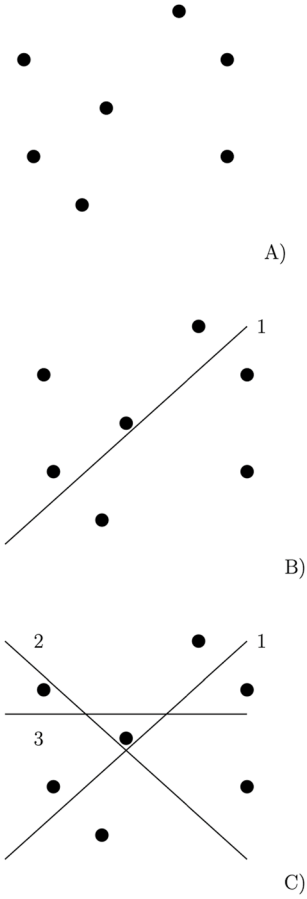

Fig. 6 Partitioning 7 points into 7 separate groups using 3 lines

Acknowledgments I acknowledge funding from the B3 project (Biodiversity Building Blocks for policy), funded by the European Union's Horizon Europe Research and Innovation Programme (Grant Agreement No. 101059592). The views and opinions expressed are those of the author only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

Author contributions D.R. conceived the study, developed the code, performed the analyses, interpreted the results, prepared the figures, and wrote the manuscript.

Funding Open access funding provided by Alma Mater Studiorum - Università di Bologna within the CRUI-CARE Agreement.

Data and code availability The `insideR` package is freely available in GitHub at: https://github.com/ducciorocchini/insideR/. All the code used to build the example is included in the package's `examples` folder. Additionally, the drone data used in the example are available on Zenodo (https://doi.org/10.5281/zenodo.17296860). The colors used in this paper are colorblind friendly. For additional information on colorblind issues, please refer to Rocchini et al. (2023, 2024).

Declarations

Conflict of interest The authors declare no conflict of interest.

References

Bacaro G, Maccherini S, Chiarucci A, Jentsch A, Rocchini D, Torri D et al (2015) Distributional patterns of endemic, native and alien species along a roadside elevation gradient in Tenerife, Canary Islands. Comm Ecol 16:223–234

Borruso G (2005) Network density estimation: analysis of point patterns over a network. In: International conference on computational science and its applications. Springer, Berlin Heidelberg, Berlin, Heidelberg, pp 126–132

Clobert J, Le Galliard JF, Cote J, Meylan S, Massot M (2009) Informed dispersal, heterogeneity in animal dispersal syndromes and the dynamics of spatially structured populations. Ecol Lett 12:197–209

Dias FS, Miller DL, Marques TA, Marcelino J, Caldeira MC, Cerdeira JO, Bugalho MN (2016) Conservation zones promote oak regeneration and shrub diversity in certified Mediterranean oak woodlands. Biol Conserv 195:226–234

Dobson AP, Rodriguez JP, Roberts WM, Wilcove DS (1997) Geographic distribution of endangered species in the United States. Science 275(5299):550–553

Enticott G, Ward K, Ashton A, Brunton L, Broughan J (2021) Mapping the geography of disease: a comparison of epidemiologists' and field-level experts' disease maps. Appl Geogr 126:102356

Gaston KJ, Jackson SF, Cantú-Salazar L, Cruz-Piñón G (2008) The ecological performance of protected areas. Annu Rev Ecol Evol Syst 39:93–113

Gatrell AC, Bailey TC, Diggle PJ, Rowlingson BS (1996) Spatial point pattern analysis and its application in geographical epidemiology. Trans Inst Br Geogr 21:256–274

Gilbert JR, Miller GL, Teng SH (1998) Geometric mesh partitioning: implementation and experiments. SIAM J Sci Comput 19:2091–2110

Goreaud F, Pélissier R (1999) On explicit formulas of edge effect correction for Ripley's K-function. J Veg Sci 10:433–438

Haines E (1994) Point in polygon strategies. In: Graphics gems, vol 4. pp 24–46

Hardy OJ, Sonké B (2004) Spatial pattern analysis of tree species distribution in a tropical rain forest of Cameroon: assessing the role of limited dispersal and niche differentiation. For Ecol Manag 197:191–202

Hormann K, Agathos A (2001) The point in polygon problem for arbitrary polygons. Comput Geom 20:131–144

Johnson LB (1990) Analyzing spatial and temporal phenomena using geographical information systems: a review of ecological applications. Landsc Ecol 4:31–43

Kolb A, Diekmann M (2004) Effects of environment, habitat configuration and forest continuity on the distribution of forest plant species. J Veg Sci 15:199–208

Korner-Nievergelt F, Sauter A, Atkinson PW, Guélat J, Kania W, Kéry M et al (2010) Improving the analysis of movement data from marked individuals through explicit estimation of observer heterogeneity. J Avian Biol 41:8–17

Liu OR, Gaines SD (2022) Environmental context dependency in species interactions. Proc Natl Acad Sci 119:e2118539119

Marini L, Bartomeus I, Rader R, Lami F (2019) Species–habitat networks: a tool to improve landscape management for conservation. J Appl Ecol 56:923–928
Nelson TA, Boots B (2008) Detecting spatial hot spots in landscape ecology. Ecography 31:556–566
Neteler M, Bowman MH, Landa M, Metz M (2012) GRASS GIS: a multi-purpose open source GIS. Environ Model Softw 31:124–130
Pebesma E (2018) Simple features for R: standardized support for spatial vector data. R J 10:439–446
Pebesma E, Bivand R (2023) Spatial data science: with applications in R. Chapman and Hall/CRC
Petrovskii S, Petrovskaya N (2012) Computational ecology as an emerging science. Interface Focus 2:241–254
Rocchini D, Nowosad J, D'Introno R, Chieffallo L, Bacaro G, Cazzolla Gatti R et al (2023) Scientific maps should reach everyone: the cblindplot R package to let colour blind people visualise spatial patterns. Eco Inform 76:102045
Rocchini D, Chieffallo L, Thouverai E, D'Introno R, Dagostin F, Donini E et al (2024) Under the mantra: 'Make use of colourblind friendly graphs'. Environmetrics 35:e2877
Roy PS, Behera MD (2002) Biodiversity assessment at landscape level. Trop Ecol 43:151–171
Self S, Overby A, Zgodic A, White D, McLain A, Dyckman C (2022) A generalization of Ripley's K function for the detection of spatial clustering in areal data. arXiv Preprint. arXiv:2204.10852
Staubach C, Schmid V, Knorr-Held L, Ziller M (2002) A Bayesian model for spatial wildlife disease prevalence data. Prev Vet Med 56:75–87
Torresani M, Rocchini D, Ceola G, de Vries JPR, Feilhauer H, MoudrÝ V et al (2024) Grassland vertical height heterogeneity predicts flower and bee diversity: an UAV photogrammetric approach. Sci Rep 14(1):809
Warton DI, Lyons M, Stoklosa J, Ives AR (2016) Three points to consider when choosing a LM or GLM test for count data. Methods Ecol Evol 7:882–890
Zobel M, Moora M, Pärtel M, Semchenko M, Tedersoo L, Opik M, Davison J (2023) The multiscale feedback theory of biodiversity. Trends Ecol Evol 38:171–182

Authors and Affiliations

Duccio Rocchini[1]

✉ Duccio Rocchini
duccio.rocchini@unibo.it

[1] BIOME Lab, Department of Biological, Geological and Environmental Sciences, Alma Mater Studiorum University of Bologna, via Irnerio 42, 40126 Bologna, Italy