TRABAJO DE FIN DE MÁSTER

Máster Interuniversitario en Estadística e Investigación Operativa UPC-UB

Título: Identificación de subpoblaciones en relación a las trayectorias de vuelo del Larus Audouinii.

Autor: Laura Julià Melis

Director: Josep Lluís Carrasco Jordan

Departamento: Fundamentos Clínicos

Universidad: Universidad de Barcelona

Convocatoria: Enero 2022

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat de Matemàtiques i Estadística

UNIVERSITAT DE BARCELONA

Máster en Estadística e Investigación Operativa
Facultad de Matemáticas y Estadística - Universidad Politécnica de Cataluña

Identificación de subpoblaciones en relación a las trayectorias de vuelo del Larus Audouinii.

Autora: Laura Julià Melis
Director: Josep Lluís Carrasco

Enero, 2022
Barcelona

Resumen

La gaviota de Audouin (*Larus audouinii*) es un ave marina oportunista que se caracteriza por tener una gran capacidad de adaptación que le permite aprovechar el alimento que la actividad pesquera les brinda por ser predecible y abundante. Sin embargo, la especie se encuentra en estado vulnerable debido a una rápida disminución de su población. Analizar las trayectorias de vuelo de las gaviotas permite detectar patrones de comportamiento y desarrollar actuaciones para su protección y conservación. Estudios anteriores revelaron que la población de la colonia del delta del Ebro ha sabido adaptar su comportamiento a la presencia/ausencia de actividad pesquera. No obstante, es posible que esta adaptación haya ocurrido de forma desigual en los individuos de la colonia.

En este trabajo se analizan los datos de las trayectorias de 36 gaviotas de Audouin de la colonia del delta del Ebro, 38090 localizaciones espaciotemporales recogidas con dispositivos GPS, con el objetivo de evaluar la posible existencia de subpoblaciones de gaviotas.

Primero, se desarrolla e implementa un algoritmo eficiente para el cálculo de la distancia de Hausdorff mediante el software `R`, logrando reducir notablemente el tiempo computacional de la operación. Luego, se lleva a cabo un análisis clúster segregando las trayectorias por los diferentes momentos de la actividad pesquera (presencia diurna, presencia nocturna y ausencia de pesca) con los que se consigue identificar gaviotas con comportamientos extremos y se confirma la necesidad de la segregación.

Palabras clave: *Larus Audouinii*, análisis de datos de trayectorias, distancia de Hausdorff, *software* `R`, análisis clúster.

Clasificación AMS: 62H11(Datos direccionales; estadísticas espaciales), 62H30 (Clasificación y discriminación; análisis de conglomerados), 92-08 (Métodos computacionales para problemas relacionados a la biología), 62-07 (Análisis de datos), 62P10 (Aplicaciones de la estadística a la biología y las ciencias médicas).

Abstract

The Audouin's Gull (*Larus audouinii*) is an opportunistic seabird characterized by a great adaptability that allows it to take advantage of the food that fishing activity provides because it is predictable and abundant. However, the species is in a vulnerable state due to a rapid decline in its population. Analyzing the patterns of flight trajectories of gulls is key to understand their behavior and develop an action plan for their protection and conservation. Previous studies revealed that the population of the colony near the Ebro Delta has been able to adapt its behavior to the presence/absence of fishing activity. But this adaptation may have occurred unevenly among individuals in the colony.

In this paper, the trajectories of 36 Audouin's gulls from the Ebro Delta colony, 38090 space-time locations collected with GPS devices, are analyzed with the aim of evaluating the possible existence of subpopulations of gulls.

First, an efficient algorithm for the computation of the Hausdorff distance is developed and implemented using the `R` software, managing to significantly reduce the computational time of the operation. Then, a cluster analysis is carried out segregating the trajectories by the different moments of the fishing activity (diurnal presence, nocturnal presence and absence) with which it is possible to identify gulls with extreme behaviors and the need for segregation is confirmed.

Key words: *Larus Audouinii*, trajectory data analysis, Hausdorff distance, `R` software, cluster analysis.

AMS classification: 62H11 (Directional data, spatial statistics), 62H30 (Classification and discrimination; cluster analysis), 92-08 (Computational methods for problems pertaining to biology), 62-07 (Data Analysis), 62P10 (Applications of statistics to biology and medical sciences).

Índice general

Índice de figuras

Capítulo 1

Introducción

La gaviota de Audouin (*Larus audouinii*) es una especie endémica de la cuenca del Mediterráneo de la que España alberga aproximadamente un 90 % de los ejemplares a nivel mundial y, en particular, en el delta del Ebro se encuentra la principal colonia reproductora de la especie (SEO/BirdLife, 2008). La gaviota se encuentra en La Lista Roja de Especies Amenazadas de la Unión Internacional para la Conservación de la Naturaleza (IUCN, 2021), incluida en la categoría de "Vulnerable" ya que desde el año 2010 la especie está experimentando una rápida reducción de la población posiblemente debida a una reducción de los descartes de los barcos pesqueros y a un incremento en la depredación de las colonias (BirdLife International, 2021).

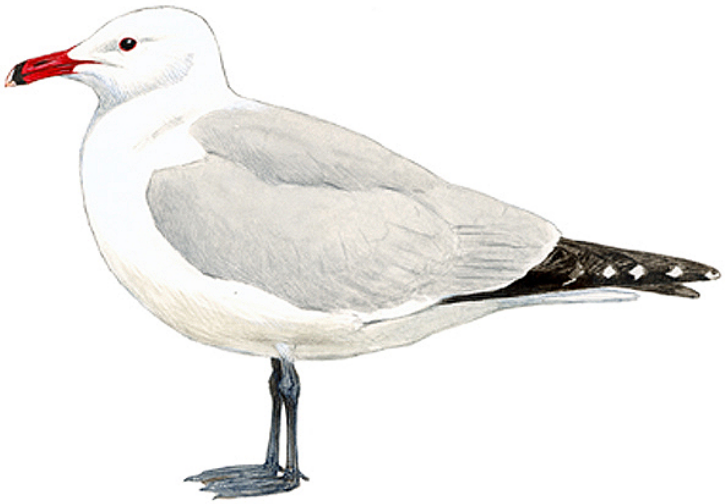

Figura 1.1: Gaviota de Audouin. Imagen extraída de SEO/BirdLife, 2008

Se trata de un ave marina oportunista que se caracteriza por tener una alta capacidad de adaptación, por lo que su patrón de conducta puede variar rápidamente con el fin de apro-

vechar de forma eficiente los recursos alimentarios disponibles (Ouled-Cheikh y col., 2020). En este contexto, la actividad humana y, especialmente la actividad pesquera, tiene implicaciones fundamentales en los procesos ecológicos de las aves oportunistas (depredación, competencia y transferencia de nutrientes entre los niveles tróficos). Como consecuencia, estudiar el comportamiento del vuelo de la gaviota de Audouin en relación a la actividad pesquera del lugar en que habita es crucial para poder entender cómo responde a las presiones humanas y así poder tomar medidas de protección y conservación de la especie.

Durante el período de incubación del año 2011 se rastrearon mediante GPS las trayectorias de un grupo de gaviotas de Audouin de la colonia de *Punta de la Banya*, en el Parque Natural del Delta del Ebro (Ouled-Cheikh y col., 2020). Tras caracterizar las trayectorias de los vuelos a partir de varios índices y analizar los datos en función de la utilización del hábitat (Cortejana Retamozo, 2020; Ouled-Cheikh y col., 2021), se descubrió que la actividad humana tenía un efecto en el comportamiento de las gaviotas.

En general, la especie pasaba más tiempo sobre el mar cuando había barcos pesqueros (realizando trayectorias más rápidas y rectas) mientras que en ausencia de actividad pesquera, la especie llevaba a cabo viajes en búsqueda de crustáceos y peces de agua dulce en los arrozales, por lo que los vuelos eran más largos y con más cambios de dirección.

Sin embargo, anteriormente se había demostrado (Ouled-Cheikh y col., 2020) que los individuos dentro de una misma población también pueden explotar de manera diferente los alimentos aportados por los humanos y que, factores como la edad, el sexo o el estado reproductivo del sujeto son factores que pueden afectar la especialización individual de las gaviotas (Phillips y col., 2017).

Por ello, el objetivo principal de este trabajo es evaluar la existencia de subpoblaciones en relación a las trayectorias de gaviotas de Audouin, de forma que se puedan identificar grupos de gaviotas con trayectorias similares y diferenciadas de otras gaviotas. Más concretamente, se desea aplicar un análisis clúster segregado por los diferentes momentos de la actividad pesquera, en el que se esperaría encontrar posiblemente dos grupos o subpoblaciones: una con las gaviotas adaptadas a la actividad pesquera de cada momento y la otra, con las que no se han especializado y siguen alimentándose en los arrozales.

La aplicación del análisis clúster requiere la utilización de una matriz de distancia apropiada según la naturaleza de los datos de estudio y, por lo que respecta a los datos en movimiento, existen varias medidas que permiten cuantificar la similitud o diferencia entre objetos (Magdy y col., 2015).

En el caso particular del análisis de trayectorias, la distancia de Hausdorff mide la similitud entre las formas geométricas de dos trayectorias, lo cual nos permitirá comparar el patrón del vuelo de las gaviotas con lo que respecta al comportamiento de alimentación en presencia y ausencia de actividad pesquera. No obstante, para identificar clústeres de gaviotas será necesario tener en cuenta la estructura jerárquica de los datos (en la que cada gaviota ha realizado varias trayectorias) y por lo tanto, las distancias entre trayectorias se deberán de agrupar por gaviota de una manera adecuada.

El cálculo de la distancia de Hausdorff es computacionalmente complejo (Taha & Hanbury, 2015) por lo que existen diferentes propuestas para su cálculo (aproximado y exacto). Consecuentemente, un segundo objetivo de este trabajo es la implementación de un algoritmo eficiente para el cálculo exacto de la distancia de Hausdorff mediante el software `R` (R Core Team, 2020).

1.1. Estructura del documento

El documento se estructura de la siguiente manera. En el Capítulo 2, se explicarán los métodos estadísticos utilizados: se describirá extensamente la técnica del análisis clúster (Sección 2.1) y la distancia de Hausdorff (Sección 2.2). En el Capítulo 3, se mostrarán los resultados obtenidos: se describirá la base de datos (Sección 3.1), se calculará la distancia de Hausdorff, comparando el tiempo computacional de varios algoritmos (Sección 3.2) y se incluirá el análisis clúster de las gaviotas de Audouin (Sección 3.3). Finalmente, en el Capítulo 4, se expondrán las conclusiones alcanzadas y sugerirán posibles mejoras y extensiones del análisis.

Capítulo 2

Metodología

En este capítulo se describirá la técnica del análisis clúster y, en particular, el *Clustering* Jerárquico Aglomerativo: el procedimiento a seguir, los criterios de enlace inter-clúster, la representación mediante el dendrograma, el número de clústeres óptimo, la representación de la solución en un espacio de dos dimensiones y cómo comparar los resultados de diversas agrupaciones. Una vez comprendida la necesidad de tener una matriz de distancias, en la Sección 2.2 se explicará la distancia de Hausdorff: la razón detrás de su utilización, cuáles son sus propiedades y limitaciones, y cómo calcularla de forma eficiente.

En general, se seguirán la notación y estructura utilizadas en el libro de Everitt y Hothorn (2011) sobre el análisis clúster, así como también los apuntes de la asignatura de Análisis Multivariante (Graffelman y col., 2021). En cuanto a la distancia de Hausdorff, se utilizarán los conceptos desarrollados en el estudio de Min y col. (2007).

2.1. Análisis Clúster

El análisis clúster o análisis de conglomerados se enmarca en el conjunto de técnicas de Análisis de Datos Multivariantes y, en particular, en los modelos de Aprendizaje no Supervisado, en los que se analizan datos no etiquetados (i.e. carentes de una variable respuesta que indique a qué grupo pertenece realmente cada observación) con la finalidad de poner de manifiesto la estructura inherente a la base de datos.

2.1.1. Definición y métodos

El término *clustering* hace referencia a un amplio abanico de algoritmos que tienen el objetivo común de hallar una partición en la que los objetos de interés (p. ej. personas, trayectorias o imágenes) se agrupen en un pequeño número de clústeres homogéneos y separados de otros

grupos. En algunas ocasiones el objetivo puede ser detectar valores extremos: objetos que no pertenecen a ningún clúster o que pertenecen a clústeres de muy baja cardinalidad (ver Figura 2.1).

En general pueden diferenciarse dos tipos de métodos de análisis de conglomerados, el *clustering* jerárquico (aglomerativo o divisivo) y el no jerárquico (algoritmos *K-means*, *PAM*, *CLARA*). La característica principal que los diferencia es el requisito u omisión de especificar el número de clústeres que se van a crear, respectivamente. Adicionalmente, existen métodos que modifican o combinan los dos anteriores como el *fuzzy clustering* (en el que los elementos pueden pertenecer a más de un clúster) y el *model based clustering*, entre otros.

En este documento únicamente se describirá el *clustering* jerárquico aglomerativo (Subsección 2.1.2), por ser el método utilizado en el análisis de las trayectorias de las gaviotas de Audouin. Para más información sobre los métodos restantes ver Gordon (1999).

La propiedad que comparten todos los métodos de *clustering* es la necesidad de medir la similitud entre las observaciones para poder llevar a cabo las agrupaciones. Esto se hace a partir de una matriz de distancias entre observaciones, cuyos valores representan la similitud o diferencia entre cada pareja de observaciones. Los algoritmos van a minimizar las distancias entre los objetos de un mismo clúster y maximizar las distancias entre los objetos de clústeres diferentes de modo que se obtendrán grupos de objetos semejantes entre sí y diferentes al resto.

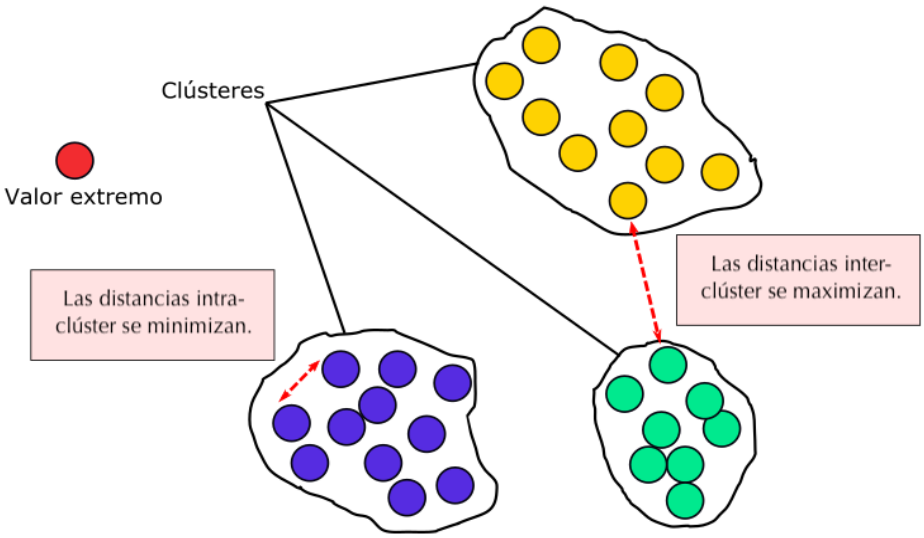

Figura 2.1: Cada círculo corresponde a un elemento y los colores son el resultado del análisis clúster, que en este caso ha identificado 3 grupos y un *outlier*.

Existen muchas medidas para caracterizar las analogías y diferencias de los objetos de la base de datos. La elección del tipo de distancia inter-individuos a utilizar va a depender de la situación experimental de cada caso.

2.1.2. Método Jerárquico Aglomerativo

Este tipo de análisis clúster produce clasificaciones jerárquicas de los datos, las cuales se van produciendo iterativamente empezando con un único grupo que incluye a todos los objetos hasta terminar con n clústeres, cada uno con un solo individuo (procedimiento divisivo) o a la inversa (procedimiento aglomerativo).

Así, dada una base de datos con n observaciones y, una vez calculada la matriz de distancias $n \times n$, el algoritmo básico para el *clustering* jerárquico aglomerativo es:

EMPEZAR

Definir n clústeres iniciales, cada uno con un único elemento.

REPETIR n-1 veces

Fusionar los dos clústeres más cercanos/similares.

Actualizar matriz de distancias.

HASTA que solo quede un clúster que incluya a todos los individuos.

TERMINAR

Nótese que la unión de dos elementos en un mismo clúster es irreversible y que no es necesario pre-especificar el número de clústeres a identificar.

Por otra parte, es importante comentar la subjetividad del método en (i) la elección de la matriz de distancias inicial [1] y (ii) en la forma de cuantificar la cercanía/similitud de los clústeres. Diferentes definiciones de las distancias inter-individuos e inter-clústeres conducirán a diferentes algoritmos y, consecuentemente, a resultados distintos.

2.1.2.1. Criterio de enlace

Para llevar a cabo el proceso iterativo de fusionar los individuos o grupos de individuos formados previamente, tal y como se indica en el algoritmo anterior, es necesario determinar cómo se cuantifica la similitud o cercanía entre dos clústeres. Existen múltiples definiciones de distancia inter-clústeres, o criterios de enlace.

[1] Esta fuente de subjetividad está presente en todos los métodos de análisis clúster.

Sea D_{AB} la distancia inter-clúster entre los clústeres $A = \{a_1, a_2, \ldots, a_{|A|}\}$ y $B = \{b_1, b_2, \ldots, b_{|B|}\}$ y sea $d(a, b)$ la distancia inter-individuos entre los objetos a y b. Los criterios más habituales son los siguientes:

- Distancia mínima o *single-link*:

$$D_{AB} = \text{mín}\{d(a, b) : a \in A, b \in B\}$$

- Distancia máxima o *complete-link*:

$$D_{AB} = \text{máx}\{d(a, b) : a \in A, b \in B\}$$

- Distancia promedio o UPGMA (*Unweighted Pair Group Method with Arithmetic mean*):

$$D_{AB} = \frac{1}{|A| \cdot |B|} \sum_{a \in A} \sum_{b \in B} d(a, b)$$

- Distancia de Ward (minimiza la suma de cuadrados intra-clúster):

$$D_{AB} = \sum_{x \in A \cup B} \left(x - c_{A \cup B}\right)^2 - \left(\sum_{a \in A} (a - c_A)^2 + \sum_{b \in B} (b - c_B)^2 \right)$$

  donde c_A, c_B y $c_{A \cup B}$ son los centros de los clústeres A, B y $A \cup B$, respectivamente.

El criterio UPGMA y el criterio de Ward suelen ser los más utilizados en la práctica debido a que son menos susceptibles al ruido y a valores extremos, por lo que generan clústeres más compensados. No obstante, la elección del enlace inter-clúster dependerá del estudio en cuestión.

2.1.2.2. Dendrograma y verificación

Este método de análisis de conglomerados produce un conjunto de clústeres estructurados como un árbol jerárquico que se puede ser representado mediante un diagrama bidimensional denominado dendrograma, el cual ilustra las uniones realizadas en cada etapa del análisis.

En la base del dendrograma se encuentran las observaciones a agrupar mientras que el eje vertical del gráfico representa la altura en la que estas agrupaciones van ocurriendo, es decir, el valor del criterio de enlace o **distancia cofenética**.

Veamos la Figura 2.2 a modo ilustrativo. Los puntos $\{1, 2, 3, 4\}$ son las observaciones a agrupar y los elementos 1 y 2 son los primeros en unirse con una distancia cofenética de magnitud A. Los elementos 3 y 4 se unen a una distancia B y, para terminar, los clústeres

$\{1, 2\}$ y $\{3, 4\}$ se fusionan en una distancia C.

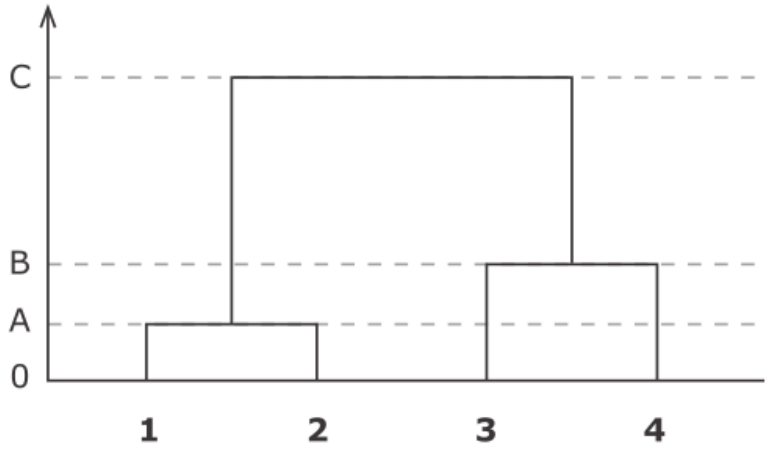

Figura 2.2: Dendrograma.

A continuación de la obtención del dendrograma, es importante verificar qué tan bien están representadas las distancias originales entre las observaciones. Una medida de la bondad del ajuste realizado es el Coeficiente de Correlación Cofenética (CCC), el cual evalúa la correlación lineal entre la distancia inter-individuos y la distancia cofenética del dendrograma.

Sean d_{ij} y c_{ij} la distancia original y la distancia cofenética entre los individuos i y j, respectivamente, y sean $\bar{d}$ y $\bar{c}$ sus respectivas medias aritméticas, entonces

$$\text{CCC} = \frac{\sum_{i<j} \left[\left(d_{ij} - \bar{d} \right) \cdot \left(c_{ij} - \bar{c} \right) \right]}{\sqrt{\sum_{i<j} \left(d_{ij} - \bar{d} \right)^2} \cdot \sqrt{\sum_{i<j} \left(c_{ij} - \bar{c} \right)^2}}$$

Por lo que respecta a su interpretación, un valor exactamente igual a 1 implica que la representación de las distancias originales entre los objetos en el dendrograma es perfecta. Esta medida puede resultar útil también para la elección del criterio de enlace.

2.1.2.3. Número óptimo de clústeres

Una vez elaborado y verificado el dendrograma, es fundamental considerar cómo seleccionar particiones específicas de los datos, o dicho de otra manera, escoger una solución con un número determinado de clústeres. Esto se lleva a cabo realizando un "corte" en el dendrograma que separe los individuos en diferentes "ramas", los clústeres. Por lo tanto, ahora la cuestión a resolver es, ¿a qué altura se debe realizar este corte?

Un enfoque habitual consiste en identificar de forma visual el intervalo de altura más amplio para el que cualquier corte dé como resultado el mismo número de clústeres. Volviendo a la Figura 2.2, esto ocurre entre las distancias cofenéticas B y C, y por lo tanto el número óptimo de clústeres es 2.

Una estrategia más formal es utilizar el método del *average silhouette*, el cual trata de maximizar la media del **coeficiente silhouette** o índice silueta (Rousseeuw, 1987). El coeficiente silhouette mide la bondad de la asignación de un objeto a un clúster. Así, se compara la distancia del objeto i con el clúster asignado en relación a la distancia al segundo clúster más cercano.

Sea A el clúster al que pertenece el objeto i, y sea C cualquier otro clúster diferente a A, entonces

$$s(i) = \frac{b(i) - a(i)}{\text{máx}\{a(i), b(i)\}}, \tag{2.1}$$

donde

- $a(i)$ es la distancia media intra-clúster del objeto i con respecto al resto de los objetos del clúster A

  $$a(i) = \frac{1}{|A| - 1} \sum_{j \in \{A \setminus i\}} d_{ij},$$

  donde d_{ij} es la distancia inter-individuo entre los objetos i y j.

- $b(i)$ es la distancia del clúster más cercano al objeto i, el clúster vecino (i.e. la segunda mejor opción para clasificar a i)

  $$b(i) = \min_{C \neq A} \{d(i, C)\},$$

  donde $d(i, C) = \frac{1}{|C|} \sum_{j \in C} d_{ij}$ es la distancia media de i con todos los objetos del clúster C.

A partir de la ecuación 2.1 es posible observar que $-1 \leq s(i) \leq 1$, para todo objeto i. Valores cercanos a 1 implican que el objeto está bien clasificado ya que la distancia intra-clúster, $a(i)$, es inferior a la distancia inter-clúster mínima, $b(i)$. Utilizando el mismo razonamiento, valores cercanos a -1 implican que sería más adecuado asignar el objeto i al clúster vecino. Por último, $s(i) = 0$ ocurrirá cuando $a(i)$ y $b(i)$ sean aproximadamente iguales y, consecuentemente, el objeto i está igual de distante a los dos clústeres.

Adicionalmente, Rousseeuw (1987) comentó en su trabajo el conflicto de definir $a(i)$ cuando el clúster únicamente contiene un objeto, por lo que definió $s(i) = 0$ cuando $|A| = 1$.

Volviendo al método *average silhouette*, se trata de computar el promedio de $s(i)$ para todos los objetos i de la base de datos, $\bar{s}(k) = \frac{1}{n}\sum_{i=1}^{n} s(i)$, donde k es el número de clústeres en los que se ha agrupado la base de datos. Así, el valor máximo de $\bar{s}(k)$ nos dará el número óptimo, k, de clústeres a determinar.

Habiendo dicho esto, es importante recalcar que determinar el número óptimo de clústeres es un proceso generalmente subjetivo que va a depender de las distancias inter-individuos y inter-clústeres empleadas así como también de información previa al análisis que pueda sugerir una idea de la estructura de grupos que los datos puedan tener.

2.1.3. Representación 2D de la solución

Luego de haber decidido cuál es el número óptimo de clústeres a determinar en la base de datos y haber realizado el corte en el dendrograma, resulta interesante representar los resultados del análisis clúster mostrando los objetos en un espacio bidimensional.

Al tratarse de un contexto multivariante, será necesario reducir la dimensionalidad de la base de datos, esto es, encontrar una matriz $\tilde{\boldsymbol{X}}_{n\times 2}$ que aproxime adecuadamente la matriz de datos original $\boldsymbol{X}_{n\times p}$ (n observaciones y p variables). El procedimiento a realizar va a depender de cómo se haya cuantificado la disimilitud entre los objetos (para más información sobre este aspecto, ver Everitt y Hothorn (2011)).

Las técnicas de **escalamiento multidimensional** (MDS por sus siglas en inglés) permiten construir una configuración de n puntos en un espacio de baja dimensionalidad (i.e. un mapa) sin usar directamente la base de datos sino la matriz de distancias. Principalmente se pueden diferenciar dos metodologías: MDS clásico y MDS no métrico.

El MDS clásico, también conocido como Análisis de Coordenadas Principales, trata de obtener una aproximación a la matriz de datos original, $\boldsymbol{X}_{n\times p}$ a partir de la matriz de distancias, $\boldsymbol{D}_{n\times n}$, mediante el teorema de la descomposición espectral.

Primeramente, se asume $\boldsymbol{D}$ matriz euclidiana y por lo tanto, semidefinida positiva, y se considera la matriz $\boldsymbol{B} = \boldsymbol{X}\boldsymbol{X}'$, a partir de la cual se pueden expresar las distancias euclidianas al cuadrado. Entonces, la idea es aproximar indirectamente la matriz de distancias $\boldsymbol{D}$, a partir de $\boldsymbol{B}$.

El algoritmo MDS clásico consta de los siguientes pasos:

1. A partir de $\boldsymbol{D} = (d_{ij})$, construir la matriz $\boldsymbol{A} = -\frac{1}{2}d_{ij}^2$.

2. Centrar la matriz $\boldsymbol{A} = (a_{ij})$ por filas y por columnas. Esto es, computar $\boldsymbol{B} = \boldsymbol{HAH}$, donde $\boldsymbol{H} = \boldsymbol{I}_n - \frac{1}{n}\mathbf{1}\mathbf{1}'$ es la matriz de centrado.

3. Tomar la descomposición espectral de $\boldsymbol{B}$

$$\boldsymbol{B} = \boldsymbol{XX'} = \boldsymbol{VD}_\lambda \boldsymbol{V'} = \sum_{i=1}^{n} \lambda_i v_i v_i',$$

con $\boldsymbol{D}_\lambda = \text{diag}\{\lambda_1, \ldots, \lambda_n\}$, λ_i valores propios, y $\boldsymbol{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n]$, $\boldsymbol{v}_i$ vectores propios.

4. Obtener las coordenadas principales a partir de $\boldsymbol{X} = \boldsymbol{V}\boldsymbol{D}_\lambda^{1/2}$.

Nótese que, $\tilde{\boldsymbol{D}} = \boldsymbol{V}_{(,1:2)}\boldsymbol{D}_{\lambda(1:2,1:2)}(\boldsymbol{V}_{(,1:2)})'$ es la aproximación de mínimos cuadrados de rango 2 para $\boldsymbol{B}$. Además, si $\boldsymbol{D}$ no es euclídea, $\boldsymbol{B}$ no es definida positiva y algunos valores propios, λ_i, serán negativos. No obstante, en Everitt y Hothorn (2011) se comenta que es posible obtener una representación de la matriz de distancias lo suficientemente buena utilizando únicamente los valores propios positivos y sus vectores propios asociados.

En general, la bondad del ajuste se puede cuantificar mediante la siguiente expresión:

$$\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n-1} \lambda_i}, \tag{2.2}$$

donde k es el rango de la aproximación.

Por consiguiente, una forma de evaluar la aproximación cuando $\boldsymbol{D}$ no es matriz euclidiana puede obtenerse modificando ligeramente la ecuación 2.2

$$\frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n-1} |\lambda_i|} \quad \text{ó} \quad \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{\lambda_i > 0} \lambda_i}.$$

2.1.4. Comparación de la estructura interna de los clústeres

En el apartado 2.1.2.3 se ha visto cómo el Coeficiente Silhouette mide la calidad del agrupamiento basándose únicamente en información interna de los datos. Por el contrario, existen también métodos de validación externa para evaluar si la solución encontrada en el análisis clúster se aproxima a la agrupación real (si se conoce) o a otra clasificación encontrada.

Una medida de semejanza entre dos agrupaciones de objetos como, por ejemplo, entre los clústeres resultantes de aplicar métodos de *clustering* es el Índice de Rand.

Dado un conjunto de n elementos, $\mathcal{L} = \{x_1, x_2, \ldots, x_n\}$, y dos agrupaciones de $\mathcal{L}$ a comparar: $\mathcal{C} = \{C_1, C_2, \ldots, C_r\}$, de tamaño r, y $\mathcal{C}' = \{C_1', C_2', \ldots, C_s'\}$, de tamaño s. Entonces, el Índice de Rand se define como (Rand, 1971):

$$R(\mathcal{C}, \mathcal{C}') = \frac{a+b}{a+b+c+d} = \frac{(a+b)}{\big(n(n-1)\big)/2}$$

donde

- a: es el número de pares de elementos que están en el mismo clúster en $\mathcal{C}$ y $\mathcal{C}'$.
- b: es el número de pares de elementos que están en clústeres distintos en $\mathcal{C}$ y $\mathcal{C}'$.
- c: es el número de pares de elementos que están en el mismo clúster en $\mathcal{C}$ y en distinto clúster en $\mathcal{C}'$.
- d: es el número de pares de elementos que están en clústeres distintos en $\mathcal{C}$ e iguales en $\mathcal{C}'$.

Se puede observar que el numerador representa el número de pares coincidentes y el denominador, el número de pares total. Por lo tanto, el índice de Rand toma valores en el intervalo $[0, 1]$ donde 0 implica que las dos agrupaciones no coinciden en ningún par de objetos y 1 indica que $\mathcal{C}$ y $\mathcal{C}'$ son exactamente iguales.

El valor esperado del índice Rand es diferente de cero incluso en el caso de asignaciones aleatorias. Por esta razón se ha propuesto una versión del índice de Rand ajustado por azar (Hubert & Arabie, 1985):

$$R_{adj} = \frac{R(\mathcal{C}, \mathcal{C}') - \mathbb{E}\left(R(\mathcal{C}, \mathcal{C}')\right)}{\text{máx}\left\{R(\mathcal{C}, \mathcal{C}')\right\} - \mathbb{E}\left(R(\mathcal{C}, \mathcal{C}')\right)}$$

Dónde $\mathbb{E}(\cdot)$ indica la esperanza del índice en caso de asignación aleatoria y se determina mediante permutaciones aleatorias de los clústeres. Nótese que esta corrección puede tomar valores negativos si el índice toma un valor inferior a la esperanza.

2.2. Distancia de Hausdorff

En la Sección 2.1 se ha visto que para poder llevar a cabo un análisis clúster es imprescindible caracterizar las analogías y diferencias entre los objetos de interés mediante la matriz de distancias inter-individuos. Pero existen múltiples formas de hacerlo y la definición de distancia va a depender de la naturaleza de los datos (p.ej. datos cualitativos) y del objetivo del estudio.

En general, una distancia $\delta_{ij} = \delta(i, j)$ entre elementos i, j de una población de tamaño n es una medida simétrica y no negativa que mide la diferencia entre ambos en base a las p

variables que los describen (Cuadras, 1988). La matriz de distancias, calculada a partir de la matriz de datos $\boldsymbol{X}_{n\times p}$, es la siguiente:

$$\Delta = \begin{pmatrix} \delta_{11} & \delta_{12} & \dots & \delta_{1n} \\ \delta_{21} & \delta_{22} & \dots & \delta_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \delta_{n1} & \delta_{n2} & \dots & \delta_{nn} \end{pmatrix},$$

donde $\delta_{ij} \geq 0$, $\delta_{ij} = \delta_{ji}$ y $\delta_{ii} = 0$.

Por lo que respecta a los objetos en movimiento, estos cambian su valor con el tiempo y su seguimiento da lugar a una secuencia de puntos espaciotemporales, denominada **trayectoria** (Magdy y col., 2015). Una trayectoria de longitud m puede ser expresada como $T = [(v_1, t_1), \dots, (v_m, t_m)]$, donde $[v_1, \dots, v_m]$ y $[t_1, \dots, t_m]$ representan las secuencias de las posiciones y los tiempos, respectivamente.

En estos casos, medir la longitud de la recta que une dos de los puntos de las trayectorias, o sus centros, no nos va a resultar útil cuando el interés reside en cuantificar cómo se diferencian con respecto a su **forma geométrica**.

Existen dos medidas que se basan en la forma geométrica de la trayectoria e ignoran la dimensión del tiempo: la distancia de Hausdorff, medida métrica que mide qué tan cerca están las formas de dos trayectorias, y la distancia de Fréchet, medida no métrica que tiene en cuenta la ubicación y el orden de los puntos a lo largo de las trayectorias.

En esta sección se dará a conocer la distancia de Hausdorff y se propondrá un algoritmo eficiente para su cálculo siguiendo los pasos del estudio de Taha y Hanbury (2015).

2.2.1. Definición y propiedades

Dadas dos trayectorias A y B, bajo la premisa de que son conjuntos no vacíos, cerrados y acotados, la distancia de Hausdorff se define como (Hausdorff, 1914):

$$H(A, B) = \max\left\{h(A, B), h(B, A)\right\} \tag{2.3}$$

donde

$$h(A, B) = \max_{a\in A}\left\{\min_{b\in B}\left\{d(a, b)\right\}\right\} \tag{2.4}$$

$$h(B, A) = \max_{b\in B}\left\{\min_{a\in A}\left\{d(a, b)\right\}\right\} \tag{2.5}$$

Generalmente las ecuaciones 2.4 y 2.5, denominadas distancia de Hausdorff dirigida hacia delante (*forward*) y hacia atrás (*backward*), respectivamente, no son iguales (i.e. no cumplen la propiedad de simetría). Además, $d(a, b)$ es cualquier distancia métrica entre a y b.

La distancia de Hausdorff satisface las siguientes propiedades:

- No negatividad: $H(A, B) \geq 0, \quad \forall A, B$
- Identidad: $H(A, B) = 0 \iff A = B$
- Simetría: $H(A, B) = H(B, A)$
- Desigualdad triangular: $H(A, B) \leq H(A, C) + H(B, C)$

Así, la distancia de Hausdorff es una distancia métrica similar a la distancia euclidiana, con la ventaja de que es aplicable a cualquier tipo de objeto.

Sin embargo, no es una medida robusta: $h(A, B)$ representa la distancia más grande desde cualquier punto extremo en A hasta su punto más cercano en B, y $h(B, A)$ representa la distancia máxima entre cualquier punto límite de B hasta el punto más cercano de A. Por este motivo, en Min y col. (2007) se propone una modificación de la distancia de Hausdorff en la que en lugar de los máximos se utilicen determinados cuantiles (p.ej. la mediana).

Para terminar, cabe mencionar que en este trabajo la distancia espacial $d(a, b)$ entre cada par de puntos de las trayectorias se calculará mediante la fórmula de Haversine (Sinnott, 1984) ya que la latitud y longitud son coordenadas esféricas, no cartesianas.

Asumiendo que la Tierra es esférica con radio R, la distancia Haversine es:

$$d_{\text{Haversine}} = R \cdot c$$

con

$$c = 2 \cdot \arcsin\left(\min\{1, \sqrt{a}\}\right)$$

$$a = \sin^2\left(\frac{lat_b - lat_a}{2}\right) + \cos(lat_a) \cdot \cos(lat_b) \cdot \sin^2\left(\frac{lon_b - lon_a}{2}\right)$$

Nótese que la distancia se expresará en las mismas unidades que el radio.

2.2.2. Algoritmo para el cálculo exacto de la distancia de Hausdorff

La complejidad computacional que existe al calcular la distancia de Hausdorff es bien conocida en la literatura actual; esta depende del tamaño del conjunto de puntos, de la densidad

y dispersión de los puntos y de que pueda ser aplicable a cualquier situación (Taha & Hanbury, 2015).

Existen los métodos heurísticos, que tratan de identificar una buena aproximación a la distancia de Hausdorff, y los métodos exactos. En este apartado se propondrá un algoritmo para el cálculo exacto de la distancia de Hausdorff para datos de trayectorias, en base al algoritmo general propuesto en Taha y Hanbury (2015).

Sean las trayectorias $A = \left[a_1, a_2, \ldots, a_n\right]$ y $B = \left[b_1, b_2, \ldots, b_m\right]$ donde $a, b \in \Re^2$ representan las coordenadas. Entonces, el algoritmo para calcular la distancia de Hausdorff dirigida (i.e. *forward* o *backward*) mediante un enfoque de búsqueda extensiva, se incluye a continuación:

Algoritmo 1: Cálculo de la Distancia de Hausdorff dirigida.

Datos: Dos trayectorias A y B
Resultado: Distancia de Hausdorff (dirigida) entre las trayectorias

```
1  cmax ← 0;
2  para a ∈ A hacer
3  |  cmin ← ∞;
4  |  para b ∈ B hacer
5  |  |  d ← d(a, b) // p.ej. distancia euclídea, Haversine, etc.
6  |  |  si d < cmin entonces
7  |  |  |  cmin ← d
8  |  |  fin
9  |  fin
10 |  si cmin > cmax entonces
11 |  |  cmax ← cmin
12 |  fin
13 fin
14 devolver cmax
```

Dos mejoras que pueden reducir notablemente el coste computacional son: (i) realizar una parada temprana del bucle interno y (ii) recorrer los puntos de forma aleatoria en ambos bucles.

La modificación (i) es posible ya que la distancia de Hausdorff dirigida trata de encontrar el máximo de las distancias mínimas (ver ecuaciones 2.4 y 2.5) por lo que en realidad el bucle interior puede pararse tan pronto como se haya encontrado una distancia inferior a la distancia de Hausdorff dirigida temporal, $cmax$.

Por otra parte, la modificación (ii) viene motivada para evitar distancias similares en ite-

raciones sucesivas y, de esto modo, posiblemente acelerar la parada temprana. Dado que aleatorizar no tiene coste, en Taha y Hanbury (2015) recomiendan aleatorizar siempre los puntos de las dos trayectorias.

Algoritmo 2: Cálculo de la Distancia de Hausdorff dirigida con las modificaciones de muestreo aleatorio y parada temprana.

```
   Datos: Dos trayectorias A y B
   Resultado: Distancia de Hausdorff (dirigida) entre las trayectorias
 1 cmax ← 0;
 2 para a ∈ aleatorizar(A) hacer
 3 |   cmin ← ∞;
 4 |   para b ∈ aleatorizar(B) hacer
 5 |   |   d ← d(a, b)
 6 |   |   si d < cmax entonces
 7 |   |   |   break;
 8 |   |   fin
 9 |   |   si d < cmin entonces
10 |   |   |   cmin ← d
11 |   |   fin
12 |   fin
13 |   si cmin > cmax entonces
14 |   |   cmax ← cmin
15 |   fin
16 fin
17 devolver cmax
```

El Algoritmo 1 tiene un coste computacional $O(n \cdot m)$, con $n = |A|$ y $m = |B|$ ya que los bucles en las líneas 2 y 4 siempre van a recorrer todos los puntos mientras que el Algoritmo 2 tiene un coste computacional $O(m)$ en el mejor de los casos (cuando la parada temprana ocurriera al iniciar el bucle todas las iteraciones) y $O(n * m)$ en el peor.

Finalmente, para obtener la distancia de Hausdorff, tal como indica la fórmula 2.3, simplemente habría que guardar el máximo de los valores obtenidos mediante el Algoritmo 2 con cada una de las direcciones.

2.3. Software

Este apartado pretende resumir brevemente qué paquetes y funciones del *software* estadístico `R` (R Core Team, 2020) se utilizarán tanto en el cálculo de la distancia de Hausdorff como

en el análisis clúster.

A continuación se muestra el código de las funciones utilizadas en cada caso:

1. Cálculo de la distancia Haversine con la función `distHaversine()` del paquete *geosphere* (Hijmans, 2021):

   ```
   distHaversine(p1, p2, r = 6378137)
   ```

   donde `p1` y `p2` son la longitud y la latitud de los puntos, y el radio de la tierra es por defecto de 6378137 metros.

2. Comparación del tiempo computacional de los algoritmos con el paquete *microbenchmark* (Mersmann, 2021):

   ```
   microbenchmark( ..., list = NULL, times = 100L, unit = NULL,
       check = NULL, control = list(), setup = NULL)
   ```

   donde `...` es la expresión a evaluar y `times`, el número de veces a evaluar la expresión

3. Realizar el *clustering* jerárquico aglomerativo y el corte del dendrograma, mediante las funciones `hclust()` y `cutree()`, respectivamente:

   ```
   hclust(d, method = "complete", members = NULL)
   ```

   donde `d` es la matriz de distancias y el argumento `method` hace referencia al criterio de enlace a utilizar.

   ```
   cutree(tree, k = NULL, h = NULL, ...)
   ```

   donde `tree` es un objeto dendrograma, como el obtenido con `hclust()`, y `k` es el número de clústeres a determinar.

4. Cálculo del Coeficiente Silhouette mediante `NbClust()` (Charrad y col., 2014):

   ```
   NbClust(data = NULL, diss = NULL, distance = "euclidean", min.nc
       = 2, max.nc = 15,  method = NULL, index = "all", alphaBeale =
       0.1)
   ```

   donde `data` es la base de datos, `diss` es la matriz de distancias y `distance` la medida de distancia a utilizar cuando `diss=NULL`.

5. Representar la solución en un gráfico bidimensional mediante MDS clásico con la función `cmdscale()` del paquete *stats*:

   ```
   cmdscale(d, k = 2, eig = FALSE, add = FALSE, x.ret = FALSE, list.
       = eig || add || x.ret)
   ```

donde `d` es la matriz de distancias, `k` es el número máximo de dimensiones a ser representadas y `eig` indica si los valores propios deben ser devueltos o no.

6. Comparar la estructura de los clústeres mediante la función `adjustedRandIndex()` del paquete *mclust* (Scrucca y col., 2016):

```
adjustedRandIndex(x, y)
```

donde `x` e `y` son los vectores de etiquetas a comparar y deben tener la misma longitud.

Finalmente, para llevar a cabo las representaciones gráficas pertinentes en cada caso, se utilizarán las funciones del paquete *ggplot2* (Wickham, 2016).

Capítulo 3

Aplicación a los datos del Larus audouinii.

En este capítulo se aplicará toda la teoría expuesta en el Capítulo 2, referente al análisis clúster y al cálculo de la matriz de distancias de Hausdorff, para analizar los datos de localización GPS de las gaviotas de Audouin con el objetivo de encontrar subpoblaciones.

3.1. Descripción de la base de datos

Antes de empezar con el análisis, es conveniente conocer cierta información sobre el conjunto de datos. Tal y como se ha mencionado en la Introducción, durante el periodo de incubación del año 2011 se rastrearon mediante GPS las trayectorias de 36 gaviotas de Audouin de la colonia de *Punta de la Banya*, en el Parque Natural del Delta del Ebro (Ouled-Cheikh y col., 2020).

Se capturaron aleatoriamente sesenta gaviotas en estado de reproducción y se las equipó con el dispositivo GPS, programado para registrar la ubicación cada 5 minutos con una precisión de 10 metros. Luego, entre una y dos semanas más tarde, treinta y seis gaviotas fueron recapturadas y, durante ese tiempo, no se produjeron condiciones climáticas adversas (p. ej. lluvia o vientos fuertes) que pudieran afectar potencialmente el comportamiento de búsqueda de alimento de las gaviotas.

Además, en esa zona existen numerosos puertos pesqueros debido a la gran abundancia de peces de la familia de los cupleidos (p.ej. sardinas o arenques) gracias a los nutrientes abocados por el río Ebro. Principalmente, se dan dos actividades de pesca diferentes y cada una ofrece distintas oportunidades de alimentación a las gaviotas.

Por un lado, la pesca de arrastre se lleva a cabo de lunes a viernes de 7:00h a 17:00h y es un tipo de pesca no selectiva que produce grandes cantidades de descartes de los cuales se aprovechan las gaviotas de Audouin. Por otro, la pesca con redes de cerco se realiza de forma nocturna de lunes a viernes con hora inicial a las 23:00h y sin hora de retorno; este método de pesca produce pocos descartes pero supone una facilitación de recursos para las gaviotas debido a que concentra una gran cantidad de peces en la superficie del agua. Finalmente, los fines de semana no hay ningún tipo de actividad pesquera.

En la Figura 3.1 se muestran dos trayectorias en el mapa, obtenidas mediante la herramienta interactiva Shiny desarrollada en Cortejana Retamozo (2020), la cual puede visitarse aquí.

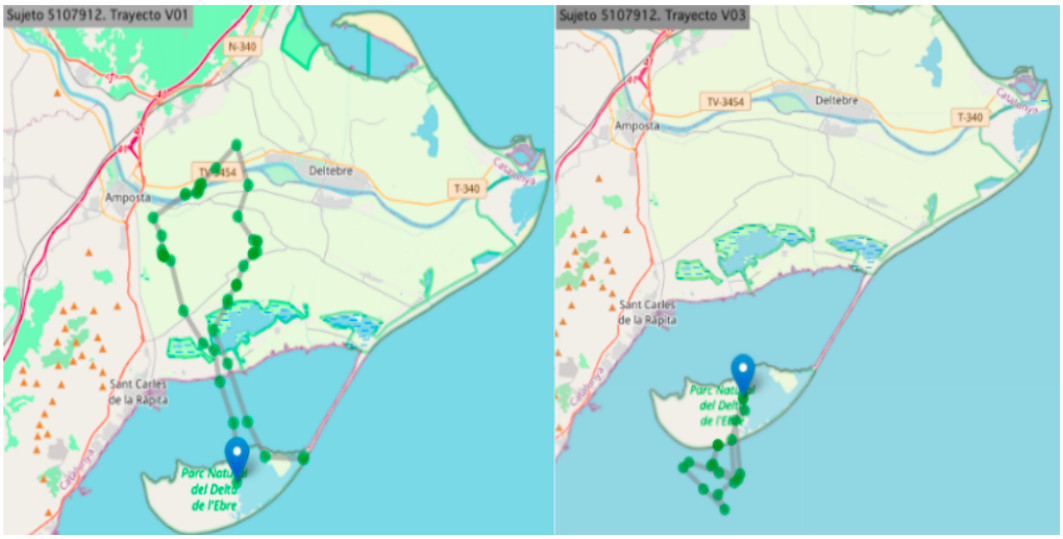

Figura 3.1: Visualización de las trayectorias V01 y V03 de la gaviota 5107912.

La base de datos original consta de 38090 filas, correspondientes a las geolocalizaciones recogidas durante las dos semanas, y 17 columnas, las variables explicativas. Sin embargo, en este estudio únicamente serán necesarias las siguientes siete variables:

- `ID_trip`: variable categórica con el número identificador de cada gaviota y su número identificador de trayectoria.
- `bird`: variable categórica que identifica la gaviota. Valores del 1 al 36.
- `trajectory`: variable categórica que identifica la trayectoria. Valores del 1 al 362.
- `LAT`: variable numérica continua que contiene la latitud de la coordenada GPS. Rango de valores entre 38,98799 y 41,42250.
- `LONG`: variable numérica continua que contiene la longitud de la coordenada GPS. Rango de valores entre $-0{,}402864$ y 2,266693.

- `time`: variable tipo fecha con la hora, minutos y segundos de la medición.
- `activity`: variable categórica con tres niveles (`No`, `Night`, `Day`) correspondientes al tipo de actividad pesquera existente en el momento de la medición.

Más detalladamente, la variable `activity` se creó de forma que toma valor `No` cuando el vuelo se inicia en fin de semana (esto incluye el lunes entre las 00:00h y las 10:00h) o de las 18:00h a las 00:00h entre semana. Si la trayectoria se inicia entre semana, `activity` toma valor `Night` si esto ocurre entre las 00:00h y las 10:00h, y `Day` entre las 10:00h y las 18:00h.

A modo ilustrativo, se incluye una pequeña muestra de la base de datos.

ID_trip	bird	trajectory	LAT	LONG	time	activity
5107912V01	1	1	40.57538	0.658506	13:29:54	No
5107912V01	1	1	40.58267	0.659979	13:34:47	No
5107912V01	1	1	40.60290	0.656322	13:39:35	No
5107912V01	1	1	40.62224	0.646901	13:44:19	No
5107912V01	1	1	40.63669	0.642940	13:49:02	No
5107912V01	1	1	40.64014	0.635187	13:53:45	No

Tabla 3.1: Primeras seis filas de la base de datos.

Análisis descriptivo.

A partir de la Figura 3.2 se puede observar que el número de trayectorias realizadas por cada gaviota difiere notablemente. Hay dos gaviotas que realizaron únicamente 2 trayectorias mientras que el máximo de trayectorias realizadas por una gaviota en esas dos semanas es 24; la media de trayectorias por gaviota es 10.

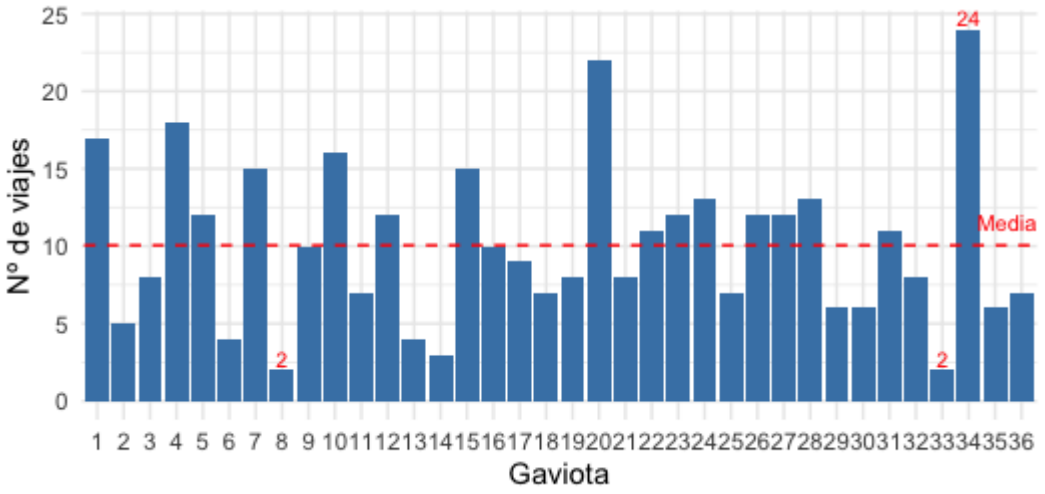

Figura 3.2: Histograma del número de trayectorias realizadas por cada gaviota.

En el gráfico circular adjuntado a continuación, se observa que las 362 trayectorias se han distribuido en los tres tipos de actividad pesquera de manera uniforme: alrededor del 33 % de las trayectorias se han iniciado en cada una de las opciones.

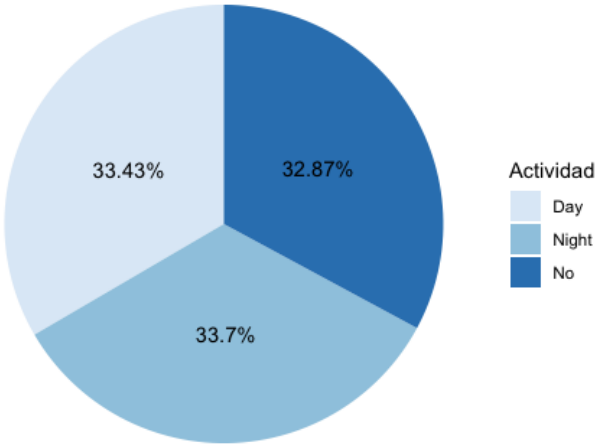

Figura 3.3: Tipo de actividad pesquera al iniciar la trayectoria.

Más concretamente, 119 trayectorias se iniciaron en ausencia de actividad pesquera, 121 en presencia de actividad pesquera diurna (de arrastre) y 122 con actividad pesquera nocturna (con redes de cerco).

Finalmente, es importante mencionar que no todas las gaviotas de Audouin efectuaron viajes en todos los niveles de la variable `activity`. Esto se puede ver en la siguiente tabla:

	Número de gaviotas	**Gaviotas ausentes**
No	33	(13, 14,18)
Night	34	(8, 12)
Day	33	(8, 33, 36)

Tabla 3.2: Resumen del número de gaviotas por tipo de actividad pesquera.

3.2. Cálculo de la distancia de Hausdorff

En esta sección se muestra cómo se ha llevado a cabo el cálculo de la distancia de Hausdorff mediante la implementación de los algoritmos vistos en el Capítulo 2.

Para empezar, en la Figura 3.4 se incluye un mapa con dos de las trayectorias de la base de datos, las cuales se utilizan de ejemplo para entender de una manera más visual cómo se

realiza el cálculo de la distancia de Hausdorff.

En el mapa, se puede observar fácilmente que la distancia de Hausdorff dirigida $h(A, B)$ es la distancia más larga entre todas las distancias más cortas de cada punto de la trayectoria A con todos los puntos de la trayectoria B, mientras que la distancia de Hausdorff dirigida hacia atrás, $h(B, A)$, representa la distancia máxima entre las distancias mínimas de los puntos de B con respecto a los de A.

Figura 3.4: Distancias de Hausdorff dirigidas entre dos trayectorias.

Finalmente, la distancia de Hausdorff entre las trayectorias A y B que se muestran en el mapa sería $H(A, B) = \text{máx}\left\{h(A, B), h(B, A)\right\} = h(B, A)$.

Por lo que respecta a su cálculo mediante el software estadístico `R` (R Core Team, 2020), se han implementado tres algoritmos diferentes para el cálculo de las distancias de Hausdorff dirigidas, siguiendo las indicaciones especificadas en la sección 2.2.

Así, en las siguientes páginas se da una breve descripción de los algoritmos para el cálculo de la distancia de Hausdorff dirigida hacia delante, $h(A, B)$, junto con el código de `R` utilizado en cada uno de ellos.

- **Algoritmo 1**

  Algoritmo *naive*, en el que se recorren los bucles por completo con el fin de llevar a cabo una búsqueda extensiva. Esto es, se calculan todas las posibles distancias entre cada par de puntos entre dos trayectorias.

```
Hausdorff_1 <- function(data){
  # Se guardan las 65703 combinaciones de 362 trayectorias tomadas de 2 en 2:
  combinaciones <- combn(1:max(data$trajectory),2)
  HD_solution <- matrix(nrow = max(data$trajectory),
                        ncol = max(data$trajectory))
  diag(HD_solution) <- 0

  # Visitamos cada par de trayectorias:
  for(t in 1:ncol(combinaciones)){
    A <- combinaciones[,t][1]
    B <- combinaciones[,t][2]
    cmax <- 0

    # Visitamos cada punto de A
    for(i in which(data$trajectory==A)){
      cmin <- Inf

      # Visitamos cada punto de B
      for(j in which(data$trajectory==B)){
        # Calculamos distancia Haversine entre i i j:
        d <- distHaversine(c(data$LONG[i], data$LAT[i]),
                           c(data$LONG[j], data$LAT[j]))
        if(d < cmin){
          cmin <- d
        }
      }
      if(cmin > cmax){
        cmax <- cmin
      }
    }
    HD_solution[A,B] <- HD_solution[B,A] <- cmax
  }
  return(HD_solution)
}
```

Código 3.1: Función para el cálculo de la distancia de Hausdorff dirigida hacia delante mediante el Algoritmo 1.

- **Algoritmo 2**

  Al algoritmo anterior se le incluye la modificación de efectuar una parada temprana en el caso en que se encuentre una distancia inferior a la distancia temporal. Las modificaciones se muestran comentadas.

```
Hausdorff_2 <- function(data){
  combinaciones <- combn(1:max(data$trajectory),2)
  HD_solution <- matrix(nrow = max(data$trajectory),
                        ncol = max(data$trajectory))
  diag(HD_solution) <- 0

  for(t in 1:ncol(combinaciones)){
    A <- combinaciones[,t][1]
    B <- combinaciones[,t][2]
    cmax <- 0

    for(i in which(data$trajectory==A)){
      cmin <- Inf
      for(j in which(data$trajectory==B)){
        d <- distHaversine(c(data$LONG[i], data$LAT[i]),
                           c(data$LONG[j], data$LAT[j]))
        # Parada temprana:
        if(d < cmax){
          if(d < cmin){cmin <- d}
          break
        }
        if(d < cmin){
          cmin <- d
        }
      }

      # Si el bucle se rompe, cmin puede valer INF, lo corregimos:
      if(cmin > cmax && cmin != Inf){
        cmax <- cmin
      }
    }
    HD_solution[A,B] <- HD_solution[B,A] <- cmax
    }
  return(HD_solution)
}
```

Código 3.2: Función para el cálculo de la distancia de Hausdorff dirigida hacia delante mediante el Algoritmo 2.

- **Algoritmo 3**

  Al algoritmo anterior se le incluye la modificación de recorrer los puntos de ambas trayectorias. Esto se hace mediante la función `sample()`.

```
Hausdorff_3 <- function(data){
  combinaciones <- combn(1:max(data$trajectory),2)
  HD_solution <- matrix(nrow = max(data$trajectory),
                        ncol = max(data$trajectory))
  diag(HD_solution) <- 0

  for(t in 1:ncol(combinaciones)){
    A <- combinaciones[,t][1]
    B <- combinaciones[,t][2]
    cmax <- 0

    # Visitamos los puntos de A en orden aleatorio
    for(i in sample(which(data$trajectory==A))){
      cmin <- Inf

      # Visitamos los puntos de B en orden aleatorio
      for(j in sample(which(data$trajectory==B))){
        d <- distHaversine(c(data$LONG[i], data$LAT[i]),
                           c(data$LONG[j], data$LAT[j]))
        if(d < cmax){
          if(d < cmin){cmin <- d}
          break
        }
        if(d < cmin){
          cmin <- d
        }
      }
      if(cmin > cmax && cmin != Inf){
        cmax <- cmin
      }
    }
    HD_solution[A,B] <- HD_solution[B,A] <- cmax
  }
  return(HD_solution)
}
```

Código 3.3: Función para el cálculo de la distancia de Hausdorff dirigida hacia delante mediante el Algoritmo 3.

Con el fin de comparar el coste computacional de cada algoritmo y evaluar la utilidad de las mejoras realizadas, se han ejecutado 10 veces cada uno de los algoritmos, usando como argumento la base de datos de las gaviotas de Audouin.

Se ha medido el tiempo empleado en cada caso con la función `microbenchmark()` del paquete *microbenchmark* (Mersmann, 2021). Por lo que respecta al *hardware*, se utilizó un CPU Intel™ Core® i9-9900 con una frecuencia básica de 3.10GHz.

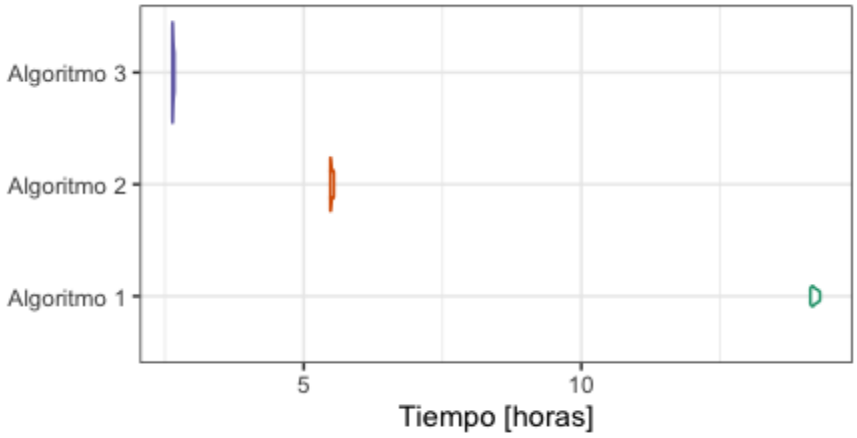

Figura 3.5: Gráfico de violín del tiempo computacional según el algoritmo.

En la Tabla 3.3 adjuntada a continuación se incluye un resumen numérico de los tiempos computacionales observados:

	Mínimo	**Media**	**Mediana**	**Máximo**
Algoritmo 1	13h 51m	13h 55m	13h 54m	14h 1m
Algoritmo 2	5h 22m	5h 24m	5h 23m	5h 26m
Algoritmo 3	2h 35m	2h 36m	2h 35m	2h 37m

Tabla 3.3: Resumen numérico del tiempo computacional por algoritmo.

Se observa que las modificaciones mejoran considerablemente el coste computacional necesario para el cálculo de la distancia de Hausdorff dirigida. Como era de esperar, el algoritmo 3 es el más eficiente y, en promedio, el algoritmo 1 y el algoritmo 2 tardan 5.37 y 2.08 veces más que el algoritmo 3, respectivamente.

Consecuentemente, el algoritmo 3 será el utilizado para el cálculo de la distancia de Hausdorff. Cabe mencionar que para calcular la distancia de Hausdorff es necesario implementar también la función para el cálculo de la distancia de Hausdorff dirigida hacia atrás y, finalmente, tomar el máximo entre los resultados de ambas funciones. Así, el tiempo compu-

tacional total será aproximadamente el doble que los tiempos especificados en la Tabla 3.3.

Dado que la función `distHaversine()` (Hijmans, 2021) utiliza por defecto el radio de la tierra en metros, este se ha dividido entre 1000 para pasarlo a kilómetros (ver Anexo A, página 67). Así, las unidades de las distancias entre trayectorias serán también kilómetros.

A continuación se incluye una pequeña muestra de la matriz de distancias de Hausdorff calculadas sobre las 362 trayectorias de las gaviotas de Audouin.

	1	**2**	**3**	**4**	**5**	**6**
1	0.000	9.360	17.395	11.901	33.441	66.101
2	9.360	0.000	9.276	3.535	33.441	66.087
3	17.395	9.276	0.000	5.808	27.351	60.310
4	11.901	3.535	5.808	0.000	32.895	65.614
5	33.441	33.441	27.351	32.895	0.000	34.332
6	66.101	66.087	60.310	65.614	34.332	0.000

Tabla 3.4: Matriz de distancias de Hausdorff entre las seis primeras trayectorias de las gaviotas de Audouin.

Para terminar, al observar la Tabla 3.5, cabe destacar que la distancia promedio es de 40.95 kilómetros y que el 75 % de las distancias se encuentran a 54.53 kilómetros o menos.

Mínimo	**Q1**	**Mediana**	**Media**	**Q3**	**Máximo**
0.69km	15.33km	32.82km	40.95km	54.53km	194.67km

Tabla 3.5: Resumen numérico de la matriz de distancias de Hausdorff.

3.3. Análisis Clúster

En este apartado se incluirán todos los resultados obtenidos al realizar el análisis de conglomerados a partir de la distancia de Hausdorff calculada en la sección anterior.

En Ouled-Cheikh y col. (2021) se concluyó que la mecánica de vuelo de las gaviotas de Audouin está muy condicionada por el tipo de actividad pesquera definida en la variable `activity`. Se observó que el comportamiento de las gaviotas en relación a sus trayectorias era notablemente diferente y, por esta razón, el análisis se segrega por esta variable. No obstante, y para dar una visión más completa, también se incluirá el análisis global sin segregar por el momento de actividad pesquera.

Por lo tanto, a partir de la matriz de distancias global, se han creado 3 matrices de distancias de la siguiente manera:

1. `activity='No'`: Matriz con las trayectorias que se iniciaron en ausencia de actividad pesquera.
2. `activity='Night'`: Matriz con las trayectorias que se iniciaron en presencia de actividad pesquera nocturna.
3. `activity='Day'`: Matriz con las trayectorias que se iniciaron en presencia de actividad pesquera diurna.

3.3.1. Matriz de distancias entre gaviotas

En primer lugar, es fundamental tener en cuenta que para tratar de identificar subpoblaciones de gaviotas se debe trabajar con matriz de distancias **entre gaviotas** y no entre trayectorias.

Entonces, es necesario construir las cuatro nuevas matrices de distancias inter-gaviotas, la matriz global y las matrices segregadas por `activity`, teniendo en cuenta la estructura jerárquica de los datos. Esto es, que las trayectorias no son independientes sino que existe un efecto "individuo"que es la gaviota que las realiza.

Las matrices de distancias entre gaviotas se calculan como la media de la distancias entre las trayectorias de cada par de gaviotas. Sea la matriz inter-trayectorias $\boldsymbol{D_{t\times t}}$ con t número de trayectorias, en cada una de las matrices de distancias inter-trayectorias se ha llevado a cabo el siguiente procedimiento:

EMPEZAR

Inicializar matriz inter-gaviotas, $\boldsymbol{M_{n\times n}}$, con n número de gaviotas.

REPETIR Para cada pareja de gaviotas (i, j), $\forall i \neq j$, $i, j = 1, \ldots, n$

Tomar las filas de $\boldsymbol{D}$ correspondientes a las trayectorias de la gaviota i.

Tomar las columnas de $\boldsymbol{D}$ correspondientes a las trayectorias de la gaviota j.

Calcular la media y guardar el valor en la matriz inter-gaviotas $\rightarrow \boldsymbol{M_{(i,j)}} = \boldsymbol{M_{(j,i)}}$

HASTA que no queden más parejas.

Asignar valor 0 a la diagonal $\rightarrow \boldsymbol{M_{(i,i)}} = 0$

TERMINAR

Como ya se ha mencionado en el análisis descriptivo (Sección 3.1), no todas las gaviotas iniciaron trayectorias en todos los escenarios de actividad pesquera. Por ello, las matrices

inter-gaviotas tienen dimensión $n_k \times n_k$, con $k = 1, 2, 3, 4$ donde $n_1 = 36, n_2 = 33, n_3 = 34, n_4 = 33$, correspondientes a la matriz global y las matrices con las trayectorias iniciadas en ausencia, presencia nocturna y presencia diurna de actividad pesquera, respectivamente.

Una vez calculadas las matrices de distancias entre gaviotas, estas pueden representarse gráficamente mediante un *heatmap* o mapa de calor (Figura 3.6), en el cual se muestra un gradiente de color proporcional al valor de la distancia entre cada par de gaviotas.

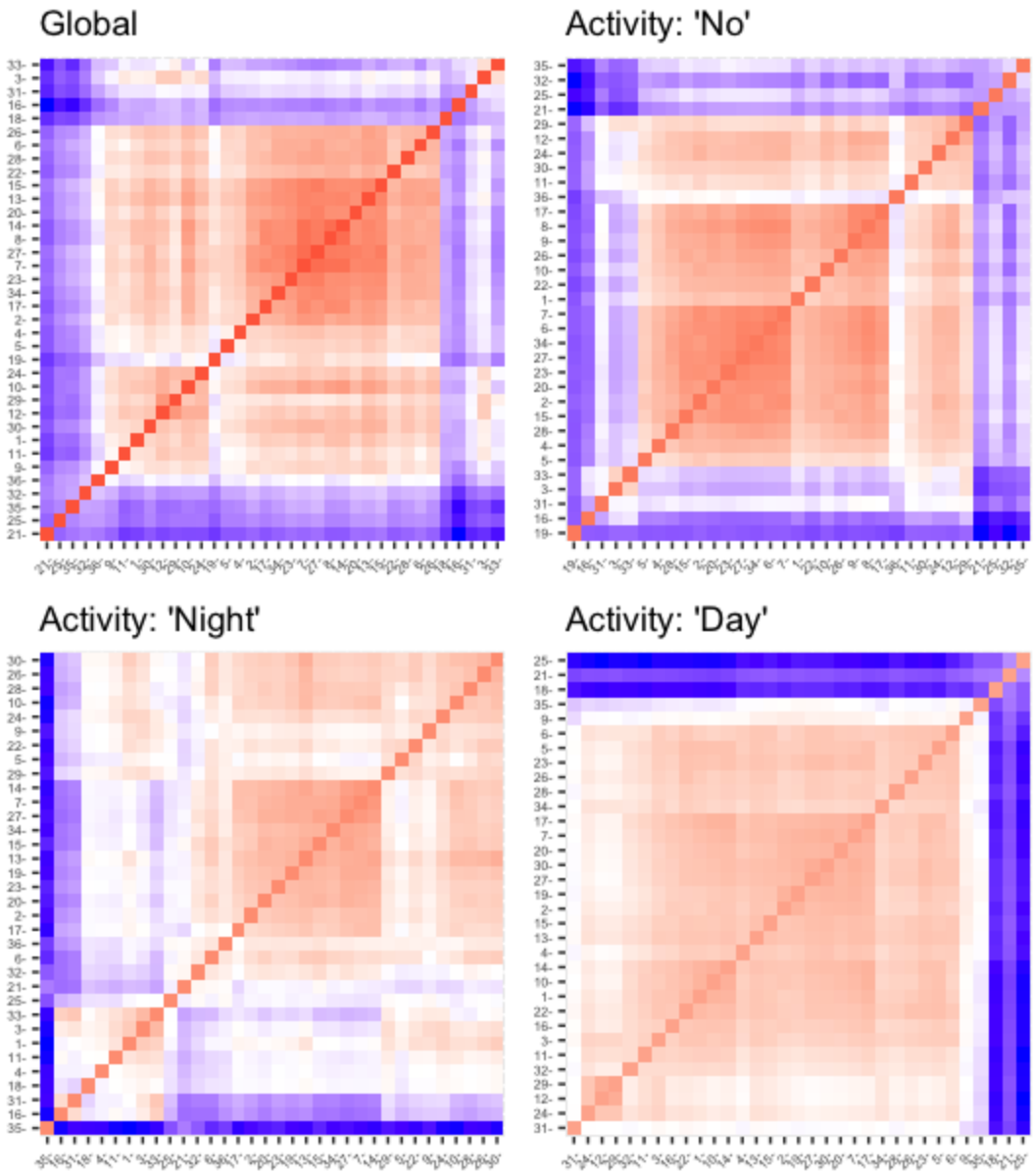

Figura 3.6: *Heatmap* de las distancias entre gaviotas.

Los gráficos han sido creados a partir de la función `fviz_dist()` del paquete *factoextra* (Kassambara & Mundt, 2020). En los ejes se encuentran las gaviotas ordenadas de manera que gaviotas similares están cerca. El color rojo corresponde a una distancia pequeña y el color azul indica una gran distancia entre las gaviotas.

Se observa, por ejemplo, que la gaviota 35 en la matriz de distancias de trayectorias iniciadas con `activity ='Night'`, ha realizado trayectorias con una distancia de Hausdorff promedio muy lejana a todas las demás gaviotas; posiblemente se trate de un valor extremo.

Además, en los cuatro mapas de calor pueden diferenciarse las zonas azules de las rojas, lo cual es indicativo de la existencia de clústeres diferenciados.

3.3.2. *Clustering* Jerárquico Aglomerativo

El siguiente paso es llevar a cabo el análisis clúster de tipo jerárquico aglomerativo a partir de las matrices de distancias entre gaviotas.

Para ello, se hace uso de la función `hclust()` en la que se ha indicado la utilización del método de Ward (`method="ward.D2"`) como criterio de enlace inter-clúster por ser un criterio menos susceptible al ruido y a valores extremos, que es ampliamente utilizado desde su primera descripción por Ward en una publicación de 1963 y que produce dendrogramas más interpretables que los otros criterios (Murtagh & Legendre, 2014), con clústeres más homogéneos y de tamaño similar (Graffelman y col., 2021).

Los dendrogramas resultantes se incluyen en la Figura 3.7 adjuntada en la página siguiente. Como ya se ha visto antes, la gaviota 35 en el análisis de las trayectorias iniciadas en presencia de actividad nocturna está separada de todas las demás a una distancia de alrededor de unos 200 kilómetros. Otros aspectos a destacar son que, en el dendrograma de actividad diurna, la separación en los dos primeros clústeres ocurre a una distancia muy grande, de aproximadamente 250 kilómetros, mientras que en el dendrograma con todas las trayectorias, la primera separación ocurre a una distancia considerablemente inferior que en los otros tres.

Con el fin de verificar qué tan bien están representadas las distancias entre gaviotas en los dendrogramas obtenidos, se ha calculado el coeficiente de correlación cofenética. En la Tabla 3.6 se incluye el coeficiente para cada uno de los dendrogramas.

Global	**Activity: 'No'**	**Activity: 'Night'**	**Activity: 'Day'**
0.74	0.74	0.46	0.95

Tabla 3.6: Coeficientes de correlación cofenética.

Se observa cómo las distancias están bastante bien representadas en el caso global y en el caso de `activity='No'` y casi perfectamente representadas para la matriz de `activity='Day'`. En cambio, para `activity='Night'` la bondad del ajuste no es tan buena y está un poco por debajo de 0.5, por lo que su dendrograma es menos fiable.

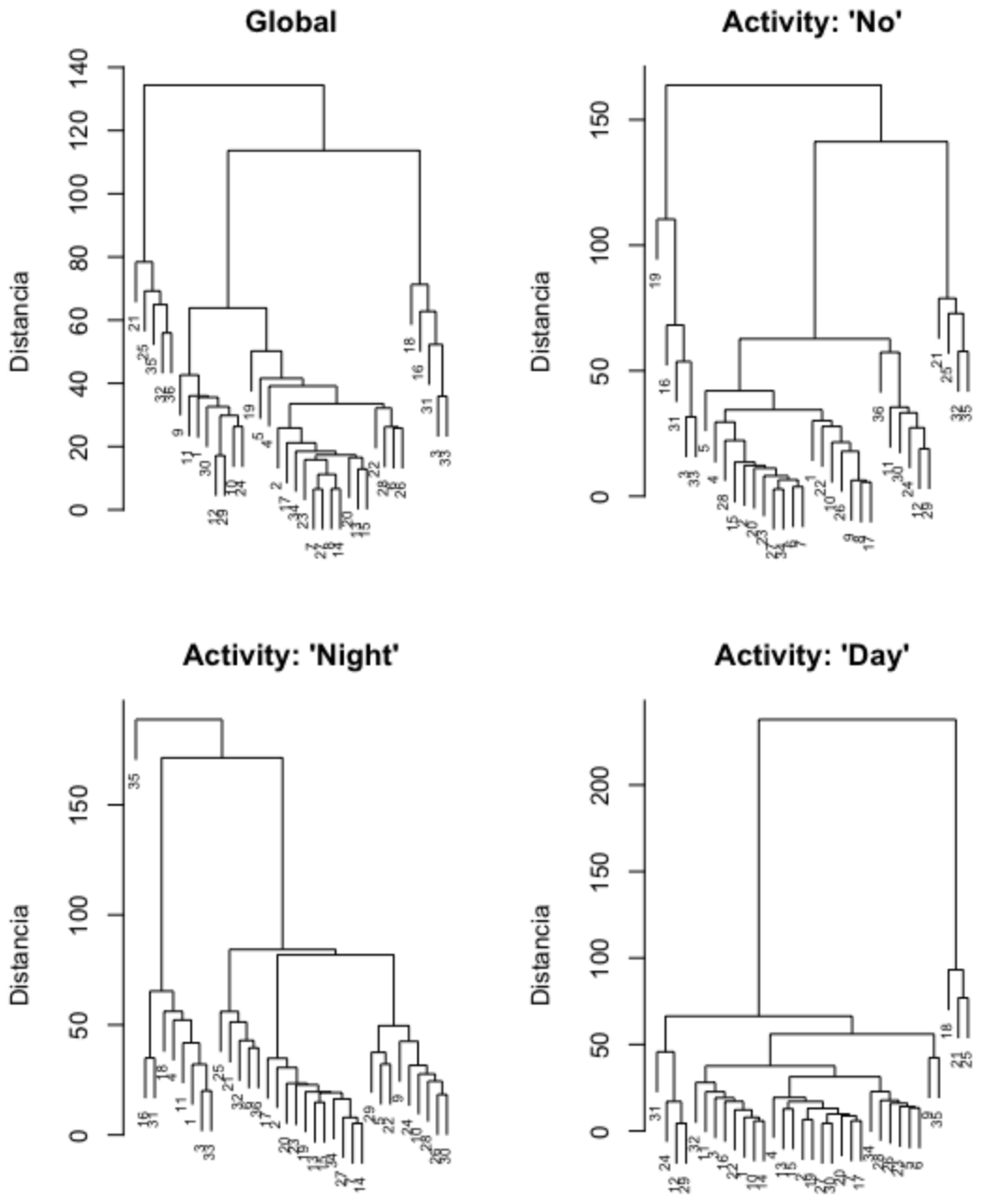

Figura 3.7: Dendrogramas para las distancias entre gaviotas obtenidos mediante el método jerárquico algomerativo con el criterio de enlace de Ward.

3.3.3. Número óptimo de clústeres: corte de los dendrogramas

Luego de haber verificado los dendrogramas obtenidos, se debe escoger una solución con un número determinado de clústeres; esto es, cortar el dendrograma.

Siguiendo la estrategia visual comentada en la Sección 2.1, parece que en general se seleccionarían dos o tres grupos. Sin embargo, a excepción de las trayectorias en presencia de actividad pesquera, esta decisión no resulta del todo evidente con este enfoque.

Consecuentemente, se han calculado los coeficientes silueta de las gaviotas con el fin de examinar cuál es el número de clústeres que maximiza su media. A continuación se comentan los resultados para cada dendrograma individualmente.

i. Matriz global de distancias entre gaviotas.

En la Figura 3.8 se observa que la media de los índices silueta es máxima si se dividen las gaviotas en siete clústeres. No obstante, se detecta un máximo local en dos clústeres y, la pérdida en la bondad de la agrupación que se obtiene al escoger dos clústeres en lugar de siete es muy pequeña (exactamente de 0.0383 unidades) en comparación al beneficio de agrupar las gaviotas en un número de clústeres pequeño.

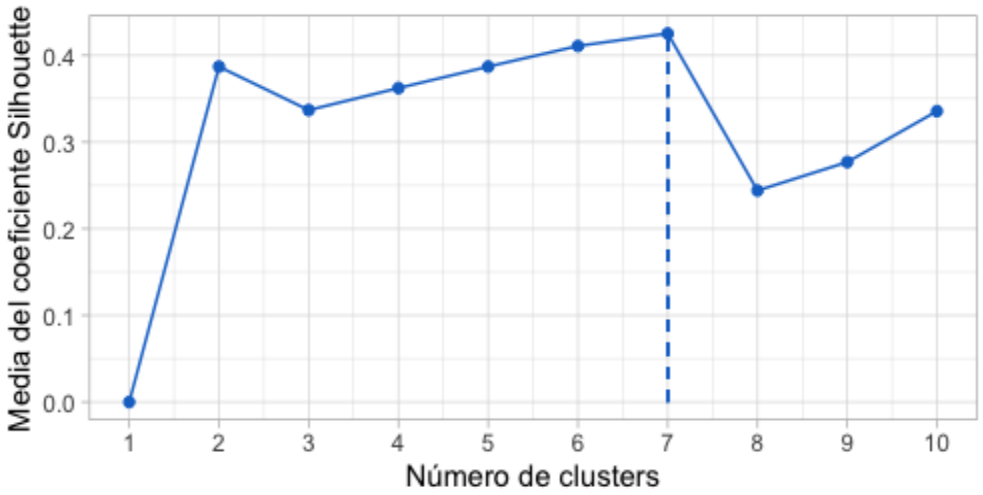

Figura 3.8: Media del coeficiente silueta según el número de clústeres para la matriz de distancias entre gaviotas global.

A pesar de esto, la media de los coeficientes silueta al tomar dos clústeres es moderada (0.387), teniendo en cuenta que coeficientes iguales 1 indican que las gaviotas están perfectamente clasificadas y 0, lo contrario.

Esto sugiere la posibilidad de que en realidad no exista una estructura clara de grupos en las gaviotas al utilizar todas sus trayectorias sin condicionarlas a la actividad pesquera.

ii. Matriz de distancias entre gaviotas en ausencia de actividad pesquera.

En este caso se obtiene también que la media es máxima al determinar siete clústeres y, de nuevo la diferencia entre las medias al optar por dos o siete clústeres es pequeña (en este caso, de 0.0822 unidades). Por lo tanto, siguiendo con el mismo razonamiento, se dividirán las gaviotas en dos grupos.

Figura 3.9: Media del coeficiente silueta según el número de clústeres para la matriz de distancias entre gaviotas y en ausencia de actividad pesquera.

La bondad de ajuste al escoger dos clústeres es 0.4415. Del mismo modo que antes, esto podría sugerir la existencia de dos subpoblaciones de gaviotas de Audouin en ausencia de actividad pesquera pero no claramente diferenciadas.

iii. Matriz de distancias entre gaviotas para actividad pesquera diurna.

A partir de la Figura 3.10, se observa que la media de los índices silueta al segregar las trayectorias por actividad pesquera diurna es máxima al dividir las gaviotas en cuatro grupos. Sin embargo, la bondad del ajuste de las asignaciones de las gaviotas al establecer dos o cuatro grupos es similar. Adicionalmente, al observar el dendrograma en la Figura 3.7 se puede ver cómo cortar el dendrograma en cuatro clústeres implica que tres de los clústeres incluirán una única gaviota.

Con todo, se decide establecer dos clústeres con los que se obtiene una bondad de asignación bastante buena (0.7424).

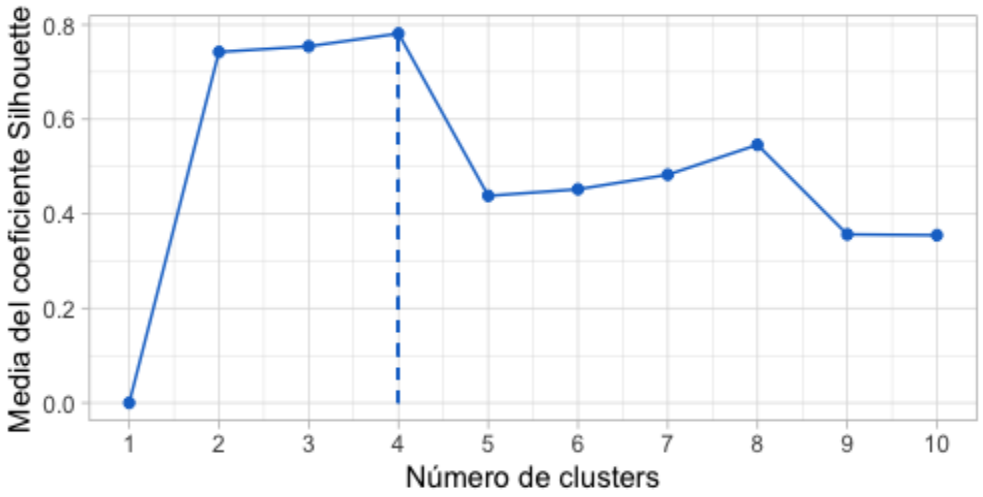

Figura 3.10: Media del coeficiente silueta según el número de clústeres para la matriz de distancias entre gaviotas y en presencia de actividad pesquera diurna.

iv. Matriz de distancias entre gaviotas para actividad pesquera nocturna.

Por último, en la Figura 3.11 se ve de forma inequívoca que el número óptimo de clústeres a establecer es dos, con una bondad de 0.67. Sin embargo, esto simplemente nos permite identificar la gaviota 35 como un valor extremo y agrupar el resto de gaviotas en un mismo clúster.

Por este motivo, se decide seleccionar tres clústeres (dos subpoblaciones y un valor extremo), con el que se obtiene una bondad de las asignaciones de exactamente 0.3674.

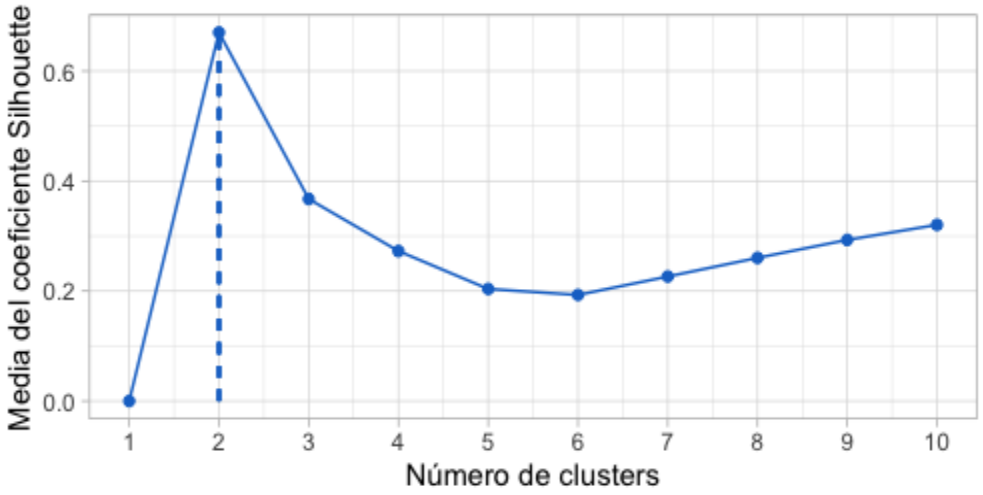

Figura 3.11: Media del coeficiente silueta según el número de clústeres para la matriz de distancias entre gaviotas y en presencia de actividad pesquera nocturna.

Y, para terminar, se adjuntan los dendrogramas con el corte que establece el número de clústeres que se ha escogido en cada caso.

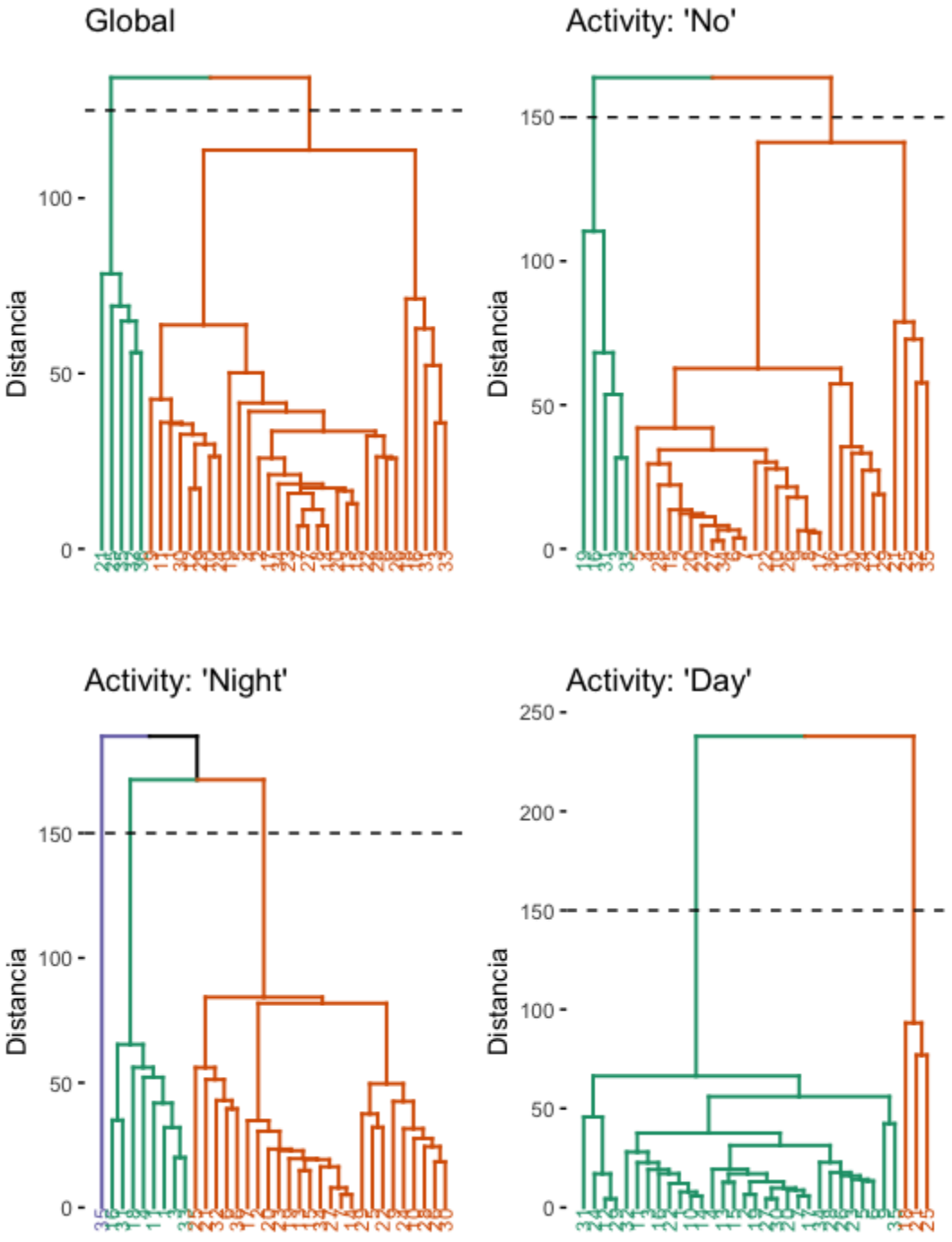

Figura 3.12: Dendrogramas con el número de clústeres determinado.

3.3.4. Representación bidimensional mediante MDS clásico

En este apartado se desea representar las soluciones del análisis clúster en un espacio de dos dimensiones. Para ello, se construye una configuración de n puntos en un espacio bidimensional a partir de las matrices de distancias entre las n gaviotas, mediante la técnica de escalamiento multidimensional clásica.

En la Figura 3.13 se muestran los cuatro gráficos bidimensionales obtenidos. Para ello se han utilizado la funciones `cmdscale()`, para el MDS clásico, y `ggscatter()` del paquete *ggpubr* (Kassambara, 2020), para la construcción de los gráficos.

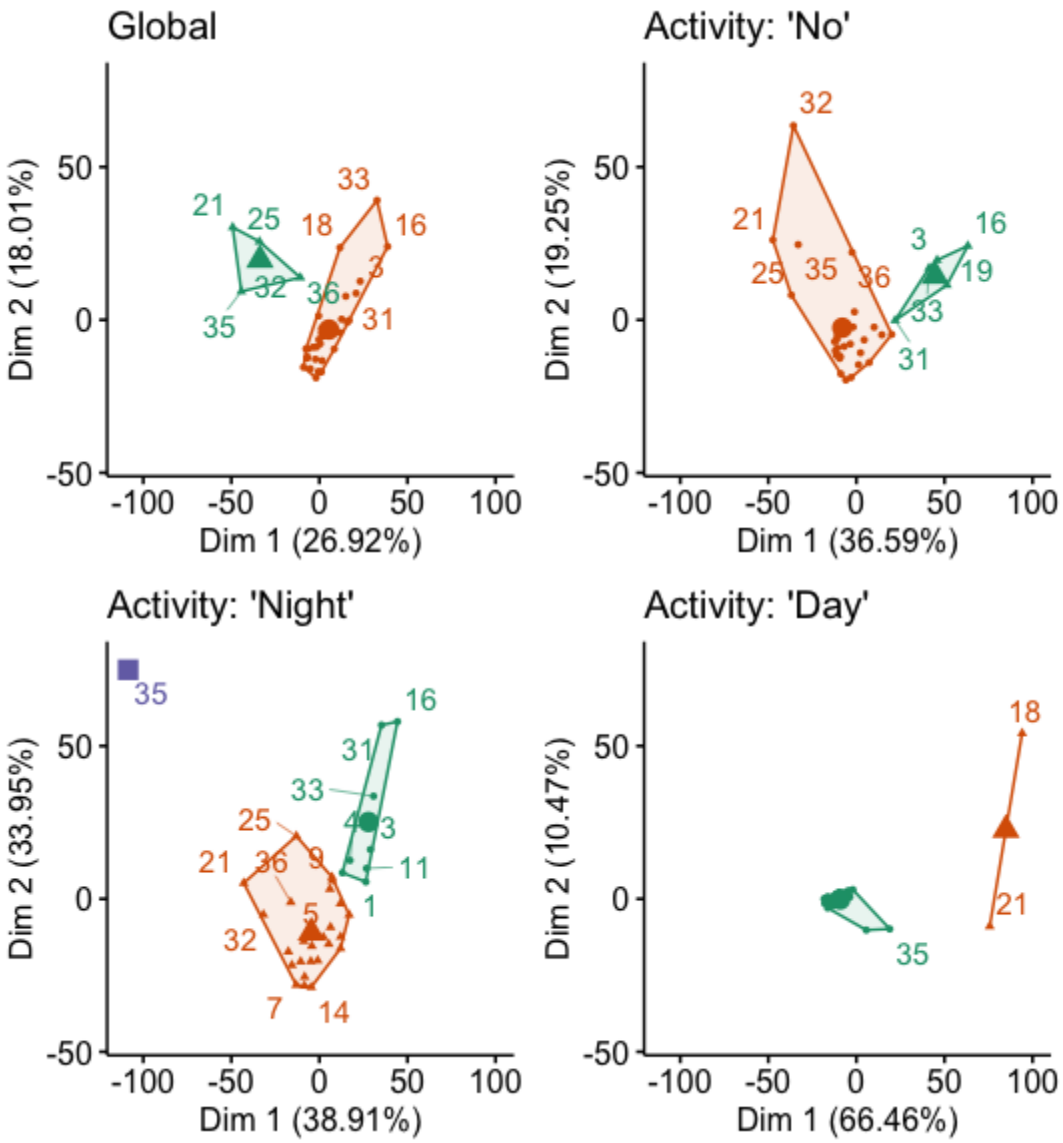

Figura 3.13: Representación de los clústeres mediante aproximación MDS clásico.

En los ejes se muestra el porcentaje de variabilidad explicada (i.e. el primer y segundo valor propio, respectivamente, multiplicados por 100) y las elipses representadas se corresponden con la envolvente convexa del conjunto de puntos de cada clúster. También se muestra el centroide de cada grupo mediante un círculo o triángulo de tamaño superior al resto.

Los centroides de los clústeres del caso global y de `Activity: 'No'` están separados por más o menos a unos 50 kilómetros, respecto a la dimensión 1. Esto apunta nuevamente al hecho de que los dos clústeres en estos casos no están claramente diferenciados, sino que la separación está siendo forzada al decidir cortar el dendrograma de esta forma.

Este hecho también puede observarse entre los dos clústeres con cardinalidad mayor a uno en el caso de actividad pesquera nocturna. Por el contrario, la distancia de los centroides de estos dos clústeres con la gaviota 35 es de 100 kilómetros o más.

Por otra parte, se observa que los centroides de los dos clústeres están más alejados en el casos de `Activity: 'Day'`, encontrándose aproximadamente a una distancia de 100 kilómetros. Destaca la separación entre los puntos 18 y 21 ya que, como se ha observado anteriormente, la bondad del ajuste era máxima cuando estas gaviotas se agrupaban en clústeres diferentes; esto nos permite definir este clúster de tres gaviotas como un grupo de gaviotas que son *outliers*.

Por último, en la Tabla 3.7 se ha incluido la bondad del ajuste de cada una de las cuatro matrices de distancias.

Global	**Sin actividad pesquera**	**Actividad nocturna**	**Actividad diurna**
0.45	0.56	0.73	0.77

Tabla 3.7: Bondad de ajuste de la aproximación bidimensional de la distancia de Hausdorff.

Se observa que las aproximaciones son bastante buenas en los dos casos en los que existe presencia de actividad pesquera. La matriz de distancias inter-gaviotas utilizando la totalidad de las trayectorias es la que tiene una bondad del ajuste peor, con un valor de 0.45.

Adicionalmente, en la Figura 3.14 incluida en la siguiente página, se han representado gráficamente las distancias de las matrices originales (las obtenidas con el cálculo de la distancia de Hausdorff) frente a su aproximación mediante en escalamiento multidimensional. La línea roja es la bisectriz, dónde deberían situarse todos los puntos en el caso en que el ajuste de la matriz de distancias fuera perfecto.

En general, se observa que las distancias se han subaproximado, es decir que en los gráficos

bidimensionales de la Figura 3.13, las distancias entre las gaviotas se muestran más cercanas a lo que en realidad deberían estar. Además, esto ocurre especialmente en las distancias cortas.

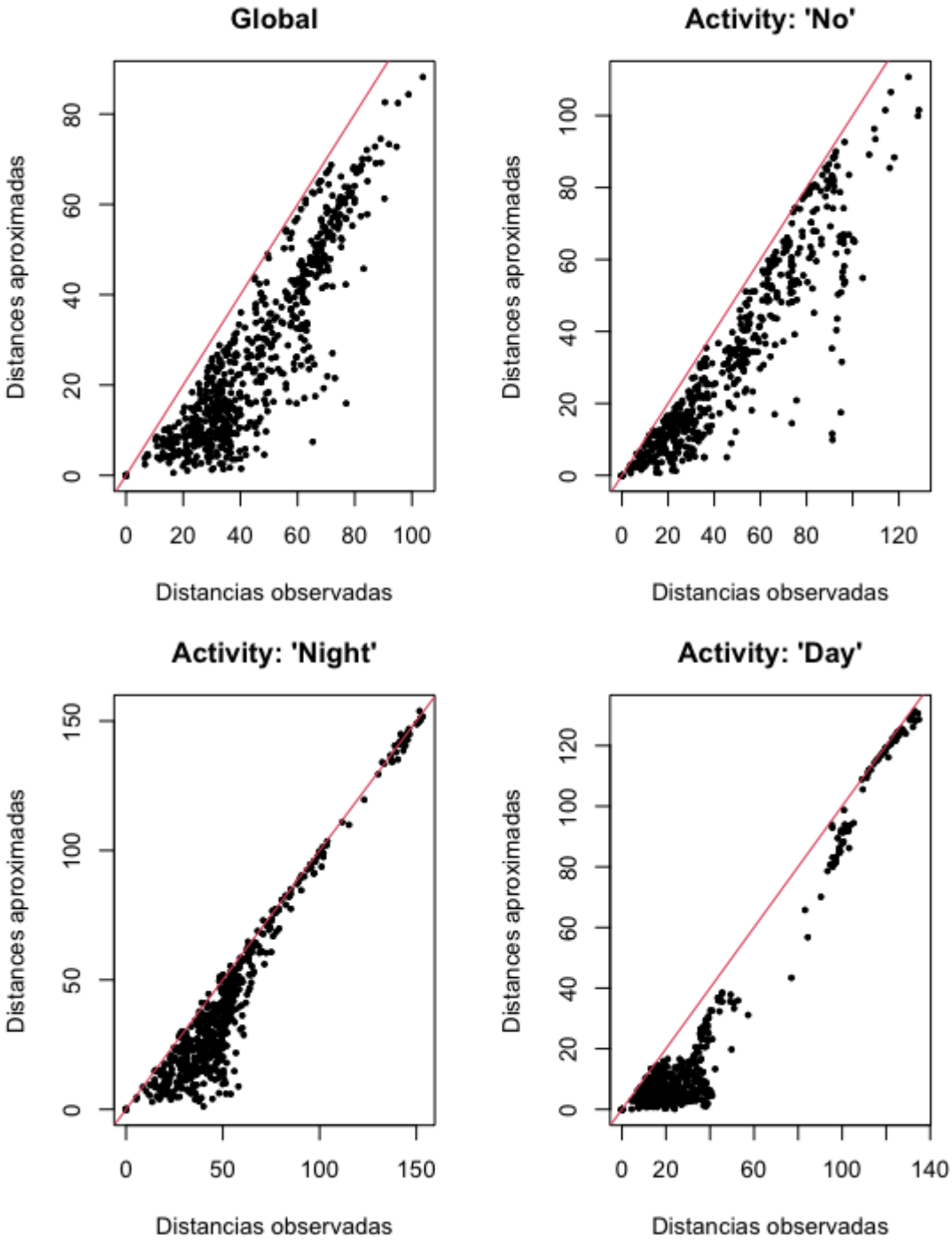

Figura 3.14: Distancias originales frente a distancias aproximadas con MDS métrico.

3.3.5. Concordancias entre clústeres

Para terminar con los resultados del análisis, se ha obtenido la matriz de concordancias entre los cuatro agrupamientos mediante el cálculo del índice de Rand ajustado (Tabla 3.8).

En el caso hipotético en que los tres índices de Rand de los datos condicionados por la actividad pesquera del momento (ausencia, presencia nocturna y presencia diurna) fueran cercanos a uno al compararlos con los datos globales, esto implicaría que la estructura detectada en los cuatro análisis sería la misma.

	Global	**No**	**Night**	**Day**
Global	1.000	-0.115	0.040	0.576
No	-0.115	1.000	0.338	-0.083
Night	0.040	0.338	1.000	-0.085
Day	0.576	-0.083	-0.085	1.000

Tabla 3.8: Índice de Rand Ajustado entre las agrupaciones establecidas.

Sin embargo, la concordancia de los tres agrupamientos con el caso global no ha sido alta. Por lo tanto, la segregación según el valor de la variable `activity` tiene sentido: la estructura de los grupos cambia según la actividad pesquera existente al iniciar la trayectoria.

Si bien la concordancia entre los clústeres de las gaviotas al no segregarlas según la actividad pesquera y bajo presencia actividad pesquera diurna es moderada (0.576), esto es únicamente debido a que dos de las tres gaviotas con comportamientos extremos en el caso diurno comparten clúster en el caso global.

Finalmente, no existe una alta concordancia entre los clústeres encontrados en los análisis segregados. Esto significa que la estructura de grupos de gaviotas en un determinado escenario de la actividad pesquera no tienen porqué ser la misma cuando el escenario es otro.

Capítulo 4

Discusión y conclusiones

En el inicio de este documento se ha enunciado el propósito principal de tratar de identificar subpoblaciones dentro de la población de gaviotas de Audouin del delta del Ebro mediante el análisis de las 362 trayectorias realizadas durante dos semanas por 36 gaviotas.

Primero, se ha calculado la distancia de Hausdorff entre las trayectorias para cuantificar la distancia entre la forma de las trayectorias de forma que distancias grandes implican formas diferentes y distancias cortas, similares.

A través de la implementación de dos pequeños cambios en el algoritmo *naive* para el cálculo de la distancia de Hausdorff exacto, se ha conseguido reducir el tiempo computacional un promedio de 11 horas y 19 minutos.

Luego, se han creado las matrices de distancias entre trayectorias en base a las diferentes condiciones de la actividad pesquera de la zona: ausencia de actividad, presencia diurna (pesca de arrastre) y presencia nocturna (pesca en redes de cerco). A continuación se han transformado las matrices inter-trayectorias a matrices de distancias inter-gaviotas, utilizando el promedio de las distancias de las trayectorias de cada gaviota.

Estas tres últimas matrices junto con la matriz inter-gaviotas de todas las trayectorias son las que se han utilizado para llevar a cabo el análisis clúster, el cual se ha realizado definiendo como criterio de enlace la distancia de mínima suma de cuadrados de Ward.

Los resultados de concordancias entre clústeres con el índice de Rand ajustado por azar indican que sí es necesario segregar el análisis en base a lo diferentes escenarios de actividad pesquera, ya que la estructura de grupos en los tres escenarios son diferentes en relación a la estructura de todas las trayectorias en conjunto. Esto concuerda con los resultados de Ouled-Cheikh y col. (2021), donde se concluyó que la mecánica del vuelo de las gaviotas de

Audouin cambiaba en función de la actividad pesquera.

En los casos de ausencia y presencia diurna de actividad pesquera se han separado las gaviotas en dos grupos mientras que en presencia de actividad nocturna se han determinado tres clústeres. Sin embargo, se ha visto que tanto en los datos de ausencia de actividad pesquera como en los de presencia nocturna, la bondad del ajuste mediante la media de los índices silueta se encontraba alrededor de 0.4. Dado que este valor no es elevado, es posible inferir que no hay una estructura de clústeres clara.

Con respecto a los datos en presencia de actividad pesquera nocturna y diurna, se han encontrado gaviotas con comportamientos extremos, es decir, con trayectorias muy diferentes a las todas las demás. Concretamente, la gaviota 35 es un *outlier* en presencia de pesca nocturna, aunque esto puede ser debido a que únicamente existe una trayectoria de esta gaviota en este momento. Bajo actividad pesquera diurna, se detecta un clúster de tres gaviotas con trayectorias claramente diferentes a las de las demás.

De todos modos, en todos los casos se han determinado los clústeres cortando los dendrogramas a una distancia cofenética de 150 kilómetros. Se trata de una distancia realmente extrema teniendo en cuenta que el 75 % de las distancias inter-trayectorias se encontraban a 55.53 kilómetros o menos.

4.1. Consideraciones metodológicas

Los resultados obtenidos ponen en manifiesto la limitación de la distancia de Hausdorff calculada. Las distancias de Hausdorff dirigidas se han calculado almacenando el máximo de todas las distancias mínimas entre cada pareja de puntos. Pero, como se ha comentado en el Capítulo 2, la medida es poco robusta con respeto a puntos periféricos de las trayectorias (i.e. valores extremos).

En la siguiente página se incluye un gráfico de caja con la distribución de los valores de las distancias. Los puntos verdes son las distancias consideradas dentro de la normalidad mientras que los puntos negros son valores extremos (aproximadamente 2800 de un total de 131044 distancias entre trayectorias).

En consecuencia, una mejora consistiría en utilizar la distancia de Hausdorff extendida (Min y col., 2007), con la que se guardaría el valor de la mediana o de otro cuantil determinado en lugar del valor máximo. En tal caso, sería necesario modificar el algoritmo 3.3 para que las distancias de Hausdorff mínimas, `cmin` se guardara, ordenara y se escogiera el cuantil deseado.

Por otra parte, la determinación de los clústeres en este trabajo se ha hecho optimizando la media de los índices silueta. Un método alternativo sería considerar un valor de distancia a partir del cuál se considere que la distancia entre dos gaviotas es grande y, por lo tanto, no sea posible que estas pertenezcan al mismo clúster.

Para terminar, una aplicación de los resultados sería la utilización de los clústeres encontrados para determinar qué fenotipos de las gaviotas son los que diferencias estos clústeres. Además, sería interesante extender el trabajo de Cortejana Retamozo (2020) para determinar si características de las trayectorias como su sinuosidad, duración o velocidad media están relacionadas con los clústeres.

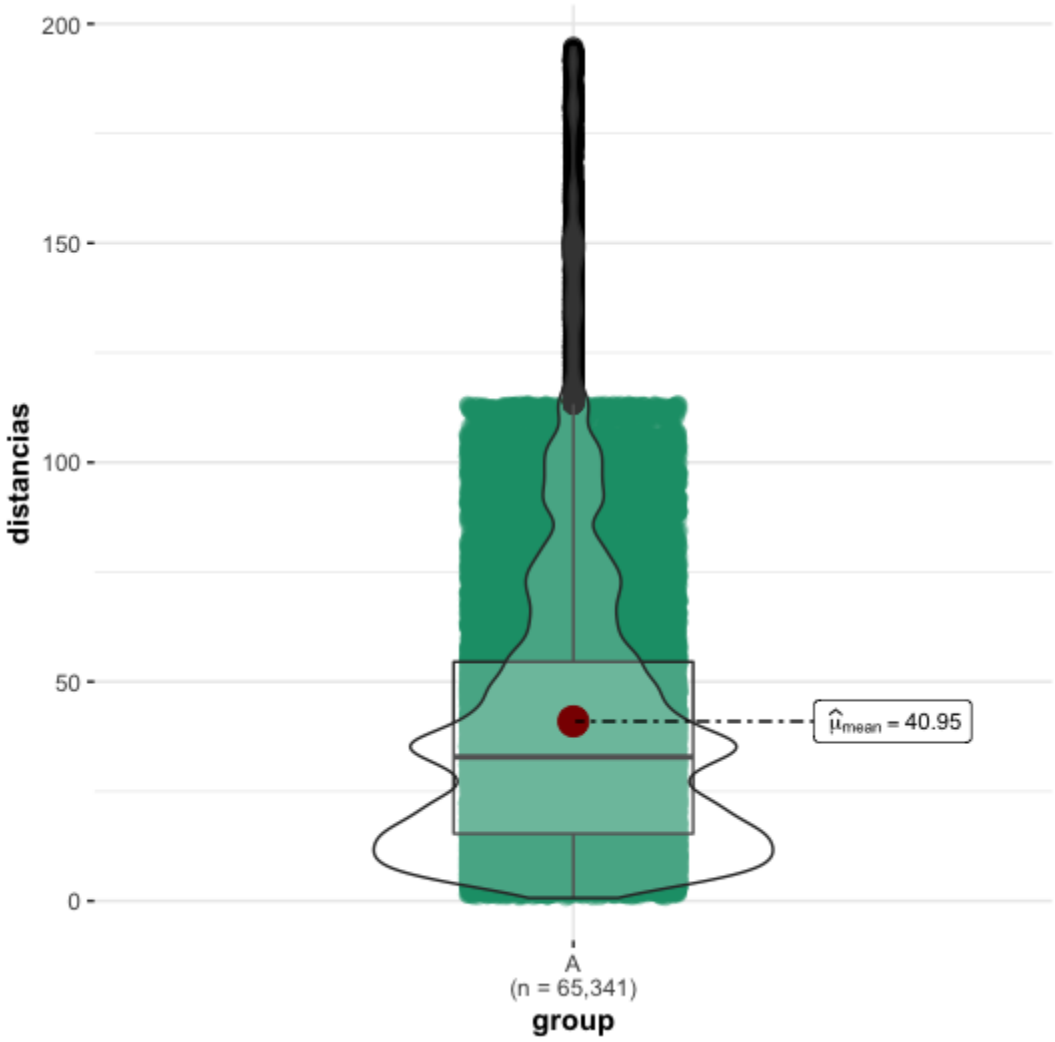

Figura 4.1: *Boxplot* y distribución de las distancias de Hausdorff entre las trayectorias.

Referencias

BirdLife International. (2021). Species factsheet: Larus audouinii [http://www.birdlife.org Última vez visitado el 2 de diciembre de 2021].

Charrad, M., Ghazzali, N., Boiteau, V. & Niknafs, A. (2014). NbClust: An R Package for Determining the Relevant Number of Clusters in a Data Set. *Journal of Statistical Software*, *61*(6), 1-36. https://doi.org/10.18637/jss.v061.i06

Cortejana Retamozo, A. (2020). *Caracterización y repetibilidad de la mecánica de las trayectorias de vuelo del Larus Audouinii* (Trabajo Final de Máster). Universitat Politècnica de Catalunya. Barcelona.

Cuadras, C. M. (1988). Distancias Estadísticas. *Estadística Española*, *30*(119), 295-378. https://doi.org/10.3354/meps12217

Everitt, B. & Hothorn, T. (2011). *An introduction to applied multivariate analysis with R.* New York, Springer. http://dx.doi.org/10.1007/978-1-4419-9650-3

Gordon, A. (1999). *Classification, 2nd Edition.* CRC Press. https://books.google.es/books?id=%5C_w5AJtbfEz4C

Graffelman, J., Salicrú, M. & Reverter, F. (2021). Apuntes de la asignatura de Análisis Multivariante del Máster en Estadística e Investigación Operativa (UPC).

Hausdorff, F. (1914). *Gründzge der Mengenlehre.* Leipzig Viet.

Hijmans, R. J. (2021). *geosphere: Spherical Trigonometry* [R package version 1.5-14]. R package version 1.5-14. https://CRAN.R-project.org/package=geosphere

Hubert, L. J. & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, *2*, 193-218.

IUCN. (2021). The IUCN Red List of Threatened Species. Version 2021-2. [https://www.iucnredlist.org. Descargado el 2/02/2021].

Kassambara, A. (2020). *ggpubr: 'ggplot2' Based Publication Ready Plots* [R package version 0.4.0.999]. R package version 0.4.0.999. https://rpkgs.datanovia.com/ggpubr/

Kassambara, A. & Mundt, F. (2020). *factoextra: Extract and Visualize the Results of Multivariate Data Analyses* [R package version 1.0.7]. R package version 1.0.7. https://CRAN.R-project.org/package=factoextra

Magdy, N., Sakr, M. A., Mostafa, T. & El-Bahnasy, K. (2015). Review on trajectory similarity measures, En *2015 IEEE Seventh International Conference on Intelligent*

Computing and Information Systems (ICICIS). https://doi.org/10.1109/IntelCIS.2015.7397286

Mersmann, O. (2021). *microbenchmark: Accurate Timing Functions* [R package version 1.4-2.11]. R package version 1.4-2.11. https://github.com/olafmersmann/microbenchmark/

Min, D., Zhilin, L. & Xiaoyong, C. (2007). Extended Hausdorff distance for spatial objects in GIS. *International Journal of Geographical Information Science*, *21*(4), https://doi.org/10.1080/13658810601073315, 459-475. https://doi.org/10.1080/13658810601073315

Murtagh, F. & Legendre, P. (2014). Ward's Hierarchical Agglomerative Clustering Method: Which Algorithms Implement Ward's Criterion? *Journal of Classification*, *31*(3), 274-295. https://doi.org/10.1007/s00357-014-9161-z

Ouled-Cheikh, J., Ramírez, F., Sánchez-Fortún, M., Cortejana, A., Sanpera, C. & Carrasco, J. L. (2021). *Fishing activities shape the flight behaviour of an opportunistic predator species* [Manuscrito enviado para publicación]. Manuscrito enviado para publicación.

Ouled-Cheikh, J., Sanpera, C., Bécares, J., Arcos, J. M., Carrasco, J. L. & Ramírez, F. (2020). Spatiotemporal analyses of tracking data reveal fine-scale, daily cycles in seabird–fisheries interactions. *ICES Journal of Marine Science*, *77*(7-8), 2508-2517. https://doi.org/10.1093/icesjms/fsaa098

Phillips, R. A., Lewis, S., González-Solís, J. & Daunt, F. (2017). Causes and consequences of individual variability and specialization in foraging and migration strategies of seabirds. *Marine Ecology Progress Series*, *578*, 117-150. https://doi.org/10.3354/meps12217

R Core Team. (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria. https://www.R-project.org/

Rand, W. M. (1971). Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, *66*(336), 846-850. https://doi.org/10.1080/01621459.1971.10482356

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, *20*, 53-65. https://doi.org/https://doi.org/10.1016/0377-0427(87)90125-7

Scrucca, L., Fop, M., Murphy, T. B. & Raftery, A. E. (2016). mclust 5: clustering, classification and density estimation using Gaussian finite mixture models. *The R Journal*, *8*(1), 289-317. https://doi.org/10.32614/RJ-2016-021

SEO/BirdLife. (2008). Adaptación de La enciclopedia de las aves de España, publicada por SEO/BirdLife y la Fundación BBVA [Disponible en https://seo.org/listado-aves-2/ Última vez visitado el 2 de diciembre de 2021].

Sinnott, R. (1984). Virtues of the Haversine. *skytel*, *68*(2), 158.

Taha, A. A. & Hanbury, A. (2015). An Efficient Algorithm for Calculating the Exact Hausdorff Distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. http://dx.doi.org/10.1109/TPAMI.2015.2408351

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org

Apéndice A

Código R

A.1. Paquetes necesarios

```
library(readxl)
library(dplyr)
library(lubridate)
library(papeR)
library(microbenchmark)
library(data.table)
library(ggplot2)
library(scales)
library(geosphere)
library(cluster)
library(factoextra)
library(gridExtra)
library(dendextend)
library(MASS)
library(ggpubr)
library(NbClust)
library(mclust)
library(ggstatsplot)
```

A.2. Lectura de datos y *preprocessing*.

```
## 1.1. Lectura de la base de datos.
# ================================
dades <- read_excel("dades_audouin_jazel.xls")

## 1.2. Transformación de los datos.
# ================================

# Selección de  variables de interés, cambio de nombre y creación de seis nuevas
# variables: daytime, diaset, type_day (weekend or working), ini_time, triptime
# (tiempo de vuelo en segundos) y maxd (distancia máxima a la colonia).

dades <- dades %>%
  select(ANILLA, fechaOfici, horaOficia, idViaje1,
         LAT, LONG, DistNido_m, DiaSem) %>%
  rename(ID = ANILLA, day = fechaOfici, time = horaOficia, trip = idViaje1,
         dcolony = DistNido_m) %>%
  mutate(daytime = as.POSIXct(paste(day, time),
                              format = "%Y-%m-%d %H:%M:%S"))  %>%
  mutate(DiaSem = factor(DiaSem,
                         levels = c("lun","mar","mie","jue","vie","sab","dom")),
         type_day = if_else(DiaSem %in% c("sab","dom"),
                            "weekend","working")) %>%
  group_by(ID, trip) %>%
  arrange(ID, trip, daytime) %>%
  mutate(ini_time = min(daytime),
         triptime = as.numeric(daytime - ini_time),
         maxd = max(dcolony))

# Creación de las variables init_hour (0h-23h), moment ("0-10","10-18","18-0"),
# weekday_salida y activity:
dades$init_hour <- hour(dades$ini_time)

dades$moment <- ifelse((dades$init_hour > 0 & dades$init_hour <= 10), "0-10",
                       ifelse((dades$init_hour > 10 & dades$init_hour <= 18),
                              "10-18", "18-0"))
dades$moment <- factor(dades$moment,levels=c("0-10","10-18","18-0"),
                       ordered = TRUE)
contrasts(dades$moment) <- contr.treatment(3)
```

```
dades$weekday_salida <- weekdays(dades$day)

dades <- dades %>% mutate(activity = ifelse(type_day == "weekend", "No",
                                            ifelse(moment == "18-0", "No",
                                                   ifelse(moment == "0-10",
                                                          "Night", "Day"))))
Sys.setlocale("LC_TIME", "English")
dades$weekday_salida<-weekdays(dades$day)

dades <- dades %>%
  mutate(activity = ifelse((weekday_salida=="Monday") & (moment=="0-10"), "No",
                           activity))
dades$activity <- factor(dades$activity, levels = c("No", "Night", "Day"))
contrasts(dades$activity) <- contr.treatment(3)

# Arreglo viaje gaviota 5107942
dades$trip[which(dades$ID == "5107942" & dades$trip =="V1b")] <- "V01"

# Nos quedamos con las 7 variables de interés
data <- dades[ ,c(1,4,3,5,6,17)]

# Creamos variables bird y trajectory
data$ID_trip <- paste(data$ID, data$trip, sep="")
data <- transform(data, bird = as.numeric(factor(ID)))
data <- transform(data, trajectory = as.numeric(factor(ID_trip)))
data <- data[,c(7,8,9,4,5,3,6)]

# Cogemos como valor de activity el momento de inicio de la trayectoria:
for(i in unique(data$trajectory)){
  data$activity[which(data$trajectory == i)] <-
    data$activity[min(which(data$trajectory == i))]
}

# Convertimos variables a factor
data$activity <- as.factor(data$activity)
str(data)
```

A.3. Análisis descriptivo

```
# 3.1. Muestra de la tabla de datos:
# ==================================
head(data)

# 3.2. Resumen numérico:
# ======================
# Número de gaviotas por activity
xx <- split(data,data$activity)
N <- c(length(unique(xx$Day$bird)),
       length(unique(xx$Night$bird)),
       length(unique(xx$No$bird)))

df2 <- data.frame(cbind("id" = unique(data$trajectory),
                        "activity"= rep(0,362)))

for(i in unique(data$trajectory)){
  df2$activity[i] <-
    as.character(unique(data$activity[which(data$trajectory == i)]))
}
df2$activity <- as.factor(df2$activity)

tabla <-papeR::summarize(df2, type ="factor")
tabla <- cbind(tabla, N)
colnames(tabla) <- c(" ",
                     "Level",
                     " ",
                     "Nº trayectorias",
                     "% trayectorias","Nºgaviotas")
tabla

# 3.3. Gráficos:
# ==============
# Pie chart de activity:
df <- as.data.frame(table(df2$activity))
df %>% ggplot(aes(x = "", y = Freq, fill = Var1)) + geom_col() +
  geom_text(aes(label = paste(round(prop.table(table(df2$activity))*100, 2),
                              "%", sep = "")),
            position = position_stack(vjust = 0.5)) +
```

```
  scale_fill_brewer(palette = "Blues") + coord_polar("y") +
  theme_void() + labs(fill = "Actividad")

# Histograma del número de trayectorias por ave
data$count<- as.numeric(substr(as.character(data$ID_trip),9,10))
df <- as.data.frame(cbind("bird" = 1:36, "trips"= rep(0,36)))
df$bird <- as.factor(df$bird)

for(i in unique(data$bird)){
  df$trips[i] <- max(data$count[data$bird == i])
}

ggplot(df, aes(x = bird, y = trips)) +
  geom_bar(stat = "identity", fill = "steelblue") + theme_minimal() +
  geom_abline(slope = 0, intercept = mean(df$trips),  col = "red", lty = 2) +
  geom_text(aes(label = ifelse(trips > 22, trips,
                               ifelse(trips < 3, trips, ""))),
       vjust = -0.3, size = 2.8, color = "Red") +
  labs(y = "Nº de viajes",
       x = "Gaviota",
       title = "Número de trayectorias por gaviota") +
  annotate("text", 35.7, mean(df$trips), vjust = -1, label = "Media",
           col = "red", size = 2.9)
```

A.4. Algoritmo para la distancia de Hausdorff dirigida.

```
### ALGORITMO 1: Calcula directamente la distancia de Hausdorff dirigida
Hausdorff_1 <- function(data){
        # Se guardan las 65703 combinaciones de 362 trayectorias tomadas de 2 en 2:
        combinaciones <- combn(1:max(data$trajectory),2)
        HD_solution <- matrix(nrow = max(data$trajectory),
        ncol = max(data$trajectory))
        diag(HD_solution) <- 0

        # Visitamos cada par de trayectorias:
        for(t in 1:ncol(combinaciones)){
                A <- combinaciones[,t][1]
                B <- combinaciones[,t][2]
                cmax <- 0

                # Visitamos cada punto de A
                for(i in which(data$trajectory==A)){
                        cmin <- Inf

                        # Visitamos cada punto de B
                        for(j in which(data$trajectory==B)){
                                # Calculamos distancia Haversine entre i i j:
                                d <- distHaversine(c(data$LONG[i], data$LAT[i]),
                                c(data$LONG[j], data$LAT[j]))
                                if(d < cmin){
                                        cmin <- d
                                }
                        }
                        if(cmin > cmax){
                                cmax <- cmin
                        }
                }
                HD_solution[A,B] <- HD_solution[B,A] <- cmax
        }
        return(HD_solution)
}
```

```
### ALGORITMO 2: Calcula la distancia de Hausdorff dirigida utilizando
###              la técnica de pausa temprana

Hausdorff_2 <- function(data){
        combinaciones <- combn(1:max(data$trajectory),2)
        HD_solution <- matrix(nrow = max(data$trajectory),
        ncol = max(data$trajectory))
        diag(HD_solution) <- 0

        for(t in 1:ncol(combinaciones)){
                A <- combinaciones[,t][1]
                B <- combinaciones[,t][2]
                cmax <- 0

                for(i in which(data$trajectory==A)){
                        cmin <- Inf
                        for(j in which(data$trajectory==B)){
                                d <- distHaversine(c(data$LONG[i], data$LAT[i]),
                                c(data$LONG[j], data$LAT[j]))
                                # Parada temprana:
                                if(d < cmax){
                                        if(d < cmin){cmin <- d}
                                        break
                                }
                                if(d < cmin){
                                        cmin <- d
                                }
                        }

                        # Si el bucle se rompe, cmin vale INF, lo corregimos:
                        if(cmin > cmax && cmin != Inf){
                                cmax <- cmin
                        }
                }
                HD_solution[A,B] <- HD_solution[B,A] <- cmax
        }
        return(HD_solution)
}
```

```
### ALGORITMO 3: Calcula la distancia de Hausdorff dirigida utilizando
###              la técnica de pausa temprana y el muestreo aleatorio
Hausdorff_3 <- function(data){
        combinaciones <- combn(1:max(data$trajectory),2)
        HD_solution <- matrix(nrow = max(data$trajectory),
        ncol = max(data$trajectory))
        diag(HD_solution) <- 0

        for(t in 1:ncol(combinaciones)){
                A <- combinaciones[,t][1]
                B <- combinaciones[,t][2]
                cmax <- 0

                # Visitamos los puntos de A en orden aleatorio
                for(i in sample(which(data$trajectory==A))){
                        cmin <- Inf

                        # Visitamos los puntos de B en orden aleatorio
                        for(j in sample(which(data$trajectory==B))){
                                d <- distHaversine(c(data$LONG[i], data$LAT[i]),
                                c(data$LONG[j], data$LAT[j]))
                                if(d < cmax){
                                        if(d < cmin){cmin <- d}
                                        break
                                }
                                if(d < cmin){
                                        cmin <- d
                                }
                        }
                        if(cmin > cmax && cmin != Inf){
                                cmax <- cmin
                        }
                }
                HD_solution[A,B] <- HD_solution[B,A] <- cmax
        }
        return(HD_solution)
}
```

```
### Comparación de los tiempos computacionales.
### ========================================

# CPU: Intel(R) Core(TM) i9-9900 3.10GHz i 32 GB de RAM.

tiempo <- microbenchmark("Algoritmo 1"= {Hausdorff_1(data)},
                         "Algoritmo 2"= {Hausdorff_2(data)},
                         "Algoritmo 3"= {Hausdorff_3(data)},
                         times = 10L)
print(tiempo)
summary(tiempo)
boxplot(tiempo, log = FALSE)
ggplot2::autoplot(tiempo, log = FALSE)

# Gráfico en horas:
temps <- tiempo
temps$time <- (tiempo$time*1.7e-11)/60

ggplot(temps, aes(x=expr, y=time, color=expr)) +
  geom_violin() +
  coord_flip() +
  labs(y = "Tiempo [horas]", x="") +
  theme_bw() +
  scale_color_brewer(palette="Dark2") +
  theme(legend.position="none")
```

A.5. Análisis clúster.

```
# Algoritmo para AGGLOMERATIVE Hierarchical clustering:
# 1. Calcular la matriz de distancia de Hasudorff entre las trayectorias (363x363)
# 2. Crear un clúster por cada gaviota, incluyendo todos los viajes que ha realizado
# 3. Actualizar la matriz de distancias (ahora matriz de distancias inter-clusters)
#    mediante los linkages "average" y "ward".
# 4. REPETIR
# 5.       Fusionar los dos clústers más cercanos
# 6.       Actualizar la matriz de distancias
# 7. HASTA que solo quede un solo clúster

## PASO 1. Cálculo de la matriz de distancias de Hausdorff.
## ========================================================
Hausdorff_BA <- function(data){
        combinaciones <- combn(1:max(data$trajectory),2)
        HD_solution <- matrix(nrow = max(data$trajectory),
        ncol = max(data$trajectory))
        diag(HD_solution) <- 0

        for(t in 1:ncol(combinaciones)){
                A <- combinaciones[,t][1]
                B <- combinaciones[,t][2]
                cmax <- 0

                # Visitamos los puntos de B en orden aleatorio
                for(i in sample(which(data$trajectory==B))){
                        cmin <- Inf

                        # Visitamos los puntos de A en orden aleatorio
                        for(j in sample(which(data$trajectory==A))){
                                d <- distHaversine(c(data$LONG[i], data$LAT[i]),
                                c(data$LONG[j], data$LAT[j]))
                                if(d < cmax){
                                        if(d < cmin){
                                                cmin <- d
                                        }
                                        break
                                }
                                if(d < cmin){
```

```
                                        cmin <- d
                                }
                        }
                        if(cmin > cmax && cmin != Inf){
                                cmax <- cmin
                        }
                }
                HD_solution[A,B] <- HD_solution[B,A] <- cmax
        }
        return(HD_solution)
}


Hdist <- pmax(Hausdorff_3(data), Hausdorff_BA(data))
colnames(Hdist) <- unique(data$trajectory)
rownames(Hdist) <- unique(data$trajectory)
Hdist <- Hdist/1000 # pasamos la distancia a kilómetros

#### Descripriva numérica de la matri de distancias
dd <- Hdist[lower.tri(Hdist)]
summary(dd)
hist(dd)

## PASO 2. Crear matriz de distancias inter-gaviotas.
## ================================================
combinaciones <- combn(1:max(data$bird),2)
avg <-  matrix(nrow = max(data$bird), ncol = max(data$bird))
diag(avg) <- 0
avg_No <- avg; avg_Night <- avg; avg_Day <- avg

for(t in 1:ncol(combinaciones)){
        A <- combinaciones[,t][1]
        B <- combinaciones[,t][2]
        d <- Hdist[unique(data$trajectory[which(data$bird==A)]),
             unique(data$trajectory[which(data$bird==B)])]

        d_No <- Hdist[unique(data$trajectory[which(data$bird==A &
                                                   data$activity=="No")]),
             unique(data$trajectory[which(data$bird==B &
```

```
                                        data$activity=="No")])]

        d_Night <- Hdist[unique(data$trajectory[which(data$bird==A &
                                                  data$activity=="Night")]),
             unique(data$trajectory[which(data$bird==B &
                                        data$activity=="Night")])]

        d_Day <- Hdist[unique(data$trajectory[which(data$bird==A &
                                                data$activity=="Day")]),
             unique(data$trajectory[which(data$bird==B &
                                        data$activity=="Day")])]

        avg[A, B] <- avg[B, A] <- mean(d)
        avg_No[A, B] <- avg_No[B, A] <- mean(d_No)
        avg_Night[A, B] <- avg_Night[B, A] <- mean(d_Night)
        avg_Day[A, B] <- avg_Day[B, A] <- mean(d_Day)
}

rownames(avg) <- rownames(avg_No) <- rownames(avg_Night) <- rownames(avg_Day) <- 1:36
colnames(avg) <- colnames(avg_No) <- colnames(avg_Night) <- colnames(avg_Day) <- 1:36

avg_No <- avg_No[complete.cases(avg_No[,1]),
                 complete.cases(avg_No[,1])]

avg_Night <- avg_Night[complete.cases(avg_Night[,1]),
                       complete.cases(avg_Night[,1])]

avg_Day  <- avg_Day[complete.cases(avg_Day[,1]),
                    complete.cases(avg_Day[,1])]

# 2.1. Representaciones gráficas (heatmap) de matrices de distancia.
p1 <- fviz_dist(dist.obj = as.dist(avg), lab_size = 5) +
  theme(legend.position = "none") +
  labs(title = "Global")

p2 <- fviz_dist(dist.obj = as.dist(avg_No), lab_size = 5) +
  theme(legend.position = "none") +
  labs(title = "Activity: 'No'")
```

```
p3 <- fviz_dist(dist.obj = as.dist(avg_Night), lab_size = 5) +
  theme(legend.position = "none") +
  labs(title = "Activity: 'Night'")

p4 <- fviz_dist(dist.obj = as.dist(avg_Day), lab_size = 5) +
  theme(legend.position = "none") +
  labs(title = "Activity: 'Day'")

grid.arrange(p1, p2, p3, p4, ncol=2)


## PASO 3. Llevar a cabo el clustering aglomerativo con hclust().
## ============================================================
cluster.ward <- hclust(as.dist(avg), method= "ward.D2")
cluster.ward_No <- hclust(as.dist(avg_No), method= "ward.D2")
cluster.ward_Night <- hclust(as.dist(avg_Night), method= "ward.D2")
cluster.ward_Day <- hclust(as.dist(avg_Day), method= "ward.D2")


# 3.1. Representar los dendogramas

par(mfrow=c(2,2), mar =c(1, 4, 3, 2)+0.1)
plot(cluster.ward, cex = 0.6, main = "Global", ylab = "Distancia")
plot(cluster.ward_No, cex = 0.6, main = "Activity: 'No'", ylab = "Distancia")
plot(cluster.ward_Night, cex = 0.6, main = "Activity: 'Night'", ylab = "Distancia")
plot(cluster.ward_Day, cex = 0.6, main = "Activity: 'Day'", ylab = "Distancia")


## PASO 4. Verificar los dendogramas.
## ==================================
# Coeficiente de Correlación Cofenética: Es una medida de la fiabilidad
# con la que un dendrograma conserva las distancias entre los pares de
# puntos de los datos originales.

cor.coph <- as.data.frame(matrix(ncol=4, nrow = 1))
colnames(cor.coph) <- c("Global", "No", "Nigth", "Day")
cor.coph[1,1] <- round(cor(cophenetic(cluster.ward), as.dist(avg))^2,2)
cor.coph[1,2] <- round(cor(cophenetic(cluster.ward_No), as.dist(avg_No))^2,2)
cor.coph[1,3] <- round(cor(cophenetic(cluster.ward_Night), as.dist(avg_Night))^2,2)
```

```
cor.coph[1,4] <- round(cor(cophenetic(cluster.ward_Day), as.dist(avg_Day))^2,2)

cor.coph

## PASO 5. Calcular el número óptimo de clusters.
## ==============================================
# El método de average silhouette: se maximiza la media de los índices silueta.
# Su valor puede estar entre -1 y 1, siendo valores altos un indicativo de que
# la observación se ha asignado al cluster correcto.

numero_clusters.ward <- NbClust(diss = as.dist(avg), distance=NULL,
                                min.nc = 2, max.nc = 10,
                                method = "ward.D2",
                                index = "silhouette")

numero_clusters.ward_No <- NbClust(diss = as.dist(avg_No), distance=NULL,
                                   min.nc = 2, max.nc = 10,
                                   method = "ward.D2",
                                   index = "silhouette")

numero_clusters.ward_Night <- NbClust(diss = as.dist(avg_Night), distance=NULL,
                                   min.nc = 2, max.nc = 10,
                                   method = "ward.D2",
                                   index = "silhouette")

numero_clusters.ward_Day <- NbClust(diss = as.dist(avg_Day), distance=NULL,
                                      min.nc = 2, max.nc = 10,
                                      method = "ward.D2",
                                      index = "silhouette")

p1 <- data.frame(n_clusters = 1:10,
                 media_silhouette = c(0,numero_clusters.ward$All.index)) %>%
  ggplot(aes(x = n_clusters, y = media_silhouette)) +
  geom_line(color="dodgerblue3") +
  geom_point(color="dodgerblue3") +
  theme_light() +
  scale_x_continuous(breaks = 1:10) +
  labs(title = "Global",
       x="Número de clusters",
```

```
       y="Media del coeficiente Silhouette") +
  geom_segment(aes(x=as.numeric(numero_clusters.ward$Best.nc[1]),
                   xend= as.numeric(numero_clusters.ward$Best.nc[1]),
                   y=0,
                   yend=as.numeric(numero_clusters.ward$Best.nc[2])),
                 colour= "dodgerblue3",
                 linetype ="dashed")

p2 <- data.frame(n_clusters = 1:10,
                 media_silhouette = c(0,numero_clusters.ward_No$All.index)) %>%
  ggplot(aes(x = n_clusters, y = media_silhouette)) +
  geom_line(color="dodgerblue3") +
  geom_point(color="dodgerblue3") +
  theme_light() +
  scale_x_continuous(breaks = 1:10) +
  labs(title = "Activity: 'No'",
       x="Número de clusters",
       y="Media del coeficiente Silhouette") +
  geom_segment(aes(x=as.numeric(numero_clusters.ward_No$Best.nc[1]),
                   xend= as.numeric(numero_clusters.ward_No$Best.nc[1]),
                   y=0,
                   yend=as.numeric(numero_clusters.ward_No$Best.nc[2])),
                 colour= "dodgerblue3",
                 linetype ="dashed")

p3 <- data.frame(n_clusters = 1:10,
                 media_silhouette = c(0,numero_clusters.ward_Night$All.index)) %>%
  ggplot(aes(x = n_clusters, y = media_silhouette)) +
  geom_line(color="dodgerblue3") +
  geom_point(color="dodgerblue3") +
  theme_light() + scale_x_continuous(breaks = 1:10) +
  labs(title = "Activity: 'Night'",
       x="Número de clusters",
       y="Media del coeficiente Silhouette") +
  geom_segment(aes(x=as.numeric(numero_clusters.ward_Night$Best.nc[1]),
                   xend= as.numeric(numero_clusters.ward_Night$Best.nc[1]),
                   y=0,
                   yend=as.numeric(numero_clusters.ward_Night$Best.nc[2])),
                 colour= "dodgerblue3",
```

```
                linetype ="dashed")

p4 <- data.frame(n_clusters = 1:10,
                media_silhouette = c(0,numero_clusters.ward_Day$All.index)) %>%
  ggplot(aes(x = n_clusters, y = media_silhouette)) +
  geom_line(color="dodgerblue3") +
  geom_point(color="dodgerblue3") +
  theme_light() +
  scale_x_continuous(breaks = 1:10) +
  labs(title = "Activity: 'Day'",
       x="Número de clusters",
       y="Media del coeficiente Silhouette") +
  geom_segment(aes(x=as.numeric(numero_clusters.ward_Day$Best.nc[1]),
                   xend= as.numeric(numero_clusters.ward_Day$Best.nc[1]),
                   y=0,
                   yend=as.numeric(numero_clusters.ward_Day$Best.nc[2])),
                colour= "dodgerblue3",
                linetype ="dashed")

grid.arrange(p1, p2, p3, p4, ncol = 2)

## PASO 6. Cortar el dendograma para generar los clústers.
## =======================================================
hc.ward <- cutree(cluster.ward, k=2)
hc.ward_No <- cutree(cluster.ward_No, k=2)
hc.ward_Night <- cutree(cluster.ward_Night, k=3)
hc.ward_Day <- cutree(cluster.ward_Day, k=2)

p1 <- fviz_dend(x = cluster.ward, k = 2, cex = 0.6,
                k_colors = c("#1B9E77", "#D95F02")) +
  geom_hline(yintercept = 125, linetype = "dashed") +
  labs(title = "Global", y = "Distancia") +
  theme(axis.text.x = element_text(size = 0.1))

p2 <- fviz_dend(x = cluster.ward_No, k = 2, cex = 0.6,
                k_colors = c("#1B9E77", "#D95F02")) +
  geom_hline(yintercept = 150, linetype = "dashed") +
  labs(title = "Activity: 'No'", y = "Distancia") +
  theme(axis.text.x = element_text(size = 0.1))
```

```
p3 <- fviz_dend(x = cluster.ward_Night, k = 3, cex = 0.6,
                k_colors = c("#7570B3","#1B9E77", "#D95F02")) +
  geom_hline(yintercept = 150, linetype = "dashed") +
  labs(title = "Activity: 'Night'", y = "Distancia") +
  theme(axis.text.x = element_text(size = 0.1))

p4 <- fviz_dend(x = cluster.ward_Day, k = 2, cex = 0.6,
                k_colors = c("#1B9E77","#D95F02")) +
  geom_hline(yintercept = 150, linetype = "dashed") +
  labs(title = "Activity: 'Day'", y = "Distancia") +
  theme(axis.text.x = element_text(size = 0.1))

grid.arrange(p1, p2, p3, p4, ncol = 2)


## PASO 7. Representación 2D con metric Multidimensional Scaling.
## ==============================================================
mds_ward <- cmdscale(avg, eig=TRUE)
mds_ward.df <- as.data.frame(mds_ward$points)
colnames(mds_ward.df) <- c("Dim.1", "Dim.2")
mds_ward.df$groups <- as.factor(hc.ward)

mds_ward_No <- cmdscale(avg_No, eig=TRUE)
mds_ward.df_No <- as.data.frame(mds_ward_No$points)
colnames(mds_ward.df_No) <- c("Dim.1", "Dim.2")
mds_ward.df_No$groups <- as.factor(hc.ward_No)

mds_ward_Night <- cmdscale(avg_Night, eig=TRUE)
mds_ward.df_Night <- as.data.frame(mds_ward_Night$points)
colnames(mds_ward.df_Night) <- c("Dim.1", "Dim.2")
mds_ward.df_Night$groups <- as.factor(hc.ward_Night)

mds_ward_Day <- cmdscale(avg_Day, eig=TRUE)
mds_ward.df_Day <- as.data.frame(mds_ward_Day$points)
colnames(mds_ward.df_Day) <- c("Dim.1", "Dim.2")
mds_ward.df_Day$groups <- as.factor(hc.ward_Day)

options(ggrepel.max.overlaps = 15)
```

```
xlab.p1 <- round(mds_ward$eig[1]/sum(mds_ward$eig[which(mds_ward$eig>0)]),4)*100
ylab.p1 <- round(mds_ward$eig[2]/sum(mds_ward$eig[which(mds_ward$eig>0)]),4)*100

p1 <- ggpar(ggscatter(mds_ward.df, x = "Dim.1",  y = "Dim.2",
                      label = rownames(mds_ward.df),
                      shape = "groups", color = "groups",
                      palette =  c("#D95F02","#1B9E77"),
                      size = 1, ellipse = TRUE, mean.point = T,
                      ellipse.type = "convex", repel = TRUE,
                      title="Global",mean.point.size=3.3,
                      show.legend.text = F),
            legend = "none") +
  labs(x = paste0("Dim 1 (", xlab.p1, "%)"),
       y = paste0("Dim 2 (", ylab.p1, "%)")) +
  xlim(-110, 100) + ylim(-45, 78)

xlab.p2 <- round(mds_ward_No$eig[1]/
                   sum(mds_ward_No$eig[which(mds_ward_No$eig>0)]),4)*100
ylab.p2 <- round(mds_ward_No$eig[2]/
                   sum(mds_ward_No$eig[which(mds_ward_No$eig>0)]),4)*100

p2 <- ggpar(ggscatter(mds_ward.df_No, x = "Dim.1",  y = "Dim.2",
                      label = rownames(mds_ward.df_No),
                      shape = "groups", color = "groups",
                      palette = c("#D95F02","#1B9E77"),
                      size = 1, ellipse = TRUE, mean.point = T,
                      ellipse.type = "convex", repel = TRUE,
                      title="Activity: 'No'", mean.point.size=3.3),
            legend = "none") +
  labs(x = paste0("Dim 1 (", xlab.p2, "%)"),
       y = paste0("Dim 2 (", ylab.p2, "%)")) +
  xlim(-110, 100) + ylim(-45, 78)

xlab.p3 <- round(mds_ward_Night$eig[1]/
                   sum(mds_ward_Night$eig[which(mds_ward_Night$eig>0)]),4)*100
ylab.p3 <- round(mds_ward_Night$eig[2]/
                   sum(mds_ward_Night$eig[which(mds_ward_Night$eig>0)]),4)*100
```

```
p3 <- ggpar(ggscatter(mds_ward.df_Night, x = "Dim.1",  y = "Dim.2",
                      label = rownames(mds_ward.df_Night),
                      shape = "groups", color = "groups",
                      palette = c("#1B9E77","#D95F02","#7570B3"),
                      size = 1, ellipse = TRUE, mean.point = T,
                      ellipse.type = "convex", repel = TRUE,
                      title="Activity: 'Night'", mean.point.size=3.3),
            legend = "none")+
  labs(x = paste0("Dim 1 (", xlab.p3, "%)"),
       y = paste0("Dim 2 (", ylab.p3, "%)")) +
  xlim(-110, 100) + ylim(-45, 78)

xlab.p4 <- round(mds_ward_Day$eig[1]/
                   sum(mds_ward_Day$eig[which(mds_ward_Day$eig>0)]),4)*100
ylab.p4 <- round(mds_ward_Day$eig[2]/
                   sum(mds_ward_Day$eig[which(mds_ward_Day$eig>0)]),4)*100

p4 <- ggpar(ggscatter(mds_ward.df_Day, x = "Dim.1",  y = "Dim.2",
                      label = rownames(mds_ward.df_Day),
                      shape = "groups", color = "groups",
                      palette = c("#1B9E77","#D95F02"),
                      size = 1, ellipse = TRUE, mean.point = T,
                      ellipse.type = "convex", repel = TRUE,
                      title="Activity: 'Day'", mean.point.size=3.3),
            legend = "none") +
  labs(x = paste0("Dim 1 (", xlab.p4, "%)"),
       y = paste0("Dim 2 (", ylab.p4, "%)")) +
  xlim(-110, 100) + ylim(-45, 78)

grid.arrange(p1, p2, p3, p4, ncol = 2)

# 7.1. Bondad de ajuste de la aproximación con MDS:
# (ev[1]+ev[2])/sum(ev[ev>0]) donde ev = valores propios

gof <- as.data.frame(matrix(ncol=4, nrow = 1))
colnames(gof) <- c("Global", "No", "Nigth", "Day")
rownames(gof) <- "GoF"
gof[1,1] <- round(mds$GOF[2], 2)
gof[1,2] <- round(mds_No$GOF[2], 2)
```

```
gof[1,3] <- round(mds_Night$GOF[2], 2)
gof[1,4] <- round(mds_Day$GOF[2], 2)

gof

# Gráficamente:
par(mfrow=c(2,2), mar =c(4, 4, 3, 2)+0.1)
plot(avg, as.matrix(dist(mds_ward$points)), pch=16,cex=.6,
     xlab = "Distancias observadas",
     ylab = "Distances aproximadas",
     main="Global")
abline(0,1, col = 2)

plot(avg_No, as.matrix(dist(mds_ward_No$points)), pch=16,cex=.6,
     xlab = "Distancias observadas",
     ylab = "Distances aproximadas",
     main="Activity: 'No'")
abline(0,1, col = 2)

plot(avg_Night, as.matrix(dist(mds_ward_Night$points)), pch=16,cex=.6,
     xlab = "Distancias observadas",
     ylab = "Distances aproximadas",
     main="Activity: 'Night'")
abline(0,1, col = 2)

plot(avg_Day, as.matrix(dist(mds_ward_Day$points)), pch=16,cex=.6,
     xlab = "Distancias observadas",
     ylab = "Distances aproximadas",
     main="Activity: 'Day'")
abline(0,1, col = 2)



# PASO 8. Comparar la estructura interna de los clusters.
## ======================================================
# ARI: is boudnded by 1, and takes the value 0 when the index
#      equals to its expected value.

xx <- split(data,data$activity)
unique(xx$No$bird)
```

```
unique(xx$Night$bird)
unique(xx$Day$bird)

# En total faltan 7 observaciones: 8, 12, 13, 14, 18, 33 i 36

hc.glob <- data.frame(hc.ward,bird=unique(data$bird))
hc.no <- data.frame(hc.ward_No,bird=unique(xx$No$bird))
hc.night <- data.frame(hc.ward_Night,bird=unique(xx$Night$bird))
hc.day <- data.frame(hc.ward_Day,bird=unique(xx$Day$bird))

hc.glob <- hc.glob[hc.glob$bird %in% c(8,12,13,14,18,33,36) == F,]
hc.no <- hc.no[hc.no$bird %in% c(8,12,13,14,18,33,36) == F,]
hc.night <- hc.night[hc.night$bird %in% c(8,12,13,14,18,33,36) == F,]
hc.day <- hc.day[hc.day$bird %in% c(8,12,13,14,18,33,36) == F,]

ARI.ward <- matrix(ncol=4, nrow = 4)
diag(ARI.ward) <- 1
colnames(ARI.ward) <- rownames(ARI.avg) <- c("Global", "No", "Nigth", "Day")

ARI.ward[1, 2] <- ARI.ward[2, 1] <- adjustedRandIndex(hc.glob$hc.ward,
                                                      hc.no$hc.ward_No)
ARI.ward[1, 3] <- ARI.ward[3, 1] <- adjustedRandIndex(hc.glob$hc.ward,
                                                      hc.night$hc.ward_Night)
ARI.ward[1, 4] <- ARI.ward[4, 1] <- adjustedRandIndex(hc.glob$hc.ward,
                                                      hc.day$hc.ward_Day)
ARI.ward[2, 3] <- ARI.ward[3, 2] <- adjustedRandIndex(hc.no$hc.ward_No,
                                                      hc.night$hc.ward_Night)
ARI.ward[2, 4] <- ARI.ward[4, 2] <- adjustedRandIndex(hc.no$hc.ward_No,
                                                      hc.day$hc.ward_Day)
ARI.ward[3, 4] <- ARI.ward[4, 3] <- adjustedRandIndex(hc.night$hc.ward_Night,
                                                      hc.day$hc.ward_Day)
round(ARI.ward,3)
```

A.6. Exploración de distancias extremas.

```
Hdist2 <- Hdist
Hdist2[upper.tri(Hdist2)] <- NA; diag(Hdist2)<- NA
dataHdist <- data.frame("distancias"=na.omit(as.vector(Hdist2)),
                        "group"=rep("A", 65341))

# Plot de las distancias:
plot(dataHdist$distancias)

# Boxplot con los valores extremos etiquetados:
ggbetweenstats(dataHdist, group, distancias, outlier.tagging = TRUE)

# Cuántos outliers exactamente:
outliers <- boxplot(dataHdist$distancias, plot=FALSE)$out
length(outliers)
```