ESTADÍSTICA PARA ESTUDIOS

DARÍO D. LARREA
LUCAS J. MINA
(COORDINADORES)

APUNTES DE CÁTEDRA · CIENCIAS EXACTAS Y NATURALES Y AGRIMENSURA

Estadística para estudios ecológicos

Un enfoque práctico utilizando R

Estadística para estudios ecológicos
Un enfoque práctico utilizando R

Estadística para estudios ecológicos : un enfoque práctico utilizando R / Darío Larrea ... [et al.] ; Editado por Irina Mariel Wandelow. - 1a ed. - Corrientes: Editorial de la Universidad Nacional del Nordeste EUDENE, 2025.
Libro digital, PDF - (Apuntes de Cátedra)

Archivo Digital: descarga
ISBN 978-950-656-267-0

1. Lenguaje. 2. Estadísticas. 3. Estudios. I. Larrea, Darío
II. Wandelow, Irina Mariel, ed.
CDD 407.2

Edición y corrección: Irina Wandelow
Diseño y diagramación: Julia Caplan
Corrección de maqueta: Facundo Alarcón

25 de Mayo 868 (CP 3400) Corrientes, Argentina.
Teléfono: (0379) 4425006
eudene@unne.edu.ar / www.eudene.unne.edu.ar

Estadística para estudios ecológicos
Un enfoque práctico utilizando R

Darío D. Larrea
Lucas J. Mina
Florencia M. Monti Areco
Néstor G. Valle
Matías I. Dufek
(autores)

apuntes de cátedra
Facultad de Ciencias Exactas y Naturales y Agrimensura

Índice

Prólogo

Los autores de esta obra llevan años utilizando el programa estadístico R como herramienta fundamental para analizar datos, interpretar resultados y obtener conclusiones válidas en sus investigaciones, lo que ha culminado en la publicación de numerosos trabajos en revistas científicas de prestigio. En *Estadística para estudios ecológicos. Un enfoque práctico utilizando R* comparten, a lo largo de los distintos capítulos, la aplicación de diversos paquetes del entorno R y detallan minuciosamente sus experiencias en el análisis de datos.

El libro comienza con una breve introducción a los conceptos básicos de R, incluyendo instrucciones para su instalación, la gestión de paquetes, la apertura de scripts y la organización de proyectos. En el capítulo 2, los autores presentan las etapas fundamentales del proceso de investigación científica, haciendo especial énfasis en el diseño experimental y la recolección de datos. El capítulo 3 describe las principales estructuras de datos en R, como vectores, distintos formatos de matrices, así como las formas de almacenamiento y visualización de datos, incluyendo la generación de gráficos.

El capítulo 4 aborda los distintos tipos de pruebas estadísticas, tanto paramétricas como no paramétricas, explicando cómo seleccionar la prueba más adecuada según las características de los datos y las hipótesis formuladas. El capítulo 5 está dedicado al análisis de la diversidad biológica a través de los índices de diversidad alfa y beta, mientras que el capítulo 6 introduce otras métricas no neutrales para evaluar la biodiversidad. El capítulo 7 presenta las curvas de Whittaker y el análisis de escalamiento multidimensional no métrico (NMDS), utilizados para comprender la estructura de los ensambles biológicos. Por su parte, el capítulo 8 examina distintos análisis orientados a evaluar la influencia de variables ambientales en la estructura de dichos ensambles. Finalmente, el capítulo 9 ofrece una introducción al modelado de nicho ecológico (MNE) en R, como herramienta para entender y predecir la distribución actual de las especies. A lo largo de estos capítulos, los autores incluyen ejemplos prácticos basados en sus propias investigaciones, incluyendo los scripts utilizados en el entorno R para realizar los análisis, aplicar pruebas e interpretar resultados.

En los dos siguientes capítulos destacan la flexibilidad y potencia de R en el ámbito de la estadística ecológica, y recopilan en un solo lugar todos los scripts mencionados entre los capítulos 4 y 9.

Este libro, fruto de un trabajo colaborativo, no sólo comparte el conocimiento y la experiencia de los autores en el uso de R, sino que también brinda a los lectores una herramienta flexible y accesible para el análisis estadístico en estudios ecológicos. Los conocimientos y experiencias aquí compartidos resultarán de gran utilidad para estudiantes, investigadores y profesionales interesados en profundizar en el uso aplicado de R.

Miryam Pieri Damborsky

Capítulo 1
Introducción al lenguaje R

Darío D. Larrea y Lucas J. Mina

R es un entorno de software y un lenguaje de programación diseñado específicamente para el análisis estadístico y gráfico. Fue desarrollado inicialmente por Ross Ihaka y Robert Gentleman en la Universidad de Auckland, Nueva Zelanda, a principios de la década de 1990.

El desarrollo de R tuvo como objetivo principal proporcionar una herramienta de análisis estadístico de libre acceso y código abierto, dirigida a estudiantes, académicos y profesionales de diversas disciplinas. Desde un enfoque general, R puede entenderse como una adaptación del lenguaje de programación S creado en los laboratorios Bell a finales de la década de 1970, con la particularidad que R es un software libre, lo que significa que está disponible para su uso de manera gratuita y sin restricciones. Actualmente, el desarrollo de R está a cargo del *R Development Core Team*.

Es importante destacar que, a diferencia de otros programas destinados a los análisis estadísticos de datos en ecología, R es un lenguaje de programación interpretado. Esto significa que los comandos se ejecutan línea por línea, a medida que se escriben, sin la necesidad de realizar una previa compilación del código, lo que ofrece mayor versatilidad y facilita la escritura de código y su comprensión para los usuarios.

Una de las características distintivas de R es su extensibilidad. El sistema central de R proporciona funcionalidades básicas para el análisis estadístico, pero también permite a los usuarios y desarrolladores crear sus propias funciones y paquetes para ampliar su funcionalidad. Esto ha llevado a una gran cantidad de paquetes disponibles para una amplia gama de aplicaciones, desde el análisis de datos espaciales y la genómica hasta el aprendizaje automático. Al presente, R se ha convertido en una herramienta fundamental en campos como la ciencia de datos, la investigación académica, la bioinformática, la ecología y muchos otros, gracias a su flexibilidad, poder y la activa comunidad de usuarios y desarrolladores.

RStudio

RStudio es un entorno de desarrollo integrado –abreviado como IDE por sus siglas en inglés (Integrated Development Environment)– y diseñado específicamente para trabajar con R. Proporciona una interfaz gráfica intuitiva que facilita la escritura, el desarrollo, la depuración y la ejecución de códigos en R. También permite la visualización de resultados y la gestión de proyectos.

La interfaz RStudio ofrece un editor de código integrado con resaltado de sintaxis, autocompletado de código, sangría automática y otras características que facilita la construcción del código. Además, cuenta con un depurador de código integrado que permite detectar y corregir errores de forma eficiente. Asimismo, RStudio permite explorar y administrar los archivos y directorios del sistema de archivos desde dentro del entorno de desarrollo, lo que facilita la gestión de proyectos y la organización de archivos de código y datos.

Conceptos básicos sobre R

Para aprovechar al máximo las capacidades del lenguaje de programación R, es esencial comprender algunos conceptos básicos que estructuran su funcionamiento. En esta sección se abordarán aspectos clave como la sensibilidad entre mayúsculas y minúsculas, la creación y ejecución de scripts, el uso de paquetes, la estructura de datos básica (vectores) y la importancia de los comentarios y diagnósticos en la escritura de código. Además, se destacarán herramientas prácticas ofrecidas por RStudio, como la gestión de proyectos y la ejecución eficiente de scripts, que facilitan la organización y reproducibilidad de los análisis. Estos conceptos sentarán las bases para un manejo adecuado de R, permitiendo desarrollar análisis robustos y bien documentados.

Sensibilidad entre mayúsculas y minúsculas. R es un lenguaje de programación *case sensitive*, es decir en la sintaxis del código se puede diferenciar entre mayúsculas y minúsculas. Por esta razón, es importante tener precaución al construir scripts, ya que, para R, no es lo mismo escribir una variable como «Hormigas» u «hormigas». En este ejemplo, las dos palabras serían consideradas como dos cadenas de texto diferentes.

Script en R. Es un archivo que contiene una serie de comandos R que pueden ser ejecutados secuencialmente. Los scripts son una forma eficiente de organizar y reproducir análisis de datos, ya que permiten documentar y automatizar tareas.

Paquetes en R (*packages* en inglés). Son extensiones de funcionalidad que amplían las capacidades del lenguaje base. Los paquetes pueden contener funciones, conjuntos de datos y documentación. Para utilizar un paquete en R, primero debe ser instalado en su sistema y luego cargado en la sesión de R con el comando library().

Ayuda en R. Todos los paquetes de R tienen asociados manuales que facilitan la escritura, ejecución e interpretación de resultados. Para acceder a estas ayudas, están disponibles

tres comandos: ?x, help(x) y help.search(x). Por ejemplo, si interesa tener más información sobre la función *vegdist*, se reemplaza el nombre de la función por las x, obteniendo los siguientes comandos: ?vegdist, help(vegdist) y help.search(vegdist).

Vectores en R. Son la estructura de datos más básica y fundamental en R. Son objetos que pueden contener datos numéricos, cadena de caracteres o datos lógicos, entre otros. Para realizar la asignación de un vector en R, se emplea el operador de asignación <-. Es recomendable, para simplificar la construcción de scripts, aprovechar el atajo de teclado Alt+- (signo menos) en RStudio. Generalmente, se evita usar = para asignaciones. Aunque el uso de = no arroja un mensaje de error, podría no realizar la asignación esperada y causar errores lógicos.

Serie de comandos. R permite escribir múltiples comandos en una misma línea. Para ello, se utiliza el ; (punto y coma) como marcador final de comando, lo que permite separar cada instrucción y facilita la escritura de scripts más concisos.

Expresiones en R. Las expresiones se construyen mediante una secuencia de comandos encerrados entre llaves {}. Estas expresiones son fundamentales en la construcción de funciones, bucles y otros elementos de la programación en R.

Caracteres pares. En R existen caracteres que siempre se usan de a pares, como las comillas (""), los paréntesis () y las llaves {}. Estos símbolos son usados para representar cadenas de caracteres o aplicar operadores sobre más de un carácter. En RStudio siempre que se escriba uno de estos caracteres se insertará automáticamente el otro, lo que facilita la construcción de los scripts.

Diagnóstico de script. El editor de script del RStudio revisa constantemente la arquitectura del código y avisa cualquier problema que encuentre en el mismo. Cuando RStudio detecta un error de sintaxis, muestra, al lado del número de línea del código, una x roja y una línea roja sobre el código. RStudio también informa sobre otros posibles problemas en el código por medio de un símbolo de admiración dentro de un triángulo amarillo.

Comentarios en RStudio. El símbolo # se utiliza para hacer comentarios en el editor de script. En RStudio, cuando se agrega # en el editor de comandos, todo lo que esté después de # será tratado como un comentario y no será ejecutado como código. Esto es útil para documentar y organizar el código, explicar lo que hace cada parte del mismo o hacer anotaciones para uno mismo o para otros que lean el código en el futuro. Los comentarios son ignorados por R cuando ejecuta el código, por lo que no afectan el resultado del análisis.

Proyectos. RStudio proporciona una excelente herramienta para organizar datos de entrada, scripts, resultados y gráficos, asociándolos a un proyecto. Esta metodología de gestión de toda la información relacionada con un estudio no sólo garantiza la seguridad de los datos y los resultados del análisis, sino que también facilita compartir esta información con otros miembros del equipo. Por este motivo, se recomienda enérgicamente realizar todos los análisis de un estudio asociados a un proyecto de R (para más detalles sobre este tema, consultar el capítulo 3).

Ejecutar el script. La mejor forma de ejecutar un script es corriendo línea por línea, así se podrá identificar en qué línea se encuentran exactamente los errores que surjan y corregirlos rápidamente. Para ello, es recomendable utilizar el atajo de teclado Ctrl+Enter. Aunque, si se está seguro de que todas las líneas del script funcionan correctamente, se puede usar el comando Ctrl+Shift+S para correr el script completo.

Convenciones de nomenclatura

En la programación, los espacios son un carácter reservado, lo que significa que no se puede usar en los nombres de las variables. Por esta razón, las convenciones de nomenclatura son importantes en programación para escribir un código legible y mantener consistencia en el estilo de escritura. En R, estas convenciones también son relevantes para crear nombres de variables claros y comprensibles. A continuación, se describen las convenciones de nomenclatura más usadas:

Camel Case. Se utiliza comúnmente para nombrar objetos o funciones. Por ejemplo, se podría tener una variable llamada estoEsUnaVariable.
Pascal Case. Al igual que en Camel Case, Pascal Case se usa para nombrar funciones o clases. Por ejemplo, se podría tener una función llamada EstoEsUnaFuncion.
Snake Case. Se utiliza principalmente para nombrar variables y columnas en data frames. Por ejemplo, se podría tener una variable llamada esto_es_una_variable.
Kebab Case. Aunque menos común, en R también se puede encontrar el uso de Kebab Case, especialmente para nombrar archivos o directorios. Por ejemplo, se podría tener un archivo llamado esto-es-una-variable.R.

El uso de estas convenciones no es obligatorio en R, pero es altamente recomendado para mejorar la legibilidad del código, tanto para otros colaboradores como para uno mismo en el futuro. No existe un estándar estricto en cuanto a la elección de una convención, ya que depende de las preferencias del equipo o del programador individual. Sin embargo, seguir una convención consistente en todo el código facilita su comprensión y mantenimiento.

Instalación de paquetes

Un paquete de R es una extensión que amplía las funcionalidades del lenguaje base, incorporando nuevas funciones, métodos de análisis, conjuntos de datos y herramientas adicionales. La mayoría de estos paquetes están disponibles en el repositorio CRAN (Comprehensive R Archive Network). Para instalarlos, los usuarios pueden acceder a la pestaña «Packages» en RStudio o ejecutar directamente la función correspondiente en la línea de comandos de R:

```
install.packages()
```

Pero hay algunos casos donde se pueden encontrar por fuera de este repositorio, en GitHub, por ejemplo, en cuyo caso se debe instalarlos valiéndose de la ayuda de una librería llamada «Remotes». Por ejemplo, para instalar el paquete datosEcoR –donde se encuentran todos los datos usados en los ejemplos de este libro, así que es muy recomendable si se desea experimentar con los scripts por su cuenta–, está alojado en GitHub:

```
install.packages("remotes")     # Se instala remotes primero

       remotes::install_github("lucasjmina/datosEcoR")
```

Primeros pasos en el lenguaje R

Lo primero que se debe hacer antes de empezar a usar R es instalarlo, sobre lo cual se hablará a continuación. Se introducirá la interfaz gráfica RStudio, que será el entorno de desarrollo o IDE, y se demostrará cómo instalar librerías.

Paso 1. Instalación de R
Para iniciar en la programación en R, se deberá descargar el programa del repositorio *The Comprehensive R Archive Network* (CRAN) e instalarlo en la computadora accediendo a la página web de CRAN (https://cran.r-project.org). En esta página se encontrarán todas las versiones disponibles para los sistemas operativos Linux, macOS y Windows.

Paso 2. Instalación de RStudio
Aunque es posible programar utilizando directamente la interfaz de R instalada en el paso anterior, es recomendable trabajar en la construcción y ejecución del código utilizando RStudio. Esta plataforma proporciona numerosas ventajas adicionales que hacen que sea una opción indispensable para la programación en R. Además de facilitar la detección de posibles errores en el código, RStudio ofrece características como resaltado de sintaxis, autocompletado de código, administración de proyectos y entorno integrado para la exploración de datos y la visualización de resultados. Su interfaz intuitiva y su amplia gama de utilidades hacen que sea una herramienta invaluable en el proceso de desarrollo de código en R, mejorando significativamente la eficiencia y la productividad del usuario. Asimismo, la comunidad de RStudio es activa y ofrece una gran cantidad de recursos, tutoriales y paquetes adicionales que pueden enriquecer aún más la experiencia de programación en R. RStudio se puede descargar e instalar desde https://posit.co/downloads/.

La primera vez que se ejecute RStudio se verán en la interfaz tres paneles principales:

- Consola: es donde se interactúa directamente con R. Se pueden escribir comandos en la consola y ver la salida inmediata de esos comandos. Es útil para probar rápidamente pequeñas porciones de código, realizar cálculos y explorar datos.
- Entorno: el ambiente muestra los objetos (como variables, funciones, paquetes cargados, etc.) que están actualmente disponibles en la sesión de R. Se puede ver una lista de estos objetos en la pestaña «Environment» en RStudio. Esta función permite tener una visión general de los datos y objetos que se está utilizando en la sesión de R. En este panel también se puede encontrar la pestaña de historial, que muestra un registro de los comandos que se ejecutaron en la consola durante tu sesión actual. Asimismo, se accede al historial desde la pestaña «History» en RStudio. Esto puede ser útil para recordar comandos que se ejecutaron previamente o para volver a ejecutar comandos anteriores.
- Salida: se refiere a los resultados de los comandos que se ejecutan en R. Esto puede incluir archivos generados, gráficos, etc. La salida se puede observar en la pestaña «Plots» como gráficos y «Files» para archivos generados. Existe otra información de salida como mensajes de error o valores de índices que se muestran típicamente en la consola.

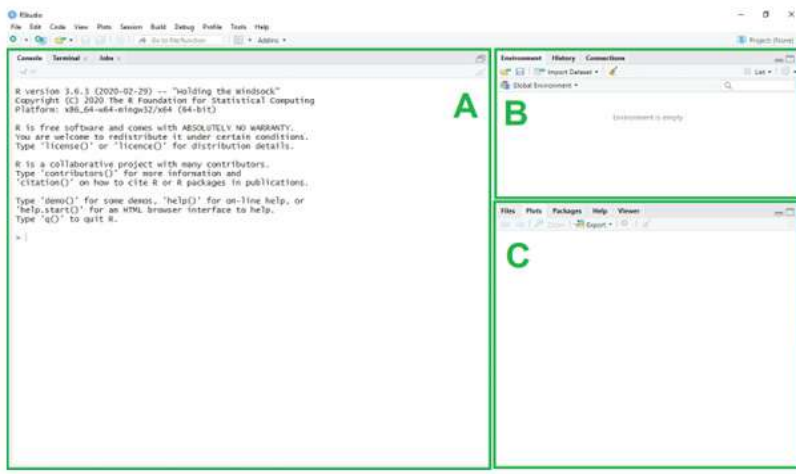

Figura 1. Representación de la interfaz de RStudio. (A) Ventana de consola. (B) Ventana de entorno, historial y conexión. (C) Ventana de archivos, visualización de gráficos, paquetes y ayuda.

Paso 3. Creación de un script

Si se presionan las teclas Ctrl+Shift+N, se abrirá un cuarto panel de trabajo en RStudio, el editor de texto. En esta área es donde se puede escribir, editar y guardar los scripts de R. Permite escribir y organizar el código de manera más estructurada que en la consola.

También proporciona características como resaltado de sintaxis, autocompletado de código y sugerencias de función, lo que facilita la escritura y la lectura del código.

RStudio no guarda automáticamente todos los scripts. Por esta razón, es recomendable que, antes de cerrar el programa, se asegure guardar las modificaciones que se realizaron al código para no perderlas. Para llevar a cabo este guardado de la forma más meticulosa posible, es recomendable usar la función de proyectos de RStudio.

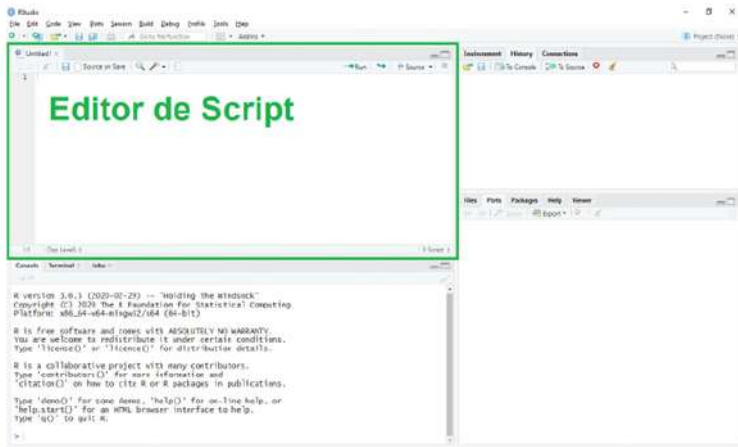

Figura 2. Representación de la interfaz de RStudio con la ventana del Editor de Scripts en actividad. En esta ventana se escriben las líneas de comandos que forman el código a ejecutar.

Paso 4. Instalación del paquete Tidyverse

Tidyverse es una colección de paquetes diseñados para trabajar de manera eficiente y coherente con datos. Estos paquetes están diseñados para abordar diferentes aspectos del análisis de datos, incluida la manipulación, visualización y modelado de datos. El Tidyverse promueve un enfoque coherente y estructurado para trabajar con datos en R, lo que facilita la escritura de código limpio, legible y reproducible.

Los principales paquetes que forman parte del Tidyverse incluyen:

a) ggplot2: para la creación de gráficos y visualizaciones de datos de alta calidad y flexibilidad.

b) dplyr: para la manipulación de datos, incluyendo filtrado, selección, transformación y agregación de datos.

c) tidyr: para la manipulación de la estructura de los datos, incluyendo la conversión entre formatos anchos y largos, y la limpieza de datos desordenados.

d) readr: para la importación de datos desde diferentes formatos de archivos, como CSV, Excel y archivos de texto plano.

e) purrr: para la programación funcional en R, incluyendo la iteración sobre listas y vectores, y la aplicación de funciones a múltiples elementos de datos.

f) tibble: para la creación y manipulación de marcos de datos tibble, una forma mejorada y moderna de trabajar con datos tabulares en R.

A continuación, se presenta la línea de código que se tiene que escribir y ejecutar en en RStudio para realizar la instalación de Tidyverse en R:

```
install.packages("tidyverse")
```

Posteriormente, se escribe y ejecuta la función library() para cargar el paquete y poder usar sus funciones:

```
library(tidyverse)
```

Estos pasos instalarán y cargarán el paquete Tidyverse en la sesión de R, lo que permitirá comenzar a trabajar con las herramientas y funciones que ofrece este conjunto de paquetes. En el capítulo 3 se detallará un poco más sobre este paquete con algunos ejemplos de su uso.

Paso 5. Creación de un proyecto de RStudio
Como se mencionó anteriormente, crear un proyecto en RStudio es una forma conveniente de organizar archivos, datos y scripts para un proyecto específico. Aquí, una guía detallada para crear un proyecto en RStudio:

1. En la barra de menú superior de RStudio hacer clic en «File» (Archivo).

2. En el menú desplegable que aparece al hacer clic en «File», seleccionar «New Project» (Nuevo Proyecto).

3. En el cuadro de diálogo que se abrirá, elegir el tipo de proyecto que se desea crear. Se puede seleccionar entre «New Directory» (Nuevo Directorio), «Existing Directory» (Directorio Existente) o «Version Control» (Control de Versiones). Para crear un nuevo proyecto desde cero, elegir «New Directory» y luego hacer clic en «New Project».

4. En el nuevo cuadro de diálogo que se abrirá, seleccionar la ubicación en la computadora donde se desea guardar el proyecto y darle un nombre. Es recomendable elegir una ubicación específica y un nombre descriptivo para el proyecto.

5. En la pestaña «Create Project» (Crear Proyecto) hacer clic para crear el proyecto en RStudio, una vez seleccionada la ubicación y el nombre del proyecto.

6. En el proyecto creado, RStudio abrirá una nueva sesión con el proyecto cargado.

7. En el panel inferior derecho de RStudio, una pestaña llamada «Files» (Archivos), donde se pueden explorar los archivos y carpetas asociados al proyecto. También en la pestaña «Environment» (Ambiente) se muestran los objetos que están actualmente disponibles en la sesión de R dentro del proyecto.

8. En una carpeta guardar todas las matrices y datos que se necesitarán para los análisis. Para crear una nueva carpeta dentro del proyecto en RStudio, seguir los pasos a continuación:

    a. En el panel de salida de RStudio, donde se muestra la pestaña «Files» (Archivos), hacer clic en la opción «New Folder» (Nueva Carpeta).

    b. En el cuadro de diálogo que se abrirá, se solicitará un nombre para la nueva carpeta (se recomienda nombrar a la carpeta que contiene las matrices como data).

    c. Luego de nombrar la carpeta, presionar «Ok» para confirmar el nombre.

Estructura de las carpetas de un proyecto

Cuando se ejecuta un análisis en R, se necesita contar con datos de entrada (*input data*). Estos datos pueden provenir de una variedad de fuentes, como archivos CSV, bases de datos, hojas de cálculo Excel, API web, o incluso generados internamente en R mediante funciones o simulaciones. A partir de esta información, los distintos análisis proporcionarán datos de salida (*output data*) en forma de resúmenes estadísticos, gráficos, tablas de resultados, archivos de texto o cualquier otra información derivada del proceso de análisis.

Si no se mantiene una organización adecuada en la gestión y estructuración de la información, los archivos dentro del proyecto pueden volverse caóticos y difíciles de manejar. Es por ello que es fundamental establecer un sistema claro y ordenado para almacenar, manipular y gestionar tanto los datos de entrada como los datos de salida en los proyectos de R.

A continuación, se presenta una forma –aunque no la única– de organizar la información en carpetas dentro del proyecto para mantener una estructura ordenada y fácil de manejar.

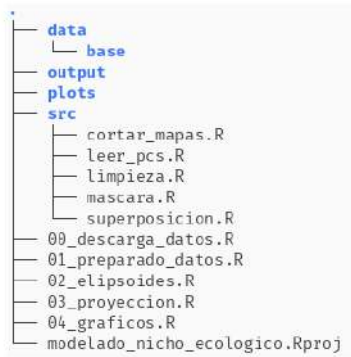

Figura 3. Estructura de carpetas y archivos en un proyecto de R. La organización incluye directorios para datos base, salidas (output), gráficos (plots) y código fuente (src) con scripts específicos para diferentes tareas. Además, se muestran los archivos principales del proyecto y el archivo de proyecto de R (Rproj).

La figura 3 detalla la estructura usada en un análisis de nicho ecológico, en la raíz del proyecto –donde se encuentra el archivo Rproj– es donde se colocarán todos los scripts utilizados para el análisis, ordenándolos con números de acuerdo con el orden en el que se deben ejecutar. En la carpeta «data» se encontrarán todas las matrices listas para los análisis, luego de su limpieza y preparado, guardando en la subcarpeta «base» las matrices *crudas*. Las carpetas «output» y «plots» son las carpetas de salida, donde se guardarán, en el caso de que existan, los archivos arrojados por los análisis y los gráficos, respectivamente.

Recapitulando

Este primer capítulo proporcionó una visión exhaustiva del lenguaje de programación R y su entorno de desarrollo, RStudio. Brindó una base sólida al ofrecer una introducción clara y accesible a R, un lenguaje fundamental en el análisis estadístico y la ciencia de datos.

La explicación detallada sobre la instalación de R y RStudio, así como los conceptos básicos del lenguaje, sientan las bases para una comprensión profunda de la programación en R. Además, al enfatizar el uso de RStudio para facilitar la escritura y depuración del código, y al recomendar el uso de convenciones de nomenclatura y una estructura ordenada de proyectos, se resalta la importancia de una práctica organizada y meticulosa en la programación.

En resumen, este enfoque proporciona herramientas esenciales para mejorar la eficiencia y la reproducibilidad en el análisis de datos.

Capítulo 2
Diseño experimental y toma de datos

Darío D. Larrea, Matías I. Dufek y Florencia M. Monti Areco

En el proceso de investigación, el diseño experimental y la recopilación de datos son etapas críticas que establecen las bases para la obtención de resultados significativos y confiables. En este capítulo se exploran los fundamentos del diseño experimental –desde la conceptualización de variables hasta la planificación de estrategias de recolección de datos– y se aborda cómo diseñar experimentos que permitan identificar relaciones causales entre variables, así como la implementación de técnicas efectivas para la recopilación precisa de datos. Además, servirá como guía y proporcionará los conocimientos y las herramientas necesarias para estructurar investigaciones sólidas y rigurosas en diversas disciplinas científicas.

Diseño experimental[1]

En el ámbito de la investigación, la experimentación sirve para identificar el efecto de un factor con significancia estadística en una respuesta. En este contexto, un experimento implica la selección de los niveles (valores) de una o más variables (factores) de entrada o independientes, y la observación de los valores de las variables de salida o dependientes. El objetivo es comprender la relación entre estas variables para optimizar el proceso subyacente. En síntesis, un diseño experimental consiste en la selección de variables y los niveles de estas variables.

1. Existen diversos tipos de diseño: 1) Preexperimental, aquellos en los que no se puede controlar completamente la variable independiente; 2) Experimental *sensu stricto*, aquellos en los que el investigador tiene un control total sobre la variable independiente; 3) Cuasiexperimental, aquellos en los que el investigador tiene un control parcial sobre la variable independiente; 4) Caso único y múltiple, aquellos que se utilizan en áreas como la salud pública, la educación y la psicología.

Para lograr un óptimo diseño experimental, se debe tener en cuenta tres cuestiones importantes: qué factores estudiar, cómo seleccionar los niveles de estos factores y qué variable dependiente seleccionar. A menudo, se tiene la libertad de seleccionar los niveles de los factores antes de la recopilación de datos, lo que aumenta la eficiencia del estudio.

Variables y niveles

En el diseño experimental, las *variables* son elementos fundamentales que se estudian y analizan para comprender fenómenos, procesos y relaciones dentro de un sistema en particular. Estas pueden variar en su naturaleza y pueden ser cuantitativas o cualitativas. La correcta identificación y distinción entre diferentes tipos de variables es crucial para el diseño y la interpretación adecuada de los estudios científicos. En este sentido, las variables dependientes e independientes son componentes esenciales en el diseño experimental.

Variable independiente[2]**.** Es aquella que se manipula o controla deliberadamente por el investigador. Es el factor que se considera que tiene un efecto sobre la variable dependiente. Es decir, la variable independiente se utiliza para predecir o explicar cambios en la variable dependiente, como el nivel de contaminación en un área, la temperatura, la presencia de depredadores o cualquier otro factor que se sospeche influya en ciertas características del ecosistema estudiado. Por ejemplo, si se está estudiando el crecimiento de una población de plantas, la cantidad de agua podría ser una variable independiente que se modifica para observar su efecto en el crecimiento de las plantas.
Variable dependiente[3]**.** Es aquella que se observa y mide en respuesta a los cambios en la variable independiente. Esta variable es la que se considera el resultado, la que se espera que cambie en respuesta a la manipulación de la variable independiente, como ser la tasa de crecimiento de una población, la diversidad de especies, la densidad poblacional de una especie, entre otros. Siguiendo el ejemplo anterior, si la cantidad de agua es la variable independiente, entonces el crecimiento de las plantas sería la variable dependiente, ya que se espera que cambie en respuesta a diferentes cantidades de agua proporcionada.

Los *niveles* de una variable, por otro lado, representan los posibles valores que puede asumir la variable formando un continuo ordenado. El nivel o escala de medición de una variable determina sus propiedades, así como las operaciones matemáticas y los procedimientos estadísticos que pueden aplicarse a dicha variable.

---

2. Según el campo de aplicación, esta variable presenta distintos nombres, como rendimiento, variable de respuesta y medida de desempeño.

3. Según el campo de aplicación, esta variable puede denominarse regresores, variable explicativa o variable predictora.

Existen cuatro escalas principales de medición: nominal, ordinal, de intervalo y de razón. Las dos primeras –nominal y ordinal– se consideran escalas categóricas, mientras que las dos últimas –intervalo y razón– son escalas numéricas. Las escalas categóricas se utilizan para medir variables cualitativas, mientras que las numéricas son adecuadas para variables cuantitativas. Estas escalas determinan el tipo de análisis de datos y pruebas de hipótesis teóricas que se pueden aplicar a una variable en particular.

Escala nominal. Representa la forma más básica de medición. En esta escala, las unidades de estudio se agrupan en categorías según una o más características distintivas y observables. Cada categoría recibe una nomenclatura. Es importante destacar que los nombres utilizados en esta escala no necesariamente corresponden a términos alfabéticos o numéricos, ya que también pueden ser números. En el contexto de las escalas nominales, estos números simplemente sirven como etiquetas o identificadores, y no tienen un significado cuantitativo. En otras palabras, los números en una escala nominal se utilizan para clasificar, no para realizar operaciones aritméticas. Los números asignados a cada categoría no reflejan un orden o jerarquía, sino que actúan como códigos de identificación. En una escala nominal, las observaciones no pueden ser ordenadas de menor a mayor, ya que todas las categorías son igualmente válidas y representan sólo diferentes valores que asume la variable medida. Por lo tanto, la magnitud de los números no es relevante en la medición nominal, lo importante es determinar si dos observaciones son idénticas o no. En resumen, la escala nominal se centra en la clasificación de datos, sin considerar la magnitud de las diferencias entre las categorías. En la escala nominal se clasifican las variables en dos o más categorías distintas. Cuando una variable nominal consta de únicamente dos categorías, se le llama *variable dicotómica*. Por ejemplo, la presencia o ausencia de determinado carácter (presencia o ausencia de plumas), el sexo (macho o hembra) son variables dicotómicas comunes. Por otro lado, cuando una variable nominal cuenta con tres o más categorías, se conoce como *variable politómica*. Ejemplos de este tipo de variables incluyen color de ojos, casta en insectos sociales, entre otros. En la práctica, los datos recopilados con este tipo de escalas generalmente consisten en conteos de frecuencias que muestran el número de eventos en cada categoría de la variable estudiada. A partir de estos datos, sólo se pueden calcular proporciones, porcentajes y razones.

Escala ordinal. Las observaciones pueden ser ordenadas jerárquicamente en relación con la característica que se está evaluando. Cuando se utilizan números, su magnitud representa el orden relativo de ese rasgo (nivel) dentro de la variable. Sin embargo, este valor no se puede emplear para ninguna estimación matemática. Es decir que las escalas ordinales –al igual que la nominal– únicamente permiten el cálculo de proporciones, porcentajes y razones, sin la posibilidad de realizar operaciones aritméticas más complejas. En este tipo de escalas, cada medición debe pertenecer obligatoriamente a uno de los niveles (categorías) de la variable. En términos generales, se puede medir en una escala ordinal una amplia variedad de atributos como el nivel de organización de un organismo (célula, tejido, sistema de órganos), el estado de desarrollo de un organismo, entre otros.

Escala de intervalo. Son más precisas ya que, además de establecer un orden o jerarquía entre las categorías, los valores indican intervalos uniformes en la medición. En este tipo de escala se puede conocer exactamente la diferencia entre los objetos medidos, de manera que cada objeto se puede representar con un valor, y la diferencia entre los valores asignados refleja la diferencia entre los objetos. Las variables medidas en escalas de intervalo proporcionan información sobre la magnitud de lo que se está midiendo. En esta escala, el cero no representa la ausencia de la característica medida, sino que es un punto de referencia arbitrario[4] a partir del cual se establecen intervalos de igual magnitud para construir la escala. Algunos ejemplos de variables con escala de intervalo son la temperatura, el tiempo, entre otras. Las escalas de intervalo admiten el cálculo de proporciones, porcentajes y razones, así como la estimación de estadísticas como la media, la mediana, la moda, el rango y la desviación estándar.

Escala de razón. El cero es real y absoluto, no arbitrario. Representa la ausencia completa de la característica en cuestión. Los números en esta escala pueden compararse como proporciones, permitiendo indicar cuántas veces es más grande un objeto que otro. Variables como abundancia de individuos, riqueza de especies, peso, longitud y masa se miden en escalas de razón. Desde una perspectiva matemática, las escalas de razón permiten realizar todo tipo de operaciones aritméticas, obtener razones y proporciones, así como estimar diversos estadísticos.

Es fundamental conocer todos los niveles de todas las variables de una investigación, ya que esto determina el tipo de análisis estadístico adecuado. En este sentido, correlacionar dos variables de intervalo requiere pruebas estadísticas diferentes a las empleadas para correlacionar dos variables ordinales.

Según las características que presentan los *valores de los niveles* de estas variables, se pueden agrupar en cualitativas (nominales y ordinales) y cuantitativas (de intervalo y razón).

Variable cualitativa. También conocidas como variables categóricas, son aquellas que representan características no numéricas o cualidades de los elementos de estudio. Estas variables no se pueden medir directamente con números y generalmente se expresan mediante etiquetas o categorías.

Variable cuantitativa. Son aquellas que representan cantidades numéricas o medidas de los elementos de estudio. Estas variables se pueden medir y expresar con números, lo

4. Los puntos de referencia suelen establecerse por medio de *convenciones* (acuerdos de la comunidad científica) para la medición. Es decir que estas escalas son un acuerdo al que ha llegado la comunidad científica para definir estándares de registro de datos. Por esta razón, pueden variar según el sistema de medición utilizado (por ejemplo, la temperatura puede ser medida en grados Fahrenheit y Centígrados).

que implica magnitudes numéricas y operaciones aritméticas significativas. Las variables cuantitativas se dividen en dos subtipos: discretas y continuas. Las variables discretas y continuas son fundamentales en el análisis de datos y la estadística.

- **Variables discretas.** Representan valores aislados y contables, generalmente enteros, que no pueden subdividirse en partes más pequeñas.
- **Variables continuas.** Pueden tomar cualquier valor dentro de un rango específico y son infinitamente divisibles, lo que les permite tener valores intermedios.

Mientras que las variables discretas se encuentran en escalones distintos, como el número de hijos en una familia o la cantidad de estudiantes en una clase, las variables continuas abarcan rangos suaves de mediciones, como la altura de una persona o la temperatura. Comprender la distinción entre estos dos tipos de variables es esencial para su análisis adecuado y la interpretación precisa de los datos.

Toma de datos

La toma de datos es el proceso de recopilación de información relevante para un estudio. Este proceso puede implicar diversas técnicas, como la observación, la captura o colecta de ejemplares, las encuestas, las entrevistas, entre otras. La toma de datos debe ser sistemática y estandarizada para garantizar la fiabilidad y validez de los resultados.

La medición de variables es el proceso de cuantificar las características o propiedades que se están estudiando. Por esta razón, es indispensable realizar una adecuada selección de variables de estudio y comprender la naturaleza de estas para poder realizar las mediciones de la manera más adecuada. Por ejemplo, algunas variables cuantitativas se pueden medir directamente, mientras que las variables cualitativas se pueden analizar mediante encuestas o entrevistas. Una vez que se han recopilado los datos, estos se representan en tablas, gráficas estadísticas e incluso se pueden estimar parámetros estadísticos como promedios, medias, entre otros.

Para asegurar la replicabilidad y la aplicabilidad de los resultados, las variables deben ser definidas con claridad, incluyendo tanto su conceptualización teórica como su operacionalización práctica.

La *operacionalización* de variables representa un paso crítico en la investigación científica, donde se define cómo medir una variable de manera precisa y observable. Este proceso esencial transforma conceptos abstractos en términos empíricos, facilitando su medición y análisis. Cada variable se convierte así en un indicador tangible que puede ser medido, recolectado, evaluado y observado. En este proceso se sustituyen variables abstractas por otras más concretas que las representen de manera efectiva.

La precisión y la sistematicidad en la recolección de datos son pilares fundamentales para garantizar la confiabilidad de los resultados en estudios científicos. Esto se logra mediante una comprensión detallada de las variables en estudio, así como de las relaciones

entre ellas y los niveles que presentan. Para alcanzar esto, se emplean protocolos estandarizados que facilitan el diseño adecuado de muestreo y la recolección de datos. Cada disciplina científica desarrolla sus propias estrategias y diseños de investigación adaptados a sus objetivos específicos. La síntesis de estos diseños y metodologías representa un desafío considerable que excede los objetivos de este capítulo. Sin embargo, a continuación, se presentan algunos aspectos generales que se tienen que considerar durante el proceso de diseño experimental.

a. *Conceptualización del estudio*. Antes de abordar el diseño experimental, es crucial entender los principios fundamentales de la temática de estudio, como las interacciones entre variables, las dinámicas entre ellas y las características del sistema que se estudiará (contexto).

b. *Hipótesis claras y específicas*. Un buen diseño experimental comienza con hipótesis bien definidas y específicas. Una *hipótesis biológica* es una proposición general que busca explicar cómo funcionan los sistemas. Estas sólo pueden verificarse de manera indirecta, es decir, por el examen de sus predicciones. Por otro lado, las *predicciones* son los resultados esperados bajo el supuesto de que la hipótesis es verdadera. Comparando estos resultados esperados con los resultados reales obtenidos de la investigación, se puede entonces rechazar (o no) la veracidad de la idea.

c. *Variables de estudio*. Identificar y definir las variables que se van a medir es esencial. Reconociendo todos los niveles de estas, lo que permitirá comprender los posibles valores que pueden asumir los datos en el estudio.

d. *Selección del diseño experimental*. Existen diferentes tipos de diseños experimentales, como el diseño de parcelas divididas o el diseño de bloques al azar. La elección del diseño depende de las preguntas de investigación y de las condiciones específicas del estudio.

e. *Replicación y aleatorización*. La replicación de las unidades experimentales[5] y la aleatorización de su asignación a tratamientos son fundamentales para reducir la variabilidad no controlada y permitir inferencias válidas sobre los efectos de los tratamientos. La inclusión de *réplicas* es esencial para obtener resultados confiables y estadísticamente significativos. El número de réplicas necesarias depende de la variabilidad de los datos. El diseño experimental debe distribuir

5. En los estudios ecológicos se pueden encontrar dos o más unidades experimentales que se agrupan en dos categorías: a) el grupo de control (conjunto de elementos de la población que no estarán sometidos a las variaciones en la variable independiente); y b) los tratamientos (grupos de elementos que estarán sometidos a distintos valores –niveles– de la variable independiente).

aleatoriamente los tratamientos y las réplicas para evitar sesgos y permitir una evaluación objetiva de los resultados.

f. *Control de variables no controlables*. Es importante controlar y/o minimizar la influencia de variables no deseadas que puedan afectar los resultados del estudio. Esto se logra con un entendimiento profundo del objeto de estudio y del contexto.

g. *Tamaño de la muestra*[6]. Determinar el tamaño adecuado de la muestra es crucial para garantizar la detección de efectos significativos y la generalización de los resultados. Este aspecto es el más complejo de definir cuando uno empieza a adentrase en los estudios científicos, además de ser el que más especificidad presenta según la disciplina. Sin embargo, cada disciplina cuenta con detallados tratados donde definen de maneras muy precisas estos aspectos.

Tipos de diseño experimental en ecología

El diseño experimental desempeña un papel crucial para garantizar la validez y confiabilidad de los resultados obtenidos. A continuación, se desarrollan los fundamentos y aplicaciones de tres enfoques metodológicos esenciales en el diseño experimental: el diseño aleatorizado, el diseño aleatorizado en bloques y el diseño factorial. Cada uno de estos métodos ofrece herramientas específicas para abordar diferentes desafíos en la planificación y ejecución de estudios, permitiendo a los investigadores optimizar la recolección de datos y la interpretación de los resultados.

Diseño aleatorizado. En este enfoque, los tratamientos se asignan aleatoriamente a las unidades experimentales. Las unidades experimentales pueden ser individuos, parcelas, o cualquier otra unidad sobre la que se esté realizando el experimento.
Diseño aleatorizado en bloques. Las unidades experimentales se agrupan en bloques homogéneos. Dentro de cada bloque, los tratamientos se asignan aleatoriamente, de modo que se reduzca la variabilidad dentro de cada bloque.
Diseño factorial. En este diseño se manipulan dos o más variables independientes simultáneamente. El diseño factorial permite evaluar cómo interactúan las variables independientes entre sí y cómo influyen en la variable de respuesta.

6. Las muestras en investigación se refieren a un subconjunto de objetos seleccionados de una población. Las muestras se utilizan en estadística cuando no es posible realizar una investigación que incluya la totalidad de los elementos de la población de estudio.

Ejemplo de diseño experimental para estudios en ecología

Imagine que interesa estudiar cómo afecta la cobertura arbórea de los bosques a la comunidad de hormigas en la provincia de Formosa. Se debe, en primera instancia, definir una hipótesis de trabajo que podría ser: «Los cambios en la cobertura arbórea afectarán a las comunidades de hormigas». A partir de esta hipótesis, se construye la siguiente predicción[7]: «Al disminuir la cobertura arbórea, disminuirá la riqueza de especies de hormigas presentes en los bosques».

Ahora que ya se tiene construida la hipótesis y predicción, se necesita estructurar un diseño experimental que permita comprobar estos supuestos planteados. Para ello, primero se debe definir el área de estudio, que en este ejemplo serán los bosques de la provincia de Formosa. También se deben caracterizar estas áreas a partir del estado de cobertura, lo que se puede hacer registrando el porcentaje de cobertura arbórea de los bosques. Este porcentaje se podría utilizar para definir tratamientos o categorías de cobertura para clasificar las áreas boscosas, por ejemplo, áreas con cobertura alta (60 al 100%) y áreas de cobertura baja (30 al 59%). Sin embargo, aún se tiene un problema, estudiar todas las áreas boscosas de la provincia no es algo factible por el tiempo y recursos que requiere lograr tal iniciativa. Por esto, lo ideal sería realizar muestreos aleatorios que sean representativos de los bosques de la provincia de Formosa. Además, es importante realizar muestreos balanceados para cada una de las categorías con el fin de poder comparar los distintos tratamientos.

Resuelto todo esto, surge una importante cuestión: ¿Cuántas réplicas (muestreos) se deberían realizar por cada categoría? Debido a la naturaleza compleja y multifactorial de los sistemas ecológicos, no existe una respuesta precisa para esta pregunta. Es decir, no existe un número único de muestras que se pueda realizar a todos los estudios ecológicos, ya que el tamaño de la muestra ideal puede variar dependiendo de varios factores, incluyendo:

- Objetivos del estudio: los estudios que buscan detectar efectos más puntuales (locales) pueden requerir tamaños de muestra más pequeños que aquellos que buscan detectar efectos grandes (regionales o continentales).
- Variabilidad de los datos: si los datos son muy variables, se pueden necesitar tamaños de muestra más grandes para detectar un efecto.
- Diseño del estudio: algunos diseños de estudio pueden requerir tamaños de muestra más grandes que otros. Por ejemplo, los estudios longitudinales que siguen a los individuos o poblaciones a lo largo del tiempo, generalmente, requieren tamaños de muestra más grandes. Esto es diferente de los estudios transversales, que toman el estado de las variables en un punto en el tiempo.
- Recursos disponibles: el tamaño de la muestra también puede estar limitado por los recursos disponibles, incluyendo el tiempo, el dinero y el personal.

7. Es importante aclarar que de una hipótesis se puede desprender más de una predicción.

Teniendo en cuenta esto, se puede definir un muestreo donde se recolecten muestras de comunidades de hormigas en diez áreas boscosas. Estas estarán distribuidas homogéneamente entre las categorías presentando el mismo número de réplicas para cada estado de cobertura (cinco en áreas con cobertura alta y cinco en áreas con cobertura baja). Es importante considerar además que estas áreas tienen que ser independientes una de la otra en el estudio. Cada grupo taxonómico tiene estandarizada la distancia mínima[8] para permitir independencia entre las unidades muestrales.

En este punto ya se cuenta con la estructura general del experimento, ahora queda definir cómo se recolectarán las muestras en campo. Esta etapa también se encuentra estandarizada para cada grupo de organismos. En el caso de las hormigas se utiliza el protocolo ALL (*Ants of the Leaf Litter Protocol*), desarrollado como propuesta para un relevamiento rápido y estandarizado de hormigas en ambientes boscosos. El esquema básico de muestreo con el protocolo ALL[9] consiste en un transecto de 200 m de longitud, con 20 puntos de muestreo distribuidos a intervalos de 10 m. En cada punto de muestreo se aplican los siguientes métodos estandarizados:

a) colecta de un metro cuadrado (1m²) de hojarasca, que luego se tamiza para extraer las hormigas mediante una trampa mini-Winkler, y

b) instalación de una trampa de caída (*pitfal*).

Este protocolo proporciona un marco sistemático y eficiente para la recolección de datos en el campo, permitiendo la obtención de información precisa sobre la diversidad y la abundancia de las poblaciones de hormigas en el área de estudio.

En este punto se puede considerar que el diseño experimental ya está completo y se puede afirmar que se obtendrán 400 muestras. Es decir, se van a muestrear bosques de la provincia de Formosa distribuidos en dos categorías. Para cada categoría, se van a colectar muestras en 5 áreas boscosas, y en cada área boscosa se tomarán 40 muestras (20 mini-Winkler + 20 pitfall). En muchos estudios ecológicos, esto es expresado en una fórmula que para el ejemplo de estudio sería:

$$400 \text{ muestras} = 2 \text{ categorías X } 5 \text{ bosques X } (20 \text{ mini-Winkler} + 20 \text{ pitfall})$$

8. Esta distancia mínima está definida para cada grupo taxonómico. En los estudios con vertebrados suelen ser de kilómetros, mientras que, cuando se estudian algunos grupos de insectos, la distancia mínima puede ser de metros.

9. La implementación de este protocolo en campo requiere de un período de tiempo de 48 horas y la participación de una a dos personas.

Recapitulando

Este segundo capítulo exploró los pilares fundamentales del diseño experimental y la toma de datos, dos etapas esenciales en cualquier proceso de investigación científica. Desde la conceptualización de variables hasta la implementación de estrategias de recolección de datos, se abordó cómo estructurar investigaciones sólidas y rigurosas en diversos campos científicos.

El diseño experimental, clave para identificar relaciones causales entre variables, implica la selección cuidadosa de variables y la identificación de sus niveles. Desde la variable independiente hasta la variable dependiente, se ha explorado cómo estas son esenciales para comprender fenómenos y procesos dentro de un sistema. Además, se examinaron las diferentes escalas de medición, desde la nominal hasta la de razón, y cómo estas determinan el tipo de análisis estadístico aplicable. Comprender estas escalas es crucial para el análisis adecuado de los datos y la interpretación precisa de los resultados.

En cuanto a la toma de datos, se destacó la importancia de la sistematicidad y la precisión en la recolección de información, así como la necesidad de definir claramente las variables y comprender su operacionalización. La replicación y la aleatorización son pilares para reducir la variabilidad no controlada y permitir inferencias válidas sobre los efectos de los tratamientos.

Finalmente, se exploraron diferentes tipos de diseños experimentales en ecología, desde el aleatorizado hasta el factorial, y cómo estos permiten evaluar interacciones entre variables y su impacto en la variable de respuesta. A través de un ejemplo de diseño experimental para estudios en ecología, se ilustró cómo estructurar un experimento para investigar el efecto de la cobertura arbórea en la comunidad de hormigas.

En resumen, se proporcionan las bases teóricas y prácticas necesarias para diseñar e implementar investigaciones científicas sólidas y rigurosas, fundamentales para avanzar en el entendimiento de los fenómenos naturales y sus interacciones.

Capítulo 3
Tipos de matrices de datos

Lucas J. Mina y Darío D. Larrea

Para realizar cualquier análisis en R, se utilizan funciones que requieren datos con estructuras específicas. Por lo tanto, es necesario preparar y organizar los datos crudos para que sean útiles en el análisis. Esta tarea, que implica limpiar y estructurar los datos, suele ser la más demandante en términos de tiempo en el proceso de análisis.

Vectores, matrices, data frames y listas

A continuación, se presentarán las estructuras de datos más importantes en R, junto con instrucciones sobre cómo crearlas y cómo acceder a sus elementos.

Vectores

Un vector es un conjunto de objetos del mismo tipo –caracteres numéricos, lógicos, entre otros– concatenados con la función c(). Por ejemplo:

```
x <- c(3, 5, 7, 0, 8)
```

Esto crea un vector formado por los números 3, 5, 7, 0 y 8. Para acceder a un valor dentro del vector, se usa x[i] (donde i es el número de índice), por ejemplo, x[3] devolverá el valor en el tercer lugar, en este caso, el número 7. Cabe aclarar que R indexa los valores a partir del 1, a diferencia de otros lenguajes que empiezan por el 0.

Ahora, ¿qué pasaría si se quisiera crear un vector de números consecutivos? Usando la función anterior, la de concatenar, se tendrán que escribir todos los números de a uno, pero existen formas más sencillas:

```
# Crea un vector con números del 1 al 10
#
x <- c(1:10)

# Crea una secuencia del 1 al 10 con 5 elementos
#
x <- seq(1, 10, lenght = 5)
```

Y así como hay varias formas de crear vectores, también existen diversas maneras para acceder a sus valores. Algunas muy utilizadas como x[1:3] devuelve los valores en el rango especificado (incluyendo los extremos) o x[-i], que permite excluir el elemento ubicado en la posición i, por ejemplo, x[-1] mostrará todos los valores menos el primero.

A partir de aquí, se mostrarán algunas operaciones útiles que se pueden hacer con los vectores. Vale la pena tener en mente, a pesar de que todavía no se ha hablado de ellas, que todas estas también son válidas para columnas de data frames o matrices, ya que se puede interpretar cada columna como un vector:

```
# Encontrar los valores del vector_1 ausentes en el vector_2
#
setdiff(vector_1, vector_2)

# Valores del vector_1 presentes en el vector_2
#
vector_1 %in% vector_2

# Valores presentes en ambos vectores
#
intersec(vector_1, vector_2)
```

En caso de utilizar exclusivamente vectores numéricos, R ofrece un par de funciones que pueden ser muy útiles en el caso de necesitar dividir vectores en n cantidad de grupos o en grupos con n elementos, suponiendo que x es vector numérico:

```
# Crear grupos con el mismo nro. n de elementos
#
cut_interval(x, lenght = n)

# Crear n grupos con, en lo posible, el mismo nro. de elementos
#
cut_number(x, n = n)
```

Matrices

Las matrices son una extensión de los vectores y funcionan como vectores multidimensionales y, al igual que estos, deben consistir en elementos del mismo tipo. Las matrices se pueden crear a través de la función matrix(). Por ejemplo:

```
x <- matrix(1:4 nrow = 2)

# Esto crea una matriz con valores del 1 al 4 y dos filas

x
     [,1] [,2]
[1,]   1    3
[2,]   2    4
```

Lo que se observa dentro de los corchetes son los índices de las filas y columnas de la matriz, y es lo que se usará para acceder a los valores dentro de ella (recuérdese un poco el juego de la batalla naval). Suponga que quiere el valor 3 y como se ve, este se encuentra en la fila 1 y columna 2, por lo tanto, tendrá que usar x[1,2]. De igual modo, es posible obtener todos los elementos de una fila (x[1,]) o de una columna (x[,1]), por lo que debe prestarse atención a la posición de la coma para cada caso.

Data frames

Los data frames son el elemento más común con el que se va a trabajar en R. Es lo más cercano a una tabla, como las que se podrían ver en programas como Excel. En este, cada campo puede tener tipos de objetos diferentes, pero cada columna debe estar compuesta por el mismo tipo. Por ejemplo, no se pueden mezclar tipos numéricos y caracteres en una misma columna, en el caso de tener números y palabras o letras en una columna, los números serán tratados como caracteres.

Para crearlos, se utilizará la función data.frame(), de forma muy similar a la usada para las matrices. En esta ocasión, se debe recordar que R rellena los data frames por columnas, es decir, el primer argumento que se pasa a la función corresponde a todos los valores de la primera columna; el segundo, a la segunda columna, y así sucesivamente. Para acceder

a los valores alojados en el data frame, se puede realizar como en las matrices, mediante índices (recuérdese x[i] o x[i,j]) o, en el caso de que las columnas tengan nombres, con la expresión x$nombre_columna (siendo x un data frame). Esto arroja como resultado un vector con todos los elementos de la columna.

Listas

Las listas son objetos en R que pueden almacenar cualquier tipo de otros objetos, como números, caracteres, vectores, matrices, data frames e incluso otras listas. Son especialmente útiles para organizar y manejar conjuntos de datos y objetos que se utilizarán posteriormente. Pero también existen algunos análisis que pueden tomar listas como entrada en sus argumentos y, a su vez, devolver una como su resultado, como es el caso de la función iNext(), que se utiliza para interpolación y extrapolación de números de Hill (ver más detalles en el capítulo 7).

Para crear una lista, nos basta con la función list(), por ejemplo:

```
x <- list(subfamilia = Myrmicinae, género = Nesomyrmex, especie =
Nesomyrmex spininodis)
```

En esta función, cada argumento es un elemento dentro de la lista, y los nombres antes del signo igual (subfamilia, género y especie) son los nombres que tendrán dentro de la lista. Por ende, para acceder a cualquiera de estos elementos, se puede usar x$genero, que nos dará como resultado *Nesomyrmex*. También se pueden usar índices, x[[i]], como se hace con otros objetos, pero aquí es necesario encerrar el índice en doble corchetes, por ejemplo, para obtener el género, al ser el segundo elemento deberíamos escribir: x[[2]].

Lectura de datos

Para empezar cualquier análisis, se deben cargar los datos necesarios en la sesión de R. Estos se pueden encontrar en distintos formatos. Aquí se describirán algunos de estos formatos y cómo leerlos dentro de R. Siempre que sea posible, se utilizará R base (las funciones por defecto, sin el uso de librerías), pero se verá que para algunos casos se deberán usar librerías.

La función genérica para leer archivos es read.table(). Existen derivados de esta función, lo único que sucede es que cambian los argumentos por defecto, así no se tienen que ingresar para leer formatos en particular:

```
tabla <- read.table("datos.txt", header = FALSE, sep = "",
row.names, dec = ".")
```

Aquí se almacenan los datos contenidos en datos.txt en la variable tabla. Los argumentos que se ven son algunos de los más usados, de los disponibles en la función read.table(), con sus valores por defecto. Es recomendable revisar los documentos de ayuda de R para ver todas las opciones. Esta recomendación es válida para cualquiera de las funciones mencionadas en este libro.

Ahora, se analizarán los argumentos aquí ejemplificados.

- header: usar o no la primera fila como nombres de columna, sus valores son TRUE o FALSE.
- sep: nos permite definir qué separador de datos es el usado en el archivo. Al estar vacío, se indica que el separado es un espacio. Si se coloca una coma entre las comillas (","), se indica que los datos están separados por comas. Existen otros separadores que se verán en el desarrollo de este capítulo.
- row.names: se utiliza para asignar nombres a las filas de una tabla en R. Este parámetro puede ser un vector que contenga el mismo número de elementos que filas tenga la tabla, o un número que indique qué columna debe usarse para los nombres de las filas, siendo comúnmente la primera columna.
- dec: para especificar cuál es el separador decimal usado.

Después de haber introducido una forma general de cómo leer datos en R, se analizarán algunos de los formatos con los que se suele trabajar y cómo importarlos a la sesión de R. Es importante aclarar que, excepto las hojas de cálculo, todos los archivos son de texto plano.

Archivos CSV

En estos archivos, los valores se encuentran separados por comas –*Comma Separated Values (CSV)*– y, para leerlos, se dispone de tres funciones (para el próximo y futuros ejemplos, suponer que se desean obtener nombres de columnas):

```
read.table("datos.csv", header = TRUE, sep = ",")   #(1)

read.csv("datos.csv", header = TRUE)                #(2)

read.csv2("datos.csv", header = TRUE)               #(3)
```

Al usar la función 1, read.table(), se debe definir el separador, detalle que no es necesario si se usan las funciones 2 o 3. Pero, ¿cuál es la diferencia entre estas dos últimas? La única diferencia es que la función 2 usa como separador decimal el punto y la función 3 usa la coma.

Archivos TSV

Los valores separados por tabulaciones –*Tab Separated Values (TSV)*– son más fáciles de interpretar si se abre el archivo en un editor de textos, como el bloc de notas de Windows. Sin embargo, ninguno de los dos formatos vistos hasta el momento es objetivamente mejor o peor que el otro. Aclarado esto, se verá que, para leerlos, otra vez se tienen tres posibilidades similares a las vistas anteriormente:

```
read.table("datos.csv", header = TRUE, sep = "\t")   #(1)

read.delim("datos.csv", header = TRUE)               #(2)

read.delim2("datos.csv", header = TRUE)              #(3)
```

Al principio se utiliza read.table() (1) y se define el separador manualmente. En este caso se indican tabulaciones (\t). La diferencia entra la opción 2 y 3 es el separador decimal que usan (la función 2 usa como separador decimal el punto y la función 3, la coma).

Hojas de cálculo

Hasta ahora, se ha visto un par de formatos en texto plano con los que lidiar sin problemas usando R base. Sin embargo, también se pueden encontrar datos o bases de datos almacenadas en hojas de cálculo creadas con programas como Excel. Para leer estos archivos, se utilizará una librería llamada readxl[1], disponible en CRAN. Una vez instalada esta librería, se procede a cargar datos desde una hoja de cálculo:

```
library(readxl)

read_excel("datos.xlsx", sheet = 1)     #(1)

excel_sheets("datos.xls")               #(2)
```

Con la línea de código número 1 se lee la primera hoja del archivo de Excel. El argumento sheet es el que permite elegir cuál de las hojas se desea leer. Para esto, se puede brindar un número o el nombre de la hoja en caso de tenerlo. Con la función número 2 se enumeran las hojas que contiene el archivo Excel. Como es el caso de otras funciones, aquí se muestra sólo una parte de las opciones disponibles.

1. Se puede instalar de forma independiente o como parte de Tidyverse. Es recomendable la segunda opción, ya que se utilizará más adelante.

Hasta ahora se han desarrollado algunas formas de importar datos a la sesión de R, basados en los formatos más utilizados en ecología. Sin embargo, existen aún más formatos y funciones disponibles. Usando otras librerías es posible incluso leer datos directamente de servicios como Drive de Google o OneDrive de Microsoft.

Trabajar con los datos

Como se mencionó al inicio del capítulo, los análisis en R requieren datos con una estructura particular, por lo tanto, en esta sección se verá cómo preparar los datos importados para los distintos análisis.

Exploración. Una vez cargados los datos dentro de R, es posible que sea de interés explorarlos, entonces se tomará como ejemplo una tabla de variables ambientales:

```
ambiente <- read.csv("data/ambiente.csv", header = TRUE)
```

Entonces: head(ambiente) permite ver las primeras filas de la tabla, aunque tails(ambiente) es similar, pero nos muestra las últimas filas.

Si se quiere saber cuántas localidades se muestrearon, se puede usar:

```
lenght(ambiente$localidad)
[1] 9
```

Con summary(ambiente) se obtienen algunas medidas resumen para cada columna. La función View(ambiente) permite ver los datos completos como una tabla.

Manipulación de los datos. No se refiere a alterar los datos en sí, sino a modificar las tablas donde estos se encuentran para que adquieran el formato apropiado para su uso.

Agregar columnas y combinar tablas. Estas son operaciones comunes y sencillas de realizar en R. Para agregar una columna a una tabla existente, basta simplemente con colocar el nombre de la variable donde se encuentra la tabla seguida del signo $ y el nombre de la columna nueva, por ejemplo:

```
tabla$suma <- rowSums(tabla)
```

Esto agrega la columna «suma» a la tabla con los valores de la sumatoria de cada fila. Es necesario aclarar que, para evitar errores, la cantidad de elementos que se agregan a la columna nueva deben ser iguales al número de filas que posee la tabla original.

Para combinar tablas, se puede hacer por columnas, colocando una tabla al lado de la otra, o por filas, colocándolas una debajo de la otra:

```
combinacion_columnas <- cbind(tabla_1, tabla_2)

combinacion_filas <- rbind(tabla_1, tabla_2)
```

En ambos casos, la cantidad de tablas que se pueden combinar utilizando las funciones cbind y rbind es ilimitada. Sin embargo, para evitar que estas funciones arrojen algún error, es necesario que todas las tablas tengan el mismo número de filas cuando se usa cbind y el mismo número de columnas cuando se usa rbind.

Reemplazar datos. Algunas veces es necesario reemplazar los datos de una tabla. Por ejemplo, en el caso de las tablas de incidencia para algunos cálculos de diversidad, donde se tienen que cambiar todos los valores numéricos por ceros y unos (presencia o ausencia). Para esto se presentan dos formas:

```
tabla[is.na(tabla)] <- 0

#Opción 1
tabla[tabla > 0] <- 1

#Opción 2
tabla_incidencia <- ifelse(tabla > 0, 1, 0)
```

Primero se asegura que la tabla no tenga NA's (en R se le asigna el valor NA a las celdas que no contienen ningún valor), reemplazándolo por cero. Luego, con la opción 1 se reemplazan todos los valores mayores a cero por uno sobrescribiendo la tabla original. En la opción 2 se crea una tabla nueva llamada tabla_incidencia. En este caso se utiliza un condicional para reemplazar todos los valores mayores a cero por uno. Se recomienda utilizar la segunda opción, ya que crear una tabla nueva facilita encontrar errores y corregirlos.

Anteriormente, se vio cómo reemplazar los valores NA por ceros. Sin embargo, en algunas ocasiones es necesario eliminar todas las filas que contengan algún valor NA. Por ejemplo, si se tiene una base de datos con todos los registros, pero se desea trabajar únicamente con aquellos que tienen una clasificación de estado de conservación, se eliminarán las filas con NA en esas celdas, ya que sólo las filas con una clasificación válida tendrán un valor en esas celdas:

```
na.omit(tabla)
```

Esto elimina la fila sin importar en qué columna se encuentra el valor NA. Algunas veces se quiere eliminar la fila solamente si el valor NA se encuentra en una columna específica, teniendo en cuenta el ejemplo anterior, se eliminarían los NA's de la columna estado_de_conservacion:

```
tabla[!is.na(tabla$estado_de_conservacion),]
```

En la expresión anterior, la función is.na() por sí sola selecciona todas las filas que tengan NA en la columna estado_de_conservacion. Pero al estar por delante un signo de exclamación (!), indica que es la operación opuesta, por lo tanto, se están seleccionando todas las filas que NO tengan NA en la mencionada columna.

Tidyverse

Tras una breve introducción en el primer capítulo sobre esta colección de paquetes, un análisis más detallado de su funcionamiento.

El uso más común que se le da a este conjunto de paquetes es para preparar los datos crudos, desde su limpieza y filtrado hasta el pivoteo de tablas, dejándolos listos para el análisis. A continuación, se muestra un ejemplo, realizando una tabla de abundancias a partir de una base de datos con registros de hormigas. En el siguiente fragmento se observará el uso de %>%, este es propio de uno de los paquetes parte de Tidyverse (magrittr), es el operador de pipe (o tubería) y permite realizar una secuencia de acciones donde el resultado de una es la entrada de la siguiente. A partir de la versión 4.1.0 de R, también está disponible una pipe nativa con la combinación |>, esta tiene algunas limitaciones con respecto a la mostrada anteriormente, pero usarla permite escribir código con menor cantidad de dependencias en librerías externas:

```
library(tidyverse)
library(readxl)

raw_data <- read_excel(base_datos, sheet = 1)

abundancia_ec <- raw_data %>%
  filter(estado_conservacion != "NA") %>%
  group_by(estado_conservacion, especie) %>%
  summarise(ABUNDANCIA = sum(abundancia)) %>%
  pivot_wider(
    names_from = estado_conservacion,
    values_from = ABUNDANCIA,
    values_fill = 0
  )
```

Primero se empieza por leer la base de datos y almacenarla en raw_data. Luego, con la función filter(), se seleccionan todas las filas que NO tengan NA como valor en la columna

estado_conservación (por eso el !=, que indica negación). A continuación, se agrupan los datos por estado de conservación y especie con la función group_by(), colocando próximos unos a los otros los elementos con un mismo valor en estas columnas. Con la función summarise() se realiza la sumatoria de las filas de abundancia que tienen el mismo estado de conservación y especie, y se lo almacena en una nueva fila llamada ABUNDANCIA. Esta función también genera que se eliminen todas las columnas que no fueron nombradas en la función group_by(). En este punto, la tabla se vería así (se muestra solamente la primera parte):

Especie	**ABUNDANCIA**
Acanthoponera mucronata	3
Acromyrmex sp1	18
Acromyrmex sp2	2
Acromyrmex sp3	1
Anochetus cf. Neglectus	1
...	...

Por último, con la función pivot_wider(), se pivota la tabla de forma que el estado de conservación quede como nombre de columnas. Cada fila corresponde a una especie, y la tabla se rellena con los valores de la columna ABUNDANCIA (es decir, la sumatoria). Con el argumento values_fill, se especifica qué valor se desea para las celdas vacías, en este caso, se selecciona cero. El resultado es la siguiente tabla (se muestra un fragmento):

Especie	**ECB**	**ECD**	**ECI**
Acanthoponera mucronata	3	0	24
Acromyrmex sp1	18	17	29
Acromyrmex sp2	2	0	0
Acromyrmex sp3	1	0	0
Anochetus cf. Neglectus	1	0	0
...	...	...	...

Cuando usamos Tidyverse, se está creando un tipo especial de tabla, propio de Tidyverse, llamado tibble. En muchos casos, principalmente en funciones de la librería vegan, no es posible usar estas tablas para los análisis, por lo que es necesario transformarlas a data frames. Para ello, se puede usar la función as.data.frame() o añadir al final del código anterior column_to_rownames(), una función que permite elegir una columna como nombres de fila (lo que transforma el tibble en un data frame, ya que los primeros no tienen nombres de fila). En este caso, el código quedaría de la siguiente manera:

```
library(tidyverse)
library(readxl)

raw_data <- read_excel(base_datos, sheet = 1)

abundancia_ec <- raw_data %>%
  filter(estado_conservacion != "NA") %>%
  group_by(estado_conservacion, especie) %>%
  summarise(ABUNDANCIA = sum(abundancia)) %>%
  pivot_wider(
    names_from = estado_conservacion,
    values_from = ABUNDANCIA,
    values_fill = 0
  ) %>%
  column_to_rownames("especie")
```

Aquí se usan los valores de columna especie como nombres de fila.

Formatos de matrices de datos

A continuación, se describirán brevemente algunos de los tipos de tablas usadas para los análisis presentados en este libro.

Matriz de abundancia
Como su nombre lo indica, cada valor contenido en la tabla corresponde a la cantidad de individuos capturados. Existen dos tipos que dependen de cómo se ubiquen las filas y columnas.

- Tipo 1 (especies por sitio): en esta tabla, cada columna corresponde a una especie y cada fila a un sitio. Los sitios pueden ser lugares geográficos, transectos, tipo de ambiente, etc. (esto también aplica para todas las tablas siguientes). Es frecuentemente utilizada en la gran mayoría de análisis.

	Especie 1	**Especie 2**	**Especie 3**	**Especie n**
Sitio 1				
Sitio 2				
Sitio 3				
Sitio n				

- Tipo 2 (sitios por especies): en este caso, en las columnas se ubican los sitios y en las filas las especies. Principalmente utilizada en análisis con el paquete iNext.

	Sitio 1	Sitio 2	Sitio 3	Sitio n
Especie 1				
Especie 2				
Especie 3				
Especie n				

Matriz de incidencia
Se obtienen generalmente a partir de las matrices de abundancia. Como se describió anteriormente, suelen ser usadas para análisis de diversidad beta. Aquí los valores representados son de presencia/ausencia, expresados con 1 (unos) y 0 (ceros), respectivamente. En estas tablas también se pueden tener los dos tipos descritos para las matrices de abundancia.

Matriz ambiental
Las filas corresponden a cada sitio y en las columnas se encuentran variables ambientales (tipo de ambiente, tratamiento, temperatura, humedad, etc.).

	Temperatura	Humedad	Tratamiento
Sitio 1			
Sitio 2			
Sitio 3			
Sitio n			

Matriz taxonómica
Es usada en los análisis de diversidad taxonómica y consiste en una tabla en la que cada columna representa una jerarquía taxonómica, los nombres de las filas corresponden a un ID que actúa como identificador de cada especie. Este ID debe ser el mismo de los nombres de las columnas en la matriz de abundancia de tipo 1.

	Género	Tribu	Subfamilia
ID 1			
ID 2			
ID n			

Matriz de rasgos

Utilizada para los análisis de diversidad funcional, en ella cada fila corresponde a una especie y cada columna a un rasgo funcional, que pueden ser continuos (como medidas morfométricas) o discretos (características, por ejemplo).

	Rasgo 1	Rasgo 2	Rasgo 3	Rasgo n
Especie 1				
Especie 2				
Especie 3				
Especie n				

Matriz geográfica

Consiste en una tabla donde las filas representan a cada sitio en estudio y las columnas a la longitud y latitud, en ese orden. Usada en análisis de Mantel.

	Longitud	Latitud
Sitio 1		
Sitio 2		
Sitio n		

Matriz para modelos

Usada para pruebas de hipótesis y modelos lineales. En esta tabla también están los sitios en las filas, pero las columnas serán cada variable que se quiere usar para el modelado, por ejemplo, riqueza, diversidad alfa, ambiente, tratamiento, etc.

	Tratamiento	Riqueza	Shannon
Sitio 1			
Sitio 2			
Sitio 3			
Sitio n			

Guardado de datos

Hasta ahora se han visto algunos tipos de objetos para almacenar datos, así como cargar y manipular los datos para su uso. Pero muchas veces es conveniente guardar estos datos ya limpios para poder usarlos en distintos scripts, sin tener que hacer todo el proceso desde cero. Para esto, R consta de algunas funciones que permiten guardar datos en distintos formatos como CSV o RDS. Aquí, se han explorado algunos tipos de objetos para almacenar datos, así como métodos para cargar y manipular datos para su uso.

El formato más común para guardar las tablas ya limpias es CSV. Este, al ser un formato en texto plano y de amplio uso, asegura una gran compatibilidad al momento de usarlo en otros scripts, incluso por fuera de R. Para hacerlo, se utiliza la función write.csv(). Supóngase que se desea guardar la tabla abundancia_ec que se limpió anteriormente con el nombre abundancia.csv dentro de la carpeta data:

```
Write.csv(abundancia_ec, file = "data/abundancia.csv")
```

Cuando se habla de archivos CSV, el separador en estos es la coma, pero ¿qué pasa si se quiere usar otro separador al guardar el archivo? Para esto, se puede usar la función write.table(), que posee un argumento (sep) que permite elegir el separador. Por ejemplo, si se quiere guardar la misma tabla de antes, pero usando tabulaciones como separador:

```
Write.table(abundancia_ec, file = "data/abundancia.tsv", sep =
"\t")
```

En cuanto a los archivos RDS, corresponden a un formato propio de R, por lo que su uso puede dificultar el uso de estos datos en otros softwares de análisis, pero posee algunas ventajas frente a los formatos de texto plano:

- Suelen ser más livianos debido a la compresión que usa R en su creación.
- Pueden guardar cualquier tipo de objeto de R, sean vectores, tablas, listas, gráficos, etc.
- Almacenan todos los atributos del objeto a la hora de su creación, aquellos atributos que no se almacenan en formatos como CSV.

Por ejemplo, para guardar nuestra tabla de abundancias como RDS basta con ejecutar:

```
saveRDS(abundancia_ec, file = "data/abundancia.rds")
```

Crear y manipular bases de datos

Estas recomendaciones tienen como finalidad facilitar el uso de los datos dentro de R.

- Evitar el uso de espacios en nombres de filas y columnas, nombrarlas usando CamelCase (NombreColumna) o snake_case (nombre_columna), por ejemplo.
- Evitar, mientras sea posible, caracteres especiales o acentos. Esto impide errores en el tipeo de los nombres, ya que R no tiene problemas a la hora de usar acentos.
- Usar nombres de filas o columnas lo más descriptivos posibles, evitando nombres como columna_1.
- Establecer reglas al crear una base de datos, es importante para el formato de horas y fechas, y apegarse a ellos. Tener distintos formatos de fecha u hora dentro de una misma tabla es una gran fuente de errores.
- Evitar colocar las unidades junto a los valores, si se tienen valores numéricos con alguna unidad de medida. En caso de ser necesario, aclarar la unidad de medida, es mejor colocarla en el nombre de la columna.
- Evitar guardarlos en la ubicación donde se encuentran los datos crudos, al preparar los datos para su uso en los análisis.

Recapitulando

Este capítulo ofreció una guía integral sobre los tipos de matrices de datos en R, fundamentales para el análisis estadístico y la manipulación de datos en este entorno, una visión detallada sobre las estructuras de datos más utilizadas en R, incluyendo vectores, matrices,

data frames y listas. Cada una de estas estructuras tiene su propio papel y utilidad en el manejo y análisis de datos.

Se adentró asimismo en el uso de vectores, que son la base de muchas otras estructuras en R, en cómo crear y acceder a ellos, y en cómo realizar operaciones básicas que son esenciales para manipular datos. Posteriormente, exploró las matrices que amplían el concepto de vectores a múltiples dimensiones y permitió una organización más compleja de los datos.

También detalló el uso de data frames, que se asemejan a tablas en programas como Excel, y cómo trabajar con ellos para manejar datos heterogéneos. A través de ejemplos prácticos, se crearon data frames, se accedió a sus elementos y se los pudo manipular. Las listas, por otro lado, se presentaron como una estructura flexible que puede almacenar una variedad de objetos, desde vectores hasta otros data frames.

Además, se abordó la lectura y escritura de datos en diferentes formatos, como CSV, TSV y hojas de cálculo, y cómo manejar estos datos una vez importados en R. Se aprendió sobre funciones clave para la limpieza, exploración y manipulación de datos, así como recomendaciones prácticas para evitar errores comunes y asegurar la integridad de los datos.

En conjunto, este capítulo brinda las herramientas y conocimientos necesarios para preparar los datos de manera efectiva para el análisis, estableciendo una base sólida para el trabajo en R.

Capítulo 4
Pruebas de hipótesis estadísticas en estudios ecológicos

Darío D. Larrea y Lucas J. Mina

Las pruebas de hipótesis estadísticas son métodos utilizados en la estadística inferencial para tomar decisiones sobre las características de una población, utilizando información obtenida de una muestra de esa población. Estas pruebas permiten evaluar si los datos proporcionan evidencia suficiente para rechazar o no una afirmación acerca de una característica o relación en la población. Es decir, estas hipótesis estadísticas son en realidad predicciones biológicas propiamente dichas y no hipótesis biológicas.

Antes de continuar, es crucial abordar los errores de tipo I y tipo II, ya que constituyen conceptos esenciales en el ámbito de las pruebas de hipótesis estadísticas. Estos errores son fundamentales para comprender la interpretación de los resultados y la validez de las conclusiones extraídas de cualquier análisis estadístico.

En el primer caso, un error de tipo I (falso positivo) ocurre cuando se rechaza incorrectamente una hipótesis nula verdadera. Es decir, concluye que hay evidencia suficiente para rechazar la hipótesis nula cuando en realidad es verdadera. La probabilidad de cometer este error es alfa (α), que es el nivel de significancia que determinamos para la prueba de hipótesis. Por ejemplo, supóngase que se realiza un estudio para investigar si una especie de planta tiene un impacto negativo en el ecosistema local. La hipótesis nula (Ho) sería que la planta no afecta negativamente al ecosistema. Si se comete un error de tipo I, se concluye que la planta sí tiene un impacto negativo (rechazando Ho) cuando en realidad no es así.

Por otro lado, un error de tipo II (falso negativo) ocurre cuando se acepta incorrectamente una hipótesis nula falsa. Es decir, concluye que no hay suficiente evidencia para rechazar la hipótesis nula cuando en realidad es falsa. La probabilidad de cometer este error es beta (β), relacionada con la potencia de la prueba y se reduce al aumentar el tamaño del muestreo. Continuando con el ejemplo anterior, si la hipótesis nula (Ho) es que la planta no tiene un impacto negativo en el ecosistema local, un error de tipo II ocurriría si se concluye que la planta no tiene un impacto negativo (no se rechaza Ho) cuando en realidad sí lo tiene, en otros términos, la planta sí afecta negativamente al ecosistema local.

En síntesis, el error de tipo I implica rechazar incorrectamente una hipótesis verdadera, mientras que el error de tipo II implica no rechazar incorrectamente una hipótesis falsa. Ambos tipos de error son importantes al interpretar los resultados de las pruebas de hipótesis estadísticas.

Proceso de prueba de hipótesis

Teniendo claro en qué consisten las pruebas de hipótesis y cuáles son los posibles errores a cometer, ahora veremos una enumeración de los pasos a seguir para llevarlas a cabo dentro de R.

1. Formulación de hipótesis estadística: se plantea una hipótesis nula (H0) y una hipótesis alternativa (H1). La hipótesis nula representa la afirmación inicial que se desea poner a prueba, mientras que la hipótesis alternativa sugiere una posible alternativa a la hipótesis nula.

2. Selección de un nivel de significación: se elige un nivel de significación (α), que es la probabilidad máxima de cometer un error tipo I al rechazar incorrectamente la hipótesis nula. El valor típicamente utilizado es 0.05.

3. Elección de la prueba estadística adecuada: se selecciona una prueba estadística apropiada según el tipo de datos y la pregunta de investigación. Por ejemplo, ANOVA, Kruskal-Wallis, entre otros.

4. Cálculo del estadístico de prueba: se calcula un estadístico de prueba a partir de los datos de la muestra.

5. Toma de decisión: se compara el valor del estadístico de prueba con un valor crítico determinado a partir de la distribución de probabilidad bajo la hipótesis nula. Si el valor del estadístico de prueba cae en la región de rechazo (área de cola) definida por el nivel de significación, se rechaza la hipótesis nula en favor de la hipótesis alternativa. De lo contrario, no se rechaza la hipótesis nula.

6. Interpretación de los resultados: se interpreta el resultado de la prueba en el contexto del problema de investigación, teniendo en cuenta las implicaciones prácticas y teóricas.

Tipos de pruebas de hipótesis estadísticas

Las pruebas de hipótesis estadísticas se pueden clasificar en dos categorías principales: paramétricas y no paramétricas. La diferencia fundamental entre ellas radica en las suposiciones que hacen sobre la distribución subyacente de los datos.

Pruebas paramétricas

Estas pruebas suponen que los datos provienen de una población con una distribución normal (Normalidad) y que la variabilidad de los errores debe ser constante en todos los niveles de la variable independiente (Homocedasticidad).

Para verificar si los datos cumplen con estos supuestos, se han desarrollado varios análisis específicos. Para evaluar la normalidad de los datos, se puede emplear la prueba de Shapiro-Wilk o la prueba de Kolmogorov-Smirnov. Estas pruebas permiten determinar si los datos siguen una distribución normal. Por otro lado, para evaluar la homogeneidad de varianza entre grupos, se pueden aplicar pruebas como la prueba de Levene o la prueba de Bartlett. Estas pruebas ayudan a verificar si las varianzas de los datos son consistentes en todos los grupos que se están comparando. Si se cumplen estos supuestos, se pueden realizar las pruebas paramétricas.

Algunas de las pruebas más comunes usadas en ecología son:

Prueba t de Student

Es una herramienta estadística fundamental para comparar las medias de dos grupos independientes o para comparar la media de una muestra con una media poblacional conocida cuando la desviación estándar de la población es desconocida. La prueba t de Student se basa en la distribución t de Student, que es una distribución de probabilidad que tiene en cuenta la variabilidad de las muestras pequeñas. Esta distribución es similar a la distribución normal, pero tiene colas más anchas, lo que la hace adecuada para muestras pequeñas.

La prueba t de Student tiene estadísticos importantes para la interpretación, *t-value*, *t-critical* y *p-value*.

- Estadística t (*t-value*): se calcula como la diferencia entre las medias de dos grupos (o entre la media de una muestra y una media poblacional conocida) dividida por una medida de variabilidad, como el error estándar.

- Valor crítico de la distribución t (*t-critical*): es el valor de la distribución t que se utiliza como punto de corte para determinar si se rechaza o no la hipótesis nula. Este valor depende del nivel de significancia (generalmente 0.05 o 0.01) y los grados de libertad, que a su vez dependen del tamaño de las muestras.

- Valor p (*p-value*): es la probabilidad de obtener una estadística t igual o más extrema que la observada en la muestra si la hipótesis nula es cierta. Un valor p bajo indica que hay evidencia suficiente para rechazar la hipótesis nula. El valor p se compara con el nivel de significancia elegido (generalmente 0.05) para tomar una decisión sobre la hipótesis nula.

Toma de decisión. Si la estadística t calculada es mayor que el valor crítico (o si el valor p es menor que el nivel de significancia), se rechaza la hipótesis nula, lo que indica que hay

una diferencia significativa entre las medias de los grupos. Es importante tener en cuenta que la prueba t asume que los datos son aproximadamente normales y que las muestras son aleatorias e independientes. Si estos supuestos no se cumplen, pueden surgir problemas en la interpretación de los resultados.

Análisis de Varianza (ANOVA)

Es utilizada para comparar las medias de tres o más grupos independientes. Su objetivo principal es determinar si al menos uno de los grupos difiere significativamente de los demás en términos de la variable que se está estudiando. El ANOVA se basa en la comparación de las variaciones entre grupos (variabilidad intergrupal), con las variaciones dentro de los grupos (variabilidad intragrupal). Si la variabilidad entre los grupos es significativamente mayor que la variabilidad dentro de los grupos, esto sugiere que al menos uno de los grupos es estadísticamente diferente de los demás.

Hay diferentes tipos de ANOVA, dependiendo del diseño experimental y el número de factores que se están estudiando:

- ANOVA de un factor: se utiliza cuando se compara la media de una variable entre tres o más grupos independientes. Por ejemplo, se podría utilizar para comparar la diversidad de especies en diferentes tipos de hábitats (bosque, pradera, desierto).

- ANOVA de dos factores: se utiliza cuando se tienen dos variables independientes (factores) y se desea evaluar su efecto en una variable dependiente. Por ejemplo, se podría utilizar para estudiar cómo la temperatura (factor 1) y la dieta (factor 2) afectan la tasa de crecimiento (variable dependiente) de una especie de insecto.

- ANOVA de medidas repetidas: se utiliza cuando las mismas unidades experimentales son medidas en diferentes momentos o bajo diferentes condiciones. Por ejemplo, se podría utilizar para comparar la tasa de supervivencia de una misma población de animales antes y después de un evento climático extremo.

El ANOVA produce dos estadísticas principales: *F-value* y *p-value*.

- F-value: compara la variabilidad entre grupos con la variabilidad dentro de los grupos. Si el valor F es grande, indica que la variación entre las medias de los grupos es significativamente mayor que la variación dentro de los grupos, lo que sugiere que al menos un grupo es estadísticamente diferente de los demás.

- p-value: es la probabilidad de obtener una estadística F igual o más extrema que la observada en la muestra, si la hipótesis nula es cierta. En el contexto del ANOVA, el valor p indica la probabilidad de que las diferencias entre las medias de los grupos sean debidas únicamente al azar.

Toma de decisión. Si el valor p es menor que un nivel de significancia predefinido (generalmente 0.05), se rechaza la hipótesis nula y se concluye que existen diferencias significativas entre los grupos. Es importante tener en cuenta que el ANOVA sólo determina si existen diferencias significativas entre los grupos, pero no identifica específicamente cuáles grupos son diferentes entre sí cuando se trabaja con más de dos grupos. En estos casos, al obtener un resultado significativo en el ANOVA, se pueden realizar pruebas de comparaciones múltiples, como la prueba de Tukey o la prueba de Scheffé. Estas pruebas permiten realizar comparaciones entre todos los pares de grupos para identificar cuáles presentan diferencias estadísticamente significativas, brindando una comprensión más detallada de las relaciones entre los grupos.

Regresión lineal

Utilizada para determinar si existe una relación significativa entre una variable dependiente y una o más variables independientes. Su objetivo principal es entender y predecir cómo cambia una variable dependiente en función de una o más variables independientes. El modelo de regresión lineal asume que la relación entre las variables es lineal, lo que significa que los cambios en la variable dependiente están linealmente relacionados con los cambios en las variables independientes.

Según el número de variables independientes usadas, se pueden clasificar las regresiones lineales en dos tipos:

- Regresión lineal simple: se emplea cuando solo hay una variable independiente que influye en la variable dependiente.
- Regresión lineal múltiple: se utiliza cuando hay dos o más variables independientes que afectan a la variable dependiente.

La regresión lineal tiene cinco estadísticas: coeficiente de regresión, coeficiente de determinación, error estándar de la estimación, *F-value* y *p-value*.

- Coeficientes de regresión (β): representan la magnitud y dirección de la relación entre las variables independientes y la variable dependiente en el modelo de regresión lineal. Cada coeficiente indica cuánto cambia la variable dependiente por cada unidad de cambio en la variable independiente correspondiente, manteniendo constantes las demás variables. Un coeficiente positivo ($\beta>0$) indica que a medida que aumenta el valor de la variable independiente, también aumenta el valor de la variable dependiente. Por el contrario, un coeficiente negativo ($\beta<0$) indica que a medida que aumenta el valor de la variable independiente, el valor de la variable dependiente disminuye.
- Coeficiente de determinación (R^2): es una medida de la proporción de la variabilidad de la variable dependiente que es explicada por el modelo de regresión.

Representa la bondad de ajuste del modelo y cuánto mejor se ajustan los datos observados a la línea de regresión. Este coeficiente varía entre 0 a 1, valores cercanos a 1 indican que el modelo explica una gran parte de la variabilidad de la variable dependiente, lo que sugiere un buen ajuste del modelo.

- Error Estándar de la Estimación (SEE): mide la variabilidad no explicada por el modelo y se utiliza para evaluar la precisión de las predicciones.

- Estadístico F (*F-value*): se utiliza para evaluar la significancia global del modelo de regresión. Compara la variabilidad explicada por el modelo (debido a la regresión) con la variabilidad no explicada (debido al error).

- Estadístico p (*p-value*): el valor p asociado al estadístico F indica la probabilidad de obtener un valor F igual o más extremo que el observado, si la hipótesis nula de que todos los coeficientes de regresión son cero es cierta.

Toma de decisión. Un valor p bajo (generalmente <0.05) sugiere que el modelo de regresión es significativo y que al menos una de las variables independientes está relacionada de manera significativa con la variable dependiente.

Análisis Multivariante de Varianza (MANOVA)

El Análisis Multivariante de Varianza es una técnica estadística utilizada para analizar la relación entre múltiples variables dependientes continuas y una o más variables independientes categóricas. El MANOVA produce varios estadísticos que ayudan a evaluar la significancia global del modelo y a comprender la naturaleza de las diferencias entre los grupos en las variables dependientes.

- Estadístico F multivariado: es un estadístico que evalúa la diferencia global entre los grupos en todas las variables dependientes simultáneamente. Un valor grande de F multivariado sugiere que al menos una de las variables dependientes tiene diferencias significativas entre los grupos.

- Valores p asociados al estadístico f multivariado: indican la probabilidad de obtener un valor del estadístico F multivariado igual o más extremo si la hipótesis nula de igualdad de medias en todas las variables dependientes es verdadera.

Toma de decisión. Un valor p pequeño (<0.05) sugiere que al menos una de las variables dependientes tiene diferencias significativas entre los grupos.

Pruebas no paramétricas

Tienen suposiciones más flexibles que las pruebas paramétricas y no hacen suposiciones sobre la distribución subyacente de los datos. Son útiles cuando los datos no cumplen con los supuestos de las pruebas paramétricas. Algunas de las pruebas no paramétricas más comunes son:

Prueba de Kruskal-Wallis

Es el equivalente no paramétrico del ANOVA, utilizada para comparar tres o más grupos independientes. Esta prueba estima un estadístico p (*p-value*) y permite determinar si las diferencias observadas entre los grupos son estadísticamente significativas o si podrían haber ocurrido por azar.

Prueba de correlación de Spearman

Es utilizada para evaluar la relación entre dos variables continuas cuando los datos no siguen una distribución normal. Esta prueba presenta dos estadísticos:

- Coeficiente de correlación de Spearman o Rho de Spearman (ρ): este coeficiente varía entre -1 a 1. Indica la fuerza y la dirección de la relación entre las dos variables. Cuanto más cercano sea el valor de ρ a 1, esto indica una correlación fuertemente positiva entre las dos variables. Es decir que, a medida que los valores de una variable aumentan, los valores de la otra variable también aumentan. Por otro lado, cuanto más cercano sea el valor de ρ a -1, existe una correlación fuertemente negativa entre las variables. Es decir que, a medida que los valores de una variable aumentan, los valores de la otra variable disminuyen. Sin embargo, cuando el valor de ρ es cercano a 0, indica la ausencia de una correlación monótona entre las dos variables.

- Valor p (*p-value*): indica la probabilidad de obtener un coeficiente de correlación de Spearman igual o más extremo que el observado si la verdadera correlación entre las variables es cero. Un valor p pequeño (<0.05) sugiere una correlación significativa, mientras que un valor p grande (>0.05) sugiere que la correlación observada podría haber ocurrido por azar.

Prueba de Mann-Whitney U

Es utilizada para comparar las medianas de dos grupos independientes cuando los datos no son normalmente distribuidos. Es una alternativa a la Prueba t de Student cuando los datos no cumplen con los supuestos de normalidad o cuando se trabaja con datos ordinales. Esta prueba estima un estadístico p (*p-value*) que permite determinar si las diferencias observadas entre los grupos son estadísticamente significativas o si podrían haber ocurrido por azar.

Análisis de Variación Permutacional Multivariante de la Varianza (PERMANOVA)

El Análisis de Variación Permutacional Multivariante de la Varianza es una extensión del MANOVA que utiliza técnicas de permutación para evaluar la significancia de las diferencias entre grupos en un diseño experimental multivariante. A diferencia del MANOVA, que asume distribuciones normales y homogeneidad de varianzas, el PERMANOVA es una técnica no paramétrica que no hace suposiciones sobre la distribución de los datos.

Esta prueba estima dos estadísticos, el valor *pseudo-F* y el *p-value*:

- Estadístico pseudo-F: mide la magnitud de las diferencias entre los grupos en la estructura multivariante de los datos. Es similar al estadístico F utilizado en el ANOVA, pero adaptado para datos multivariantes.

- Valor p (*p-value*): indican la probabilidad de obtener un estadístico de prueba igual o más extremo bajo la hipótesis nula de que no hay diferencias entre los grupos. Un valor p pequeño sugiere que las diferencias observadas son estadísticamente significativas.

Tendencias en ecología

En las últimas décadas, en ecología se ha extendido el uso de los Modelos Lineales Generalizados (GLM, por sus siglas en inglés). Estos modelos son una herramienta estadística ampliamente utilizada para analizar datos que pueden no cumplir con los supuestos de los modelos lineales tradicionales, como la normalidad y la homogeneidad de varianzas. Los GLM permiten modelar relaciones entre variables predictoras y una variable de respuesta que pueden ser no normales, como datos de conteo, presencia/ausencia, proporciones o variables categóricas. Además, para enfrentar desafíos particulares en el análisis de datos ecológicos, se han desarrollado extensiones y variantes de los GLM, tales como los Modelos Lineales Mixtos (GLMM) y los Modelos Lineales Generalizados Aditivos (GAM). Profundizar conceptualmente en estos modelos excede los alcances de este libro; sin embargo, es esencial mencionarlos y brindar al menos una explicación básica sobre la ejecución de los Modelos Lineales Generalizados (GLM) en R.

Medir e interpretar las pruebas de hipótesis en R

Es importante destacar que la interpretación de estos análisis se respalda principalmente en los p valores. Generalmente, los valores de p menor a 0.05 expresan significancia estadística.

Pruebas paramétricas

En las pruebas de hipótesis paramétricas es importante evaluar la normalidad y la homocedasticidad. Estos parámetros en R pueden ser evaluados con las siguientes pruebas estadísticas y gráficos. En todos estos análisis se busca evaluar si existen diferencias significativas observadas. Es decir, si el valor p es menor que un nivel de significancia predefinido (generalmente 0.05).

Prueba de Shapiro-Wilk

Utilizada para verificar la normalidad de los datos, se puede realizar a través de una función incluida en R base, por lo que no es necesario descargar librerías adicionales.

Input

Paso 1. Para esta prueba y para todas las presentes en este capítulo, se usarán tablas con distintos datos, pero todas corresponden al tipo matriz para modelos.

Paso 2. Supóngase que se tiene una variable llamada datos que se desea evaluar para la normalidad, usar la prueba de Shapiro-Wilk para evaluar la normalidad:

```
resultado_shapiro <- shapiro.tesshapt(datos)
```

Output

Paso 1. Visualizar los resultados:

```
print(resultado_shapiro)
        Shapiro-Wilk normality test
data:  datos$Taxa_S
W = 0.96155, p-value = 0.6319
```

Paso 2. Para visualizar la normalidad, usar el gráfico Q-Q (cuantil-cuantil) si los datos siguen una distribución normal:

```
qqnorm(datos)
qqline(datos)
```

Paso 3. Interpretar la prueba de Shapiro-Wilk:

- Valor p: si el valor p es mayor que el nivel de significancia elegido (generalmente 0.05), no hay suficiente evidencia para rechazar la hipótesis nula de que los datos provienen de una distribución normal. Es decir, una *p-value* alta sugiere que los datos siguen una distribución normal.
- Gráfico de Q-Q (cuantil-cuantil): si los puntos del gráfico Q-Q se ajustan aproximadamente a la línea diagonal, sugiere que los datos se ajustan a una distribución normal. Sin embargo, si los puntos se desvían significativamente de la línea diagonal, podría indicar que los datos no siguen una distribución normal.

Prueba de Levene

Nos permite verificar la homogeneidad de la varianza, en este caso es preciso instalar la librería «car».

Input

Paso 1. Supóngase que se tienen dos grupos (A y B) y se quiere evaluar la homocedasticidad.

Paso 2. Usar la prueba de Levene para evaluar la homocedasticidad:

```
library(car)

resultado_levene <- leveneTest(datos ~ grupo, data = datos)
```

Output

Paso 1. Imprimir el resultado:

```
print(resultado_levene)

Levene's Test for Homogeneity of Variance (center =
median)
      Df F value Pr(>F)
group  2  2.1094 0.1559
      15
```

Paso 2. Usar el gráfico de Residuos vs. Ajustes para visualizar la homocedasticidad:

```
modelo <- lm(datos ~ grupo, data = datos)
plot(modelo, which = 1)
```

Paso 3. Para interpretar la prueba de Levene:

- Valor p: si el valor p es mayor que el nivel de significancia elegido (generalmente 0.05), no hay suficiente evidencia para rechazar la hipótesis nula de que las varianzas son iguales en todos los grupos. Es decir, una *p-value* alta sugiere homocedasticidad.
- Gráfico de Residuos vs. Ajustes: en el gráfico de Residuos vs. Ajustes, si los puntos están dispersos aleatoriamente alrededor de la línea horizontal en cero, sugiere que la varianza es constante en todos los niveles de la variable

independiente. Sin embargo, si los puntos forman un patrón específico (por ejemplo, un embudo), sugiere que la homocedasticidad podría no cumplirse.

Paso 4. Si cumplen estos supuestos, se puede realizar alguna de las siguientes pruebas paramétricas.

Prueba t de Student

Adecuada para comparar dos grupos de muestras pequeñas e incluida en R base a través de la función t.test().

Input

Paso 1. Supóngase que se tienen dos conjuntos de datos que se quieren comparar, llamémoslos grupo1 y grupo2.

Paso 2. Cargar estos datos desde un archivo CSV, TXT, una base de datos u otro origen de datos. Por ejemplo, si se tienen los datos en un archivo CSV llamado datos.csv, se pueden cargar de la siguiente manera:

```
datos <- read.csv("datos.csv")
```

Paso 3. Para realizar la prueba t de Student en R, emplear la función t.test():

```
resultado <- t.test(grupo1, grupo2)
```

Paso 4. Los vectores grupo1 y grupo2 contienen los datos. Si los datos no están en un formato adecuado para comparar directamente (por ejemplo, si están en un marco de datos y se necesita seleccionar una columna específica), se puede realizar así:

```
resultado <- t.test(datos$columna_grupo1, datos$columna_grupo2)
```

Output

Paso 1. Una vez realizada la prueba t de Student, ver los resultados al imprimir el objeto resultado, lo que mostrará los resultados, con el valor de t, el valor p y otros detalles relevantes:

```
print(resultado)
        Welch Two Sample t-test
data:  datos$Taxa_S[datos$TRATAMIENTO == "ECB"] and
datos$Taxa_S[datos$TRATAMIENTO == "ECD"]
```

```
t = 3.3651, df = 8.6431, p-value = 0.008815
alternative hypothesis: true difference in means is not
equal to 0
95 percent confidence interval:
  3.181152 16.485515
sample estimates:
mean of x mean of y
 26.16667  16.33333
```

Paso 2. Si se desea visualizar los resultados de la prueba t de Student, hacerlo mediante gráficos. Por ejemplo, se puede crear un diagrama de caja para mostrar la distribución de los datos en cada grupo. Esto creará un diagrama de caja con los datos de los dos grupos, donde cada caja representa la distribución de los datos en cada grupo:

```
boxplot(grupo1, grupo2, names = c("Grupo 1", "Grupo 2"), col =
c("blue", "red"))
```

ANOVA

Utilizar ANOVA para comparar las medias de tres o más grupos independientes.

Input

Paso 1. Cargar los datos. Supóngase que se tiene un conjunto de datos en el que se quiere realizar un ANOVA. Se pueden cargar los datos desde un archivo CSV, una base de datos u otro origen de datos. Por ejemplo, si se tiene los datos en un archivo CSV llamado datos.

```
datos <- read.csv("datos.csv")
```

Paso 2. Para el cálculo del ANOVA, realizar un ANOVA en R y usar la función aov(). Esta función ajusta un modelo lineal a los datos y realiza un ANOVA para comparar las medias de varios grupos:

```
modelo_anova <- aov(respuesta ~ tratamiento, data = datos)
```

Paso 3. Respuesta es la variable dependiente, es decir, la variable que se quiere analizar, y tratamiento es la variable independiente, variable que define los grupos.

Output

Paso 1. Una vez que se haya realizado el ANOVA, ver los resultados imprimiendo el objeto modelo_anova. Esto mostrará un resumen del ANOVA, que incluye la tabla de análisis de varianza con los valores F, los valores p y otras estadísticas relevantes:

```
print(summary(modelo_anova))

            Df Sum Sq Mean Sq F value Pr(>F)
TRATAMIENTO  2  293.4  146.72   4.079 0.0385 *
Residuals   15  539.5   35.97
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Paso 2. En caso de que el ANOVA muestre diferencias significativas entre los grupos, es posible que se quieran realizar pruebas *post hoc* para determinar cuáles grupos difieren entre sí. Para realizar pruebas *post hoc* en R, usar funciones como TukeyHSD() o pairwise.t.test(). Por ejemplo, con TukeyHSD(), se calculará las diferencias entre los grupos y sus intervalos de confianza:

```
posthoc <- TukeyHSD(modelo_anova)
```

Paso 3. Si se desea visualizar los resultados del ANOVA, hacerlo mediante gráficos. Por ejemplo, crear un diagrama de barras para mostrar las medias de cada grupo. Esto creará un diagrama de barras con las medias de cada grupo en el eje y, y los tratamientos en el eje x:

```
barplot(tapply(datos$respuesta, datos$tratamiento, mean),
        names.arg = levels(datos$tratamiento),
        ylab = “Media de respuesta”,
        xlab = “Tratamiento”,
        col = “lightblue”)
```

Regresión lineal

Para llevar a cabo una regresión lineal se utiliza la función lm(), disponible en R base. Nótese, en el paso 2, el primer argumento de la función; este se conoce como fórmula e indica «variable dependiente ~ variable independiente».

Input

Paso 1. Cargar los datos. Supóngase que se tiene un conjunto de datos en el que se quiere realizar una regresión lineal, se cargan los datos desde un archivo CSV, una base de datos u otro origen de datos. Por ejemplo, si se tiene los datos en un archivo CSV llamado datos.csv, puede cargarse así:

```
datos <- read.csv(“datos.csv”)
```

Paso 2. Para ajustar un modelo de regresión lineal en R, usar la función lm(). Esta función ajusta una línea recta a los datos que minimiza la suma de los cuadrados de las diferencias entre los valores observados y los valores predichos por el modelo:

```
modelo <- lm(respuesta ~ tratamiento, data = datos)
```

Paso 3. Respuesta es la variable dependiente, es decir, la variable que se quiere analizar, y tratamiento, la variable independiente que define los grupos.

Output

Paso 1. Para la interpretación del modelo, una vez que se haya ajustado el modelo de regresión lineal, ver un resumen del modelo imprimiendo el objeto modelo. Esto mostrará un resumen del modelo de regresión lineal, que incluye los coeficientes de regresión, los valores p, el coeficiente de determinación (R-cuadrado) y otras estadísticas relevantes:

```
print(summary(modelo))

Call:
lm(formula = Taxa_S ~ TRATAMIENTO, data = datos)

Residuals:
    Min      1Q  Median      3Q     Max
-10.333  -4.083   0.250   4.542   7.833

Coefficients:
               Estimate Std. Error t value     Pr(>|t|)
(Intercept)      26.167      2.448  10.687 0.0000000207 ***
TRATAMIENTOECD   -9.833      3.462  -2.840       0.0124 *
TRATAMIENTOECI   -5.833      3.462  -1.685       0.1127
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 5.997 on 15 degrees of freedom
Multiple R-squared:  0.3523,    Adjusted R-squared:  0.2659
F-statistic: 4.079 on 2 and 15 DF,  p-value: 0.03849
```

Paso 2. Si se desea visualizar el modelo de regresión lineal, hacerlo mediante un diagrama de dispersión junto con la línea de regresión ajustada. Por ejemplo, crear un diagrama de dispersión para visualizar la relación entre la variable independiente y la variable dependiente, junto con la línea de regresión. Esto creará un diagrama de dispersión con los datos y la línea de regresión ajustada en rojo:

```
plot(datos$variable_independiente, datos$variable_dependiente,
     xlab = "Variable independiente",
     ylab = "Variable dependiente",
     main = "Regresión lineal")

abline(modelo, col = "red")
```

MANOVA

Prueba también incluida en R base, mediante la función manova(). Aquí también se utiliza la notación de fórmula (paso 2) para establecer la relación entre las variables.

Input

Paso 1. Crear un diagrama de dispersión con los datos y la línea de regresión ajustada en rojo.

Paso 2. Cargar los datos. Supóngase que se tiene un conjunto de datos en el que se quiere realizar un MANOVA, cargar estos datos desde un archivo CSV, una base de datos u otro origen de datos. Por ejemplo, si se tiene los datos en un archivo CSV llamado datos.csv, pueden cargarse así:

```
datos <- read.csv("datos.csv")
```

Paso 3. Para ajustar un modelo MANOVA en R, usar la función manova(). Esta función ajusta un modelo multivariante que evalúa la relación entre múltiples variables dependientes y una o más variables independientes:

```
modelo_manova <- manova(cbind(variable_dependiente1, variable_
dependiente2) ~ variable_independiente, data = datos)
```

Paso 4. variable_dependiente1 y variable_dependiente2 son las variables que se quiere analizar, y variable_independiente es la variable que define los grupos.

Output

Paso 1. Una vez que se haya ajustado el modelo MANOVA, ver un resumen del modelo imprimiendo el objeto modelo_manova. Esto mostrará un resumen del modelo MANOVA, que incluye las estadísticas multivariadas como Pillai's trace, Wilks' lambda, Hotelling-Lawley trace y Roy's largest root, junto con los valores p y otras estadísticas relevantes:

```
print(summary(modelo_manova))

            Df Pillai approx F num Df den Df Pr(>F)
TRATAMIENTO  2  0.385   1.7879      4     30 0.1573
Residuals   15
```

Paso 2. En caso de que el MANOVA muestre diferencias significativas entre los grupos, realizar pruebas post hoc para determinar cuáles grupos difieren entre sí en cada variable dependiente. Para realizar pruebas post hoc en R, se puede usar funciones como TukeyHSD() o pairwise.manova(). Por ejemplo, con TukeyHSD(). Esto calculará las diferencias entre los grupos y sus intervalos de confianza para cada variable dependiente:

```
posthoc <- TukeyHSD(modelo_manova)
```

Paso 3. Si se desea visualizar los resultados del MANOVA, hacerlo mediante gráficos. Por ejemplo, crear un gráfico de barras para mostrar las medias de cada variable dependiente por grupo. Esto creará un diagrama de barras con las medias de las variables dependientes para cada grupo en el eje y, y las categorías de la variable independiente en el eje x:

```
barplot(colMeans(datos[, c("variable_dependiente1",
"variable_dependiente2")]),
        names.arg = levels(datos$variable_independiente),
        ylab = "Media de variables dependientes",
        xlab = "Variable independiente",
        col = "lightblue")
```

Pruebas no paramétricas

En los análisis no paramétricos también se busca evaluar si existen diferencias significativas. Es decir, si el valor p es menor que un nivel de significancia predefinido (generalmente 0.05).

Prueba de Kruskal-Wallis

Nos permite comparar tres o más grupos independientes y es posible llevarla a cabo sin necesidad de instalar librerías mediante kruskal.test().

Input

Paso 1. Cargar los datos. Supóngase que se tiene un conjunto de datos en el que se quiere realizar una prueba de Kruskal-Wallis, cargar estos datos desde un archivo CSV, una base de datos u otro origen de datos. Por ejemplo, si se tienen los datos en un archivo CSV llamado datos.csv, pueden cargarse así:

```
datos <- read.csv("datos.csv")
```

Paso 2. Para realizar la prueba de Kruskal-Wallis en R, usar la función kruskal.test(). Esta función realiza una prueba no paramétrica para determinar si hay diferencias significativas entre las medianas de dos o más grupos:

```
resultado_kruskal <- kruskal.test(variable_dependiente ~
variable_independiente, data = datos)
```

Paso 3. variable_dependiente es la variable que se quiere analizar y variable_independiente, la variable que define los grupos.

Output

Paso 1. Una vez que se haya realizado la prueba de Kruskal-Wallis, ver los resultados imprimiendo el objeto resultado_kruskal. Esto mostrará los resultados de la prueba de Kruskal-Wallis, incluyendo el valor de estadístico de la prueba y el valor p:

```
print(resultado_kruskal)

        Kruskal-Wallis rank sum test

data:  Taxa_S by TRATAMIENTO
Kruskal-Wallis chi-squared = 6.3117, df = 2, p-value = 0.0426
```

Paso 2. Si la prueba de Kruskal-Wallis muestra diferencias significativas entre los grupos, realizar pruebas post hoc para determinar cuáles grupos difieren entre sí. Algunas opciones para pruebas post hoc incluyen pruebas de comparaciones múltiples como Dunn's test o la prueba de Conover-Iman. Por ejemplo, con la prueba de Dunn's test (usando el paquete dunn.test):

```
install.packages("dunn.test")
library(dunn.test)

resultado_dunn <- dunn.test(datos$variable_dependiente,
g = datos$variable_independiente, method = "bonferroni")
```

Prueba de correlación de Spearman

En datos con una distribución distinta a la normal, podemos llevar a cabo una prueba de correlación utilizando cor.test(), disponible en R base.

Input

Paso 1. Cargar los datos. Supóngase que se tienen dos variables que se quieren correlacionar, cargar estos datos desde un archivo CSV, una base de datos u otro origen de datos. Por ejemplo, si se tienen los datos en un archivo CSV llamado datos.csv, pueden cargarse así:

```
datos <- read.csv("datos.csv")
```

Paso 2. Para la ejecución de la prueba de Correlación de Spearman, usar la función cor.test() con el método Spearman. Esta función calcula la correlación de rango de Spearman entre dos variables:

```
resultado_spearman <- cor.test(datos$variable1,
datos$variable2, method = "spearman")
```

Paso 3. variable1 y variable2 son las dos variables que se quieren correlacionar. Por ejemplo, si estás tratando de correlacionar el IHH con la riqueza:

```
resultado_spearman <- cor.test(datos$tRiqueza, datos$IHH,
method = "spearman")
```

Output

Paso 1. Una vez que se haya realizado la prueba de correlación de Spearman, ver los resultados imprimiendo el objeto resultado_spearman.

Paso 2. Se mostrarán los resultados de la prueba de correlación de Spearman, incluyendo el coeficiente de correlación de Spearman, el valor p y otras estadísticas relevantes:

```
print(resultado_spearman)

        Spearman's rank correlation rho

data:  datos_moscas$Riqueza and datos_moscas$IHH
S = 4934.1, p-value = 0.6076
alternative hypothesis: true rho is not equal to 0
sample estimates:
        rho
-0.09768012
```

Prueba de Mann-Whitney U

Es una prueba no paramétrica para determinar si hay diferencias significativas entre las distribuciones de dos grupos.

Input

Paso 1. Cargar los datos. Supóngase que se tienen dos grupos y se quiere comparar las distribuciones de una variable entre estos grupos, cargar estos datos desde un archivo CSV, una base de datos u otro origen de datos. Por ejemplo, si se tienen los datos en un archivo CSV llamado datos.csv, pueden cargarse así:

```
datos <- read.csv("datos.csv")
```

Paso 2. La ejecución de la prueba de Mann-Whitney U usa la función wilcox.test():

```
resultado_mann_whitney <- wilcox.test(variable_dependiente ~
grupo, data = datos)
```

Paso 3. variable_dependiente es la variable que se quiere comparar entre los dos grupos, y grupo es la variable que define los grupos.

Output

Paso 1. Para ver los resultados e imprimir el objeto resultado_mann_whitney.

Paso 2. Se mostrarán los resultados de la prueba de Mann-Whitney U e incluirá el estadístico U de Mann-Whitney, el valor p y otras estadísticas relevantes:

```
print(resultado_mann_whitney)

        Wilcoxon rank sum test with continuity correction

data:  Taxa_S by TRATAMIENTO
W = 33.5, p-value = 0.01612
alternative hypothesis: true location shift is not equal to 0
```

PERMANOVA (Permutational Multivariate Analysis of Variance)

Input

Paso 1. Cargar los datos. Supóngase que se tiene un conjunto de datos multivariados en el que se desea realizar una PERMANOVA, cargar estos datos desde un archivo CSV, una base de datos u otro origen de datos. Por ejemplo, si se tiene los datos en un archivo CSV llamado datos.csv, pueden cargarse así:

```
datos <- read.csv("datos.csv")
```

Paso 2. Para la carga de paquetes y realizar la PERMANOVA, es necesario cargar el paquete vegan en R, que proporciona funciones para análisis de ecología de comunidades:

```
library(vegan)
```

Paso 3. Para realizar la PERMANOVA en R, se puede usar la función adonis() del paquete vegan. Esta función realiza una PERMANOVA basada en distancias, que es útil para datos multivariados:

```
resultado_permanova <- adonis(datos ~ grupo, data = datos,
permutations = 999)
```

Paso 4. Datos es una matriz o un marco de datos que contiene las variables multivariadas, y grupo es la variable que define los grupos.

Output

Paso 1. Una vez que se haya realizado la PERMANOVA, ver los resultados imprimiendo el objeto resultado_permanova.

Paso 2. Se mostrarán los resultados de la PERMANOVA, incluyendo la estadística F, el valor p y otras estadísticas relevantes:

```
print(resultado_permanova)

Permutation: free
Number of permutations: 999

Terms added sequentially (first to last)
```

```
            Df SumsOfSqs  MeanSqs F.Model      R2 Pr(>F)
TRATAMIENTO  2   0.15135 0.075674  3.4418 0.31456  0.046 *
Residuals   15   0.32980 0.021987         0.68544
Total       17   0.48115                  1.00000
---
Signif. Codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
...
```

Modelo Lineal Generalizado (GLM)

Input

Paso 1. Cargar los datos. Supóngase que se tiene un conjunto de datos en el que se quiere ajustar un GLM, cargar estos datos desde un archivo CSV, una base de datos u otro origen de datos. Por ejemplo, si se tienen los datos en un archivo CSV llamado datos.csv, pueden cargarse así:

```
datos <- read.csv(“datos.csv”)
```

Paso 2. Para ajustar un GLM en R, usar la función glm(). Esta función permite especificar una función de enlace y una distribución de errores adecuada para los datos:

```
modelo <- glm(variable_dependiente ~ variable_independiente1
+ variable_independiente2,
              data = datos, family = family(distribution =
“distribution_name”))
```

Paso 3. variable_dependiente es la variable de respuesta que se está tratando de modelar; variable_independiente1 y variable_independiente2 son las variables predictoras, y distribution_name es la distribución de errores que se desea utilizar en el modelo. Por ejemplo, si se está ajustando un modelo con una distribución de errores binomial y una función de enlace logit:

```
modelo <- glm(variable_dependiente ~ variable_independiente1 +
variable_independiente2,
              data = datos, family = binomial(link = “logit”))
```

Output

Paso 1. Una vez que se haya ajustado el modelo GLM, ver un resumen del modelo imprimiendo el objeto modelo. Esto mostrará un resumen del modelo GLM, que incluye los coeficientes estimados, los errores estándar, los valores z, los valores p y otras estadísticas relevantes:

```
print(summary(modelo))
```

Paso 2. Para hacer gráficos de diagnóstico y evaluar la adecuación del modelo, algunos gráficos comunes incluyen gráficos de residuos y gráficos Q-Q.

- Gráfico de residuos: este comando genera un gráfico de residuos estandarizados vs. los valores ajustados por el modelo. Los residuos estandarizados son los residuos divididos por su desviación estándar, lo que ayuda a identificar valores atípicos y patrones en los residuos:

```
plot(modelo, which = 1)
```

- Gráfico Q-Q: este comando genera un gráfico cuantil-cuantil (Q-Q plot) de los residuos estandarizados contra cuantiles teóricos de una distribución normal. Ayuda a evaluar si los residuos se distribuyen de manera aproximadamente normal:

```
plot(modelo, which = 2)
```

Paso 3. Para visualizar el gráfico de la relación entre variables, se puede hacer mediante un diagrama de dispersión con la línea de regresión ajustada. Este comando genera un diagrama de dispersión de los datos y traza la línea de regresión ajustada por el modelo GLM. Esto permite visualizar la relación entre la variable independiente y la variable dependiente, teniendo en cuenta el modelo ajustado:

```
plot(datos$variable_independiente, datos$variable_dependiente)
abline(modelo, col = “red”)
```

Recapitulando

Al finalizar este capítulo, resulta fundamental que se sintetice y reflexione sobre las herramientas y conceptos que se han abordado en relación con las pruebas de hipótesis estadísticas en estudios ecológicos. El objetivo ha sido el de proporcionar una comprensión integral y aplicable de cómo estas pruebas se implementan y su relevancia en la investigación ecológica.

En primer lugar, se destacó la importancia de formular hipótesis claras y precisas, distinguiendo entre la hipótesis nula (H_0) y la hipótesis alternativa (H_1). Este proceso es fundamental para estructurar adecuadamente las pruebas estadísticas. Asimismo, se ha subrayado la relevancia de seleccionar un nivel de significancia apropiado y de elegir la prueba estadística correcta en función de los datos y el contexto de la investigación. Estas decisiones son cruciales para asegurar la validez y la fiabilidad de los resultados obtenidos.

Además, se abordaron los errores de tipo I y tipo II, conceptos clave que influyen en la interpretación de cualquier prueba de hipótesis. Comprender estos errores permite apreciar la importancia de equilibrar el riesgo de rechazar una hipótesis verdadera (error de tipo I) y el riesgo de no rechazar una hipótesis falsa (error de tipo II). Estos conceptos nos orientan a diseñar estudios que minimicen el impacto de estos errores y maximicen la potencia de las pruebas.

En la sección dedicada a las pruebas paramétricas y no paramétricas, se ha explicado cómo seleccionar la prueba adecuada según las características de los datos. Las pruebas paramétricas, como la prueba t de Student y ANOVA, requieren supuestos específicos sobre la distribución de los datos. Por otro lado, las pruebas no paramétricas, como la prueba de Kruskal-Wallis y la prueba de Mann-Whitney U, ofrecen alternativas útiles cuando esos supuestos no se cumplen.

También se introdujo el PERMANOVA, una herramienta valiosa para el análisis multivariante en ecología, y los Modelos Lineales Generalizados (GLM), que han revolucionado la forma en que se abordan los datos ecológicos. Estos modelos avanzados permiten manejar una amplia gama de tipos de datos y complejas relaciones entre variables, lo que es esencial en el análisis de fenómenos ecológicos multifacéticos.

Finalmente, al concluir este capítulo, es evidente que el dominio de las pruebas de hipótesis estadísticas es indispensable para llevar a cabo investigaciones ecológicas robustas. La capacidad de seleccionar y aplicar adecuadamente estas pruebas, junto con una interpretación cuidadosa de los resultados, proporciona herramientas poderosas para desentrañar los complejos patrones y relaciones en los ecosistemas. A medida que se avanza en la exploración de métodos estadísticos más sofisticados, es crucial mantener una comprensión sólida de los fundamentos discutidos aquí para seguir contribuyendo al avance del conocimiento en la disciplina.

Capítulo 5
Índices para medir la diversidad biológica

Matías I. Dufek y Darío D. Larrea

La diversidad biológica o biodiversidad[1] hace referencia a la variedad de vida a nivel de genes, especies y ecosistemas. La complejidad de la estructura de las comunidades de un sitio en particular puede describirse mediante diferentes índices de diversidad, utilizados para analizar la magnitud de los cambios en la composición de especies entre sitios o a lo largo de gradientes temporales. Además, es una herramienta fundamental para evaluar los procesos de fragmentación ambiental producto de la actividad antrópica. En este sentido, una adecuada comprensión de la biodiversidad de un área es indispensable para el diseño y gestión de programas de conservación.

Componentes de la diversidad biológica

La diversidad biológica tiene tres componentes: la diversidad alfa, beta y gamma. La alfa (α) es la diversidad de especies a nivel local, la beta (β) se entiende como las diferencias de composición entre las unidades de estudio de la diversidad alfa y, por último, la gamma[2] (γ) es la diversidad de especies a nivel regional.

Estos componentes se pueden medir con distintos índices y proporcionan información indispensable para comprender la estructura y dinámica de las comunidades y sus respuestas a perturbaciones naturales y disturbios antrópicos.

Diversidad alfa

Los métodos para medir la diversidad alfa pueden clasificarse en dos grandes grupos:

1. La diversidad biológica debe ser tratada más seriamente como un recurso global para ser preservado. Esto resulta indispensable ya que gran parte de la diversidad se está perdiendo de forma irreversible debido a la extinción causada por la destrucción de los hábitats naturales, especialmente en los trópicos.

2. La diversidad se puede calcular como la suma de la diversidad alfa promedio + la diversidad beta.

1. Métodos basados en la medición del número de especies. En este grupo se encuentran la riqueza específica (S), Chao de segundo orden (Chao 2), Jackknife de primer orden (Jack1), Jacknife de segundo orden (Jack 2), Bootstrapping, entre otros, siendo los índices mencionados los más utilizados actualmente en ecología.

2. Métodos basados en la estructura de la comunidad. Estos a su vez se pueden clasificar en dos subgrupos:

    a) Índices basados en dominancia, le dan más valor a la representatividad de las especies con mayor abundancia sin evaluar la contribución del resto de las especies, como el índice de Simpson y el índice de Berger-Parker.

    b) Índices basados en la equidad, expresan la uniformidad de los valores de importancia a través de todas las especies de la unidad muestral, como los índices de Shannon-Wiener y de Pielou.

Índices basados en la dominancia

El índice de Simpson, simbolizado tradicionalmente con la letra *D*, es uno de los índices de dominancia más usado en los estudios de ecología. Este índice expresa la probabilidad de que dos individuos tomados al azar de una muestra sean de la misma especie. Los valores obtenidos varían entre cero y uno. Valores cercanos a uno representan la dominancia de una especie por sobre las demás, es decir, son ecosistemas con menos diversidad o con mayor homogeneidad de las especies.

$$D = \sum_{i=1}^{S} p_i^2$$

En la fórmula, *S* representa el número de especies, y *pi* es la proporción de cada especie.

Cuando el índice de dominancia de Simpson es menor a uno, indica mayores valores de diversidad. Esta relación inversa entre el valor del índice y el nivel de diversidad del área hace que la interpretación del índice resulte un poco confusa. Por esta razón, muchos autores sugieren el uso del inverso de Simpson, que se considera un buen indicador de la diversidad de un área. El inverso de Simpson se calcula empleando la siguiente fórmula:

$$C_{inv} = \frac{1}{D}$$

El índice de Berger Parker, simbolizado con la letra *d*, es un índice de dominancia que varía entre cero y uno; cuanto más se acerca a 1, significa que mayor es la dominancia y menor la equidad.

$$d = \frac{N_{max}}{N_T}$$

En la fórmula, N_{max} representa el número de individuos de la especie más abundante, mientras que N_T es la abundancia total de individuos, contemplando todas las especies colectadas.

Valores cercanos a 1 de este índice se interpretan como un aumento en la dominancia y una disminución de la equidad.

Los valores altos de este índice, al igual que los obtenidos con el índice de Simpson, indican valores bajos de diversidad.

Índices basados en la equidad

Los índices más utilizados en estudios ecológicos basados en la equidad son Shannon-Wiener y de Pielou.

El índice Shannon-Wiener, simbolizado por la letra *H´*, es el índice de equidad más utilizado en los estudios ecológicos. Este índice expresa la uniformidad de los valores de importancia a través de todas las especies de la muestra y presenta valores positivos que van desde cero hasta el logaritmo de *S* (riqueza de especies).

$$H' = -\sum_{i=1}^{S} p_i \ln p_i$$

En la fórmula, *S* es el número de especies totales colectadas, mientras que *pi* es la abundancia relativa de la especie i, que se calcula como la proporción de individuos de la especie i (*ni*) respecto del total de individuos (*N*), como se detalla en la siguiente fórmula:

$$pi = \frac{n_i}{N}$$

Valores altos de este índice indican que los ambientes presentan una diversidad elevada.

El índice de Pielou, por su parte, es simbolizado tradicionalmente con la letra *J´*. Se calcula por medio de la proporción de la diversidad observada con relación a la máxima diversidad esperada.

$$J' = \frac{H'}{ln\ (S)}$$

En la fórmula, *H´* es el índice de Shannon-Wiener, y *S* es el número total de especies presentes (riqueza de especies). Este índice expresa el grado de uniformidad en la distribución de las abundancias entre especies, es decir, valores altos de este índice indican que los ambientes presentan una diversidad elevada.

Índices de diversidad verdadera

La diversidad verdadera, también conocida como los *índices de Hill*, es una forma de medir la biodiversidad que se expresa en términos de órdenes de diversidad (Q). Los índices de Hill unifican varios índices de diversidad tradicionales en un solo marco teórico, lo que facilita la comparación entre diferentes comunidades o tratamientos. Asimismo, son sólidos matemáticamente y tienen una amplia aplicabilidad en diferentes contextos ecológicos, desde estudios locales hasta comparaciones a escala regional o global. Estos índices son herramientas valiosas en la ecología debido a su capacidad para proporcionar medidas claras, intuitivas y comparables de la diversidad biológica, además de su flexibilidad para enfatizar diferentes aspectos de la biodiversidad mediante la elección de diferentes valores de q. Esto los convierte en una potente herramienta para los estudios ecológicos que evalúan la estructura y la función de las comunidades biológicas.

La *diversidad de orden Q=0 (riqueza de especies)* es un parámetro que simplemente cuenta el número de especies presentes (S) en la comunidad, sin tener en cuenta su abundancia. Este índice es equivalente a la riqueza de especies.

$$D^0 = S$$

La *diversidad de orden Q=1 (exponencial* del índice de Shannon) es un parámetro que toma en cuenta tanto la riqueza de especies como la abundancia relativa de cada especie. Este índice es sensible a las especies raras y comunes y se calcula utilizando el exponencial del índice de Shannon.

$$D^1 = p_i$$

En la fórmula, *pi* es la proporción de la especie *i*.

La *diversidad de orden Q=2 (índice de Simpson)* es un enfoque da una mayor importancia a las especies dominantes, es decir, a las más abundantes. Se calcula como el inverso del índice de Simpson.

$$D^2 = \frac{1}{D}$$

Diversidad beta

La diversidad beta expresa el grado de diferencia entre unidades muestrales (disimilitud) y evidencia un cambio biótico a través de gradientes ambientales espaciales o temporales. Por esta razón, muchos de los índices de diversidad beta se construyen a partir de *índices de similitud* (*s*) y quedan relacionadas estas aproximaciones en la siguiente fórmula:

$$D_\beta = 1 - s$$

Índices de similitud

Los índices de Jaccard y Sørensen son ampliamente utilizados para la valoración de la similitud en la composición de las comunidades y, en consecuencia, su complemento (la disimilitud) es la aproximación más usada para explicar la diversidad beta. Estos dos índices presentan fórmulas de calcular muy similares, diferenciándose solamente en la mayor importancia que le da el índice de Sørensen a las especies compartidas.

Fórmula del índice de Jaccard:

$$Ja = \frac{c}{a + b - c}$$

Fórmula del índice de Sørensen:

$$Sø = \frac{2c}{a + b}$$

En las fórmulas, *a* es el número de especies exclusivas de la comunidad A, *b* es el número de especies exclusivas de la comunidad B, y *c* es el número de especies compartidas por las dos comunidades. Estos índices varían de cero a uno; valores cercanos a uno indican mayor similitud entre dos comunidades.

La variación de estos índices se puede dar por medio de dos procesos, en otros términos, la diversidad beta se puede dividir o particionar en dos componentes:

1. El recambio de especies, es decir que entre la comunidad A y B las diferencias están mediadas por el reemplazo total o parcial de las especies. El ejemplo clásico de este proceso ocurre en ambientes muy contrastantes, como cuando comparamos bosques y pastizales.

2. La pérdida/ganancia (diferencias de riqueza), es decir que entre las comunidades A y B lo que regula su diferencia es la pérdida/ganancia de especies entre las comunidades. Este componente es conocido como anidamiento, debido a que uno de los ambientes es producto de una simplificación de la estructura del otro ambiente. Ejemplos clásicos para evidenciar este componente de la diversidad beta son las comparaciones de un mismo ambiente en periodos de tiempos distintos o diferentes sitios con niveles de actividad antrópica (fragmentación del hábitat).

Estos elementos se relacionan de la siguiente forma: para beta en base Jaccard, β*jtu* es la diversidad beta explicada por el reemplazo de especies, y β*jne* es la diversidad beta explicada por la pérdida/ganancia de especies.

$$\beta_{Jac} = beta.jtu + beta.jne$$

Para beta en base Sørensen, *betaSIM* es la diversidad beta explicada por el reemplazo de especies, y *betaSNE* es la diversidad beta explicada por la pérdida/ganancia de especies.

$$\beta_{SØR} = betaSIM + betaSNE$$

Medir e interpretar los índices de diversidad alfa en R

En R se pueden estimar fácilmente todos los métodos basados en la medición del número de especies (riqueza específica, Chao 2, Jacknife de primer orden, Jacknife de segundo) empleando el siguiente script.

Input

Paso 1. Definir la carpeta donde se encuentran los datos:

```
Mydir = ("C:/Users/Desktop/Datos")
setwd (Mydir)
```

Paso 2. Para cargar las bases de datos[3] que se empleará para los análisis, se usan dos tablas de datos, una de abundancia tipo 1 (datos-estudio) y otra de variables ambientales (variables):

```
datos-estudio = read.table("Datos.txt", header = TRUE, sep =
"\t", row.names = 1)
Variables = read.table("Variables.txt", header = TRUE, sep =
"\t", row.names = 1)
```

Paso 3. Para cargar la librería utilizada para los análisis, en este caso será *vegan:*

```
library(vegan)
attach(Variables)
```

3. Es importante tener en cuenta que las filas en las dos bases de datos que se cargarán tienen que ser iguales.

Paso 4. Correr el análisis para estimar la riqueza:

```
specpool(datos-estudio, Tratamiento)
```

Output

Paso 1. Observar la siguiente tabla:

	Species	Chao	chao.se	jack1	jack1.se	jack2	boot	boot.se	n
Ambiente1	75	126.2	21.08	107	23.63	120.5	89.77	11.33 3	3
Ambiente2	52	93.48	19.59	76.66	18.049	87.16	63.37	8.55 3	3
Ambiente3	57	110.48	25.93	82.33	19.27	93.5	68.59	9.37	3

Paso 2. Interpretar los resultados. La interpretación de estos resultados es muy intuitiva. Los valores de *species* representan la riqueza de especies observadas para cada tratamiento, mientras que los valores de *Chao*, *Jack1, Jack2* y *Boot* representan las estimaciones de la riqueza para cada tratamiento.

Medición de índices basados en dominancia

Los más utilizados en estudios ecológicos son el índice de Simpson y de Berger-Parker. El procedimiento para la estimación de estos índices se detalla a continuación.

Input

Paso 1. Definir la carpeta donde se encuentran los datos:

```
Mydir = ("C:/Users/Desktop/Datos")
setwd (Mydir)
```

Paso 2. Para cargar las bases de datos que se empleará para los análisis, utilizar una matriz de datos de abundancia de tipo 1 (especies por sitio):

```
Datos = read.table("Datos.txt", header = TRUE, sep = "\t", row.
names = 1)setwd (Mydir)
```

Paso 3. Para cargar la librería utilizada para los análisis, utilizar BiodiversityR y vegan:

```
library(BiodiversityR)
 library(kableExtra)
```

```
D <- diversityresult(abundancia$conservacion, index = "Simpson",
method = "each site")
```

Paso 4. Para correr el análisis y estimar los índices de diversidad basados en la equidad:

```
B = diversityresult(Datos, index=c("Berger"), method=c("each
site"))
iD = diversityresult(Datos, index=c("inverseSimpson"),
method=c("each site"))
```

Paso 5. Para confeccionar una tabla con los índices calculados:

```
    indicesdomi = data.frame(D, iD, B)

kable(indicesdomi, format = "markdown", col.names = c("Simpson",
"Inverso de Simpson", "Berger"))
```

Output

Paso 1. Observar la siguiente tabla:

	Simpson	**Inverso de Simpson**	**Berger**
Ambiente1	0.93077829	14.4463356	0.12946191
Ambiente2	0.87280776	7.862115	0.27067995
Ambiente3	0.83436931	6.0375284	0.34324554

Paso 2. Interpretar los resultados. El valor de Simpson varía entre 0 y 1. Un valor de cero (0) indica una diversidad infinita (todos los individuos son de diferentes especies). Si el valor es uno (1), indica mayor diversidad (todos los individuos son de una misma especie).

El valor del inverso de Simpson (1/D) varía desde 1 hasta el número total de especies en la muestra. En este índice, el valor de uno (1) indica que toda la comunidad está compuesta por una sola especie. Un valor alto indica una alta diversidad y significa que hay muchas especies presentes y que los individuos están distribuidos más equitativamente entre estas especies.

El valor de Berger-Parker varía de 0 a 1. Un valor de cero (0) indica que todas las especies tienen la misma abundancia (alta diversidad), mientras que un valor uno (1) indica que una sola especie domina completamente la comunidad (baja diversidad).

El inverso del índice de Berger-Parker (1/d) también se usa a veces para facilitar la interpretación, donde un valor más alto indica mayor diversidad.

Medición de los índices basados en la equidad
Estos índices se calculan mediante el siguiente procedimiento.

Input

Paso 1. Definir la carpeta donde se encuentran los datos:

```
Mydir = ("C:/Users/Desktop/Datos")
setwd (Mydir)
```

Paso 2. Para cargar las bases de datos que se emplearán para los análisis, y de la misma forma que en los análisis de dominancia, usar una tabla de abundancia tipo 1:

```
Datos = read.table("Datos.txt", header = TRUE, sep = "\t",
row.names = 1)
```

Paso 3. Para cargar la librería utilizada para los análisis, utilizar BiodiversityR y vegan:

```
library(BiodiversityR)
library(kableExtra)
```

Paso 4. Para correr el análisis y estimar los índices de diversidad basados en la equidad:

```
H = diversityresult(Datos, index=c("Shannon"), method=c
("each site"))
JP = H/log(specnumber(Datos))
```

Paso 5. Para confeccionar una tabla con los índices calculados:

```
indicesequi = data.frame(H, JP)

kable(indicesequi, format = "markdown", col.names = c("H,
Shannon", "JP, Pielou", "iD, inverseSimpson"))
```

Output

Paso 1. Observar la siguiente tabla:

	H, Shannon	JP, Pielou
Ambiente1	3.0597763	0.71315819
Ambiente2	2.6244421	0.64912413
Ambiente3	2.5077268	0.63162278

Paso 2. Interpretar los resultados. El valor de Shannon (H) generalmente varía entre 1.5 y 3.5 para la mayoría de las comunidades ecológicas, aunque puede ser menor o mayor en casos extremos. Valores bajos (cercanos a 0) indican baja diversidad, donde una o pocas especies predominan. Valores altos indican alta diversidad, donde hay muchas especies con una distribución más equitativa de individuos.

El valor de Pielou (J) varía entre 0 y 1. Un valor de cero (0) indica que la comunidad está completamente dominada por una sola especie (mínima equitatividad), mientras que un valor de uno (1) indica que todas las especies tienen la misma abundancia (máxima equitatividad).

Medición de los índices de diversidad verdadera

Para calcular la diversidad verdadera de órdenes Q=0, Q=1 y Q=2, se necesitan las matrices de abundancia. A continuación, se describe cómo calcular estos índices:

Input

Paso 1. Definir la carpeta donde se encuentran los datos:

```
Mydir <- ("C:/Users/Desktop/Datos")
setwd (Mydir)
```

Paso 2. Para cargar las bases de datos que se emplearán para los análisis, usar una tabla de abundancia de tipo 1:

```
Datos <- read.table("abundancia.txt", header = TRUE,
sep = "\t", row.names = 1)
```

Paso 3. Para cargar la librería utilizada para los análisis:

```
library(vegan)
library(ggplot2)
library(reshape2)
```

Paso 4. Función para calcular diversidad verdadera de orden Q:

```
diversidad_q <- function(Datos, q) {
  if (q == 0) {
    return(specnumber(Datos))
  } else if (q == 1) {
    return(exp(diversity(Datos, index = “shannon”)))
  } else if (q == 2) {
    return(diversity(Datos, index = “invsimpson”))
  } else {
    # Calcular la diversidad verdadera para cualquier
otro valor de q
    return((rowSums(Datos^q))^(1/(1 - q)))
  }
}
# Calcular diversidades para Q=0, Q=1, Q=2, Q=3
diversidad_q0 <- diversidad_q(Datos, 0)
diversidad_q1 <- diversidad_q(Datos, 1)
diversidad_q2 <- diversidad_q(Datos, 2)
```

Output

Paso 1. Crear una tabla con los resultados:

```
resultados <- data.frame(Categoria = rownames(Datos),
  Q0 = diversidad_q0,
  Q1 = diversidad_q1,
  Q2 = diversidad_q2,

# Mostrar la tabla
print(resultados)
```

Paso 2. Observar la tabla de resultado:

	Q0	Q1	Q2
ECB	73	21.32279	14.446336
ECI	55	13.36206	7.717607
ECD	51	12.16131	6.017080

Paso 3. Convertir la tabla a formato largo para ggplot2:

```
resultados_melt <- melt(resultados, id.vars = “Categoria”,
variable.name = “OrdenQ”, value.name = “Diversidad”)
resultados_melt$OrdenQ <- as.numeric(sub(“Q”, “”, resultados_
melt$OrdenQ))
```

Paso 4. Crear una gráfica con ggplot2:

```
ggplot(resultados_melt, aes(x = OrdenQ, y = Diversidad, color =
Categoria, group = Categoria)) +
  geom_line(size = 1.2) +
  geom_point(size = 3) +
  labs(title = “Diversidad Verdadera por Orden Q y Categoría”,
       x = “Orden Q”,
       y = “Diversidad Verdadera”,
       color = “Categoría”) +
  theme_minimal()
```

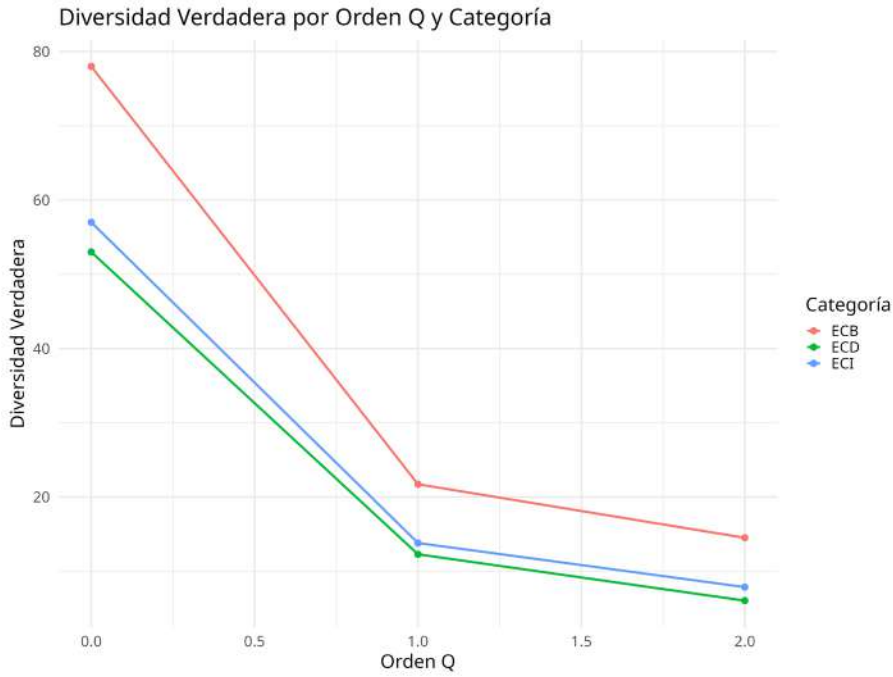

Figura 1. Gráfico de diversidad para tres ambientes en distintos estados de conservación: bueno (ECB), intermedio (ECI) y degradado (ECD).

De este modo se calcula la diversidad verdadera observada, lo que no debería presentar ningún problema en la mayoría de los casos, pero a veces, sobre todo en muestreos biológicos, el esfuerzo muestral entre sitios no es el mismo y dificulta la comparación de estos índices observados. Para solucionarlo, puede valerse de la función estimadeD(), de la librería iNext, para calcular índices mediante rarefacción y extrapolación, basándose en un mismo tamaño de muestra o en una misma cobertura muestral (esta última es la más usada en ecología). Aquí también se puede usar una matriz de abundancia, pero de tipo 2 (sitios por especies):

```
library(iNEXT)

estimateD(datos, datatype = “abundance”, base = “coverage”,
level = NULL)
```

Con level = NULL, la función calcula automáticamente el nivel de cobertura de muestra, pero también puede tomar un valor entre 0 y 1.

```
     m      Method     Order.q     SC        qD    qD.LCL
qD.UCL
ECB 3754.000 Extrapolation  0 0.993636 96.460124 71.921513
120.998735
ECB 3754.000 Extrapolation  1 0.993636 22.128519 20.732545
23.524492
ECB 3754.000 Extrapolation  2 0.993636 14.555693 13.720227
15.391159
ECI 1822.372   Rarefaction  0 0.993636 54.307913 49.632720
58.983106
ECI 1822.372   Rarefaction  1 0.993636 13.741831 13.030140
14.453521
ECI 1822.372   Rarefaction  2 0.993636  7.855878  7.390363
8.321393
ECD 2068.921 Extrapolation  0 0.993636 61.495989 47.443440
75.548537
ECD 2068.921 Extrapolation  1 0.993636 12.478139 11.415921
13.540358
ECD 2068.921 Extrapolation  2 0.993636  6.048698  5.443428
6.653969
```

En estos resultados, el valor de nuestros índices se encuentra en la columna qD, y como se observa en Method. Para los sitios ECB y ECD, se realizó una extrapolación, lo que significa que la cobertura muestral de nuestros datos era menor que la elegida (SC = 0,99) en estos sitios y en ECI, una rarefacción porque en este sitio la cobertura muestral era mayor a la elegida para el cálculo.

Paso 5. Interpretar los resultados. La interpretación de los índices de diversidad verdadera de órdenes Q0, Q1 y Q2 se realiza de la siguiente manera:

- Orden 0 (q=0): mide solo la riqueza de especies. Es útil cuando se quiere saber cuántas especies hay en total, sin considerar la abundancia de estas. Tomando como ejemplo los resultados obtenidos, se puede decir que los ambientes ECB tienen mayor diversidad de orden Q0 (73).

- Orden 1 (q=1): mide una combinación de riqueza y equitatividad. Es útil para obtener una medida equilibrada de diversidad que refleje tanto el número de especies como su distribución equitativa. Es sensible tanto a especies comunes como raras. En los resultados se aprecia que la comunidad ECB tiene una diversidad equivalente a una comunidad con aproximadamente 21.32 especies equitativamente distribuidas.

- Orden 2 (q=2): pone más énfasis en las especies abundantes. Es útil para entender la diversidad en términos de dominancia, es decir, cómo las especies más comunes afectan la estructura de la comunidad. En el ejemplo, la comunidad tiene una diversidad equivalente a una comunidad con aproximadamente 14.44 especies dominantes.

Medir e interpretar los índices de diversidad beta en R

En R, los índices de Jaccard y Sørensen se utilizan para la valoración de la similitud en la composición de las comunidades y, en consecuencia, su complemento (la disimilitud).

Input

Paso 1. Definir la carpeta donde se encuentran los datos:

```
Mydir = ("C:/Users/Desktop/Datos")
setwd (Mydir)
```

Paso 2. Para cargar las bases de datos que se emplearán para los análisis, se utilizará una matriz de abundancia de tipo 1.

```
Datos = read.table("Datos.txt", header = TRUE, sep = "\t", row.
names = 1)
```

Paso 3. Para cargar la librería utilizada para los análisis, se usará betapart:

```
library(betapart)
```

Paso 4. Para convertir la matriz de datos de abundancia en matriz de datos de incidencia (presencia ausencia):

```
Datos = ifelse(Datos>0, 1, 0)
```

Paso 5. Para calcular las medidas de diversidad beta de sitios múltiples y las matrices de disimilitud por pares:

```
Beta.coreAM = betapart.core(Datos)
```

Paso 6. Los siguientes comandos permiten calcular la beta partición de los distintos índices de diversidad beta:

- Beta partición en base Jaccard:

```
Multi.jac = beta.multi(Beta.coreAM, index.family="jac")
Dist.jac = beta.pair(Beta.coreAM, index.family="jac")
```

- Beta partición en base Sørensen:

```
Multi.sor = beta.multi(Beta.coreAM, index.family="sor")
Dist.sor = beta.pair(Beta.coreAM, index.family="sor")
```

Paso 7. Para visualizar los resultados en tablas:

```
Multi.jac
Dist.jac

Multi.sor
Dist.sor
```

Output

Paso 1. Beta partición en base Jaccard:

```
Multi.jac
$beta.JTU
[1] 0.62311558

$beta.JNE
[1] 0.063076891

$beta.JAC
[1] 0.68619247

Dist.jac
$beta.jtu
          Ambiente1 Ambiente2
Ambiente2      0.48000000
Ambiente3      0.54794521 0.62337662
```

```
$beta.jne
         Ambiente1 Ambiente2
Ambiente2       0.091428571
Ambiente3        0.097216085 0.018598685

$beta.jac
         Ambiente1 Ambiente2
Ambiente2       0.57142857
Ambiente3       0.64516129 0.64197531
```

Paso 2. Beta partición en base Sørensen:

```
Multi.sor
$beta.SIM
[1] 0.45255474

$beta.SNE
[1] 0.069738249

$beta.SOR
[1] 0.52229299

Dist. sor
$beta.sim
         Ambiente1 Ambiente2
Ambiente2       0.31578947
Ambiente3       0.37735849 0.45283019

$beta.sne
         Ambiente1 Ambiente2
Ambiente2       0.084210526
Ambiente3       0.098831986 0.019897084

$beta.sor
         Ambiente1 Ambiente2
Ambiente2       0.40000000
Ambiente3       0.47619048 0.47272727
```

Paso 3. Interpretar los resultados. Aunque los valores específicos pueden diferir ligeramente entre los índices de diversidad beta en base Jaccard y Sørensen, ambos proporcionan una medida de la disimilitud entre dos comunidades. Valores más cercanos a 1 indican mayor disimilitud, mientras que valores más cercanos a 0 indican mayor similitud.

- Sensibilidad: el índice de diversidad beta en base Sørensen tiende a ser más sensible a la presencia de especies comunes que el índice de beta en base Jaccard, ya que le da más importancia a las especies compartidas (multiplicando C por 2 en el numerador).

- Beta partición: valores altos del componente de reemplazo de especies (βjtu o betaSIM) indican un alto grado de reemplazo de especies entre las comunidades, es decir, las comunidades tienen especies diferentes. Por otro lado, valores altos de anidamiento (βjne o betaSNE) indican que la estructura de las comunidades está altamente anidada, es decir, las comunidades con menos especies son subconjuntos de aquellas con más especies.

La interpretación conjunta de estos componentes proporciona una visión más completa de la estructura de la diversidad beta en un paisaje, permitiendo identificar las causas subyacentes de las diferencias en la composición de especies entre las comunidades.

Recapitulando

Este capítulo exploró de manera exhaustiva los diversos índices utilizados para medir la diversidad biológica, así como desglosó sus aplicaciones en la evaluación de la riqueza de especies y la equidad dentro de las comunidades ecológicas.

Los tres componentes esenciales de la biodiversidad (alfa, beta y gamma), cada uno representado por métodos y fórmulas específicos que proporcionaron una visión integral sobre la estructura y dinámica de las comunidades biológicas.

Desde los índices de dominancia como el de Simpson y Berger-Parker hasta los enfoques basados en la equidad como Shannon-Wiener y Pielou, este capítulo ofreció una guía detallada para interpretar estos índices en el análisis de la biodiversidad. Además, detalló el uso de R para medir y analizar estos índices, facilitando su aplicación en estudios ecológicos y proporcionando herramientas prácticas para la investigación.

En resumen, el capítulo resaltó la importancia de seleccionar y aplicar adecuadamente estos índices para una comprensión profunda de la biodiversidad y su gestión efectiva en contextos de conservación.

Capítulo 6
Otras formas de medir la biodiversidad

Lucas J. Mina y Darío D. Larrea

En el capítulo anterior se discutió qué es la biodiversidad, sus componentes y algunos métodos clásicos para medirla, como la riqueza específica, los índices de Shannon y Simpson, entre otros. Estas formas de medir la diversidad se consideran neutrales, lo que significa que consideran de forma igual a todas las especies de una comunidad. Sin embargo, esto puede presentar limitaciones, ya que no tienen en cuenta diferencias entre especies como sus rasgos ecológicos, morfológicos, fisiológicos, biogeográficos o funcionales.

Diversidad taxonómica

La diversidad taxonómica[1] es una forma de medir la biodiversidad que reconoce las diferencias entre las categorías taxonómicas anidadas de las especies. Es decir, el valor de diversidad de la comunidad estará en función de su estructura taxonómica. Las comunidades con especies pertenecientes a distintos géneros tendrán una mayor diversidad que aquellas cuyas especies pertenecen al mismo género. Esta diversidad será aún mayor si estos géneros pertenecen a distintas familias.

Para medir la diversidad taxonómica, en general, es necesario contar con información filogenética de los grupos estudiados. Sin embargo, a pesar de los avances en áreas como la sistemática y la biología molecular, esta información no siempre está disponible. Por ello, destaca el índice de distintividad taxonómica propuesto por Warwick y Clarke, ya

1. El término diversidad taxonómica tiene dos acepciones en ecología. Originalmente, hacía referencia a la composición de especies en un área. Sin embargo, a partir de los trabajos de Clarke y Warwick, este concepto define análisis que consideran la estructura de las relaciones taxonómicas entre las especies, lo que resulta en la generación de índices específicos de diversidad taxonómica. Para evitar confusiones en el uso de la terminología, lo ideal sería hablar de diversidad taxonómica únicamente cuando se utiliza el enfoque de Clarke y Warwick.

que para su cálculo no se requieren distancias filogenéticas, sino únicamente conocer las jerarquías taxonómicas.

Medir e interpretar la distintividad taxonómica en R

A continuación, se verán los pasos necesarios para la medición de la distintividad taxonómica en R, utilizando funciones disponibles en la librería BiodiversityR. Para ello, son necesarias dos matrices, una de abundancia y otra taxonómica.

Input

Paso 1. Cargar las librerías necesarias:

```
library(BiodiversityR)
library(ggplot)
library(ggrepel)
```

Las dos últimas librerías son usadas para producir gráficos, no son estrictamente necesarias para los cálculos de diversidad.

Paso 2. Para cargar los datos necesarios, una tabla de abundancia de especies con sitios de estudio (matriz de abundancia tipo 1) y otra con la jerarquía taxonómica (matriz taxonómica), es importante que la tabla de jerarquía taxonómica tenga una primera columna de identificación que coincida con los nombres de las columnas de la tabla de abundancia:

```
abundancia <- read.table("tabla_abundancia.txt", header =
TRUE, row.names = 1, sep = "\t")

taxonomia <- read.table("tabla_taxa.txt", header = TRUE,
row.names = 1, sep = "\t")
```

Paso 3. Para calcular las distancias taxonómicas. La función plot nos muestra un dendograma de las distancias taxonómicas:

```
dist_taxonomia <- taxa2dist(taxonomia)

plot(hclust(dist_taxonomia), hang = 1)
```

Paso 4. Cálculo diversidad taxonómica:

```
div_taxonómica <- taxondive(abundancia, dist_toxonomica)
div_taxonomica
```

Output

Paso 1. Observar el resultado:

	Species	Delta	Delta*	Lambda+	Delta+	S Delta+
ESC-T1	28.000	84.330	89.959	454.219	89.220	2498.1
ESC-T2	34.000	87.580	90.763	429.166	91.310	3104.5
PCII-T1	25.000	86.111	90.435	385.076	91.083	2277.1
PCII-T2	20.000	83.502	89.190	416.707	92.237	1844.7
RCH-T1	31.000	83.227	87.412	485.796	88.548	2745.0
RCH-T2	19.000	79.832	85.113	572.869	87.719	1666.7
ESQ-T1	14.000	87.765	93.715	330.576	93.132	1303.8
ESQ-T2	10.000	78.947	87.097	816.667	85.000	850.0
EBA-T1	28.000	88.078	92.744	511.341	89.815	2514.8
EBA-T2	24.000	86.908	91.069	590.212	88.406	2121.7
PCI-T1	28.000	88.558	94.218	450.177	90.146	2524.1
PCI-T2	18.000	82.119	86.681	401.341	87.418	1573.5
VED-T1	19.000	85.747	88.389	614.420	85.965	1633.3
VED-T2	12.000	78.333	84.794	458.706	88.258	1059.1
PAN-T1	22.000	81.698	89.600	475.511	89.827	1976.2
PAN-T2	18.000	77.192	84.548	645.318	86.438	1555.9
PIN-T1	14.000	74.160	82.323	524.846	89.011	1246.2
PIN-T2	13.000	72.348	79.253	810.733	77.564	1008.3
Expected		96.727	85.969		89.482	

Paso 2. Interpretar los resultados. Como resultado de la función taxondive, se obtiene una tabla con la cantidad de especies y los valores de Delta, Delta*, Lambda+ y Delta+ para cada sitio.

- Delta o diversidad taxonómica: indica la distancia taxonómica entre dos individuos elegidos al azar, toma en cuenta la abundancia y puede incluir individuos de la misma especie. Esto significa que es sensible al esfuerzo de muestreo e indica qué tan relacionados están los individuos entre sí.

- Delta* o distintividad taxonómica: es el promedio de las distancias taxonómicas entre todos los individuos de un sitio, exceptuando a los de la misma especie. Con esto se elimina parcialmente su dependencia en la abundancia y representa las relaciones taxonómicas entre especies de los sitios sin considerar su riqueza o abundancia.

- Delta+ o distintividad taxonómica promedio: calculada como el promedio de la distancia taxonómica entre cualquier especie en el sitio, dividido entre el

número total de rutas taxonómicas. Este índice es equivalente a los anteriores, pero considera únicamente la incidencia de las especies, lo que lo hace menos sensible al esfuerzo de muestreo.

- Lambda+ o variación de la distintividad taxonómica: es la varianza del promedio de las distancias taxonómicas entre especies. Expresa la variación de las distancias que conectan a cada par de especies en la estructura taxonómica. Refleja la equitatividad taxonómica, por lo que nos indica si ciertos taxones están sobre o subrepresentados.

En el caso de Delta, Delta* y Delta+, un valor más alto indica una mayor diversidad en el sitio, ya que refleja una mayor separación taxonómica entre los individuos. Por otro lado, en el caso de Lambda+, valores altos indican que la mayoría de las especies están concentradas en pocos taxones, mientras que valores bajos sugieren una distribución más equitativa de las especies entre los taxones.

Paso 3. Para graficar la diversidad taxonómica:

```
ggplot(div_taxonomica, aes(x = Species, y = Dplus)) +
  geom_point() +
  ylab("Δ+") +
  xlab("Riqueza") +
  geom_hline(aes(yintercept = EDplus), linetype = "dotted") +
  geom_ribbon(aes(ymax = EDplus + sd.Dplus * 2, ymin = EDplus - sd.
Dplus * 2), fill = NA, color = "black")+
  geom_text_repel(aes(label = row.names(div_taxonomica)), size =
3.5, color = "blue3") +
  theme_classic()
```

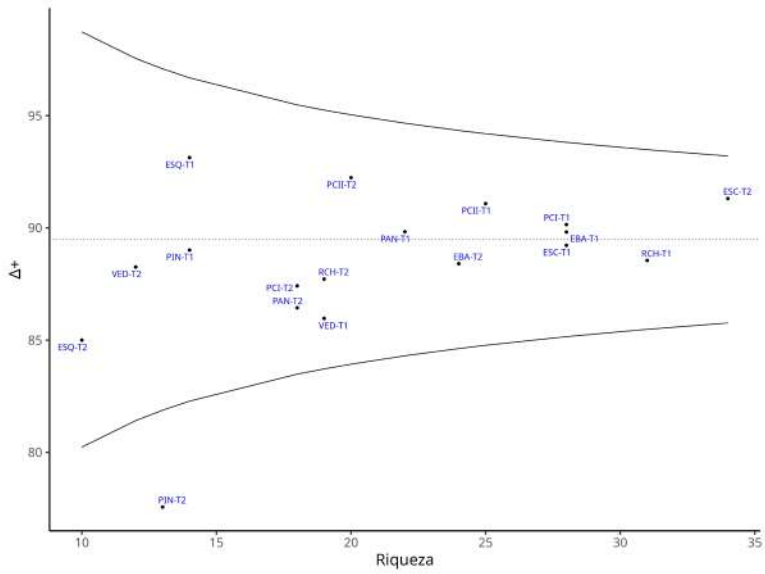

Figura 1. Gráfico de diversidad taxonómica.

En la figura 1, el gráfico muestra la diversidad taxonómica (Delta+) de cada sitio en relación con su riqueza de especies. La línea de puntos representa el valor esperado de diversidad, mientras que el embudo indica el intervalo de confianza. Es interesante observar que se puede tener mayor diversidad taxonómica en sitios con menor riqueza de especies. Esto sugiere que, a pesar de tener menos especies, estos sitios poseen una mayor variedad de géneros y/o familias.

Para el gráfico anterior, se usó la librería ggplot, que es un poco más completa de utilizar que los gráficos base, pero permite mayor flexibilidad para modificar el gráfico a nuestro gusto. También es posible hacerlo utilizando gráficos base de R con plot(div_taxonomica).

Diversidad funcional

La diversidad funcional es una medida no neutral de la diversidad, pero en este caso no se tienen en cuenta las relaciones filogenéticas, sino los rasgos funcionales.

Los rasgos funcionales son características fenotípicas o de comportamiento observables que afectan el desempeño de las especies y/o los procesos del ecosistema. La diversidad funcional mide el grado en el que las especies de una comunidad varían en sus rasgos. Aunque se mencionó que no se tienen en cuenta las relaciones filogenéticas, vale la pena destacar que los rasgos funcionales suelen estar relacionados con la taxonomía, por lo que existe algo de similitud en los métodos numéricos usados por ambos índices.

Los índices más comunes para medir la diversidad funcional se detallan a continuación:

- Riqueza funcional (FRic). Es el volumen o área de espacio funcional ocupado por las especies de la comunidad. En el caso de dos rasgos, es el área del polígono cuyos vértices son las especies más extremas (con más distancias entre rasgos). Debido a esto, un aumento de este índice nos indica un aumento de la diversidad funcional. Cabe aclarar que, para su cálculo, es necesario al menos una especie más que rasgos funcionales.

- Uniformidad funcional (FEve). Este índice combina la uniformidad del espaciado entre especies en el espacio funcional y la uniformidad de sus abundancias. Mide la consistencia del largo de las ramas del MST (*Minimun Spanning Tree* o árbol de expansión mínimo) luego de ponderarlas por la suma de las abundancias relativas. Varía entre 1 y 0; un valor de 1 nos indica que la comunidad es perfectamente uniforme con una baja diversidad funcional. Por otro lado, valores cercanos a 0 indican que la uniformidad disminuye y la diversidad funcional aumenta.

- Divergencia funcional (FDiv). Es el grado en que las especies más abundantes están alejadas del centroide del ensamble en ese espacio funcional, estandarizado al promedio de la distancia al centroide no ponderado de las especies en el perímetro del polígono. Su valor se incrementa al aumentar la diversidad y es más alto cuando las especies más abundantes tienen rasgos extremos. Es independiente del volumen de rasgos.

- Dispersión Funcional (FDis). Mide la distancia promedio al centroide del ensamble en el espacio funcional, ponderada por las abundancias relativas. Conceptualmente es similar a FDiv, pero los mecanismos de medición son diferentes. Por último, a medida que la diversidad aumenta, también lo hace el valor de este índice.

Medir e interpretar los índices diversidad funcional en R

Ahora que ya se sabe qué es lo que se entiende por diversidad funcional, en esta sección se verá cómo calcularla e interpretar los resultados arrojados por las funciones de la librería FD. Para esto, se volverá a utilizar la tabla de abundancia, pero esta vez junto con una de rasgos.

Input

Paso 1. Para el cálculo de los índices de diversidad funcional en R, usar la librería FD disponible en CRAN:

```
library(FD)
```

Paso 2. En cuanto a los datos, como ya se mencionó, se necesita una matriz de abundancia de tipo 1 y una matriz de rasgos. En estas, los nombres de las especies (filas en la tabla de rasgos y columnas en la de abundancia) deben ser iguales y estar en el mismo orden.

```
# ---- Carga de datos ----

selva_01 <- read.table("data/selva_1b.txt", header = TRUE, sep =
"\t")
selva_02 <- read.table("data/selva_2b.txt", header = TRUE, sep =
"\t")
selva_03 <- read.table("data/selva_3b.txt", header = TRUE, sep =
"\t")
```

Paso 3. Para el análisis, se utiliza la función *dbFD*. Esta función nos brinda una lista que contiene los valores de los distintos índices de diversidad funcional (FRic, FEve, FDiv y FDis). Más adelante, nos detendremos en cada uno de los índices y su interpretación. Por ahora, se transformarán estas listas en una tabla para observar los resultados más fácilmente:

```
# Diversidad funcional ambiente 1
resultados_ambiente1 <- dbFD(selva_01)
resultados_ambiente1

# Diversidad funcional ambiente 2
resultados_ambiente2 <- dbFD(selva_02)
resultados_ambiente2

# Diversidad funcional para el ambiente 3
resultados_ambiente3 <- dbFD(selva_03)
resultados_ambiente3
```

Output

Paso 1. Observar el resultado:

```
df_datos <- data.frame(
  Ambiente = c("Ambiente1", "Ambiente2", "Ambiente3"),
  FEve = c(resultados_ambiente1$FEve, resultados_ambiente2$FEve,
resultados_ambiente3$FEve),
  FDiv = c(resultados_ambiente1$FDiv, resultados_ambiente2$FDiv,
```

```
resultados_ambiente3$FDiv),
  FDis = c(resultados_ambiente1$FDis, resultados_ambiente2$FDis,
resultados_ambiente3$FDis),
  FRic = c(resultados_ambiente1$FRic, resultados_ambiente2$FRic,
resultados_ambiente3$FRic)
)
df_datos
```

Paso 2. Obtener el siguiente data.frame:

```
   Ambiente      FEve      FDiv      FDis                      FRic
1 Ambiente1 0.9189994 0.8540726 0.2855491 0.00000000000116613814307
2 Ambiente2 0.8687807 0.8795479 0.2865776 0.00000002346762775835205
3 Ambiente3 0.9191980 0.8592282 0.2668836 0.00000000000000004207213
```

Paso 3. Interpretar los resultados. En su mayoría, excepto para el caso de uniformidad funcional (FEve), un aumento numérico en el valor del índice revela una mayor diversidad funcional. Es decir que las áreas con valores más altos de Fdiv, FDis y FRic indican lugares con mayor diversidad funcional. Por otro lado, valores bajos de FEve, mayor diversidad funcional al comparar distintas áreas.

Recapitulando

En este capítulo, se amplió la comprensión de la biodiversidad al explorar formas adicionales y complementarias de medirla, más allá de los métodos clásicos de diversidad. Mientras que en capítulos anteriores se abordaron la riqueza específica y los índices de diversidad clásicos, como Shannon y Simpson, aquí se ha enfocado en medidas que proporcionan una visión más matizada y enriquecida.

Primero, se introdujo la diversidad taxonómica, una métrica que va más allá de la simple riqueza de especies al considerar las relaciones filogenéticas entre ellas. Y se ha detallado el proceso para calcular e interpretar la diversidad taxonómica utilizando herramientas en R, proporcionando un paso a paso práctico para aplicar estos conceptos en estudios reales.

En adición se ha explorado la diversidad funcional, una medida crucial que se enfoca en los rasgos funcionales de las especies en lugar de su filiación taxonómica. Este enfoque nos proporciona una perspectiva diferente sobre la biodiversidad al considerar cómo las especies afectan y participan en los procesos ecosistémicos. La diversidad funcional, a diferencia de la taxonómica, no se basa en relaciones filogenéticas, sino en las características observables que determinan el desempeño ecológico de las especies.

En conjunto, estas metodologías avanzadas enriquecen nuestra capacidad para medir e interpretar la biodiversidad. Esta perspectiva multifacética es fundamental para diseñar estrategias de conservación efectivas y para estudiar la dinámica de los ecosistemas con mayor profundidad.

Capítulo 7
Estudio de la estructura de la comunidad

Lucas J. Mina y Matías I. Dufek

Hasta este punto, se han explorado distintos tipos y formas de calcular índices de diversidad. Entre todos ellos, la riqueza de especies, junto con los índices clásicos de diversidad, que permiten reconocer aspectos importantes de las comunidades biológicas. Sin embargo, estos últimos no logran mostrar qué especies contribuyen más a esa diversidad. Esto se debe a que en la gran mayoría de las comunidades siempre habrá especies dominantes (muy abundantes) y especies raras.

Este capítulo explorará algunos análisis que permitan conocer esa estructura, como las curvas de Whittaker, así como otros para comparar la composición de dos o más comunidades, como el Escalado multidimensional no métrico (NMDS por sus siglas en inglés, *Non-metric multidimensional scaling*). Se usará como ejemplo a las comunidades de hormigas clasificadas según su estado de conservación (alto, medio o bajo).

Curvas de Whittaker

También conocidas como curvas de rango-abundancia, permiten ver, de forma gráfica, los patrones de distribución de las especies de una comunidad. En estas curvas, las especies se ordenan de mayor a menor abundancia, asignándoles un rango dependiente de esta ordenación, de ahí su nombre rango-abundancia.

Construir e interpretar curvas de Whittaker en R

Ahora se verá cómo elaborar estas curvas mediante la librería BiodiversityR. Para su confección, se necesitan dos tablas: una que contenga las especies por sitio de muestreo y otra que indique a qué clasificación corresponde cada sitio; en este caso, como se mencionó anteriormente, se usará el estado de conservación.

Input

Paso 1. Las librerías necesarias son las siguientes:

```
library(tidyverse)
library(readxl)
library(ggrepel)
library(BiodiversityR)
library(datosEcoR)
```

Paso 2. Para este análisis, se usarán los datos de «abundancia» y «ambiente» del paquete, que contienen tablas con los formatos que se ven en la figura 1, que se corresponden con una matriz de abundancia de tipo 1 y una matriz ambiental.

```
data(abundancia)
data(ambiente)
```

A

curvas_whittaker.R | especies | ambiente

Filter Cols: « ‹ 1-50 › »

	Acanthoponera mucronata	Acromyrmex cf. lundii	Anochetus
Estancia El Bagual	24	1	
Estancia Quintana	0	0	
Estancia San Carlos	0	0	
Pampa del Indio	0	0	
Parque Nacional Chaco I	0	0	
Parque Nacional Chaco II	3	0	
Puerto Antequera	0	0	
Reserva los Chaguares	0	0	
Vedia	0	0	

B

curvas_whittaker.R | especies | ambiente

Filter

	estado_conservacion
Estancia El Bagual	ECI
Estancia Quintana	ECI
Estancia San Carlos	ECB
Pampa del Indio	ECD
Parque Nacional Chaco I	ECI
Parque Nacional Chaco II	ECB
Puerto Antequera	ECD
Reserva los Chaguares	ECB
Vedia	ECD

Figura 1. A) Tabla de especies por sitio; B) Tabla de clasificación de sitios.

Output

Paso 1. Una vez que se tengan estas tablas, se puede comenzar el análisis y realizar una curva de Whittaker para todas las especies, sin tener en cuenta la clasificación de los sitios. En este caso, esto se hace únicamente con fines exploratorios:

```
# ---- Rangos de abundancia ----

rank_abundancia <- rankabundance(abundancia$localidad)

rankabunplot(rank_abundancia, scale = “abundance”)  #Fig. 7.2
```

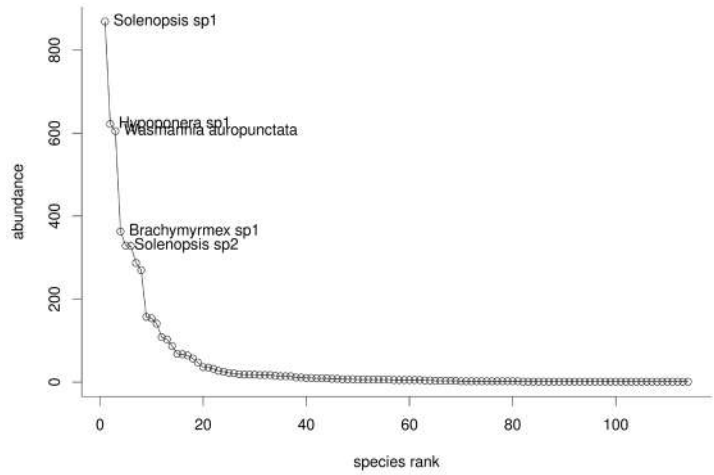

Figura 2. Curva de Whittaker generada con la función rankabunplot.

Paso 2. Como se puede ver en la figura 2, este gráfico no resulta muy atractivo y tampoco aporta la información que se estaba buscando, que era comparar las composiciones de los tres sitios. A continuación, se verá cómo solucionar ambos problemas.

Primero, se empieza por correr la función rankabundcomp, que permite calcular las curvas de rango-abundancia para distintos subconjuntos (en este caso, para cada estado de conservación). Esta función genera un gráfico automáticamente, pero se lo ignorará y almacenarán los resultados en una variable:

```
rank_abundancia_amb <- rankabuncomp(
  abundancia$localidad,
  y = ambiente,
  factor = “estado_conservacion”,
  legend = FALSE
)
```

Paso 3. En este caso, la tabla de abundancias por sitio, denominada *especies*, contiene la información sobre la frecuencia de las especies en cada ambiente. El argumento *ambiente* indica la clasificación de los sitios, mientras que el argumento *factor* especifica qué columna de la tabla ambiente se utilizará para generar los subconjuntos (esta columna debe ser un factor, sino lo fuera, la función arroja un error). Con estos resultados, se dispone de todo lo necesario para crear los gráficos. Sin embargo, se puede agregar una columna para marcar las especies únicas de cada comunidad, con el siguiente fragmento de código (esto es completamente opcional):

```
# Marcamos especies únicas
especies_unicas <- lapply(
  unique(levels(ambiente$estado_conservacion)),
  function(group) {
    setdiff(subset(rank_abundancia_amb, Grouping == group)$species,
            subset(rank_abundancia_amb, Grouping != group)$species)
  }
)
especies_unicas <- unlist(especies_unicas)

rank_abundancia_amb$unique <- ifelse(
  rank_abundancia_amb$species %in% especies_unicas, TRUE, FALSE
)
```

Paso 4. Confeccionar tres gráficos, uno para cada estado de conservación, y se empieza por definir una función para no repetir el código, ya que los tres gráficos son similares:

```
curva_whittaker <- function(x, group, scale, color, mark.unique =
FALSE) {
  if ("unique" %in% names(x)) {
    unique_num <- nrow(
      subset(x, Grouping == group & unique == TRUE)
    )
  }

  label <- sprintf("%s especies únicas", unique_num)
  xmax <- max(x[["rank"]])
  ymax <- max(x[[scale]])

  plot <- ggplot(x, aes(x = rank, y = .data[[scale]])) +
    coord_cartesian(
      xlim = c(0, xmax),
      ylim = c(0, ymax)
    ) +
    geom_point(
      data = subset(x, Grouping == group),
      size = 3,
      shape = 1,
      color = color
    ) +
    scale_shape_manual(
      name = NULL,
```

```
      breaks = c("unique"),
      label = c(label),
      values = c(19)
    ) +
    geom_line(
      data = subset(x, Grouping == group),
      color = color
    ) +
    geom_text_repel(
      data = subset(x, Grouping == group & labelit == TRUE),
      aes(label = species),
      hjust = 0,
      nudge_x = 3,
      size = 3
    ) +
    theme_classic()

  if (mark.unique) {
    plot <- plot +
      geom_point(
      data = subset(x, Grouping == group & unique == TRUE),
      aes(shape = "unique"),
      color = color,
      size = 3,
    ) +
    theme(legend.position = c(.7, .5))
  }
  return(plot)
}
```

Paso 5. Esta función permite hacer los gráficos de la siguiente forma:

```
curva_whittaker(x, group, scale, color, mark.unique = FALSE)
```

Paso 6. Donde: *x* son los resultados de rankabunplot, que se calculó más arriba; *group* es el subconjunto que se desea graficar, en este caso, alguno de los estados de conservación; *scale* indica qué escala[1] usar, puede ser abundance, proportion, accumfreq,

1. La escala abundance usa la abundancia de especies; proportion, la abundancia proporcional; accumfreq, la acumulación de la abundancia proporcional y logabun, el logaritmo en base 10 de la abundancia.

logabun, rankfreq; *color* es el color de la curva; *mark.unique*, con valores TRUE o FALSE, permite elegir si se quiere indicar o no las especies únicas, el valor por defecto es FALSE.

Paso 7. Graficar las tres curvas y guardarlas basta con:

```
curva_ecb <- curva_whittaker(rank_abundancia_amb, "ECB",
"abundance", "darkgreen", mark.unique = TRUE)
curva_eci <- curva_whittaker(rank_abundancia_amb, "ECI",
"abundance", "orange", mark.unique = TRUE)
curva_ecd <- curva_whittaker(rank_abundancia_amb, "ECD",
"abundance", "red", mark.unique = TRUE)

curva_ecb   # Fig. 7.3.A
curva_eci   # Fig. 7.3.B
curva_ecd   # Fig. 7.3.C

ggsave("./plots/curva_ecb.png", plot = curva_ecb)
ggsave("./plots/curva_eci.png", plot = curva_eci)
ggsave("./plots/curva_ecd.png", plot = curva_ecd)
```

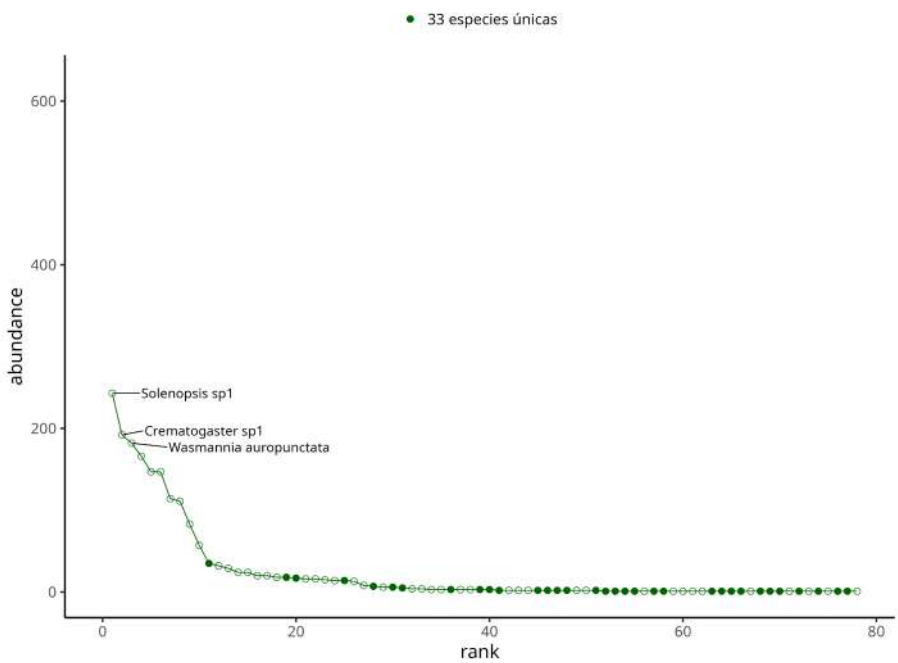

Figura 3. Curvas de Whittaker, estado de conservación bueno (ECB).

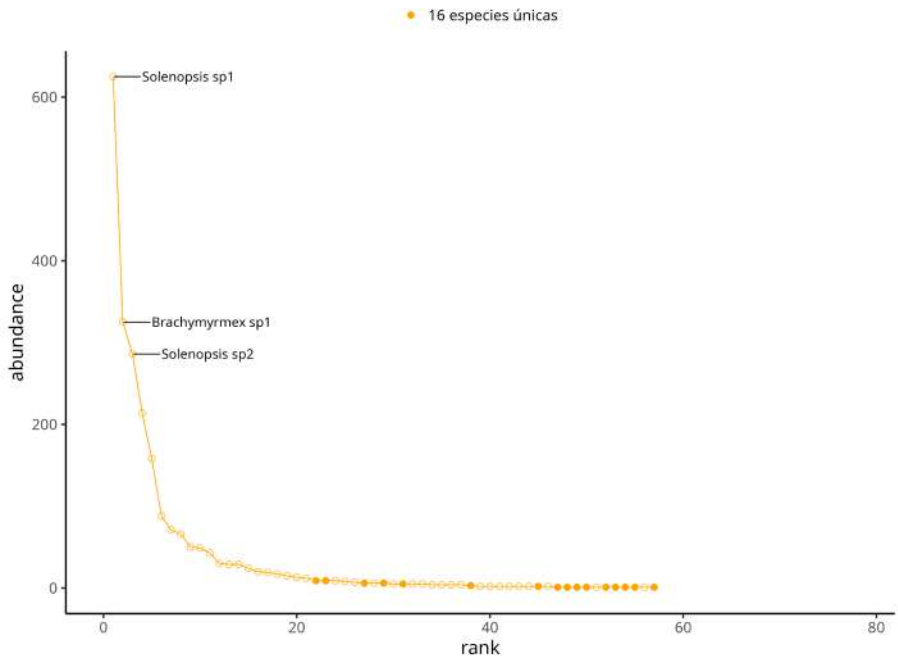

Figura 4. Curvas de Whittaker, estado de conservación intermedio (ECI).

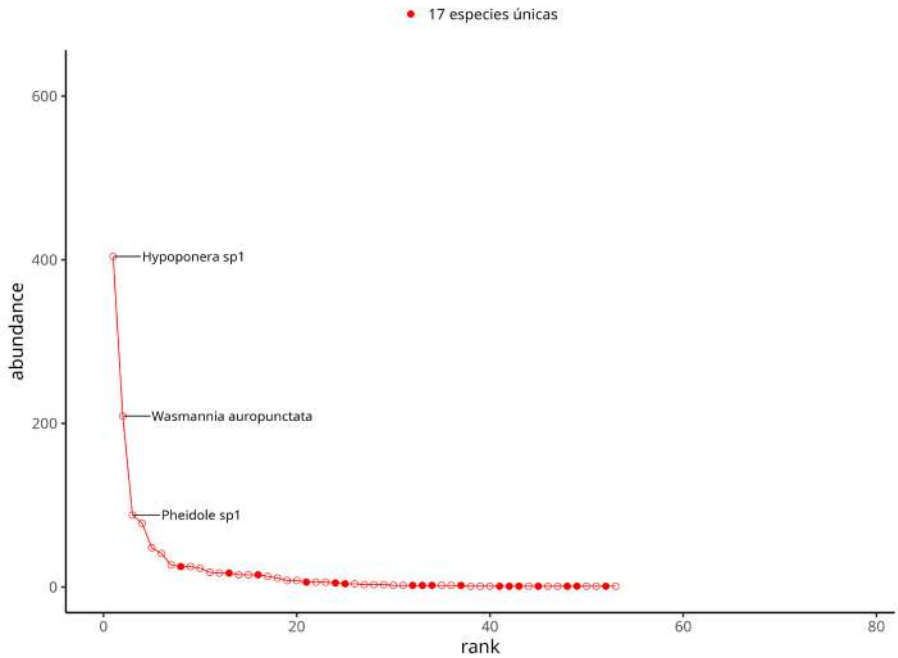

Figura 5. Curvas de Whittaker, estado de conservación degradado (ECD)

Paso 8. Confeccionar un gráfico con todas las curvas juntas:

```
ggplot(rank_abundancia_amb, aes(x = rank, y = abundance, color =
Grouping)) +
  geom_line() +
  geom_point(size = 2.5) +
  labs(color = "", shape = "") +
  scale_color_manual(breaks = c("ECB", "ECI", "ECD"), values =
c("darkgreen", "orange", "red"))+
  theme_classic() +
  theme(legend.position = "top")
```

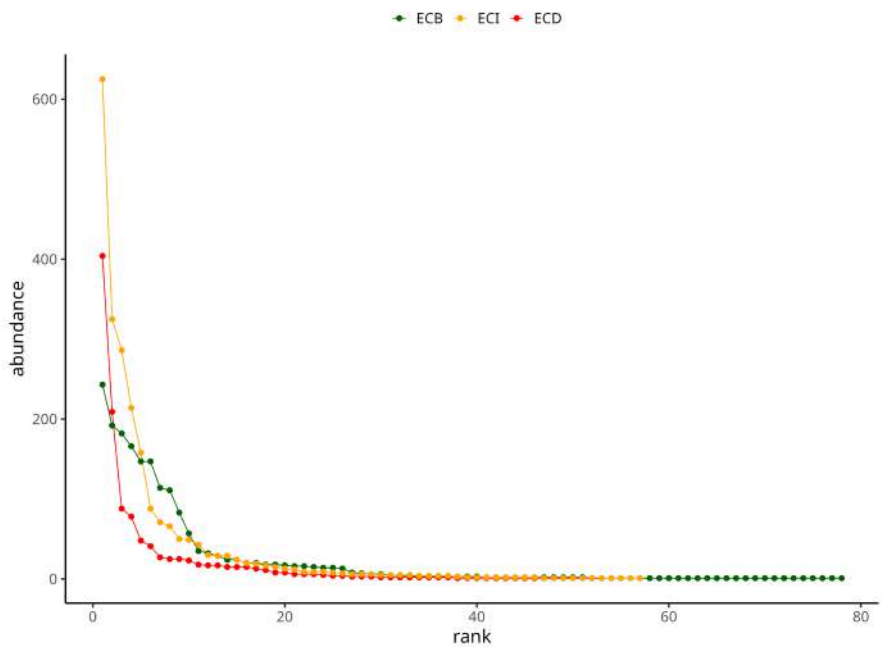

Figura 6. Curvas de Whittaker para ambientes con estados de conservación bueno, intermedio y degradado.

Paso 9. Interpretar los resultados. Como se mencionó anteriormente, las curvas de Whittaker o de rango-abundancia permiten observar gráficamente la estructura de las comunidades en estudio y ofrecen una sencilla interpretación. Comunidades uniformes tendrán curvas más «planas», similares a la observada en la figura 3A. En cambio, si existen especies dominantes, adquirirá la forma que se puede ver en las figuras 3B y C, a veces conocida como forma de «palo de golf». Cuanto mayor sea la dominancia de esa o esas

especies, más acentuada estará dicha forma. Por ejemplo, en nuestro caso, si bien tanto en ambientes ECI y ECD hay dominancia de especies, se puede ver que en ambientes ECI (figura 3B) es mayor que en los ambientes ECD (figura 3C).

Curvas de acumulación

Las curvas de acumulación son un tipo de gráfico que relaciona la cantidad de especies (o riqueza) con una unidad de esfuerzo de muestreo como, por ejemplo, número de sitios de muestreo. Estas curvas permiten estimar la cantidad máxima de especies de un ambiente, así como también evaluar y comparar nuestros esfuerzos de muestreo.

Construir e interpretar las curvas de acumulación en R
Para su confección, sólo es necesaria una tabla con la abundancia de especies por unidad de muestreo, en este caso, la unidad serán los sitios de recolección. Y, como se verá, el script es bastante sencillo.

Input

Paso 1. Utilizar una matriz de abundancia de tipo 1:

```
# ---- Librerias ----

library(BiodiversityR)
library(ggplot2)

# ---- Carga de datos ----

abundancia_esp <- read.csv("./data/datos_abundancia.csv",
row.names = 1)

# ---- Curva de acumulación ----

# Dimensiones de los datos
dim(abundancia_esp)

curva <- specaccum(abundancia_esp)
```

Paso 2. La función specaccum devuelve el siguiente objeto, donde *sites* es el número de sitios (unidad de muestra), *richness* corresponde a la riqueza de especies correspondiente al número de sitios y *sd*, la desviación estándar:

```
Sites     1.00000  2.00000  3.00000  4.00000  5.00000  6.00000
7.00000   8.00000   9
Richness 30.33333 48.47222 62.05952 73.23810 82.92857 91.61905
99.58333 107.00000 114
sd        8.74325  9.49255  9.60442  9.33264  8.76685  7.87409
6.49098   4.37163   0
```

Paso 3. Para graficar la curva, es posible hacerlo usando los gráficos base de R, con la función plot(), pero en este caso se usará ggplot. Antes, se debe convertir el resultado anterior en un data.frame:

```
datos_sp <- data.frame(
  Sitios = curva$sites,
  Riqueza = curva$richness,
  SD = curva$sd
)
```

Output

Paso 1. Graficar:

```
ggplot(datos_sp, aes(x = Sitios, y = Riqueza)) +
  geom_ribbon(aes(ymin = Riqueza - SD, ymax = Riqueza + SD), fill =
"grey90") +
  scale_x_continuous(breaks = datos_sp$Sitios) +
  geom_line(color = "blue") +
  theme_classic()
```

Con esto obtenemos el siguiente gráfico.

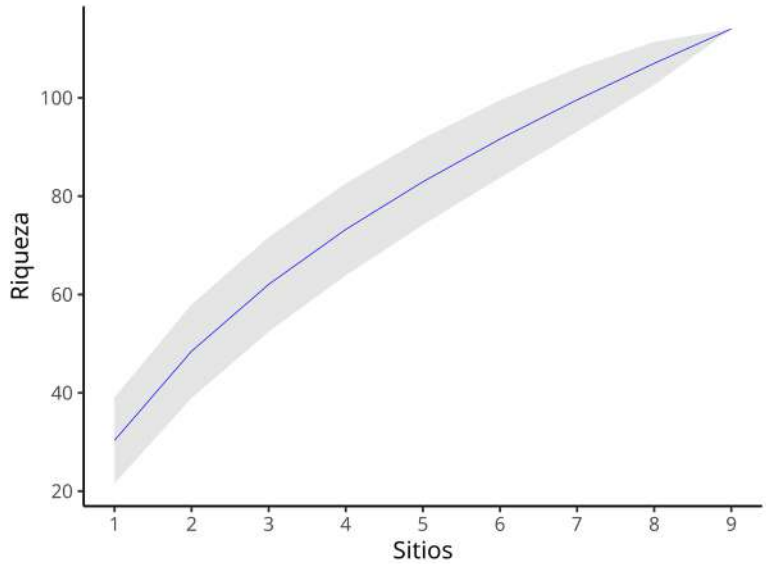

Figura 7. Curva de acumulación.

Paso 2. Interpretar los resultados. Como se ve en el gráfico, a medida que aumenta la cantidad de sitios (o lo que es lo mismo, el esfuerzo de muestreo), la curva tiende a ascender. Esto indica que se están recolectando especies nuevas con cada sitio de muestreo. Generalmente, estas curvas comienzan con un rápido ascenso, debido a que al principio se recolectan las especies más comunes y, a medida que aumenta el esfuerzo de muestreo, se comienzan a recolectar especies más raras, por lo que el ascenso de la curva disminuye, llegando a un punto en el que adquiere un comportamiento asintótico. Esto último indica que, sin importar cuánto aumente el muestreo, no estará añadiendo una cantidad significativa de especies al inventario. Es por esto por lo que estas curvas son muy valiosas para ayudar a determinar la fiabilidad de muestreo y también para estimar el esfuerzo de muestreo necesario para obtener un inventario de especies representativo.

Curvas de rarefacción

Aunque las curvas de acumulación que se han visto anteriormente son una herramienta valiosa, no están exentas de problemas. Uno de ellos es que, al usar sitios como medida de esfuerzo muestral, no se tiene en cuenta las diferencias entre estos sitios que puedan afectar la colecta (como, por ejemplo, la estructura vegetal o variables microclimáticas).

Aquí es donde entran en juego las curvas de rarefacción, las cuales establecen una relación entre el número de especies y el número de individuos. Esta corrección facilita una comparación de la riqueza de muestras con distinto tamaño.

Construir e interpretar las curvas de rarefacción en R

Confeccionar curvas de rarefacción en R es bastante sencillo, gracias a la librería iNext, como se verá, con un par de líneas de código es suficiente para obtener resultados. Lo más importante, como se mencionó en capítulos anteriores, es el formato de los datos de entrada. La función admite datos tanto de abundancia como de incidencia.

Input

Paso 1. Para realizar estas curvas en R, se usará el paquete *iNext*:

```
library(iNEXT)
```

Paso 2. Este ejemplo compara las curvas de rarefacción, de diversidad verdadera, entre tres ambientes, con distintos estados de conservación: bueno, intermedio y degradado. La función iNEXT() admite dos tipos de datos para el cálculo, de abundancia e incidencia. En este caso se usarán datos de abundancia, que consisten simplemente en una tabla donde las filas corresponden a cada especie, las columnas a cada uno de los ambientes y los valores a la abundancia (matriz de abundancia de tipo 2):

```
abundancia_ec <- read.csv("./data/abundancia_ec.csv",
row.names = 1)
```

Output

Paso 1. Cargados estos datos, ahora basta con:

```
inext_abundancia <- iNEXT(abundancia_ec, q = c(0, 1, 2),
datatype = "abundance")
```

Paso 2. El argumento q permite elegir qué orden q de los números de Hill se quiere calcular. Recuérdese que se está trabajando con la diversidad verdadera, por lo que se calcula q de orden 0 (riqueza), 1 (Shannon) y 2 (inverso de Simpson). Con estos datos, se realizarán dos tipos de gráficos, uno separando las curvas por ambiente (figura 6A), comparando los órdenes *q*, y otro separando los gráficos por orden *q* (figura 6B), comparando los ambientes.

Paso 3. Para el gráfico por ambiente, estado de conservación en este caso:

```
plot_ec <- ggiNEXT(inext_abundancia, type = 1, facet.var =
"Assemblage") +
  theme_classic(base_size = 10) +
  theme(legend.position = "bottom")
plot_ec
```

Paso 4. Para el gráfico por orden q:

```
plot_orderq <- ggiNEXT(inext_abundancia, type = 1, facet.var
= "Order.q") +
  theme_classic(base_size = 10) +
  theme(legend.position = "bottom")
plot_orderq
```

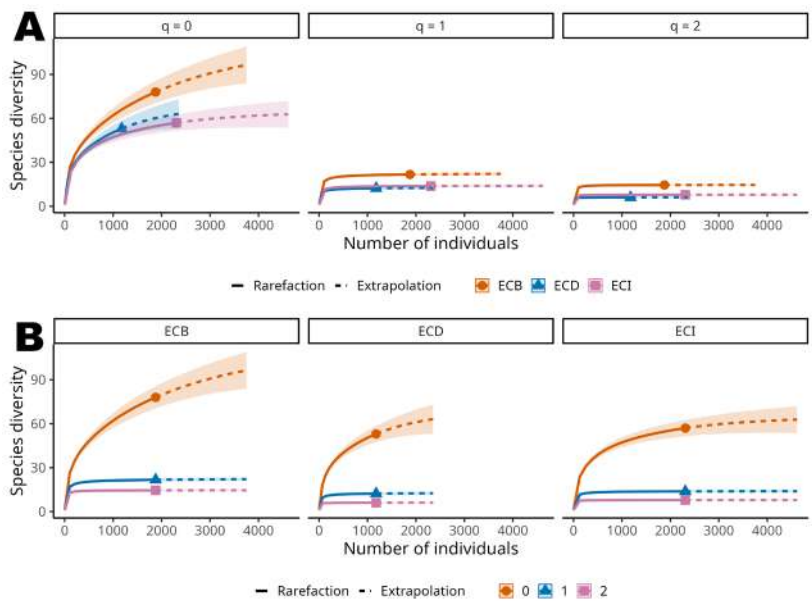

Figura 8. Curvas de rarefacción, la zona sombreada corresponde al intervalo de confianza. A) divididas por ambiente; B) divididas por orden q.

Paso 5. Interpretar los resultados. La interpretación de estas curvas es similar a las de las curvas de acumulación. Recuérdese que con la rarefacción se hace una relación por el número de individuos. Aquí también se observa que las curvas ascienden rápidamente en un principio, pero a medida que aumentan los individuos colectados, este ascenso se

desacelera. Al usar *iNEXT*, también se estima una extrapolación (línea punteada en los gráficos de la figura 5), lo que permite estimar cuántos individuos se deberían capturar para tener un inventario fiable de la diversidad del ambiente. Esta estimación permite comprobar cuándo la curva comienza a tener un comportamiento asintótico, como se puede observar en la curva de orden q o del ambiente ECI.

Escalado multidimensional no métrico (NMDS)

NMDS es una técnica de ordenación. Este análisis utiliza como base a las matrices de distancia y su objetivo principal es representar gráficamente en un número predeterminado de ejes, generalmente dos, datos con relaciones complejas o con gran cantidad de ellas. No pretende conservar las distancias entre los elementos, pero sí sus relaciones de ordenación.

Estimar e interpretar los NMDS en R

Ahora se verá un ejemplo de cómo realizar un análisis NMDS, utilizando funciones de la librería BiodiversityR. Para lo cual se utilizan dos tablas, una de abundancia y otra con datos ambientales. Y se construirán gráficos utilizando ggplot, en vez de los gráficos base de R.

Input

Paso 1. Las librerías necesarias son las siguientes:

```
library(BiodiversityR)
library(ggrepel)
library(ggforce)
library(concaveman)
```

Paso 2. Para este análisis, se necesitan dos tablas de datos. En ambas, las filas corresponden a la unidad muestral (en este caso, transectas compuestas por 20 trampas cada una). En la primera tabla, las columnas representan las especies capturadas (matriz de abundancia tipo 1) y en la segunda, las variables ambientales (matriz ambiental). Es muy importante que en ambas tablas los nombres de las filas sean los mismos y estén en el mismo orden. Ahora se procede a cargar los datos:

```
sitios <- read.csv("./datos/sitios.csv", row.names = 1,
header = TRUE)
ambiente <- read.csv("./datos/ambiente.csv", row.names = 1,
header = TRUE)
```

Paso 3. Se pasa al análisis, pero antes se establece un número de *seed* para asegurar que los resultados sean reproducibles. Se pueden probar distintos números para observar cómo afecta esto al resultado.

```
set.seed(19950922)

resultado_nmds <- metaMDS(sitios, distance = "bray", K = 2)
#(1)
resultado_nmds$stress   #(2)
```

Paso 4. El argumento *K* es la cantidad de dimensiones que se quiere, en la gran mayoría de casos se utilizan dos, ya que facilita la interpretación; la línea (2) devuelve el valor de *stress* del análisis y permite juzgar qué tan buena es la solución alcanzada por el NMDS. Valores menores a 0.10[2] son perfectamente aceptables, entre 0.10 y 0.20 se pueden dar errores y mayores a 0.20, su interpretación no es confiable.

Este análisis se puede complementar con *anosim*, que permite conocer si existen diferencias significativas entre las comunidades en estudio. Para este ejemplo, entre los ambientes con distintos estados de conservación:

```
dist_sitios <- vegdist(sitios)
anosim_sitios <- anosim(dist_sitios, ambiente$estado_
conservacion, distance = "bray")
summary(anosim_sitios)
```

Output

Paso 1. Esto da como resultado un valor de significancia (p). Si este es menor a 0.05, significa que sí existen diferencias significativas entre los sitios; de lo contrario, si p es mayor a 0.05, no hay diferencias significativas. En este caso el resultado es:

```
Call:
anosim(x = dist_sitios, grouping = ambiente$estado_conservacion,
distance = "bray")
Dissimilarity: bray

ANOSIM statistic R: -0.00218
      Significance: 0.454
...
```

2. Un valor de estrés menor a 0.05 indicaría una representación excelente, pero esto es muy difícil de lograr con datos de muestreo que no sean representativos.

Paso 2. Graficar el NMDS utilizando ggplot*:*

```
# Creamos un data.frame con los resultados
puntos_nmds <- as.data.frame(resultado_nmds$points)
puntos_nmds$CONSERVACION <- ambiente$estado_conservacion

# Agregamos siglas para nombres de los sitios y guardamos el valor
de stress
puntos_nmds$SITIO <- c(
  “EBA-T1”, “EBA-T2”, “EQN-T1”, “ESC-T1”, “ESC-T2”, “PIN-T1”,
“PCHI-T1”, “PCHI-T1”,
  “PCHII-T1”, “PCHII-T2”, “PANT-T1”, “PANT-T2”, “RCH-T1”, “RCH-T2”,
“VED-T1”
)
estres <- sprintf(“Stress = %s”, round(resultado_nmds$stress, 2))

# Graficamos
plot_nmds <- ggplot(puntos_nmds, aes(x = MDS1, y = MDS2)) +
  ggtitle(“NMDS”) +
  geom_point(aes(shape = CONSERVACION), size = 3) +
  scale_shape_manual(
    name = “”,
    breaks = c(“ECB”, “ECI”, “ECD”),
    labels = c(“Conservado”, “Intermedio”, “Degradado”),
    values = c(15, 16, 17)
  ) +
  geom_mark_hull(
    aes(group = CONSERVACION, linetype = CONSERVACION),
    concavity = 10,
    radius = 0,
    expand = 0,
    show.legend = FALSE
  ) +
  scale_linetype_manual(values = c(“solid”, “dashed”, “dotted”)) +
  annotate(“text”, x = +Inf, y = +Inf, label = estres, hjust = 1,
vjust = 1) +
  theme_classic()
plot_nmds

# Opcionalmente podemos añadir el nombre de los sitios
plot_nmds +
  geom_text_repel(
```

```
  aes(label = SITIO),
  box.padding = 0.5,
  size = 3.5,
  colour = "blue",
)
```

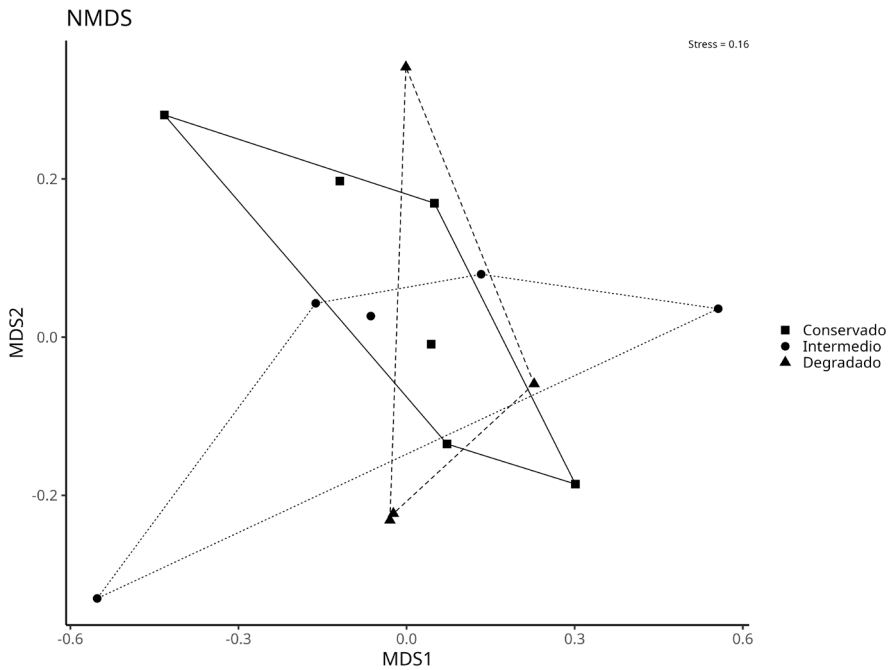

Figura 9. Gráfico NMDS.

Paso 3. Interpretar los resultados. En el gráfico se puede ver que el valor de estrés es 0.16, el cual no es excelente pero aun así es aceptable. Sin embargo, lo más relevante que se puede notar es cómo los polígonos, que representan cada estado de conservación, se superponen. Esto sugiere que no hay diferencias significativas entre estos sitios. En el caso de que las hubiera, los polígonos estarían alejados unos de otros o su superposición sería relativamente pequeña. Esta interpretación se respalda con los resultados obtenidos con *anosim* (p = 0.454), lo que también indica que no hay diferencias significativas entre sitios.

Recapitulando

Este capítulo abordó el estudio de la estructura de las comunidades biológicas mediante técnicas avanzadas de análisis de diversidad.

Primero, introdujo las curvas de Whittaker, que visualizan la distribución de especies según su abundancia y permiten identificar patrones de dominancia. Estas curvas proporcionan

una perspectiva detallada de cómo se distribuyen las especies dentro de una comunidad y facilitan la comparación entre diferentes ambientes.

Continuó con las curvas de acumulación, que relacionan el número de especies con el esfuerzo de muestreo, permitiendo evaluar la efectividad de este esfuerzo y estimar la riqueza total de especies. A diferencia de estas, las curvas de rarefacción ajustan la riqueza de especies según el número de individuos, corrigiendo las diferencias en el tamaño de las muestras.

Finalmente, el NMDS (Escalado multidimensional no métrico) se presentó como una herramienta para representar gráficamente relaciones complejas entre comunidades, facilitando la interpretación de la estructura comunitaria a través de múltiples dimensiones.

Cada técnica complementa a las otras, ofreciendo una comprensión integral de la diversidad y distribución de las especies en diversos contextos de conservación.

Capítulo 8
Evaluación de variables ambientales

Darío D. Larrea y Lucas J. Mina

Las variables ambientales son factores determinantes (físicos, químicos y/o biológicos) que modelan la estructura, función y dinámica de los ecosistemas. Pueden ser tanto factores abióticos (como temperatura, humedad, luz, pH, salinidad) como factores bióticos (como la estructura de la vegetación). Estas variables pueden modificarse a lo largo del tiempo y el espacio, determinando la abundancia de las especies y sus patrones de distribución.

Las variables ambientales de temperatura, presión, viento, humedad, precipitación y fotoperiodo definen condiciones limitantes de una determinada área o región, es decir, el clima, al que se adaptan los organismos. En otras palabras, los organismos evolucionan y se adecuan a condiciones específicas del entorno, por lo que ciertos factores pueden limitar su distribución geográfica.

Existen otras variables del entorno importantes para los organismos como las condiciones edáficas e hídricas. Las condiciones edáficas se refieren a las características del suelo, como la textura, la composición química, la estructura y la capacidad de retención de agua. El tipo de suelo influye en la disponibilidad de nutrientes para las plantas, la filtración del agua y la estabilidad del hábitat para la fauna del suelo. Por otro lado, las condiciones hídricas hacen referencia a la disponibilidad y calidad del agua. Las variables hídricas incluyen el volumen de precipitación, la humedad del suelo, la disponibilidad de agua superficial y la salinidad. Estas variables influyen en la distribución de especies acuáticas y terrestres, así como en los patrones de migración y reproducción.

Estas variables proporcionan el contexto necesario para entender cómo los organismos interactúan con su entorno y cómo los ecosistemas responden a los cambios ambientales.

En algunos estudios ecológicos resulta fundamental comprender cómo las variables ambientales afectan a los organismos. En el análisis estadístico de estos estudios existen varias técnicas que permiten evaluar la relación entre variables ambientales y la estructura de las comunidades biológicas, de los cuales se hablará en este capítulo.

- Análisis de regresión.
- Análisis de correlación.
- Análisis de componentes principales (PCA).
- Análisis de correspondencia canónica (CCA).
- Test de Mantel.

A continuación, se mostrará cómo llevar a cabo los distintos análisis mencionados en R, y cómo interpretar sus resultados.

Análisis de regresión

Este método permite cuantificar la relación entre una o más variables ambientales (como temperatura, humedad, pH del suelo, etc.) y las características de las comunidades biológicas (como diversidad, riqueza de especies, biomasa, etc.).

Input

Paso 1. Cargar los datos. Supóngase que se tiene un conjunto de datos en un archivo CSV llamado datos_ecologicos.csv, al que se pueden cargar los datos en R utilizando la función read.csv(). Esta tabla es una matriz para modelos:

```
datos <- read.csv("datos_ecologicos.csv")
```

Paso 2. Visualizar los datos. Se pueden crear gráficos para visualizar la relación entre las variables utilizando la función plot().

```
plot(datos$variable_independiente, datos$variable_dependiente)
```

La variable independiente pueden ser la diversidad, riqueza de especies, biomasa. La variable dependiente pueden ser temperatura, humedad, pH del suelo.

Paso 3. Ajustar el modelo de regresión. Para ajustar un modelo de regresión lineal, se puede utilizar la función lm(), donde *lm* significa modelo lineal:

```
modelo <- lm(variable_dependiente ~ variable_independiente,
data = datos)
```

Output

Paso 1. Evaluar el modelo. Se puede obtener un resumen del modelo utilizando la función summary():

```
summary(modelo)
```

Paso 2. Interpretar los resultados. El resumen del modelo proporciona información sobre los coeficientes de regresión, la significancia estadística y la bondad de ajuste del modelo.

Paso 3. Visualizar el modelo. Crear gráficos para visualizar el ajuste del modelo utilizando la función plot():

```
plot(datos$variable_independiente, datos$variable_dependiente)
abline(modelo, col = "red")
```

Paso 4. Los coeficientes de regresión estimados se encuentran bajo la columna *Estimate* en el resumen del modelo. Cada coeficiente representa la magnitud del efecto de la variable independiente sobre la variable dependiente. Por ejemplo, un coeficiente de 2.5 significa que, en promedio, la variable dependiente aumenta en 2.5 unidades por cada unidad de cambio en la variable independiente, manteniendo constantes todas las demás variables.

La columna *Pr(>|t|)* muestra los valores p asociados a cada coeficiente. Estos valores p indican la significancia estadística del coeficiente. Un valor p menor que el nivel de significancia (usualmente 0.05) sugiere que el coeficiente es significativamente diferente de cero.

Análisis de correlación

Se utiliza para determinar si existe una relación estadísticamente significativa entre dos variables, como la temperatura del agua y la densidad de población de un organismo acuático. La *correlación de Pearson* y la *correlación de Spearman* son métodos comunes para este propósito.

Input

Paso 1. Cargar los datos. Supóngase que se tiene un conjunto de datos en un archivo CSV llamado datos_ecologicos.csv, cargar los datos en R utilizando la función read.csv():

```
datos <- read.csv("datos_ecologicos.csv")
```

Paso 2. Calcular la matriz de correlación. Para calcular la matriz de correlación entre las variables, utilizar la función cor():

```
matriz_cor <- cor(datos)
```

Output

Paso 1. Visualización de datos:

a) Visualizar la matriz de correlación. Para visualizar la matriz de correlación, utilizar la función corrplot() del paquete corrplot:

```
install.packages("corrplot")
library(corrplot)
corrplot(matriz_cor, method = "circle")
```

b) Otras formas de visualización, como un mapa de calor:

```
corrplot(matriz_cor, method = "color")
```

Paso 2. Identificar relaciones significativas. Para identificar relaciones significativas, observar los coeficientes de correlación y su significancia estadística. Calcular la significancia utilizando la función cor.test():

```
resultado_prueba <- cor.test(datos$variable1, datos$variable2)
resultado_prueba
```

Paso 3. Interpretar los resultados. La matriz de correlación muestra los coeficientes de correlación entre pares de variables. Los valores pueden variar entre -1 y 1, donde un valor de 1 indica una correlación positiva perfecta y un valor de -1 indica una correlación negativa perfecta. Por otor lado, un valor cercano a 0 indica una correlación débil o nula.

Análisis de Componentes Principales (PCA)

Este análisis se utiliza para reducir la dimensionalidad de un conjunto de datos, lo que permite identificar patrones y tendencias en la relación entre múltiples variables ambientales y la estructura de las comunidades biológicas. El PCA puede ayudar a identificar qué variables ambientales explican la mayor parte de la variabilidad observada en las comunidades biológicas.

Input

Paso 1. Cargar los datos. Supóngase que se tiene un conjunto de datos en un archivo CSV llamado datos_ecologicos.csv, cargar los datos en R utilizando la función read.csv(). En este caso, usar una matriz para modelos únicamente con valores numéricos:

```
datos <- read.csv("datos_ecologicos.csv")
```

Paso 2. Seleccionar las variables para el PCA. En el PCA es común estandarizar las variables para que tengan una media de cero y una desviación estándar de uno. Usar la función scale() para estandarizar las variables:

```
datos_estandarizados <- scale(datos)
```

Paso 3. Realizar el PCA. Para realizar el PCA, utilizar la función prcomp():

```
pca <- prcomp(datos_estandarizados, scale = TRUE)
```

Output

Paso 1. Explorar los resultados del PCA y obtener información sobre las componentes principales y su contribución a la variabilidad total utilizando la función summary():

```
summary(pca)
```

Paso 2. Visualizar los resultados del PCA, así como la proporción de la variabilidad explicada por cada componente principal, utilizando un gráfico de barras.

```
plot(pca)
```

Paso 3. Seleccionar componentes principales. Para seleccionar un número determinado de componentes principales que expliquen una cantidad suficiente de la variabilidad total, por ejemplo, se pueden seleccionar los primeros k PC que expliquen, en conjunto, al menos el 70-80% de la variabilidad total.

Paso 4. Proyectar los datos en el espacio de las componentes principales. Utilizar la función predict() para proyectar los datos originales en el espacio de las componentes principales:

```
datos_proyectados <- predict(pca, newdata =
datos_estandarizados)
```

Paso 5. Interpretar los resultados. Las componentes principales (PC) representan las direcciones de máxima variabilidad en los datos. La proporción de variabilidad explicada por cada PC se encuentra en la salida del resumen del PCA (esta proporción esta explicada como un porcentaje). Pueden interpretarse los PC en función de las variables originales para entender qué características de los datos contribuyen más a la variabilidad observada.

Análisis de Correspondencia Canónica (CCA)

En los estudios ecológicos a menudo se investigan gradientes ambientales, como la altitud, la latitud o la disponibilidad de nutrientes. El análisis de correspondencia canónica (CCA) permite evaluar cómo varían las comunidades biológicas a lo largo de estos gradientes y cómo estas variaciones están relacionadas con las variables ambientales.

Input

Paso 1. Cargar los datos. Usar dos tablas, una matriz de abundancia de tipo uno y una matriz ambiental.

```
abundancia <- read.csv("abundancia.csv")
ambiente <- read.csv("ambiente.csv")
```

Paso 2. Ejecución del CCA. Para realizar el CCA, utilizar la función cca() del paquete vegan:

```
install.packages("vegan")
library(vegan)

cca_resultado <- cca(abundancia ~ ., ambiente)
```

Output

Paso 1. Explorar los resultados del CCA. Obtener información sobre la importancia de las variables ambientales y biológicas utilizando la función summary():

```
summary(cca_resultado)
```

Paso 2. Visualizar los resultados del CCA. Para visualizar la relación entre las variables ambientales y biológicas, utilizar gráficos de ordenación (ordination plots):

```
plot(cca_resultado)
```

Paso 3. Seleccionar variables ambientales significativas. Para identificar las variables ambientales más importantes, utilizar pruebas de significancia, como el test de permutación:

```
anova_resultado <- anova.cca(cca_resultado)
significativas <- anova_resultado$ANOVA[,"Pr(>F)"] < 0.05
variables_ambientales_significativas <- rownames(anova_
resultado$ANOVA)[significativas]
```

Paso 4. Interpretar los resultados. El CCA identifica las variables ambientales que explican la mayor parte de la variabilidad en las variables biológicas. Pueden interpretar las relaciones entre las variables ambientales y biológicas en función de la proximidad en el gráfico de ordenación.

Test de Mantel en R

Este test evalúa la relación entre dos matrices de distancia, como una matriz de distancia biológica (basada en la similitud entre comunidades biológicas) y una matriz de distancia ambiental (basada en la similitud entre las variables ambientales). El test de Mantel determina si estas dos matrices están correlacionadas, lo que sugiere que las variables ambientales están relacionadas con la estructura de las comunidades biológicas. Este enfoque es especialmente útil cuando se investigan múltiples variables ambientales y se busca identificar cuáles tienen el mayor impacto en las comunidades biológicas estudiadas.

Input

Paso 1. Cargar los datos. Supóngase que se tienen dos matrices de distancia: una matriz de distancia biológica y una matriz de distancia ambiental. Cargar estas matrices en R.

a) Matriz biológica (matriz de abundancia de tipo 1):

```
matriz_biologica <- read.csv("matriz_biologica.csv", header =
TRUE, row.names = 1)
```

b) Matriz de distancia ambiental (matriz geográfica):

```
matriz_ambiental <- read.csv("matriz_ambiental.csv", header =
TRUE, row.names = 1)
```

Paso 2. Calcular las matrices de distancia:

```
dist_abundancia <- vegdist(matriz_biologica, method = “bray”)
dist_geografica <- as.dist(distm(matriz_ambiental, fun =
distHaversine))
```

Paso 3. Calcular la matriz de correlación entre las matrices de distancia. Utilizar la función mantel() del paquete vegan para calcular el coeficiente de correlación de Mantel entre las dos matrices:

```
install.packages(“vegan”)
library(vegan)

resultado_mantel <- mantel(dist_abundancia, dist_geografica,
method = “pearson”, permutations = 999)
```

Output

Paso 1. Explorar los resultados del test de Mantel y obtener información sobre el coeficiente de correlación de Mantel y su significancia estadística:

```
resultado_mantel
```

Paso 2. Visualizar los resultados (opcional). Para visualizar las matrices de distancia y la relación entre ellas, utilizar gráficos de dispersión o mapas de calor.

Paso 3. Interpretar los resultados. El coeficiente de correlación de Mantel (rM) proporciona una medida de la asociación entre las dos matrices de distancia. Un valor de rM cercano a 1 indica una correlación positiva perfecta, mientras que un valor cercano a -1 indica una correlación negativa perfecta. Además, el valor p asociado al test de Mantel indica la significancia estadística de la correlación observada. Un valor p bajo (por ejemplo, < 0.05) sugiere que la correlación es significativa.

Recapitulando

Este capítulo examinó la influencia crucial de las variables ambientales en la estructura y dinámica de los ecosistemas. Estas variables, tanto abióticas como bióticas, como temperatura, humedad, pH y salinidad, determinan la distribución y abundancia de las especies, y su comprensión es esencial para la ecología.

Así también, abordó diversas técnicas estadísticas para analizar estas relaciones. El análisis de regresión y la correlación permiten cuantificar e identificar relaciones significativas entre variables ambientales y características de las comunidades. El Análisis de Componentes Principales (PCA) y el Análisis de Correspondencia Canónica (CCA) facili-

tan la reducción de la dimensionalidad y la evaluación de patrones en la variabilidad de los ecosistemas. Además, el test de Mantel ayuda a evaluar la asociación entre matrices de distancia biológica y ambiental.

La aplicación práctica de estos métodos en R, desde la carga y visualización de datos hasta la interpretación de resultados, ha demostrado cómo realizar análisis rigurosos. Estos enfoques son fundamentales para comprender cómo las variables ambientales afectan la organización y distribución de las especies, y son herramientas clave para la gestión y conservación de la biodiversidad.

Capítulo 9
Modelado de nicho ecológico

Néstor G. Valle y Lucas J. Mina

Este capítulo describe los pasos para construir un modelo de nicho ecológico utilizando el paquete ellipsenm[1], junto con otros paquetes complementarios (raster, kuenm, rgl, ggplot2), en el lenguaje de programación R. La elección de este paquete se debe a su sencillez a la hora de estimar la amplitud del nicho y por no requerir otras técnicas de modelización, ya que utiliza sólo registros de ocurrencia de especies.

Nicho ecológico

El término nicho ecológico ha sido utilizado por numerosos autores; sin embargo, su definición es muy discutida y se ha reformulado varias veces a lo largo del tiempo. El concepto de nicho fue empleado por primera vez por el ecólogo estadounidense Joseph Grinnell (1924), quien se refiere al papel de una especie en su comunidad y su respuesta a factores abióticos y bióticos en su entorno. Más tarde, Charles Elton (1927) lo definió como el lugar en el ambiente biótico, su relación con el alimento y su interacción con otros organismos. Por lo tanto, estos dos autores califican el nicho ecológico como el lugar que ocupa o el papel que desempeña una especie en el medio.

Treinta años después, Evelyn Hutchinson (1957) estableció que el nicho de una especie se define como un «hipervolumen n-dimensional» que describe todas las condiciones ambientales y los recursos que permiten a una especie sobrevivir y reproducirse. En otras palabras, el nicho ecológico de una especie abarca todos los aspectos de su interacción con el medio, incluida su dieta, hábitat, tolerancias ambientales y relaciones con otras especies (competencia, depredación y mutualismo). Distinguió además entre nicho fundamental y nicho

1. Ellipsenm es un paquete de R para la caracterización de nichos ecológicos mediante elipsoides. Incluye opciones de calibración y selección de modelos, réplicas y proyecciones, además de la superposición de nichos.

realizado: el nicho fundamental se refiere al conjunto completo de condiciones abióticas en las que puede vivir una especie, mientras que el nicho realizado está constituido por las condiciones ambientales existentes en el área más las interacciones biológicas.

Diagrama BAM

En el contexto de la modelización de nichos, Soberón y Peterson (2005) utilizan un diagrama de Venn para representar el espacio geográfico y ecológico y explicar cómo influyen en la distribución la interacción de factores bióticos y abióticos y la accesibilidad de una región (figura 1). La región A representa el espacio ecológico donde se dan las condiciones abióticas adecuadas para la supervivencia de una especie; es la expresión de lo que Hutchinson (1957) definió como nicho fundamental (NF). La región B representa las relaciones bióticas interespecíficas positivas o negativas que influyen en la presencia de una especie, y la región M es la región accesible a la especie, ya sea por la capacidad de dispersión intrínseca de los individuos o por la introducción deliberada o accidental resultante de actividades antropogénicas. En resumen, la presencia de una especie será más probable en aquellos lugares donde se cumplan determinados requisitos, como la accesibilidad para la llegada y donde las condiciones bióticas y abióticas sean favorables para el incremento poblacional de dicha especie.

El diagrama está delimitado por la intersección de B, A y M, que corresponde al nicho realizado (NR), el espacio geográfico y ecológico donde la especie está presente; la intersección de A y B (G1) representa la existencia de factores bióticos y abióticos favorables, pero no es accesible.

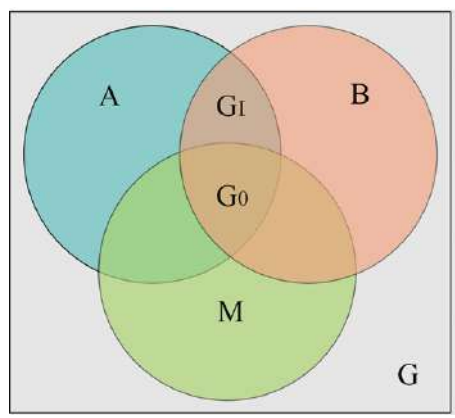

Figura 1. Diagrama de Venn ilustrando las interacciones de los factores que determinan la distribución geográfica de una especie. (A) Área abiótica idónea; (B) Área biótica idónea; (M) Área accesible para la especie; (GI) Área con condiciones ambientales y bióticas favorables, pero fuera de la accesibilidad de la especie; (G0) Área de distribución ocupada, que cuenta con condiciones ambientales y bióticas favorables y accesible, con presencia efectiva de la especie (Soberón y Peterson, 2005; Soberón, Osorio-Olvera, L. y Townsend Peterson, 2017).

Modelo de nicho ecológico

El área de distribución de una especie se documenta con base en la información procedente de publicaciones, de bases de datos disponibles en la red y de ejemplares recolectados en campañas y depositados en colecciones científicas. Esta área puede estimarse utilizando modelos predictivos de distribución (MDS) o de nicho ecológico (MNE) que relacionan las ocurrencias puntuales conocidas con un conjunto de variables independientes, generalmente climáticas, y mediante un algoritmo se obtiene una representación del espacio ecológico que luego se proyecta en un espacio geográfico, prediciendo así el grado de idoneidad de los lugares y delimitando el área de distribución potencial de la especie, incluso en localidades donde aún no ha sido detectada (figura 2).

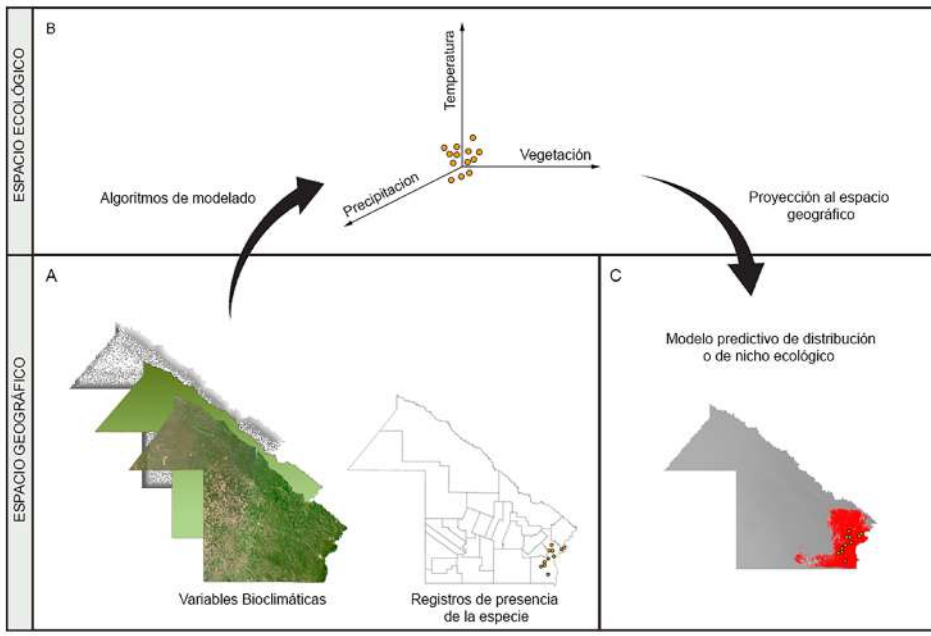

Figura 2. Representación del proceso de modelado del nicho ecológico y de la distribución geográfica de las especies. A) Registros de variables ambientales y lugares donde se ha registrado la especie. B) Espacio ecológico, donde se utilizan algoritmos matemáticos para combinar la distribución real conocida y un conjunto de variables independientes con el fin de proyectar la distribución pasada o futura de la especie. C) Modelo creado por el algoritmo, trasladado al espacio geográfico. Los círculos amarillos representan los registros de ocurrencia de la especie en la provincia del Chaco y en rojo el área de idoneidad climática.

Los modelos de distribución y nicho ecológico generan una estimación del área de presencia real o potencial. La modelización suele incluir sólo factores abióticos, y rara vez incorpora factores de dispersión o interacciones bióticas como predictores, pero estos también afectan a la distribución de las especies. Por ejemplo, para las especies fitófagas, la vegetación constituye una parte importante de su nicho ecológico; la presencia de su

planta huésped es esencial para el desarrollo satisfactorio del ciclo vital y el establecimiento de una población. La implementación de estos modelos de distribución ha tenido un incremento significativo en los últimos años, ya que tienen una amplia aplicación dada su utilidad para abordar distintas problemáticas. Así, los modelos predictivos se utilizan para resolver cuestiones relacionadas con la evaluación, planificación y toma de decisiones en materia de conservación y protección de especies amenazadas, en programas de manejo ambiental, en el hallazgo de especies endémicas y raras, en la evaluación del riesgo asociado a la ampliación de la distribución de las especies invasoras que puedan causar graves daños económicos y en el diseño de reservas.

Modelización de nichos ecológicos (MNE) en R

Para realizar el análisis de modelización, es necesario llevar a cabo una búsqueda exhaustiva de registros de ocurrencia de especies y variables ambientales en función de sus requerimientos ecológicos. Los datos de ocurrencia son recopilados de diferentes fuentes bibliográficas, colecciones de museos, herbarios, plataformas digitales (ejemplo, GBIF, speciesLink, etc.), que permiten un acceso rápido y hacen factible el análisis de taxones a gran escala en periodos de tiempo relativamente acotados. Respecto a las variables ambientales, existen varias fuentes o plataformas que proporcionan datos climáticos (ejemplo, WorldClim[2], CHELSA, EarthEnv, etc.). Si bien el cambio climático puede tener efectos directos sobre la abundancia y distribución específica, es necesario considerar a los factores bióticos (ejemplo, plantas hospederas).

En este capítulo, el espécimen utilizado para ejemplificar los pasos de modelado de nicho ecológico es *Lepturges (Lepturges) limpidus* Bates, 1872, un cerambícido exclusivo del Neotrópico, distribuido desde México hasta el sur del Paraguay (Valle y Simões, 2022).

Input

Paso 1. Descargar los datos. Las variables climáticas pueden descargarse directamente de WorldClim u obtenerlas desde de R. Se utilizan variables con escenario para el presente, con una resolución espacial de 2,5 arcmin (~4,6 km en el Ecuador) proyectadas en WGS84. Además, se consideró a su única planta hospedera, *Catostemma fragrans* Benth., especie nativa de las Guayanas, como predictor biótico.

El *shapefile* de la región Neotropical propuesto por Morrone (2014) se utilizó como capa de proyección y se descargó mediante la función download.file. Las variables climáticas de WorldClim se ajustaron para esta región biogeográfica en respuesta a los registros de ocurrencia:

2. WorldClim es el repositorio más conocido y ampliamente utilizado en ecología predictiva.

```
# Variables ambientales
mapas_worldclim <- "https://biogeo.ucdavis.edu/data/worldclim/
v2.1/base/wc2.1_2.5m_bio.zip"

tmp <- tempfile()
download.file(mapas_worldclim, destfile = tmp)
unzip(tmp, exdir = file.path("data/base/worldclim", bio_all))
unlink(tmp)

# Shapefile neotropico
tmp <- tempfile()
download.file(
  "https://figshare.com/ndownloader/articles/3569361/versions/2",
  destfile = tmp
)
unzip(tmp, exdir = "data/base/mascara_neotropico")
unlink(tmp)
```

Paso 2. Separar las variables. De las 19 variables extraídas de WorldClim, 4 (bio 8, bio 9, bio 18 y bio 19) fueron excluidas con el siguiente script por artificios espaciales conocidos entre celdas de cuadrícula adyacentes, y las capas en tres grupos: bio_all, con todas las capas; bio_temp, con las capas de variables de temperatura y bio_prec, con las capas de variables de precipitación:

```
for(i in c(8, 9, 18, 19)) {
  path <- file.path("./data/base/worldclim/bio_all", sprint-
f("wc2.1_2.5m_bio_%s.tif", i))
  file.remove(path)
}
dir.create("./data/base/worldclim/bio_temp")
dir.create("./data/base/worldclim/bio_prec/")
for (i in c(1:11)) {
  if (i == 8 || i == 9) {
    next
  }
  path <- file.path("./data/base/worldclim/bio_all", sprint-
f("wc2.1_2.5m_bio_%s.tif", i))
  file.copy(from = path, to = "./data/base/worldclim/bio_temp/")
}
for (i in c(12:17)) {
  path <- file.path("./data/base/worldclim/bio_all", sprint-
f("wc2.1_2.5m_bio_%s.tif", i))
  file.copy(from = path, to ="./data/base/worldclim/bio_prec/")
}
```

Paso 3. Preparar los datos necesarios (mapas, tablas, PCAs y máscaras):

```
# ---- Librerias ----

library(raster)
library(kuenm) # https://github.com/marlonecobos/kuenm
library(ellipsenm) # https://github.com/marlonecobos/ellipsenm/
```

Paso 4. Depurar los datos. En los siguientes fragmentos de código se definen dos funciones que se utilizan para la depuración de los datos de ocurrencias (limpieza) y para la eliminar las áreas de calibración (cortar_mapas):

```
# ---- Limpieza ----

limpieza <- function(datos, region, thin_dist) {
  dentro <- data.frame(
    datos,
    inside = !is.na(raster::extract(region, datos[-1]))
  )
  dentro <- dentro[dentro$inside == TRUE, ][1:3]
  dentro <- dentro[!is.na(dentro$Lat) & !is.na(dentro$Long), ]
  unicos <- unique(dentro)
  thin_data(unicos, longitude = "Long", latitude = "Lat", thin_
distance = thin_dist)
}

# ---- Máscara ----

mascara <- function(ocurrencias, variables, salida, distancia) {
  variables <- raster::stack(
    list.files(file.path(variables, "Initial"), pattern = "pc_", full.
names = TRUE)
  )
  buffer_oc <- buffer_area(
    ocurrencias,
    longitude = "Long",
    latitude = "Lat",
    buffer_distance = distancia
  )
```

```
  mascara <- raster::mask(raster::crop(variables, buffer_oc),
buffer_oc)
  dir.create(salida)
  raster::writeRaster(mascara, filename = file.path(salida, "pc"),
format = "ascii", bylayer = TRUE)
}
```

Paso 5. Representar. El conjunto de datos recopilados para identificar las áreas de idoneidad climática de *L. limpidus* alcanzó un total de 98 registros de presencia y para su planta hospedadora se recopilaron 177 ocurrencias. Todas las locaciones se deben representar en una matriz de datos que muestre este orden: nombre de la especie, longitud y latitud (figura 3).

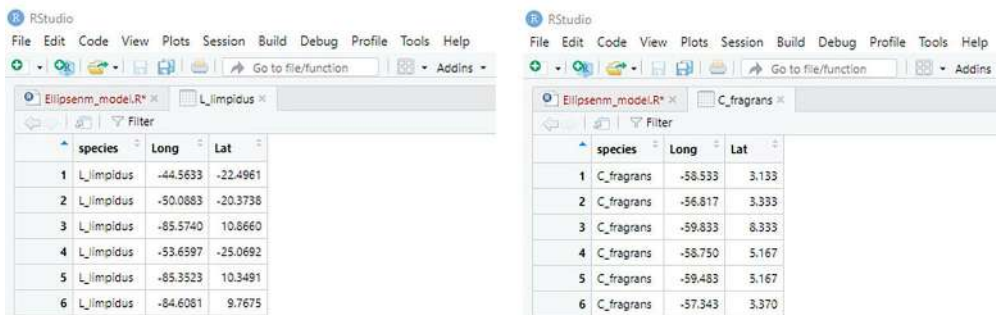

Figura 3. Ventana de RStudio donde se muestran las tablas de registro de presencia de *Lepturges limpidus* y *Catostemma fragrans* cargadas en formato CSV.

```
# ---- Limpieza ocurrencias ----

# Carga de datos
oc_l_limpidus <- read.csv("./data/base/ocurrencias_l_limpidus.csv")
oc_c_fragrans <- read.csv("data/base/ocurrencias_c_fragrans.csv")
region_neo <- raster("./data/bio/bio_all.tif", lyrs = 1)
plot(region_neo, col = "grey70", legend = FALSE)

# Limpieza
oc_l_limpidus_neot <- limpieza(oc_l_limpidus[1:3], region_neo, 10)
points(oc_l_limpidus_neot$Long, oc_l_limpidus_neot$Lat)
write.csv(oc_l_limpidus_neot, file = "data/ocurrencias/l_limpidus.
csv")
```

```
oc_c_fragrans_neot <- limpieza(oc_c_fragrans, region_neo, 10)
points(oc_c_fragrans_neot$Long, oc_c_fragrans_neot$Lat, col =
“seagreen”)
write.csv(oc_c_fragrans_neot, file = “data/ocurrencias/c_fragrans_
hp.csv”)
```

Paso 6. Al depurar el número de registros de ocurrencia se obtuvo como resultado final un total de 42 registros para *L. limpidus* y 28 presencias para *C. fragrans*. Se utilizó una distancia de adelgazamiento espacial de 10 km (figura 4).

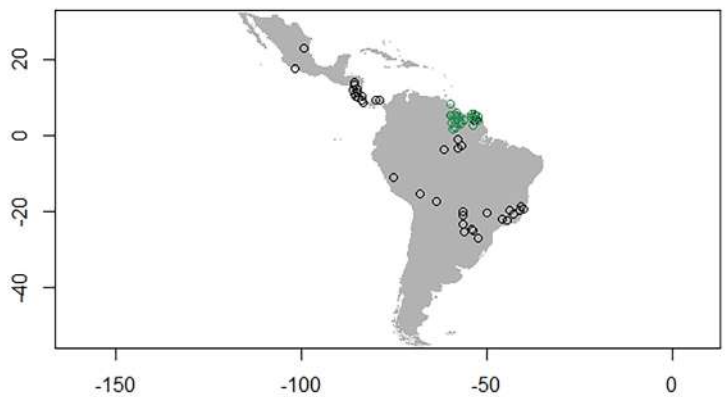

Figura 4. Capa de proyección mostrando las locaciones depuradas de *L. limpidus* (círculos negros) y *C. fragrans* (círculos verdes).

Paso 7. Con la finalidad de mejorar la eficiencia y precisión del modelo, se realizó un análisis de componentes principales (en español ACP, en inglés PCA), lo que permite reducir la dimensionalidad de los datos y eliminar los problemas de multicolinealidad.

Paso 8. Se probaron tres conjuntos ambientales distintos para evitar sesgos en la combinación de variables empleadas para caracterizar la centralidad de nicho de las especies. El conjunto 1 incluyó 15 variables, el conjunto 2 sólo variables de temperatura y el conjunto 3, únicamente variables de precipitación. Los resultados del PCA se guardaron en formato ascii, un archivo de texto plano que puede abrirse en cualquier editor de texto:

```
# ---- PCA ----

# Todas las variables
kuenm_rpca(
  variables = raster::stack(“./data/bio/bio_all.tif”),
  var.scale = TRUE,
```

```
  write.result = TRUE,
  out.format = "ascii",
  out.dir = "./data/pcs_all",
  n.pcs = 3 )

# Temperatura
kuenm_rpca(
  variables = raster::stack("./data/bio/bio_temp.tif"),
  var.scale = TRUE,
  write.result = TRUE,
  out.format = "ascii",
  out.dir = "./data/pcs_temp",
  n.pcs = 3 )

# Precipitación
kuenm_rpca(
  variables = raster::stack("./data/bio/bio_prec.tif"),
  var.scale = TRUE,
  write.result = TRUE,
  out.format = "ascii",
  out.dir = "./data/ pcs_prec",
  n.pcs = 3 )
```

Paso 9. Como resultado se conservaron los tres primeros componentes de cada conjunto, ya que explicaban acumulativamente > 88% de la varianza. Asímismo, para delimitar el área de calibración (es decir, «M», la región accesible a la especie), se enmascararon y recortaron las capas de componentes principales incluyendo un buffer[3] (50 km) que rodeaba los registros de presencia conocidos:

```
# ---- Máscara ----
pcs <- c("./data/pcs_all", "./data/pcs_prec", "./data/pcs_temp")
for (carpeta in pcs) {
  salida <- file.path(carpeta, "l_limpidus")
  mascara(oc_l_limpidus_neot, carpeta, salida, 50)
}
for (carpeta in pcs) {
  salida <- file.path(carpeta, "c_fragrans")
  mascara(oc_c_fragrans_neot, carpeta, salida, 50)
}
```

3. El tamaño del buffer se define en relación con la capacidad de dispersión conocida de la especie en estudio.

Paso 10. Construcción de los elipsoides. Para caracterizar los nichos ambientales de *L. limpidus* y *C. fragans*, se utilizaron elipsoides como modelos de nicho climático. Este método resulta óptimo al momento de interpretar el espacio de parámetros multidimensional de las variaciones del hábitat y establecer una clasificación aproximada de los registros según pertenezcan a poblaciones fuente o sumidero[4].

Paso 11. Con el fin de comprobar si la planta hospedadora podría estar limitando la distribución de *L. limpidus*, se calculó la superposición de nichos elipsoidales considerando la unión de las condiciones ambientales relevantes tanto para *L. limpidus* como para *C. fragans*[5]. El proceso se replicó 1.000 veces, y los valores de superposición observados se compararon con los encontrados para pares de elipsoides aleatorios:

```
# Definición funcion para leer los pcs generados anteriormente
leer_pcs <- function(carpeta, tipo = NULL) {
  carpetas <- list.files(carpeta, full.names = TRUE)
  nombres <- list.files(carpeta)
  raster_list <- lapply(carpetas, function(x) {
    raster <- raster::stack(
      list.files(x, pattern = "pc_", full.names = TRUE)
    )
    names(raster) <- paste0(names(raster), tipo)
    raster
  })
  names(raster_list) <- nombres
  raster_list
}

#Definición función para calculo de elipses
superposicion <- function(ocs_hp, ocs_cb, vars_hp, vars_cb) {

  nicho_1 <- overlap_object(
    ocs_hp, species =  "species",
    longitude = "Long", latitude = "Lat",
    method = "covmat", level = 95,
    variables = vars_hp)
```

4. El concepto de población fuente y sumidero es importante en la conservación de especies, ya que puede influir en las estrategias de gestión y conservación.

5. Existen diferentes tipos de superposición: all, full y back_union. Se optó por la tercera opción porque se pretende medir la superposición de nichos elipsoidales considerando sólo la unión de las condiciones ambientales relevantes para las dos especies en estudio.

```
  nicho_2 <- overlap_object(
    ocs_cb, species =  “species”,
    longitude = “Long”, latitude = “Lat”,
    method = “covmat”, level = 95,
    variables = vars_cb)

  ellipsoid_overlap(
    nicho_1, nicho_2, overlap_type = “back_union”,
    significance_test = TRUE, replicates = 1000,
    confidence_limit = 0.05)
}


# ---- Elipsoides ----

# Variables
pcs_all <- leer_pcs(“data/pcs_all/”)
pcs_prec <- leer_pcs(“data/pcs_prec/”)
pcs_temp <- leer_pcs(“data/pcs_temp/”)

# Ocurrencias
oc_l_limpidus <- read.csv(“./data/ocurrencias_L_limpidus.csv”)
oc_c_fragrans <- read.csv(“./data/ocurrencias_c_fragrans.csv”)

# Superposición con prueba de significación
overlap_all <- superposicion(
  ocs_hp = oc_c_fragrans_neot,
  ocs_cb = oc_l_limpidus_neot,
  vars_hp = pcs_all$c_fragrans,
  vars_cb = pcs_all$l_limpidus
)
overlap_prec <- superposicion(
  ocs_hp = oc_c_fragrans_neot,
  ocs_cb = oc_l_limpidus_neot,
  vars_hp = pcs_prec$c_fragrans,
  vars_cb = pcs_prec$l_limpidus
)
overlap_temp <- superposicion(
  ocs_hp = oc_c_fragrans_neot,
  ocs_cb = oc_l_limpidus_neot,
  vars_hp = pcs_temp$c_fragrans,
  vars_cb = pcs_temp$l_limpidus
```

```
)

dir.create("./output/overlaps", recursive = TRUE)
saveRDS(overlap_all, "./output/overlaps/overlap_all.rds")
saveRDS(overlap_prec, "./output/overlaps/overlap_prec.rds")
saveRDS(overlap_temp, "./output/overlaps/overlap_temp.rds")
```

Paso 12. Proyección del modelo:

```
# ---- Carga de datos ----

# Se agraga un sufijo a cada capa del raster (_all, _prec, _temp)
# dependiendo del conjunto de predictors climáticos.

pcs_all <- leer_pcs("./data/pcs_all", tipo = "_all")
pcs_prec <- leer_pcs("./data/pcs_prec", tipo = "_prec")
pcs_temp <- leer_pcs("./data/pcs_temp", tipo = "_temp")

# ---- Preparación para análisis ----

dir.create("./output/projection_model/splits", recursive = TRUE)
data_split <- split_data(
  oc_cerambicido, method = "random", longitude = "Long",
  latitude = "Lat", train_proportion = 0.75,
  save = TRUE, name = "./output/projection_model/splits/ocurren-
cias"
)

# Juntando las variables de cerambicido
# para luego armar los sets
vars_cerambicido <- raster::stack(
  pcs_all$l_limpidus, pcs_prec$l_limpidus, pcs_temp$l_limpidus
)

sets <- list(
  set_1 = c("pc_1_all", "pc_2_all", "pc_3_all"),
  set_2 = c("pc_1_prec", "pc_2_prec", "pc_3_prec"),
  set_3 = c("pc_1_temp", "pc_2_temp", "pc_3_temp")
)

sets_vars_cerambicido <- prepare_sets(vars_cerambicido, sets)
```

```
# ---- Calibracion ----

metodos <- c("covmat", "mve1")
calibracion <- ellipsoid_calibration(
  data_split, species = "species", longitude = "Long",
  latitude = "Lat", variables = sets_vars_cerambicido,
  methods = metodos, level = 99, selection_criteria = "S_OR_P",
  error = 5, iterations = 500, percentage = 50,
  output_directory = "output/projection_model/calibration_pcs"
)
res_calibracion <- read.csv(
  "./output/projection_model/calibration_pcs/selected_
parameterizations.csv"
)
res_calibracion
```

Paso 13. Para construir los modelos elipsoidales, existen diferentes métodos disponibles: covmat, mve1 y mve2. En este ejemplo se utilizó covmat, basado en el centroide y una matriz de covarianzas de las variables, y mve1, que genera un elipsoide que minimiza el volumen sin perder datos.

Por otra parte, el rendimiento de los modelos se evaluó en función de la significación estadística «ROC parcial», la tasa de omisión (E = 5%) y la prevalencia. Para calcular la métrica ROC parcial, se utilizaron 500 iteraciones Bootstrap, con un 50% de los datos de prueba y un 5% de error de incertidumbre. La prevalencia se calculó en el espacio geográfico y ambiental.

Los parámetros finales se eligieron basándose en los modelos mejor evaluados, se produjeron 10 réplicas utilizando el 75% de los datos.

Paso 14. Por último, para convertir el modelo de idoneidad continuo de Maxent en un mapa binario de «presencia-ausencia» de condiciones ambientales idóneas, se utilizó un valor umbral basado en la presencia mínima de entrenamiento, asumiendo un porcentaje de error en las ocurrencias del 5%:

```
# ---- Modelo ----

modelo <- ellipsoid_model(
  data = oc_cerambicido, species = "species",
  longitude = "Long", latitude = "Lat",
  raster_layers = pcs_temp$Initial, method = "covmat", level = 99,
  replicates = 10, prediction = "suitability",
```

```
  return_numeric = TRUE, format = "GTiff", overwrite = FALSE,
  output_directory = "output/projection_model/model_pcs"
)
# Umbral del modelo
modelo_mean <- raster(
"./output/projection_model/model_pcs/mean_suitability_
calibration_l_limpidus.tif"
)

# Threshold del 5%
# Se tomó el valor más bajo recuperado y se utilizó para
transformar el modelo en binario.
puntos_cerambicido <- extract(modelo_mean, oc_cerambicido[-1])
modelo_mean_binario <- (modelo_mean > min(puntos_cerambicido))
plot(modelo_mean_binario)
writeRaster(
  modelo_mean_binario,
  "./output/projection_model/final_model_thrs_5.tif",
)
library(readr)
nuevo <- read_csv("data/ocurrencias/l_limpidus_nr.csv")
View(l_limpidus_nr)
points(oc_l_limpidus_neot$Long, oc_l_limpidus_neot$Lat, col =
"black", cex = 0.3)
points(nuevo$Long, nuevo$Lat, col = "red", cex = 0.3)
```

Paso 15. La distancia de Mahalanobis calcula la distancia entre ocurrencias, siendo un método sencillo para detectar errores espaciales. Los valores de idoneidad en los modelos de envoltura elipsoidal representan la distancia de Mahalanobis desde el óptimo (es decir, el centroide del elipsoide), de modo que los valores máximos estarán cerca del centroide y los mínimos cerca de la periferia del elipsoide:

```
# Crear tabla de especies alien y native de l_limpidus
# ---- Ocurrencias con status ----

l_limpidus_nr <- read.csv("./data/base/l_limpidus_nr.csv")
l_limpidus_nr$status  <- "alien"
oc_l_limpidus_neot$status <- "native"
l_limpidus_status <- rbind(oc_l_limpidus_neot, l_limpidus_nr)
write.csv(l_limpidus_status, file = "./data/ocurrencias/l_limpidus_
status.csv", row.names = FALSE)
```

```
# ---- Distancia Mahalanobis ----

cb_dist <- data.frame(
  l_limpidus_status,
  dist = extract(modelo_mean, l_limpidus_status[2:3])
)
cb_dist <- na.omit(cb_dist)
write.csv(cb_dist, file = "./output/dist_mahalanobis.csv")
```

Output

Paso 1. Construcción de los gráficos:

```
# ---- Librerias ----

library(rgl)
library(ggplot2)

# ---Gráficos ---

# Overlaps
plot_over <- function(overlap) {
  plot_overlap(
  overlap, niche_col = c( "brown1", "deepskyblue"),
  data_col = c("brown1", "deepskyblue"), background = FALSE,
  change_labels = TRUE, xlab = "", ylab = "", zlab = "", legend= T)

}
plot_over(overlap_all) rgl.snapshot( "plots/overlap_all.png", fmt =
"png", top = TRUE )
plot_over(overlap_prec) rgl.snapshot( "plots/overlap_prec.png", fmt
= "png", top = TRUE )
plot_over(overlap_temp) rgl.snapshot( "plots/overlap_temp.png", fmt
= "png", top = TRUE )

# Histogramas
# La función plot_hist crea gráficos usando R base.
# La función gg_hist lo hace utilizando ggplot2

plot_hist <- function(overlap){
```

```
  hist(overlap@significance_results$union_random$Niche_1_vs_2$over-
lap,
    breaks = 5, main = "Overlap HP x Cb", xlab = "Overlap",
    xlim = c(0, 1), ylim = c(0, 650)
  )
  abline(v = quantile(overlap@significance_results$union_random$Ni-
che_1_vs_2$overlap, 0.05),
    col = "red", lwd = 2, lty = 2
  )
  abline(v = overlap@union_overlap$overlap[1], col = "lightblue",
lwd = 2)
  legend("topright", bty = "n", legend = c("Observed", "5% CL"),
    col = c("lightblue", "red"), lty = c(1, 2), lwd = 2
  )
}
plot_hist(overlap_all)
plot_hist(overlap_prec)
plot_hist(overlap_temp)

gg_hist <- function(overlap, file) {
  datos <- data.frame(
    Overlap = overlap@significance_results$union_random$Niche_1_
vs_2$overlap
  )
  observado <- overlap@union_overlap$overlap
  cl <- quantile(overlap@significance_results$union_random$Niche_1_
vs_2$overlap, 0.05)
  plot <- ggplot(datos) +
    ylab("Frequency") +
    coord_cartesian(xlim = c(0, 1), ylim = c(0,650)) +
    geom_histogram(aes(x = Overlap), bins = nclass.Sturges(da-
tos[,1]), color = "black", fill = "gray85") +
    geom_vline(
      aes(xintercept = cl, linetype = "5% CL", color = "5% CL")
    ) +
    geom_vline(
      aes(xintercept = observado, linetype = "Observed", color =
"Observed")
    ) +
    scale_color_manual(
      name = "",
      values = c("red", "blue")
```

```
    ) +
    scale_linetype_manual(
      name = "",
      values = c("dashed", "solid")
    ) +
    theme_classic()
  ggsave(file, plot = plot, width = 7, height = 7)
  plot
}
gg_hist(overlap_all, "plots/overlap_all.svg")
gg_hist(overlap_prec, "plots/overlap_prec.svg")
gg_hist(overlap_temp, "plots/overlap_temp.svg")

# Mahalanobis

hist(cb_dist$dist)
abline(v = cb_dist$dist[cb_dist$status == "alien"], col = "red", lwd
= 1)

maha_plot <- ggplot(cb_dist, aes(x = dist)) +
  ylab("Frequency") +
  xlab("Distance") +
  geom_histogram(bins = nclass.Sturges(cb_dist $dist), color =
"black", fill = "gray") +
  geom_vline(data = subset(cb_dist, status == "alien"), aes(xinter-
cept = dist), color = "red") +
  theme_classic()
maha_plot
ggsave("./plots/mahalanobis_hist.svg", plot = maha_plot, height = 7,
width = 7)
```

Paso 2. Láminas que se obtienen del output:

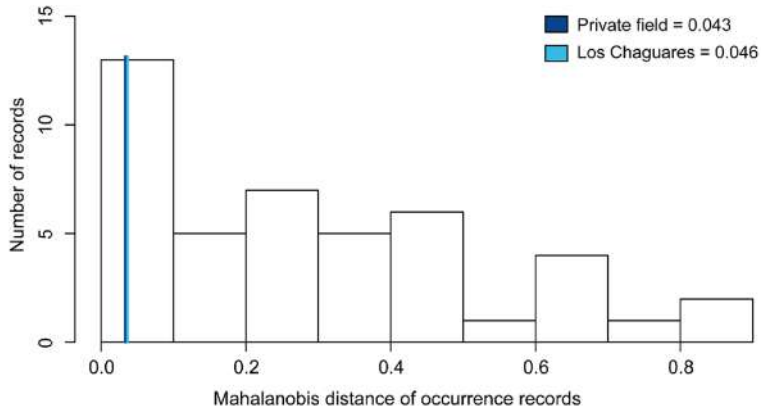

Figura 5. Cálculo de la distancia entre ocurrencias de Mahalanobis.

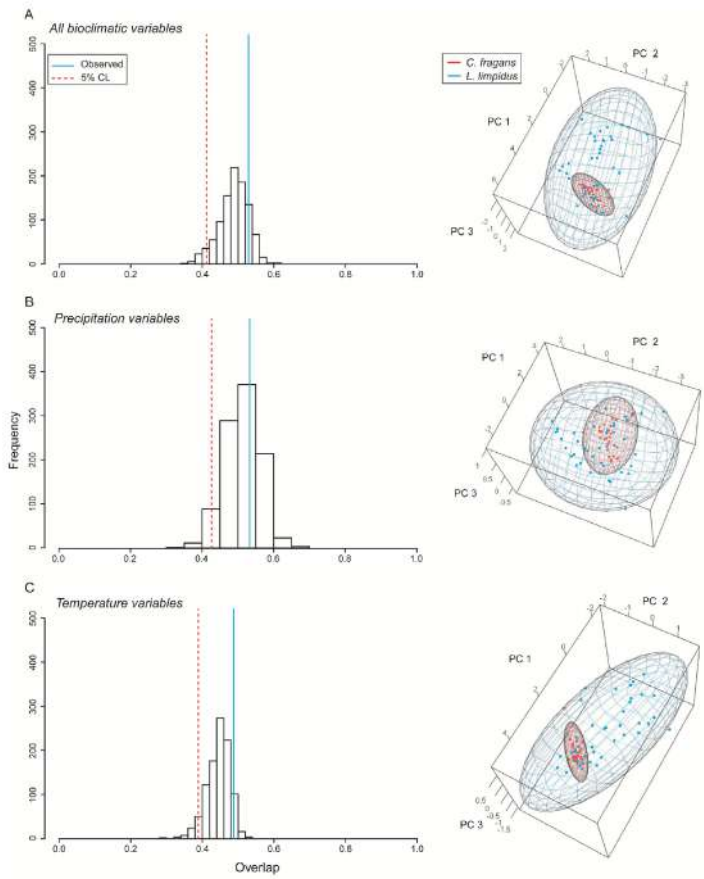

Figura 6. Elipsoides representando el nicho de *C. fragans*, en rojo, y *L. limpidus* en azul para A) todas las variables bioclimáticas, B) las variables de precipitación y C) las variables de temperatura.

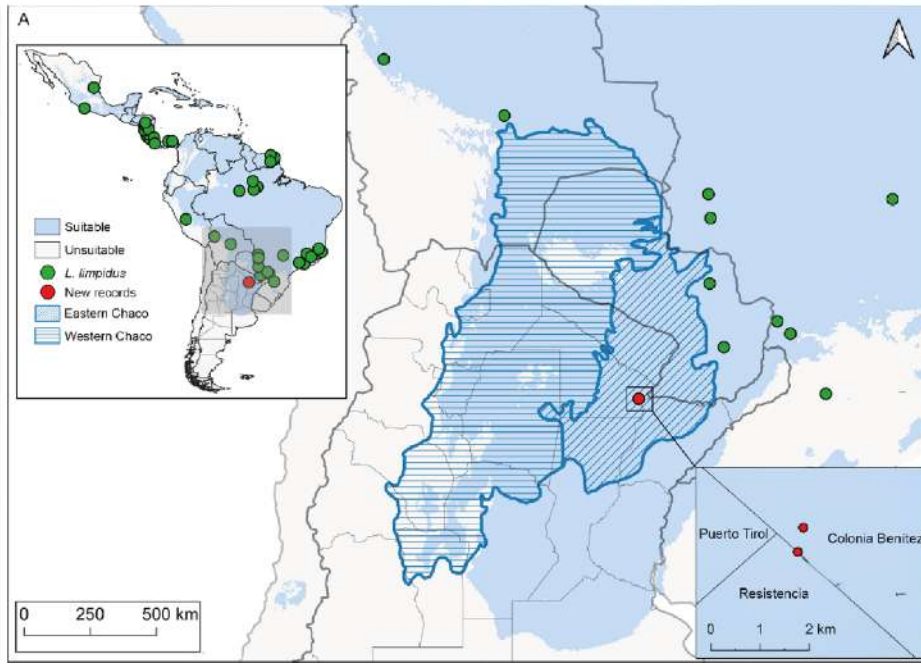

Figura 7. Mapa de idoneidad. Zonas idóneas en celeste y no idóneas en blanco. Los puntos en rojo indican los nuevos registros de *L. limpidus*.

Paso 3. Interpretación de los resultados. Estos modelos permiten no sólo entender la distribución actual de las especies, sino también prever su expansión futura y evaluar las posibles áreas de riesgo. Comprender la interacción entre las condiciones climáticas y los factores ecológicos es fundamental para diseñar estrategias efectivas de conservación y control, especialmente en el contexto de la creciente propagación de especies no nativas y/o exóticas.

Teniendo en cuenta el ejemplo, se puede concluir que la construcción de modelos elipsoidal reveló una amplia idoneidad climática para *Lepturges (Lepturges) limpidus* en la región Neotropical, sugiriendo que los nuevos registros en Argentina no son anomalías climáticas, sino indicativos de un potencial establecimiento. Además, la falta de dependencia exclusiva de su planta huésped conocida sugiere una mayor plasticidad ecológica de la especie, lo que podría facilitar su expansión a nuevas áreas. Estos hallazgos subrayan la necesidad de realizar encuestas sistemáticas y aplicar medidas de control para prevenir una mayor propagación y posibles impactos ecológicos negativos.

Recapitulando

El modelado de nicho ecológico es una herramienta fundamental para comprender la distribución de las especies en función de sus interacciones con el medio ambiente. A lo

largo de la historia, el concepto de nicho ecológico ha evolucionado, desde las primeras definiciones centradas en el papel de una especie en su comunidad hasta la concepción moderna de un «hipervolumen n-dimensional» que integra todas las condiciones abióticas y bióticas necesarias para la supervivencia y reproducción de una especie.

El uso de modelos de nicho ecológico permite estimar la distribución potencial de una especie, lo que es crucial para la conservación, el manejo ambiental y la predicción de la expansión de especies invasoras. Al emplear técnicas como el análisis de componentes principales y la superposición de elipsoides, se pueden generar representaciones más precisas del nicho ecológico, lo que mejora la eficacia de las acciones de conservación. La capacidad de integrar datos climáticos, de ocurrencia y de interacciones bióticas en modelos predictivos hace que esta metodología sea indispensable en la biogeografía y en la toma de decisiones ambientales.

Capítulo 10
La importancia de los análisis ecológicos con R

Florencia Monti Areco, Matías I. Dufek y Darío D. Larrea

La ecología, como disciplina científica, ha experimentado una evolución significativa desde su definición por Ernst Haeckel en 1869. Originalmente, centrada en la observación y la descripción de los organismos y sus ambientes, la ecología se enfocaba en registrar y clasificar la biodiversidad a partir de observaciones directas. Este enfoque descriptivo, aunque crucial en sus inicios, resultaba limitado para abordar la complejidad de las interacciones entre especies y los procesos ecológicos subyacentes. Con el tiempo, la necesidad de una comprensión más profunda y cuantitativa llevó a la incorporación de herramientas estadísticas y modelos matemáticos en la investigación ecológica.

Durante la segunda mitad del siglo XX, los ecólogos comenzaron a integrar métodos estadísticos para analizar grandes volúmenes de datos, lo que permitió una comprensión más precisa de los patrones y procesos ecológicos. El uso de técnicas estadísticas avanzadas, como la regresión, el análisis de varianza y las pruebas de hipótesis, facilitó la validación de teorías y modelos ecológicos, y ofreció herramientas para identificar patrones en la diversidad de especies y las dinámicas poblacionales. La capacidad de analizar datos de manera rigurosa permitió a los ecólogos no sólo describir la biodiversidad, sino también comprender cómo los factores bióticos y abióticos interactúan para dar forma a los ecosistemas.

El advenimiento de la informática trajo consigo una nueva era de análisis ecológicos. Con la llegada de softwares especializados, como Past, EstimateS y PRIMER-e, los ecólogos tuvieron acceso a herramientas avanzadas para analizar la diversidad de especies, modelar nichos ecológicos y evaluar patrones espaciales. Estos programas ofrecieron soluciones específicas para preguntas ecológicas complejas, aunque a menudo requerían licencias costosas y estaban limitados a un rango específico de análisis.

En este contexto, R emergió como una alternativa poderosa y accesible. R es un software de código abierto que proporciona una plataforma flexible para el análisis de datos, con una amplia gama de paquetes estadísticos desarrollados por la comunidad científica. La capacidad de R para realizar desde análisis simples hasta modelos complejos ha revolucionado la

investigación ecológica, permitiendo a los investigadores personalizar y adaptar sus análisis según las necesidades específicas de sus estudios.

Una de las principales ventajas de R es su capacidad para promover la transparencia y la reproducibilidad en la investigación científica. Al ser un software de código abierto, R permite a los investigadores compartir sus scripts y datos, facilitando la revisión y la reproducción de resultados. Esta apertura es esencial para garantizar la validez de los hallazgos científicos y fomentar la colaboración en la comunidad ecológica global.

A pesar de su creciente popularidad, la mayoría de los recursos y documentación de R están disponibles en inglés, lo que representa una barrera significativa para muchos investigadores de habla hispana. La falta de recursos en español limita la capacidad de los investigadores noveles para acceder y utilizar eficazmente este software, lo que puede ralentizar el progreso en la investigación ecológica en comunidades hispanohablantes. Un manual detallado en español no sólo facilitaría el aprendizaje y el uso de R, sino que también promovería una mayor inclusión y diversidad en la ciencia, permitiendo que un mayor número de investigadores participen en la investigación ecológica.

Además de facilitar el acceso a R, la creación de recursos en español fomentaría una mayor colaboración entre investigadores de diferentes regiones y contextos lingüísticos. La diversidad en la investigación ecológica es crucial para abordar desafíos globales, como la pérdida de biodiversidad y el cambio climático. Un enfoque inclusivo y colaborativo permitirá a los ecólogos compartir conocimientos, intercambiar ideas y desarrollar soluciones innovadoras para proteger y conservar nuestros ecosistemas.

La importancia de los análisis ecológicos con R se manifiesta en la capacidad de este software para abordar preguntas complejas sobre la biodiversidad y la dinámica de los ecosistemas. Desde el análisis de la estructura de comunidades y la evaluación de patrones espaciales hasta la modelización de las dinámicas poblacionales y la predicción de cambios ambientales, R ofrece herramientas poderosas para comprender y gestionar la biodiversidad en un mundo en constante cambio. La disponibilidad de recursos en español y la promoción de la participación inclusiva en la ciencia son pasos esenciales para avanzar en la investigación ecológica y enfrentar los desafíos ambientales globales de manera efectiva.

En conclusión, R se ha consolidado como una herramienta esencial para la investigación ecológica, proporcionando a los científicos la capacidad de realizar análisis detallados y personalizados. La creación de recursos en español y la promoción de la colaboración internacional son cruciales para maximizar el impacto de R en la ciencia ecológica y garantizar que la investigación sobre la biodiversidad y los ecosistemas sea accesible, transparente y efectiva en todo el mundo.

Anexo

A continuación, se compilan todos los scripts desarrollados en los distintos capítulos de esta obra, permitiendo una revisión completa y accesible de los contenidos presentados. Cada capítulo ofrece una visión profunda y técnica sobre el tema tratado, proporcionando al lector herramientas y conocimientos valiosos.

Capítulo 4. Pruebas de hipótesis estadísticas en estudios ecológicos

Prueba de normalidad de Shapiro-Wilk

```
resultado_shapiro <- shapiro.tesshapt(datos)

print(resultado_shapiro)

qqnorm(datos)
qqline(datos)
```

Prueba de Levene

```
library(car)

resultado_levene <- leveneTest(datos ~ grupo, data = datos)
print(resultado_levene)

# Gráfico de residuos vs. ajustes
modelo <- lm(datos ~ grupo, data = datos)
plot(modelo, which = 1)
```

Prueba t de Student

```
datos <- read.csv(“datos.csv”)

resultado <- t.test(grupo1, grupo2)
resultado

boxplot(grupo1, grupo2, names = c(“Grupo 1”, “Grupo 2”), col = c(“-
blue”, “red”))
```

ANOVA

```
datos <- read.csv(“datos.csv”)

modelo_anova <- aov(respuesta ~ tratamiento, data = datos)
summary(modelo_anova)

# Prueba posthoc
posthoc <- TukeyHSD(modelo_anova)

barplot(
    tapply(datos$respuesta, datos$tratamiento, mean),
    names.arg = levels(datos$tratamiento),
    ylab = “Media de respuesta”,
    xlab = “Tratamiento”,
    col = “lightblue”
)
```

Regresión lineal

```
datos <- read.csv(“datos.csv”)

modelo <- lm(respuesta ~ tratamiento, data = datos)
summary(modelo)

plot(
    datos$variable_independiente, datos$variable_dependiente,
    xlab = “Variable independiente”,
    ylab = “Variable dependiente”,
    main = “Regresión lineal”
)

abline(modelo, col = “red”)
```

MANOVA (Multivariate Analysis of Variance)

```
datos <- read.csv("datos.csv")

modelo_manova <- manova(cbind(variable_dependiente1, variable_depen-
diente2) ~ variable_independiente, data = datos)
summary(modelo_manova)

# Prueba posthoc
posthoc <- TukeyHSD(modelo_manova)

barplot(
    colMeans(datos[, c("variable_dependiente1", "variable_dependien-
te2")]),
    names.arg = levels(datos$variable_independiente),
    ylab = "Media de variables dependientes",
    xlab = "Variable independiente",
    col = "lightblue"
)
```

Prueba de Kruskal-Wallis

```
datos <- read.csv("datos.csv")

resultado_kruskal <- kruskal.test(variable_dependiente ~ variable_
independiente, data = datos)
resultado_kruskal

# Prueba posthoc
install.packages("dunn.test")
library(dunn.test)

resultado_dunn <- dunn.test(datos$variable_dependiente, g = datos$-
variable_independiente, method = "bonferroni")
```

Prueba de correlación de Spearman

```
datos <- read.csv("datos.csv")

resultado_spearman <- cor.test(datos$variable1, datos$variable2,
method = "spearman")
resultado_spearman
```

Prueba de Mann-Whitney U

```
datos <- read.csv("datos.csv")
resultado_mann_whitney <- wilcox.test(variable_dependiente ~ grupo,
data = datos)
resultado_mann_whitney
```

PERMANOVA (Permutational Multivariate Analysis of Variance)

```
datos <- read.csv("datos.csv")

library(vegan)

resultado_permanova <- adonis(datos ~ grupo, data = datos, permuta-
tions = 999)
resultado_permanova
```

Modelo lineal generalizado (GLM)

```
datos <- read.csv("datos.csv")

modelo <- glm(
    variable_dependiente ~ variable_independiente1 + variable_inde-
pendiente2,
    data = datos, family = binomial(link = "logit")
)
summary(modelo)

plot(modelo, which = 1)    # Gráfico de residuos
plot(modelo, which = 2)    # Gráfico Q-Q

plot(datos$variable_independiente, datos$variable_dependiente)
abline(modelo, col = "red")
```

Diversidad alfa - Riqueza

```
Mydir = (“C:/Users/Desktop/Datos”)
setwd (Mydir)

datos-estudio = read.table(“Datos.txt”, header = TRUE, sep = “\t”,
row.names = 1)
variables = read.table(“Variables.txt”, header = TRUE, sep = “\t”,
row.names = 1)

library(vegan)
attach(variables)

specpool(datos-estudio, Tratamiento)    # Riqueza
```

Índices basados en dominancia

```
Mydir = (“C:/Users/Desktop/Datos”)
setwd (Mydir)

Datos = read.table(“Datos.txt”, header = TRUE, sep = “\t”, row.names
= 1)

library(BiodiversityR)
library(kableExtra)

D = diversityresult(Datos, index=c(“Simpson”), method=c(“each
site”))
B = diversityresult(Datos, index=c(“Berger”), method=c(“each site”))
iD = diversityresult(Datos, index=c(“inverseSimpson”), method=-
c(“each site”))

indicesdomi = data.frame(D, iD, B)
kable(indicesdomi, format = “markdown”, col.names = c(“Simpson”,
“Inverso de Simpson”, “Berger”))
```

Índices basados en equidad

```
Mydir = (“C:/Users/Desktop/Datos”)
setwd (Mydir)

datos = read.table(“Datos.txt”, header = TRUE, sep = “\t”, row.names
= 1)

library(BiodiversityR)
library(kableExtra)

H = diversityresult(datos, index=c(“Shannon”), method=c(“each
site”))
JP = H/log(specnumber(datos))

indicesequi = data.frame(H, JP)
kable(indicesequi, format = “markdown”, col.names = c(“H, Shannon”,
“JP, Pielou”, “iD, inverseSimpson”))
```

Diversidad beta

```
Mydir = (“C:/Users/Desktop/Datos”)
setwd (Mydir)

datos = read.table(“Datos.txt”, header = TRUE, sep = “\t”, row.names
= 1)

library(betapart)

datos = ifelse(datos>0, 1, 0)

Beta.coreAM = betapart.core(Datos)
```

Beta partición en base Jaccard

```
Multi.jac = beta.multi(Beta.coreAM, index.family=”jac”)
Dist.jac = beta.pair(Beta.coreAM, index.family=”jac”)
```

Beta partición en base Sørensen

```
Multi.sor = beta.multi(Beta.coreAM, index.family=”sor”)
Dist.sor = beta.pair(Beta.coreAM, index.family=”sor”)
```

Distintividad taxonómica

```
library(BiodiversityR)
library(ggplot)
library(ggrepel)

abundancia <- read.table("tabla_abundancia.txt", header = TRUE, row.
names = 1, sep = "\t")
taxonomia <- read.table("tabla_taxa.txt", header = TRUE, row.names =
1, sep = "\t")

dist_taxonomia <- taxa2dist(taxonomia)
plot(hclust(dist_taxonomia), hang = 1)

div_taxonómica <- taxondive(abundancia, dist_toxonomica)
div_taxonomica

ggplot(div_taxonomica, aes(x = Species, y = Dplus)) +
    geom_point() +
    ylab("Δ+") +
    xlab("Riqueza") +
    geom_hline(aes(yintercept = EDplus), linetype = "dotted") +
    geom_ribbon(aes(ymax = EDplus + sd.Dplus * 2, ymin = EDplus -
sd.Dplus * 2), fill = NA, color = "black") +
    geom_text_repel(aes(label = row.names(div_taxonomica)), size =
3.5, color = "blue3") +
    theme_classic()
```

Diversidad funcional

```
library(FD)

selva_01 <- read.table("data/selva_1b.txt", header = TRUE, sep =
"\t")
selva_02 <- read.table("data/selva_2b.txt", header = TRUE, sep =
"\t")
selva_03 <- read.table("data/selva_3b.txt", header = TRUE, sep =
"\t")
```

```
# Diversidad funcional ambiente 1
resultados_ambiente1 <- dbFD(selva_01)
resultados_ambiente1

# Diversidad funcional ambiente 2
resultados_ambiente2 <- dbFD(selva_02)
resultados_ambiente2

# Diversidad funcional para el ambiente 3
resultados_ambiente3 <- dbFD(selva_03)
resultados_ambiente3

df_datos <- data.frame(
    Ambiente = c("Ambiente1", "Ambiente2", "Ambiente3"),
    FEve = c(resultados_ambiente1$FEve, resultados_ambiente2$FEve,
resultados_ambiente3$FEve),
    FDiv = c(resultados_ambiente1$FDiv, resultados_ambiente2$FDiv,
resultados_ambiente3$FDiv),
    FDis = c(resultados_ambiente1$FDis, resultados_ambiente2$FDis,
resultados_ambiente3$FDis),
    FRic = c(resultados_ambiente1$FRic, resultados_ambiente2$FRic,
resultados_ambiente3$FRic)
)
df_datos
```

Capítulo 7. Estudio de la estructura de la comunidad

Curvas de Whittaker

```
library(tidyverse)
library(readxl)
library(ggrepel)
library(BiodiversityR)

base_datos <- Sys.getenv("DB_LOCAL_HORMIGAS")
raw_data <- read_excel(base_datos, sheet = 1)

# Primero se extrae lo que nos interesa todo junto así nos asegura-
mos de que
# cuando se separen las tablas ambas tengan el mismo nombre y orden
de filas
```

```
datos <- raw_data %>%
    filter(estado_conservacion != “NA”) %>%
    group_by(localidad, estado_conservacion, especie) %>%
    summarise(ABUNDANCIA = sum(abundancia)) %>%
    pivot_wider(
      names_from = especie,
      values_from = ABUNDANCIA,
      values_fill = 0
    ) %>%
    column_to_rownames(“localidad”)

especies <- datos %>%
    select(!estado_conservacion)

ambiente <- datos %>%
    select(estado_conservacion) %>%
    mutate(estado_conservacion = factor(estado_conservacion))

# ---- Rangos de abundancia ----

rank_abundancia <- rankabundance(especies)
rankabunplot(rank_abundancia, scale = “abundance”)  #Fig. 7.2

rank_abundancia_amb <- rankabuncomp(
    especies,
    y = ambiente,
    factor = “estado_conservacion”,
    legend = FALSE
)
```

Marcamos especies únicas

```
especies_unicas <- lapply(
    unique(levels(ambiente$estado_conservacion)),
    function(group) {
        setdiff(subset(rank_abundancia_amb, Grouping == group)$spe-
cies,
            subset(rank_abundancia_amb, Grouping != group)$species)
    }
)
especies_unicas <- unlist(especies_unicas)
```

```
rank_abundancia_amb$unique <- ifelse(
    rank_abundancia_amb$species %in% especies_unicas, TRUE, FALSE
)

curva_whittaker <- function(x, group, scale, color, mark.unique =
FALSE) {
    if ("unique" %in% names(x)) {
        unique_num <- nrow(
            subset(x, Grouping == group & unique == TRUE)
        )
    }

    label <- sprintf("%s especies únicas", unique_num)
    xmax <- max(x[["rank"]])
    ymax <- max(x[[scale]])

    plot <- ggplot(x, aes(x = rank, y = .data[[scale]])) +
        coord_cartesian(
            xlim = c(0, xmax),
            ylim = c(0, ymax)
        ) +
    geom_point(
        data = subset(x, Grouping == group),
        size = 3,
        shape = 1,
        color = color
    ) +
    scale_shape_manual(
        name = NULL,
        breaks = c("unique"),
        label = c(label),
        values = c(19)
    ) +
    geom_line(
        data = subset(x, Grouping == group),
        color = color
    ) +
    geom_text_repel(
        data = subset(x, Grouping == group & labelit == TRUE),
         aes(label = species),
         hjust = 0,
         nudge_x = 3,
```

```
        size = 3
    ) +
    theme_classic()

    if (mark.unique) {
    plot <- plot +
        geom_point(
        data = subset(x, Grouping == group & unique == TRUE),
        aes(shape = "unique"),
        color = color,
        size = 3,
    ) +
    theme(legend.position = c(.7, .5))
    }
    return(plot)
}
curva_whittaker(x, group, scale, color, mark.unique = FALSE)

curva_ecb <- curva_whittaker(rank_abundancia_amb, "ECB", "abundan-
ce", "darkgreen", mark.unique = TRUE)
curva_eci <- curva_whittaker(rank_abundancia_amb, "ECI", "abundan-
ce", "orange", mark.unique = TRUE)
curva_ecd <- curva_whittaker(rank_abundancia_amb, "ECD", "abundan-
ce", "red", mark.unique = TRUE)

curva_ecb
curva_eci
curva_ecd

ggsave("./plots/curva_ecb.png", plot = curva_ecb)
ggsave("./plots/curva_eci.png", plot = curva_eci)
ggsave("./plots/curva_ecd.png", plot = curva_ecd)

ggplot(rank_abundancia_amb, aes(x = rank, y = abundance, color =
Grouping)) +
    geom_line() +
    geom_point(size = 2.5) +
    labs(color = "", shape = "") +
    scale_color_manual(breaks = c("ECB", "ECI", "ECD"), values =
c("darkgreen", "orange", "red"))+
    theme_classic() +
    theme(legend.position = "top")
```

Curvas de acumulación

```
# ---- Librerias ----

library(BiodiversityR)
library(ggplot2)

# ---- Carga de datos ----

abundancia_esp <- read.csv("./data/datos_abundancia.csv", row.names
= 1)

# ---- Curva de acumulación ----

# Dimensiones de los datos
dim(abundancia_esp)

curva <- specaccum(abundancia_esp)

datos_sp <- data.frame(
    Sitios = curva$sites,
    Riqueza = curva$richness,
    SD = curva$sd
)

ggplot(datos_sp, aes(x = Sitios, y = Riqueza)) +
    geom_ribbon(aes(ymin = Riqueza - SD, ymax = Riqueza + SD), fill =
"grey90") +
    scale_x_continuous(breaks = datos_sp$Sitios) +
    geom_line(color = "blue") +
    theme_classic()
```

Curvas de rarefacción

```
library(iNEXT)

abundancia_ec <- read.csv("./data/abundancia_ec.csv", row.names = 1)

inext_abundancia <- iNEXT(abundancia_ec, q = c(0, 1, 2), datatype =
"abundance")

# Gráficos
```

```
# Separando por ambiente
plot_ec <- ggiNEXT(inext_abundancia, type = 1, facet.var = “Assem-
blage”) +
    theme_classic(base_size = 10) +
    theme(legend.position = “bottom”)
plot_ec

# Separando por orden q
plot_orderq <- ggiNEXT(inext_abundancia, type = 1, facet.var = “Or-
der.q”) +
    theme_classic(base_size = 10) +
    theme(legend.position = “bottom”)
plot_orderq
```

NMDS (Escalado Multi-dimensional No Métrico)

```
library(BiodiversityR)
library(ggrepel)
library(ggforce)
library(concaveman)

sitios <- read.csv(“./datos/sitios.csv”, row.names = 1, header =
TRUE)
ambiente <- read.csv(“./datos/ambiente.csv”, row.names = 1, header =
TRUE)

set.seed(19950922)

resultado_nmds <- metaMDS(sitios, distance = “bray”, K = 2)     #(1)
resultado_nmds$stress   #(2)

dist_sitios <- vegdist(sitios)
anosim_sitios <- anosim(dist_sitios, ambiente$estado_conservacion,
distance = “bray”)
summary(anosim_sitios)

# Creamos un data.frame con los resultados
puntos_nmds <- as.data.frame(resultado_nmds$points)
puntos_nmds$CONSERVACION <- ambiente$estado_conservacion

# Agregamos siglas para nombres de los sitios y guardamos el valor
de stress
```

```
puntos_nmds$SITIO <- c(
    “EBA-T1”, “EBA-T2”, “EQN-T1”, “ESC-T1”, “ESC-T2”, “PIN-T1”,
“PCHI-T1”, “PCHI-T1”,
    “PCHII-T1”, “PCHII-T2”, “PANT-T1”, “PANT-T2”, “RCH-T1”,
“RCH-T2”, “VED-T1”
)
estres <- sprintf(“Stress = %s”, round(resultado_nmds$stress, 2))

# Graficamos
plot_nmds <- ggplot(puntos_nmds, aes(x = MDS1, y = MDS2)) +
    ggtitle(“NMDS”) +
    geom_point(aes(shape = CONSERVACION), size = 3) +
    scale_shape_manual(
        name = “”,
        breaks = c(“ECB”, “ECI”, “ECD”),
        labels = c(“Conservado”, “Intermedio”, “Degradado”),
        values = c(15, 16, 17)
    ) +
    geom_mark_hull(
        aes(group = CONSERVACION, linetype = CONSERVACION),
        concavity = 10,
        radius = 0,
        expand = 0,
        show.legend = FALSE
    ) +
    scale_linetype_manual(values = c(“solid”, “dashed”, “dotted”)) +
    annotate(“text”, x = +Inf, y = +Inf, label = estres, hjust = 1,
vjust = 1) +
    theme_classic()
plot_nmds

# Opcionalmente podemos añadir el nombre de los sitios
plot_nmds +
    geom_text_repel(
        aes(label = SITIO),
        box.padding = 0.5,
        size = 3.5,
        colour = “blue”,
    )
```

Análisis de regresión

```
datos <- read.csv("datos_ecologicos.csv")

plot(datos$variable_independiente, datos$variable_dependiente)

modelo <- lm(variable_dependiente ~ variable_independiente, data =
datos)
summary(modelo)

plot(datos$variable_independiente, datos$variable_dependiente)
abline(modelo, col = "red")
```

Análisis de correlación

```
datos <- read.csv("datos_ecologicos.csv")

matriz_cor <- cor(datos)

install.packages("corrplot")
library(corrplot)

corrplot(matriz_cor, method = "circle")
corrplot(matriz_cor, method = "color")

resultado_prueba <- cor.test(datos$variable1, datos$variable2)
resultado_prueba
```

Análisis de componentes principales (PCA)

```
datos <- read.csv("datos_ecologicos.csv")

datos_estandarizados <- scale(datos)

pca <- prcomp(datos_estandarizados, scale = TRUE)
summary(pca)
plot(pca)

datos_proyectados <- predict(pca, newdata = datos_estandarizados)
```

Análisis de Correspondencia Canónica (CCA)

```
datos <- read.csv("datos_ecologicos.csv")

variables_ambientales <- datos[, c("variable_ambiente1", "variable_
ambiente2", ...)]  # Selecciona las variables ambientales
variables_biologicas <- datos[, c("variable_biologica1", "variable_
biologica2", ...)]  # Selecciona las variables biológicas

library(vegan)

cca_resultado <- cca(variables_biologicas ~ variables_ambientales)
summary(cca_resultado)
plot(cca_resultado)

anova_resultado <- anova.cca(cca_resultado)
significativas <- anova_resultado$ANOVA[,"Pr(>F)"] < 0.05
variables_ambientales_significativas <- rownames(anova_resultado$ANO-
VA)[significativas]
```

Test de Mantel en R

```
matriz_biologica <- read.csv("matriz_biologica.csv", header = TRUE,
row.names = 1)
matriz_ambiental <- read.csv("matriz_ambiental.csv", header = TRUE,
row.names = 1)

library(vegan)

resultado_mantel <- mantel(matriz_biologica, matriz_ambiental, me-
thod = "pearson", permutations = 999)
resultado_mantel
```

Capítulo 9. Modelado de nicho ecológico

```
# Variables ambientales
mapas_worldclim <- "https://biogeo.ucdavis.edu/data/worldclim/v2.1/
base/wc2.1_2.5m_bio.zip"

tmp <- tempfile()
```

```
download.file(mapas_worldclim, destfile = tmp)
unzip(tmp, exdir = file.path("data/base/worldclim", bio_all))
unlink(tmp)

# Shapefile neotropico
tmp <- tempfile()
download.file(
    "https://figshare.com/ndownloader/articles/3569361/versions/2",
    destfile = tmp
)
unzip(tmp, exdir = "data/base/mascara_neotropico")
unlink(tmp)

for(i in c(8, 9, 18, 19)) {
    path <- file.path("./data/base/worldclim/bio_all", sprint-
f("wc2.1_2.5m_bio_%s.tif", i))
    file.remove(path)
}
dir.create("./data/base/worldclim/bio_temp")
dir.create("./data/base/worldclim/bio_prec/")
for (i in c(1:11)) {
    if (i == 8 || i == 9) {
        next
    }
    path <- file.path("./data/base/worldclim/bio_all", sprint-
f("wc2.1_2.5m_bio_%s.tif", i))
    file.copy(from = path, to = "./data/base/worldclim/bio_temp/")
}
for (i in c(12:17)) {
    path <- file.path("./data/base/worldclim/bio_all", sprint-
f("wc2.1_2.5m_bio_%s.tif", i))
    file.copy(from = path, to ="./data/base/worldclim/bio_prec/")
}

# ---- Librerias ----

library(raster)
library(kuenm) # https://github.com/marlonecobos/kuenm
library(ellipsenm) # https://github.com/marlonecobos/ellipsenm/

# ---- Limpieza ----
```

```
limpieza <- function(datos, region, thin_dist) {
    dentro <- data.frame(
        datos,
        inside = !is.na(raster::extract(region, datos[-1]))
     )
    dentro <- dentro[dentro$inside == TRUE, ][1:3]
    dentro <- dentro[!is.na(dentro$Lat) & !is.na(dentro$Long), ]
    unicos <- unique(dentro)
    thin_data(unicos, longitude = “Long”, latitude = “Lat”, thin_
distance = thin_dist)
}

# ---- Máscara ----

mascara <- function(ocurrencias, variables, salida, distancia) {
    variables <- raster::stack(
        list.files(file.path(variables, “Initial”), pattern = “pc_”,
full.names = TRUE)
    )
    buffer_oc <- buffer_area(
        ocurrencias,
        longitude = “Long”,
        latitude = “Lat”,
        buffer_distance = distancia
    )
    mascara <- raster::mask(raster::crop(variables, buffer_oc),
buffer_oc)
    dir.create(salida)
    raster::writeRaster(mascara, filename = file.path(salida, “pc”),
format = “ascii”, bylayer = TRUE)
}

# ---- Limpieza ocurrencias ----

# Carga de datos
oc_l_limpidus <- read.csv(“./data/base/ocurrencias_l_limpidus.csv”)
oc_c_fragrans <- read.csv(“data/base/ocurrencias_c_fragrans.csv”)
region_neo <- raster(“./data/bio/bio_all.tif”, lyrs = 1)
plot(region_neo, col = “grey70”, legend = FALSE)

# Limpieza
oc_l_limpidus_neot <- limpieza(oc_l_limpidus[1:3], region_neo, 10)
```

```
points(oc_l_limpidus_neot$Long, oc_l_limpidus_neot$Lat)
write.csv(oc_l_limpidus_neot, file = "data/ocurrencias/l_limpidus.
csv")

oc_c_fragrans_neot <- limpieza(oc_c_fragrans, region_neo, 10)
points(oc_c_fragrans_neot$Long, oc_c_fragrans_neot$Lat, col = "sea-
green")
write.csv(oc_c_fragrans_neot, file = "data/ocurrencias/c_fragrans_
hp.csv")

# ---- PCA ----

# Todas las variables
kuenm_rpca(
    variables = raster::stack("./data/bio/bio_all.tif"),
    var.scale = TRUE,
    write.result = TRUE,
    out.format = "ascii",
    out.dir = "./data/pcs_all",
    n.pcs = 3
)

# Temperatura
kuenm_rpca(
    variables = raster::stack("./data/bio/bio_temp.tif"),
    var.scale = TRUE,
    write.result = TRUE,
    out.format = "ascii",
    out.dir = "./data/pcs_temp",
    n.pcs = 3
)

# Precipitación
kuenm_rpca(
    variables = raster::stack("./data/bio/bio_prec.tif"),
    var.scale = TRUE,
    write.result = TRUE,
    out.format = "ascii",
    out.dir = "./data/ pcs_prec",
    n.pcs = 3
)
```

```
# ---- Máscara ----
pcs <- c("./data/pcs_all", "./data/pcs_prec", "./data/pcs_temp")
for (carpeta in pcs) {
    salida <- file.path(carpeta, "l_limpidus")
    mascara(oc_l_limpidus_neot, carpeta, salida, 50)
}
for (carpeta in pcs) {
    salida <- file.path(carpeta, "c_fragrans")
    mascara(oc_c_fragrans_neot, carpeta, salida, 50)
}

# Definición funcion para leer los pcs generados anteriormente
leer_pcs <- function(carpeta, tipo = NULL) {
    carpetas <- list.files(carpeta, full.names = TRUE)
    nombres <- list.files(carpeta)
    raster_list <- lapply(carpetas, function(x) {
        raster <- raster::stack(
        list.files(x, pattern = "pc_", full.names = TRUE)
    )
    names(raster) <- paste0(names(raster), tipo)
    raster
    })
    names(raster_list) <- nombres
    raster_list
}

#Definición función para calculo de elipses
superposicion <- function(ocs_hp, ocs_cb, vars_hp, vars_cb) {

    nicho_1 <- overlap_object(
        ocs_hp, species =  "species",
        longitude = "Long", latitude = "Lat",
        method = "covmat", level = 95,
        variables = vars_hp)

    nicho_2 <- overlap_object(
        ocs_cb, species =  "species",
        longitude = "Long", latitude = "Lat",
        method = "covmat", level = 95,
        variables = vars_cb)

    ellipsoid_overlap(
```

```
        nicho_1, nicho_2, overlap_type = “back_union”,
        significance_test = TRUE, replicates = 1000,
        confidence_limit = 0.05)
}


# ---- Elipsoides ----

# Variables
pcs_all <- leer_pcs(“data/pcs_all/”)
pcs_prec <- leer_pcs(“data/pcs_prec/”)
pcs_temp <- leer_pcs(“data/pcs_temp/”)

# Ocurrencias
oc_l_limpidus <- read.csv(“./data/ocurrencias_L_limpidus.csv”)
oc_c_fragrans <- read.csv(“./data/ocurrencias_c_fragrans.csv”)

# Superposición con prueba de significación
overlap_all <- superposicion(
    ocs_hp = oc_c_fragrans_neot,
    ocs_cb = oc_l_limpidus_neot,
    vars_hp = pcs_all$c_fragrans,
    vars_cb = pcs_all$l_limpidus
)
overlap_prec <- superposicion(
    ocs_hp = oc_c_fragrans_neot,
    ocs_cb = oc_l_limpidus_neot,
    vars_hp = pcs_prec$c_fragrans,
    vars_cb = pcs_prec$l_limpidus
)
overlap_temp <- superposicion(
    ocs_hp = oc_c_fragrans_neot,
    ocs_cb = oc_l_limpidus_neot,
    vars_hp = pcs_temp$c_fragrans,
    vars_cb = pcs_temp$l_limpidus
)

dir.create(“./output/overlaps”, recursive = TRUE)
saveRDS(overlap_all, “./output/overlaps/overlap_all.rds”)
saveRDS(overlap_prec, “./output/overlaps/overlap_prec.rds”)
saveRDS(overlap_temp, “./output/overlaps/overlap_temp.rds”)
```

```
# ---- Carga de datos ----

# Se agraga un sufijo a cada capa del raster (_all, _prec, _temp)
# dependiendo del conjunto de predictors climáticos.

pcs_all <- leer_pcs("./data/pcs_all", tipo = "_all")
pcs_prec <- leer_pcs("./data/pcs_prec", tipo = "_prec")
pcs_temp <- leer_pcs("./data/pcs_temp", tipo = "_temp")

# ---- Preparación para análisis ----

dir.create("./output/projection_model/splits", recursive = TRUE)
data_split <- split_data(
    oc_cerambicido, method = "random", longitude = "Long",
    latitude = "Lat", train_proportion = 0.75,
    save = TRUE, name = "./output/projection_model/splits/ocurren-
cias"
)

# Juntando las variables de cerambicido
# para luego armar los sets
vars_cerambicido <- raster::stack(
    pcs_all$l_limpidus, pcs_prec$l_limpidus, pcs_temp$l_limpidus
)

sets <- list(
    set_1 = c("pc_1_all", "pc_2_all", "pc_3_all"),
    set_2 = c("pc_1_prec", "pc_2_prec", "pc_3_prec"),
    set_3 = c("pc_1_temp", "pc_2_temp", "pc_3_temp")
)

sets_vars_cerambicido <- prepare_sets(vars_cerambicido, sets)

# ---- Calibracion ----

metodos <- c("covmat", "mve1")
calibracion <- ellipsoid_calibration(
    data_split, species = "species", longitude = "Long",
    latitude = "Lat", variables = sets_vars_cerambicido,
    methods = metodos, level = 99, selection_criteria = "S_OR_P",
    error = 5, iterations = 500, percentage = 50,
    output_directory = "output/projection_model/calibration_pcs"
```

```
)
res_calibracion <- read.csv(
    "./output/projection_model/calibration_pcs/selected_parameteri-
zations.csv"
)
res_calibracion

# ---- Modelo ----

modelo <- ellipsoid_model(
    data = oc_cerambicido, species = "species",
    longitude = "Long", latitude = "Lat",
    raster_layers = pcs_temp$Initial, method = "covmat", level = 99,
    replicates = 10, prediction = "suitability",
    return_numeric = TRUE, format = "GTiff", overwrite = FALSE,
    output_directory = "output/projection_model/model_pcs"
)
# Umbral del modelo
modelo_mean <- raster(
    "./output/projection_model/model_pcs/mean_suitability_calibra-
tion_l_limpidus.tif"
)

# Threshold del 5%
# Se tomó el valor más bajo recuperado y se utilizó para transformar
el modelo en binario.
puntos_cerambicido <- extract(modelo_mean, oc_cerambicido[-1])
modelo_mean_binario <- (modelo_mean > min(puntos_cerambicido))
plot(modelo_mean_binario)
writeRaster(
    modelo_mean_binario,
    "./output/projection_model/final_model_thrs_5.tif",
)
library(readr)
nuevo <- read_csv("data/ocurrencias/l_limpidus_nr.csv")
View(l_limpidus_nr)
points(oc_l_limpidus_neot$Long, oc_l_limpidus_neot$Lat, col =
"black", cex = 0.3)
points(nuevo$Long, nuevo$Lat, col = "red", cex = 0.3)

# Crear tabla de especies alien y native de l_limpidus
# ---- Ocurrencias con status ----
```

```
l_limpidus_nr <- read.csv(“./data/base/l_limpidus_nr.csv”)
l_limpidus_nr$status  <- “alien”
oc_l_limpidus_neot$status <- “native”
l_limpidus_status <- rbind(oc_l_limpidus_neot, l_limpidus_nr)
write.csv(l_limpidus_status, file = “./data/ocurrencias/l_limpidus_
status.csv”, row.names = FALSE)

# ---- Distancia Mahalanobis ----

cb_dist <- data.frame(
    l_limpidus_status,
    dist = extract(modelo_mean, l_limpidus_status[2:3])
)
cb_dist <- na.omit(cb_dist)
write.csv(cb_dist, file = “./output/dist_mahalanobis.csv”)

# ---- Librerias ----

library(rgl)
library(ggplot2)

# ---Gráficos ---

# Overlaps
plot_over <- function(overlap) {
    plot_overlap(
    overlap, niche_col = c( “brown1”, “deepskyblue”),
    data_col = c(“brown1”, “deepskyblue”), background = FALSE,
    change_labels = TRUE, xlab = “”, ylab = “”, zlab = “”, legend=
T)

}
plot_over(overlap_all) rgl.snapshot( “plots/overlap_all.png”, fmt =
“png”, top = TRUE )
plot_over(overlap_prec) rgl.snapshot( “plots/overlap_prec.png”, fmt
= “png”, top = TRUE )
plot_over(overlap_temp) rgl.snapshot( “plots/overlap_temp.png”, fmt
= “png”, top = TRUE )

# Histogramas
# La función plot_hist crea gráficos usando R base.
```

```
# La función gg_hist lo hace utilizando ggplot2

plot_hist <- function(overlap){
    hist(overlap@significance_results$union_random$Niche_1_vs_2$over-
lap,
        breaks = 5, main = “Overlap HP x Cb”, xlab = “Overlap”,
        xlim = c(0, 1), ylim = c(0, 650)
    )
    abline(v = quantile(overlap@significance_results$union_random$Ni-
che_1_vs_2$overlap, 0.05),
        col = “red”, lwd = 2, lty = 2
    )
    abline(v = overlap@union_overlap$overlap[1], col = “lightblue”,
lwd = 2)
    legend(“topright”, bty = “n”, legend = c(“Observed”, “5% CL”),
        col = c(“lightblue”, “red”), lty = c(1, 2), lwd = 2
    )
}
plot_hist(overlap_all)
plot_hist(overlap_prec)
plot_hist(overlap_temp)

gg_hist <- function(overlap, file) {
    datos <- data.frame(
        Overlap = overlap@significance_results$union_random$Niche_1_
vs_2$overlap
    )
    observado <- overlap@union_overlap$overlap
    cl <- quantile(overlap@significance_results$union_random$Niche_1_
vs_2$overlap, 0.05)
    plot <- ggplot(datos) +
        ylab(“Frequency”) +
        coord_cartesian(xlim = c(0, 1), ylim = c(0,650)) +
        geom_histogram(aes(x = Overlap), bins = nclass.Sturges(da-
tos[,1]), color = “black”, fill = “gray85”) +
        geom_vline(
            aes(xintercept = cl, linetype = “5% CL”, color = “5%
CL”)
        ) +
        geom_vline(
            aes(xintercept = observado, linetype = “Observed”, color
= “Observed”)
```

```
        ) +
        scale_color_manual(
            name = "",
            values = c("red", "blue")
        ) +
        scale_linetype_manual(
            name = "",
            values = c("dashed", "solid")
        ) +
        theme_classic()
    ggsave(file, plot = plot, width = 7, height = 7)
    plot
}
gg_hist(overlap_all, "plots/overlap_all.svg")
gg_hist(overlap_prec, "plots/overlap_prec.svg")
gg_hist(overlap_temp, "plots/overlap_temp.svg")

# Mahalanobis

hist(cb_dist$dist)
abline(v = cb_dist$dist[cb_dist$status == "alien"], col = "red", lwd
= 1)

maha_plot <- ggplot(cb_dist, aes(x = dist)) +
    ylab("Frequency") +
    xlab("Distance") +
    geom_histogram(bins = nclass.Sturges(cb_dist $dist), color =
"black", fill = "gray") +
    geom_vline(data = subset(cb_dist, status == "alien"), aes(xin-
tercept = dist), color = "red") +
    theme_classic()
maha_plot
ggsave("./plots/mahalanobis_hist.svg", plot = maha_plot, height = 7,
width = 7)
```

Glosario

Aleatorización. Asignación aleatoria de tratamientos a unidades experimentales para reducir la variabilidad no controlada.
Diseño experimental. Proceso de planificación y estructuración de un experimento para identificar relaciones causales entre variables.
Escalas de medición. Nominal, ordinal, de intervalo y de razón. || Determinan el tipo de análisis estadístico aplicable.
Mini-Winkler. Método de recolección de hormigas en la hojarasca del suelo.
Muestra. Conjunto de observaciones recopiladas para el estudio.
Muestreo aleatorio. Selección de muestras de manera aleatoria para representar la población.
Muestreo balanceado. Distribución homogénea de muestras entre categorías o tratamientos.
Niveles de una variable. Posibles valores que puede asumir una variable formando un continuo ordenado.
Operacionalización de variables. Proceso de definir cómo medir una variable de manera precisa y observable.
Pitfall. Método de recolección de hormigas mediante trampas enterradas en el suelo.
Protocolo de muestreo. Procedimiento estandarizado para la recolección de datos en el campo.
Replicación. Inclusión de múltiples unidades experimentales para obtener resultados confiables.
Tamaño de la muestra. Número de observaciones recopiladas para el estudio. || Determinado por objetivos, variabilidad de los datos, diseño del estudio y recursos disponibles.
Toma de datos. Proceso de recopilación de información relevante para el estudio.
Transecto. Método de muestreo que involucra el recorrido de una línea recta a través de un área de estudio.
Variables. Elementos fundamentales estudiados y analizados para comprender fenómenos y procesos dentro de un sistema.
Variable continua. Pueden tomar cualquier valor dentro de un rango específico. || Infinitamente divisibles.
Variable cualitativa. Representan características no numéricas o cualidades de los elementos de estudio.
Variable cuantitativa. Representan cantidades numéricas o medidas de los elementos de estudio.
Variable dependiente. Variable observada y medida en respuesta a los cambios en la variable independiente.
Variable discreta. Valores aislados y contables. || Generalmente enteros.
Variable independiente. Factor manipulado o controlado por el investigador, con efecto sobre la variable dependiente.

Referencias bibliográficas

BADII, M., Castillo Rodríguez, M., Wong, A. y Villalpando, P. (2007). Diseños experimentales e investigación científica. *Innovaciones de Negocios, 4*(8), 283-330.

BOLKER, B. M. (2008). *Ecological models and data in R*. Princeton University Press.

BORCARD, D., Gillet, F. y Legendre, P. (2011). *Numerical Ecology with R*. Springer.

BOUZA, C. N. y Covarrubias, D. (2005). Estimación del índice de Diversidad de Simpson en m sitios de muestreo. *Revista Investigación Operacional, 26*(2), 187-197.

CALDERÓN, J. y Moreno, C. (2019). Diversidad beta como disimilitud: su partición en componentes de recambio y diferencias en riqueza. En *La biodiversidad en un mundo cambiante: Fundamentos teóricos y metodológicos para su estudio* (pp. 203-222). Universidad Autónoma del Estado de Hidalgo/Libermex.

DALGAARD, P. (2008). 69 Statistics and Computing. En *Introductory Statistics R*. Springer.

ELTON, C. S. (1927). *Animal ecology, by Charles Elton; with an introduction by Julian S. Huxley*. Macmillan Co.

GRINNELL, J. (1927). Geography and Evolution. *Ecology*, *5*(3), 225-229.

HUTCHINSON, G. E. (1957). Concluding Remarks. *Cold Spring Harbor Symposia on Quantitative Biology*, 22(0), 415-427.

LEARDI, R. (2013). Experimental Design. *Data Handling in Science and Technology*, *28*, 9-53.

MORENO, C. E. (2000). *Métodos para medir la biodiversidad* (Vol. 1). Manuales y tesis SEA. Cyted, Orcyt, SEA.

MORRONE, J.J. (2014). Biogeographical regionalisation of the Neotropical region. *Zootaxa*, *3782*(1), 1-110.

PÉREZ-HERNÁNDEZ, C. X. (2019). Distintividad taxonómica: Evaluación de la diversidad en la estructura taxonómica en los ensambles. En *La biodiversidad en un mundo cambiante: Fundamentos teóricos y metodológicos para su estudio* (pp. 285-306). Universidad Autónoma del Estado de Hidalgo/Libermex.

SOBERÓN, J. y Townsend Peterson, A. (2005). Interpretation of models of fundamental ecological niches and species' distributional areas. *Biodiversity Informatics*, 2, 1-10.

SOBERÓN, J., Osorio-Olvera, L. y Townsend Peterson, A. (2017). Diferencias conceptuales entre modelación de nichos y modelación de áreas de distribución. *Revista Mexicana de Biodiversidad*, *88*(2), 437-441.

STEINBERG, D. M. y Hunter, W. G. (1984). Experimental design: Review and comment. *Technometrics, 26*(2), 71-97.

SWENSON, N. G. (2014). *Functional and phylogenetic ecology in R*. Springer.

VALDIVIESO SERRANO, L. (1991). Escalas de Medición. *Pro Mathematica, 5*(9-10), 53-67.

VALLE, N. y Simões, M. (2022). New Distributional Records and Characterization of the Climatic Niche of *Lepturges* (*Lepturges*) *limpidus* Bates, 1872 (Coleoptera, Cerambycidae): Sink or Source Population? *Insects*, *13*(11), 1069.

VILLARREAL, H., Álvarez, M., Córdoba, S., Escobar, F., Fagua, G., Gast, F., Mendoza, H., Ospina, M. y Umaña, A. M. (2004). *Manual de métodos para el desarrollo de inventarios de biodiversidad.* Programa Inventarios de Biodiversidad, Instituto de Investigación de Recursos Biológicos Alexander von Humboldt.

WILSON, E. O. (1988). *Biodiversity*. National Academy of Sciences.

Recursos en línea

Institute for Statistics and Mathematics of WU (s.f.). *The Comprehensive R Archive Network [CRAN]*. https://cran.r-project.org/

The R Foundation (s.f.). *The R Journal*. https://journal.r-project.org/

Sobre autores y autora

Darío Daniel Larrea. Profesor en Biología, licenciado en Ciencias Biológicas y doctor en Biología (FaCENA-UNNE). Actualmente, se desempeña como docente en Zoología Agrícola, Biología de los Invertebrados (FaCENA-UNNE) y da cursos de posgrado. Su línea de investigación comprende el estudio de la diversidad, biogeografía y sistemática de hormigas (*Formicidae*), con énfasis en especies de ambientes naturales y antropizados del NEA. Lidera proyectos regionales en etología, biogeografía y biodiversidad de insectos, siendo autor prolífico de artículos, libros, capítulos de libros y disertante en reuniones científicas nacionales e internacionales. Desde su rol docente y como investigador, asesora a tesistas de licenciatura, coordina pasantías y dirige becarios de grado y posgrado. Al presente, se desempeña como editor de la Revista de la Sociedad Entomológica Argentina (RSEA).

Lucas Javier Mina. Técnico electromecánico y estudiante avanzado de la Licenciatura en Ciencias Biológicas (FaCENA-UNNE). Actualmente, se desempeña como pasante en el Laboratorio de Biología de los Artrópodos y en el Laboratorio de Zoología Agrícola, donde participa activamente en proyectos de investigación vinculados a la diversidad, biogeografía y ecología de artrópodos. Su formación técnica se complementa con el desarrollo y optimización de herramientas informáticas aplicadas al análisis de datos biológicos, incluyendo el mantenimiento y la mejora de código en lenguaje R y el diseño de bases de datos para estudios de biodiversidad. Además, colabora con investigadores de distintas unidades académicas brindando soporte en el procesamiento y análisis estadístico de datos. También ha participado activamente en congresos y reuniones científicas de carácter nacional e internacional.

Florencia Monti Areco. Licenciada en Ciencias Biológicas y docente investigadora de la Facultad de Ciencias Exactas, Naturales y Agrimensura de la UNNE. Desarrolla su labor académica en las cátedras de Biología de los Invertebrados, Protozoología y Zoología Agrícola, y su investigación se especializa en la ecología y biología del protozooplancton en ambientes acuáticos continentales del noreste argentino. Ha participado activamente

en proyectos científicos, reuniones especializadas y publicaciones académicas, incluyendo artículos científicos, libros y capítulos de libro. Sus investigaciones han contribuido al estudio de taxones poco explorados en ecosistemas dulceacuícolas –como cnidarios y lofoforados–, así como a la caracterización de la biodiversidad regional, principalmente en humedales de la Reserva Natural Iberá.

Néstor Gerardo Valle. Licenciado en Ciencias Biológicas y profesor en Biología (FaCENA-UNNE). Actualmente, cursa el Doctorado en Ciencias Biológicas en la misma casa de estudios, en el marco de una beca cofinanciada por el Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) y la UNNE. Su línea de investigación se orienta al análisis de los ensambles de *Cerambycidae* (Coleoptera) en formaciones boscosas del Chaco Oriental Húmedo. Es autor de artículos en revistas científicas, capítulos de libros, y ha participado activamente en congresos y reuniones científicas de carácter nacional e internacional.

Matías Ignacio Dufek. Licenciado en Ciencias Biológicas (FaCENA-UNNE), profesor universitario en Ciencias Biológicas (HUM-UNNE) y doctor en Biología (FaCENA-UNNE). Investigador asistente del CONICET (CECOAL-CONICET-UNNE), docente en las asignaturas Biología de los Artrópodos y Entomología (FaCENA-UNNE), y de cursos de posgrado y actualización. Su línea de investigación comprende el estudio del efecto de la antropización sobre la estructura comunitaria y el rol ecológico de moscas sarcosaprófagas (Diptera: *Calyptratae*). Es autor de artículos científicos, capítulos de libros y numerosas presentaciones en congresos. Lleva a cabo tareas de divulgación científica y formación de recursos humanos. Actualmente, se desempeña como editor del Boletín de la Sociedad Entomológica Argentina (BSEA) y de la Revista de la Sociedad Entomológica Argentina (RSEA).

Estadística para estudios ecológicos.
Un enfoque práctico utilizando R se compuso y diagramó en Eudene, en el mes de julio de 2025.

Esta obra ofrece una guía teórica y práctica para el análisis estadístico en estudios ecológicos mediante el uso del entorno R. A través de ejemplos reales, los autores presentan un recorrido integral por las herramientas más relevantes: desde el diseño experimental y la organización de datos hasta pruebas estadísticas, análisis de biodiversidad y modelado de nicho ecológico. La inclusión de scripts comentados y recursos en línea convierte a este libro en una herramienta valiosa para estudiantes, investigadores y profesionales de las ciencias biológicas y áreas afines.

Producto de años de experiencia y publicaciones científicas, este manual promueve una práctica analítica sólida y accesible, alineada con los desafíos actuales de la investigación ecológica y la necesidad urgente de comprender los sistemas naturales en un contexto de cambio global.

APUNTES DE CÁTEDRA ● CIENCIAS EXACTAS Y NATURALES Y AGRIMENSURA

EUDENE

Universidad Nacional del Nordeste

ISBN 978-950-656-267-0