



Creación de un paquete en el software estadístico R para una prueba de hipótesis Bayesiana no paramétrica para muestras pareadas

Kevin Ortiz González
Karol Michelle Sandoval

Universidad del Valle
Facultad de Ingeniería, Escuela de Estadística
Santiago de Cali, Colombia
2022

Creación de un paquete en el software estadístico R para una prueba de hipótesis Bayesiana no paramétrica para muestras pareadas

Kevin Ortiz González
Karol Michelle Sandoval

Trabajo de grado presentado como requisito parcial para optar al título de:
Estadístico(a)

Directora:
Ph. D. Luz Adriana Pereira Hoyos
Codirector:
Ph. D. Cesar Andrés Ojeda Echeverry

Universidad del Valle
Facultad de Ingeniería, Escuela de Estadística
Santiago de Cali, Colombia
2022

Dedicatoria y lema

Dedicado a nuestros padres, Walter Ortiz (q. e. p. d.), Paola González, Raúl Sandoval y María Burbano, quienes son los principales promotores de nuestros sueños y que con su amor, paciencia y apoyo incondicional nos permitieron alcanzar este logro.

“La paciencia y perseverancia tienen un efecto mágico ante el que las dificultades desaparecen y los obstáculos se desvanecen”.

- John Quincy Adams.

Agradecimientos

A nuestros padres, quienes nos inspiran cada día a ser mejores personas y superar cada obstáculo que se presente en el camino. Gracias por los consejos, el apoyo, la dedicación, paciencia y sobretodo por el amor incondicional que nos han expresado durante todos estos años y nos han convertido en lo que somos actualmente. Infinitas gracias por siempre creer en nosotros y fomentar el deseo de superación y triunfo en la vida, sin ustedes no hubiera sido posible este logro.

A nuestra directora de trabajo de grado, Luz Adriana Pereira Hoyos, quien con mucha paciencia y dedicación nos acompañó gratamente durante todo este ciclo, rememorándonos la importancia de tener en cuenta esos valiosos conceptos adquiridos a lo largo de nuestra carrera profesional y brindándonos acceso a nuevos conocimientos.

A nuestro codirector, César Andrés Ojeda Echeverry, que con su colaboración y lineamientos nos permitió direccionar el desarrollo de este trabajo de una manera más adecuada.

A la Universidad del Valle, a la Escuela de Estadística y, en general, a todos los profesores que contribuyeron a nuestro aprendizaje y formación como profesionales de esta maravillosa carrera. Nuestro más profundo agradecimiento para cada uno de ustedes por hacer parte de una de las profesiones más bonitas e importantes de la sociedad, porque así como dice Elena Poniatowska: *“los profesores se desprenden de cuanto tienen y de cuanto saben, porque su misión es esa: dar”*.

Finalmente, quisiéramos agradecer a aquellas personas que hoy no están con nosotros, a Walter Ortiz, Olga Mosquera, Daniel Sandoval e Italo Burbano, por todas las enseñanzas, valores, principios y el amor que nos brindaron, siempre vivirán en nuestros corazones y estarán presentes en cada paso que demos.

Resumen

En este trabajo de grado se lleva a cabo la creación de un paquete estadístico en el *software* R para la implementación de la prueba de hipótesis Bayesiana no paramétrica para muestras pareadas propuesta por Pereira et al. (2020), permitiendo a la comunidad científica y académica su uso como alternativa ante técnicas paramétricas, que tienen como principal limitante el cumplimiento de supuestos, y no paramétricas, en las que correlaciones negativas conllevan a una baja potencia de la prueba. Así, se presenta en primera instancia una descripción a detalle de la prueba, con los fundamentos más importantes que dieron lugar a su origen y el algoritmo utilizado para su programación en el *software* R. Posteriormente, se presentan los pasos requeridos para la creación del paquete y las funciones que lo integran. Finalmente, se ilustra su funcionamiento a través de ejemplos, considerando datos pareados simulados y reales.

Palabras clave: software R, muestra pareada, estadística Bayesiana, estadística no paramétrica, prueba de hipótesis.

Abstract

In this undergraduate thesis, a statistical package is created in R software for the implementation of the Bayesian Nonparametric hypothesis testing procedure for paired samples proposed by Pereira et al. (2020), allowing the scientific and academic community to use it as an alternative to parametric techniques, whose main limitation is the fulfillment of assumptions, and nonparametric techniques, in which negative correlations lead to a low power of the test. Thus, a detailed description of the test is presented in the first instance, with the most important fundamentals that gave rise to its origin and the algorithm used for its programming in the software R. Subsequently, the steps required for the creation of the package and the functions that integrate it are presented. Finally, its operation is illustrated through examples, considering simulated and real paired data.

Keywords: R software, paired sample, Bayesian statistics, non-parametric statistics, hypothesis testing.

Contenido

Resumen	7
1. Introducción	1
1.1 Planteamiento del problema	1
1.2 Justificación	2
1.3 Objetivos de investigación	3
1.3.1 Objetivo general	3
1.3.2 Objetivos específicos	3
1.4 Antecedentes	3
2. Marco teórico	6
2.1 Conceptos relacionados con el <i>software</i> R	6
2.1.1 Software	6
2.1.2 Software R	6
2.1.3 Funciones de R	6
2.1.4 Paquetes en R	7
2.1.5 Git	9
2.1.6 GitHub	9
2.2 Conceptos Bayesianos fundamentales	9
2.2.1 Estadística Bayesiana	9
2.2.2 Distribución a priori	9
2.2.3 Distribución a posteriori	10
2.2.4 Pruebas de hipótesis Bayesianas	10
2.2.5 Estadística Bayesiana no paramétrica	11
2.2.6 Proceso Dirichlet	11
2.2.7 Modelo de mezcla Dirichlet	12
2.2.8 Distribuciones a priori <i>spike-slab</i>	13
2.3 Algoritmos para estimación a posteriori	14
2.3.1 Algoritmo MCMC	14
2.3.2 Muestreador de Gibbs	14
2.3.3 Algoritmo de Metrópolis	15
2.3.4 Método <i>slice sampling</i> propuesto por Walker (2007)	16

2.4	Comparación entre distribuciones	17
2.4.1	Función <i>shift</i>	17
2.5	Prueba de hipótesis BNP para muestras pareadas	20
2.5.1	Algoritmo de la prueba BNP para muestras pareadas	24
3.	Metodología	27
3.1	Creación de las funciones en el software R	27
3.2	Creación del paquete en el software R	36
3.2.1	Proyecto en R	36
3.2.2	Directorio de trabajo	36
3.2.3	Tipo de proyecto	36
3.2.4	Detalles del paquete	37
3.2.5	Archivos esenciales del paquete	38
3.2.6	Estructuración del código en R y la documentación	39
3.2.7	Archivo <i>DESCRIPTION</i>	40
3.2.8	Carga de datos	41
3.2.9	Viñetas de las funciones	42
3.2.10	Archivo <i>README</i>	42
3.2.11	Carpeta de pruebas	43
3.2.12	Git - GitHub	43
3.3	Archivos de datos	51
3.3.1	Datos 1 - Simulación de una mezcla normal bivariada	51
3.3.2	Datos 2 - Calidad del aire	51
3.3.3	Datos 3 - Pérdida de peso por dieta	51
3.3.4	Datos 4 - Simulación de una mezcla normal bivariada	52
4.	Resultados	53
4.1	Paquete BNPPairedSamples	53
4.2	Ilustración del funcionamiento del paquete <i>BNPPairedSamples</i>	54
4.2.1	Simulación estadística #1 - Mezcla Gaussiana bivariada	54
4.2.2	<i>Airquality</i> - Paquete <i>datasets</i>	56
4.2.3	<i>Weightloss</i> - Paquete <i>datarium</i>	58
4.2.4	Simulación estadística #2 - Mezcla Gaussiana bivariada	61
5.	Conclusiones y recomendaciones	63
5.1	Conclusiones	63
5.2	Limitaciones	63
5.3	Recomendaciones	64
	Bibliografía	65

1 Introducción

1.1. Planteamiento del problema

En diversos estudios es usual que se tenga como objetivo principal comprobar la efectividad de una intervención o tratamiento, hecho que conlleva a tomar mediciones de una o más variables de interés en una muestra de individuos en dos momentos de tiempo específicos, lo cual se conoce como muestras pareadas o dependientes (Romero 2013). Para la comparación estadística de estas muestras se realizan pruebas de hipótesis, las cuales desde el enfoque paramétrico usualmente son desarrolladas a través de la prueba t de Student (1908). Sin embargo, los supuestos paramétricos de esta clase de pruebas, como el requerimiento de simetría en la distribución, pueden llegar a limitar el alcance de las investigaciones. No obstante, se han implementado nuevas técnicas en el ámbito de las pruebas no paramétricas, entre las que resalta la prueba de los rangos con signo de Wilcoxon (1945), para la cual se encuentra disponible su versión bayesiana, propuesta por Benavoli et al. (2014). Aún así, uno de los inconvenientes que han presentado las pruebas mencionadas anteriormente se debe a que el signo negativo de la correlación entre las mediciones (antes y después) puede tener un impacto desfavorable en la potencia de la prueba (Zimmerman 1997). Ante esto, Girón et al. (2003) establecieron una alternativa a la prueba t de Student, la cual se basa en la representación de la dependencia entre las mediciones observadas por medio de un modelo jerárquico, no obstante los supuestos paramétricos siguen vigentes y nuevamente pueden condicionarse como limitantes para el investigador. Desde el punto de vista de las pruebas Bayesianas no paramétricas poco se ha estudiado; se tienen trabajos como el de Filippi et al. (2016) y Filippi & Holmes (2017), en los que se ha hecho énfasis en pruebas de hipótesis sobre el coeficiente de correlación, haciendo uso de distribuciones a priori dadas por procesos Dirichlet y árboles de Pólya respectivamente. Por tal motivo, Pereira et al. (2020) desarrollaron una prueba de hipótesis Bayesiana no paramétrica (aplicable a variables de respuesta continuas) que flexibiliza los supuestos de las pruebas tradicionales y permite comparar las distribuciones en su totalidad y no tan solo por los parámetros de localización y escala.

Ahora bien, a nivel general esta prueba tiene su fundamento en las muestras pareadas y, en este sentido, es importante destacar que su aplicación resulta útil en diversas áreas del conocimiento donde se requiere identificar el efecto producido por una determinada medida. De cualquier modo, es muy frecuente que no se pueda llevar a cabo la aplicación de pruebas

tradicionales debido al incumplimiento de los supuestos paramétricos, bien sea por el tamaño reducido de la muestra o por la naturaleza misma de los datos, es por esto que resulta conveniente la aplicación de una prueba de hipótesis que flexibilice estos supuestos, como es el caso de esta prueba. En este trabajo se propone el desarrollo de un paquete estadístico en el *software* R que permita llevar a cabo dicho test, de manera práctica y funcional. Este paquete se encontrará documentado a detalle, posibilitando al usuario final un entendimiento completo de las funciones configuradas en el *software* a partir de la prueba de hipótesis desarrollada por Pereira et al. (2020).

1.2. Justificación

En un sentido práctico, los investigadores se encuentran frecuentemente interesados en observar si las condiciones experimentales de dos muestras de datos están asociadas o influyen en los resultados obtenidos en un estudio (Soriano 2015); esto ha generado que la estadística inferencial, la cual tiene su base en la información muestral, juegue un papel fundamental en diversas áreas del conocimiento (Sprent & Smeeton 2001). En particular, las pruebas de hipótesis resultan ser útiles dentro de este ámbito, ya que permiten la toma de decisiones con respecto a las inferencias realizadas sobre la población de estudio basadas en la evidencia, al considerar un error aleatorio.

Teniendo en cuenta la aplicabilidad que ha tenido el análisis inferencial a lo largo del tiempo, se ha pensado en el diseño de programas estadísticos que estén encaminados no sólo a la comunidad estadística, sino también a otras comunidades académicas y científicas que los requieran (Genolini et al. 2019). Ahora bien, en vista de los constantes avances a nivel tecnológico y de la internet, autores como de Leeuw (2011) mencionan que esto conlleva a que los científicos promuevan cada vez más el uso de *softwares* estadísticos de código abierto y la investigación replicable; en este sentido, uno de los lenguajes de programación que ha destacado en este campo y generado un gran impacto en la actualidad es R, esto debido a que los investigadores contribuyen frecuentemente con nuevas funcionalidades y modificaciones que proporcionan la solución más actualizada y completa posible (Jank 2011).

En términos generales, la eficiencia del lenguaje de programación R va más allá de solo permitir al usuario acceder al código fuente; dentro de las ventajas adicionales que resaltan se encuentran su desarrollo por más de dos décadas, su acceso libre, su interpretación sencilla y de alto nivel y su lenguaje orientado a objetos que está direccionado a estadísticos y profesionales de disciplinas relacionadas (Sueur 2018). Otros autores, por su parte, exponen que es imposible encasillar a R únicamente como un programa de análisis de datos, pues su flexibilidad y colectividad han hecho que se extienda su uso a otras áreas de aplicación (Boehmke 2016). Finalmente, en cuanto al alcance del programa, Ohri (2014) indica que para el 2014 la cantidad estimada de usuarios era aproximadamente de 2 millones a nivel mundial,

hecho que resulta ser muy conveniente para divulgar la prueba de hipótesis desarrollada en este trabajo, pues su reciente publicación no cuenta con un código formal abierto que pueda ser replicado por la comunidad, impidiendo así la obtención de los resultados requeridos.

1.3. Objetivos de investigación

1.3.1. Objetivo general

Crear un paquete en el *software* estadístico R para la aplicación de una prueba de hipótesis Bayesiana no paramétrica para muestras pareadas.

1.3.2. Objetivos específicos

- Describir detalladamente la prueba de hipótesis Bayesiana no paramétrica para muestras pareadas propuesta por Pereira et al. (2020).
- Establecer las funciones y documentación necesarias que posibiliten la aplicación de la prueba, de tal forma que se tenga un conocimiento completo de su funcionamiento en el lenguaje de programación R.
- Ilustrar mediante ejemplos las funciones dispuestas en el paquete haciendo uso del lenguaje de programación R.

1.4. Antecedentes

Aunque los inicios de R se remontan a 1993, año en el que Ross Ihaka y Robert Gentleman crearon este potencial *software* para ser usado en sus clases de laboratorio, no fue sino hasta 1995 que sus creadores dispusieron el código fuente de forma libre, bajo los términos de la licencia general GNU de la Fundación de Software Libre (Ihaka 1998). A partir de entonces el gran desarrollo de este *software* ha sido notable, particularmente gracias a su extensa colección de paquetes integrada por voluntarios e investigadores (Xia et al. 2018), constituyendo así la manera en la que este se encuentra actualizado respecto a los procedimientos analíticos y gráficos dentro del amplio mundo de la estadística y otras disciplinas que se han ido integrando paulatinamente. En este sentido, la creación de paquetes en R ha suscitado una cantidad considerable de estudios a su alrededor. Es así que autores como Chambers (2008) y Wickham (2015) han realizado escritos enfocados a la organización y construcción de paquetes en R; Chambers (2008) en su libro “Software for Data Analysis: Programming with R” dedica un capítulo relacionado con los paquetes en R, refiriéndose a la organización del código, la documentación del código, las modificaciones que pueden

realizarse sobre paquetes existentes, pruebas del paquete y los problemas que pueden surgir en la instalación de algún paquete en especial y, en cuanto a Wickham (2015), su libro “R Packages: Organize, Test, Document, and Share your Code” detalla con mayor profundidad la compilación del código en R y el uso de *Git* y *Github* para la publicación del paquete.

Por otro lado, al tiempo que voluntarios contribuían en buena medida con nuevas funcionalidades para R y proporcionaban actualizaciones pertinentes a las ya existentes a mediados de los años 90, a los creadores del *software* les era imposible atender a todas las solicitudes satisfactoriamente; por tal razón, se estableció en 1997 un equipo desarrollador más grande, el cual estaba compuesto por once integrantes (Ihaka 1998). Además de atender las solicitudes de forma periódica, algunos de estos integrantes desarrollaron documentos de valor agregado para la comunidad de usuarios de R, como es el caso de Leisch (2009), quien años después exhibiría un tutorial práctico para la creación de paquetes en el *software*: no solo se presenta la forma en la que puede iniciarse un paquete a partir de un conjunto de funciones, sino que también se discute el modo en el que la programación orientada a objetos y las fórmulas de S (lenguaje de programación base para R) pueden ser utilizadas con el objetivo de darle el estilo y la apariencia habitual al código en R. Todo lo que el autor da a conocer en dicho tutorial es ejemplificado a través de funciones que permiten llevar a cabo un análisis de regresión lineal estándar, posibilitando al lector una interpretación paso a paso de todos los requerimientos referentes a la creación de paquetes en R.

Con respecto a las actualizaciones que se realizan sobre los paquetes, estas implican mejoras sobre el *software* R a nivel general, razón por la cual es vital tener la información tan actualizada y completa como sea posible. Con ese objetivo en mente es que los autores Wiley & Wiley (2016) analizan en el capítulo *Writing a Package* un total de cuatro extensiones actuales que permiten desarrollar los paquetes en R en su totalidad: *devtools*, que gestiona el desarrollo del paquete; *roxygen2*, que facilita la redacción en la documentación requerida por el *software* y *testthat* y *covr*, los cuales facilitan el control de la calidad del paquete creado.

Ahora bien, luego de abordar la creación de paquetes en términos generales resulta importante ahondar en la parte específica, particularmente en lo relacionado con las temáticas de la estadística Bayesiana no paramétrica que resultan ser aspectos cruciales en la prueba de hipótesis utilizada en este trabajo y, por tal razón, algunos de los conceptos aquí mencionados serán detallados en la sección 2. Como primera medida, Jara et al. (2011) desarrollaron el *DPpackage* con el propósito de permitir la implementación de modelos Bayesianos no paramétricos y semiparamétricos, esto debido a que en muchas ocasiones resulta necesario relajar supuestos paramétricos para ganar flexibilidad en la modelación y robustez en lo que se refiere a la mala especificación del modelo de probabilidad y, en este sentido, una posible solución desde el contexto Bayesiano es asociar una distribución a priori en un espacio de funciones; sin embargo, las distribuciones a posteriori que abarcan

los espacios de funciones resultan ser muy complejas y, ante esta situación, este paquete proporciona un conjunto de programas simples que facilitan la ejecución de modelos para la estimación de las densidades marginal y condicional, datos de regresión binaria, datos longitudinales y agrupados haciendo uso de modelos lineales mixtos generalizados y la elicitación del parámetro de precisión del proceso Dirichlet a priori. Por otra parte, Krypotos et al. (2017) desarrollaron el *condir package*, que se consolida como una herramienta clave para investigadores que utilizan datos de condicionamiento, ya que proporciona una serie de técnicas de contraste de hipótesis desde el enfoque frecuentista y Bayesiano (facilitando el cálculo del factor de Bayes) con la finalidad de identificar diferencias entre las respuestas de los individuos al ser sometidos a diferentes estímulos, como imágenes o sonidos que generan desagrado. Así mismo, Canale (2017) creó el *msBP package* en R a fin de implementar un nuevo método basado en los polinomios de Bernstein multiescala para densidades y, de este modo, contribuir a la inferencia Bayesiana no paramétrica multiescala, pues esta técnica supera el inconveniente principal de los árboles de Pólya, que hace referencia a la obtención de densidades extremadamente puntiagudas aún en presencia de densidades reales suaves, y conserva muchas de las ventajas de los modelos de mezclas de procesos Dirichlet, particularmente una adecuada estimación de la densidad ante regiones planas y/o concentradas. Por su parte, Carrasco et al. (2017) desarrollaron el *rNPBST package*, que proporciona un conjunto de pruebas no paramétricas y Bayesianas en un sólo repositorio para propósitos como pruebas de aleatoriedad, pruebas de bondad de ajuste o análisis de dos y múltiples muestras, constituyéndose como una herramienta muy útil en este campo de la estadística. Por último, resalta el paquete *dirichletprocess*, desarrollado por J. Ross & Markwick (2020), usado para realizar análisis Bayesianos no paramétricos considerando procesos Dirichlet, permitiendo a usuarios ejecutar estas funcionalidades genéricas en vez de tener que generar sus propios algoritmos; entre las funciones que componen este paquete, destacan: a) la estimación de densidades, partiendo de modelos de mezcla Dirichlet; b) clusterización, para evidenciar formas de agrupación de los datos y c) establecimiento de distribuciones a priori dentro de modelos de mezcla jerárquicos. Esta última función se encuentra relacionada con los fundamentos teóricos de la prueba de hipótesis desarrollada por Pereira et al. (2020), ya que al trabajar con modelos Bayesianos se asocian distribuciones a priori a los parámetros y, este caso al tratarse de un proceso Dirichlet, permite transformar la integral de la densidad del modelo de mezcla infinito en una suma infinita; a su vez estos modelos hacen parte de un conjunto más grande denominados modelos de mezcla jerárquicos.

2 Marco teórico

Esta sección está enfocada en presentar la terminología necesaria para la comprensión de la prueba de hipótesis Bayesiana no paramétrica para muestras pareadas propuesta por Pereira et al. (2020) a nivel estadístico y su configuración en el *software* R.

2.1. Conceptos relacionados con el software R

2.1.1. Software

De acuerdo con Kuhn (2008), el término *software* hace referencia a todos aquellos componentes no físicos de una computadora, lo cual incluye su sistema operativo y todos los programas incorporados. A través de instrucciones preprogramadas, el software permite la ejecución del hardware (procesadores, monitores y controladores) y, en consecuencia, la computadora puede operar de una manera correcta.

2.1.2. Software R

Este proyecto gratuito y de código abierto proporciona un lenguaje para la realización de cálculos interactivos que se encuentran apoyados por técnicas de organización de datos, gráficos, simulaciones, entre otras funcionalidades que se encuentran a disposición del usuario (Chambers 2008). Así mismo, se trata de un lenguaje de programación orientado a objetos que fue diseñado por Ross Ihaka y Robert Gentleman en 1992 y es considerado una gran herramienta en lo que respecta al análisis estadístico.

2.1.3. Funciones de R

Son las piezas básicas que componen al software R. En síntesis, estas funciones permiten reducir la duplicación de código en R, mediante la automatización en forma recursiva de tareas generalizadas. Con excepción de unas cuantas, las funciones en R están conformadas de forma general por tres partes: a) *body()*, que es el código que se encuentra dentro de la función, el cual da las instrucciones necesarias para llevar a cabo la tarea; b) *formals()* o

argumentos de la función, necesarios para que esta se ejecute y c) *environment()*, que hace referencia a la localización o ambiente en el que opera la función (Boehmke 2016).

2.1.4. Paquetes en R

De acuerdo con Wiley & Wiley (2016) un paquete es la forma fundamental y básica a partir de la cual se documenta, comparte y distribuye el código en R; estas extensiones son consideradas colecciones de funciones y conjuntos de datos desarrolladas por la comunidad del entorno R para cumplir una tarea específica. Para su instalación se requiere del comando *install.package()* y mediante la función *library()* se obtiene acceso a todo lo que contienen.

Estos paquetes se encuentran alojados ya sea en el CRAN (*The Comprehensive R Archive Network*) o en algún repositorio y cuentan de forma interna con un conjunto de archivos esenciales que permiten un adecuado funcionamiento para el usuario, según lo expone Wickham (2015). Estos son:

- **Archivo *DESCRIPTION*:** expone los metadatos del paquete, entre los que se encuentran su nombre, la versión, los autores (incluyendo aquí el mantenedor del paquete), una breve descripción generalizada, el tipo de licencia asociado y la especificación de funciones de otros paquetes (o paquetes en su totalidad) que deben ser importadas. Para todos los paquetes este archivo se encuentra disponible en el recuadro inferior derecho de la interfaz por defecto de RStudio.
- **Archivo *NAMESPACE*:** se trata de un archivo necesario para una instalación exitosa del paquete por parte de los usuarios de R, ya sea que el paquete se encuentre alojado directamente en el CRAN o deba ser instalado de algún repositorio. Este archivo permite al paquete relacionarse con cualquier otro, de modo tal que puedan importarse funciones específicas o el paquete relacionado en su totalidad. Así mismo, provee la posibilidad de exportar las funciones propias del paquete creado con la finalidad de ponerlas a disposición de los usuarios de R.
- **Directorio *man*:** en este directorio se almacena toda la documentación necesaria asociada a las funciones creadas y alojadas dentro del paquete. Cada uno de los archivos aquí dispuestos pueden concretarse con mayor facilidad a partir de la librería *roxygen2*, la cual suministra un amplio conjunto de etiquetas para la organización de la información, entre las que destacan:
 1. **@title:** el título que se despliega cuando el usuario realiza la llamada por medio de las funciones de ayuda.
 2. **@description:** descripción corta del objetivo que cumple la función.

3. **@param:** cada uno de los argumentos (obligatorios u opcionales) que son ingresados a la función y permiten ejecutarla satisfactoriamente. Estos argumentos también se deben describir brevemente.
4. **@return:** especifica todo lo que retorna la función al ser ejecutada.
5. **@importFrom:** son todas aquellas funciones de otros paquetes que se enlazan e importan para un adecuado funcionamiento del paquete propio.
6. **@note:** observaciones, recomendaciones y/o sugerencias que se le dan al usuario previo al uso de alguna función del paquete. Es un campo opcional.
7. **@examples:** ejemplos que se proveen para el uso de las funciones.
8. **@export:** código en R de cada función contenida en el paquete.

Lo que se encuentra desplegado finalmente bajo estas etiquetas y dispuesto en la carpeta *man*, se ve reflejado al realizar la instalación del paquete en R a través del recuadro de ayuda (*Help*) ubicado en la sección inferior derecha (ver Figura 2-1).

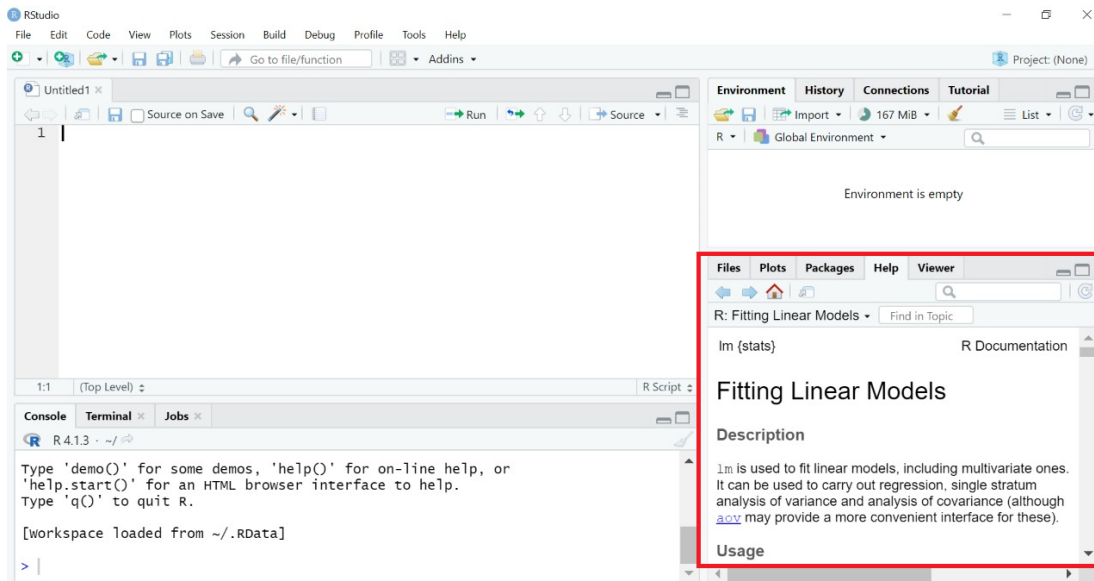


Figura 2-1: Sección de ayuda en RStudio. **Fuente:** Elaboración propia.

- **Directorio *R*:** en este directorio se agregan todas las funciones de R que se quieren disponer al público en el paquete, las cuales se deben organizar en documentos con extensión de archivo *.R*. La librería *roxygen2* facilita esta tarea, permitiendo combinar el código en R con la documentación asociada a cada función en un mismo archivo.

2.1.5. Git

De acuerdo con Chacon & Straub (2014) Git es un sistema de control de versiones distribuido desarrollado por Linus Torvalds en el 2005 que facilita el trabajo colaborativo, ya que permite registrar los cambios de los archivos compartidos en repositorios de código, es decir, gestionar las distintas versiones del código de los proyectos.

2.1.6. GitHub

GitHub es un portal de alojamiento usado por millones de desarrolladores y para infinidad de proyectos colaborativos, ya que es el mayor proveedor de almacenamiento de repositorios de Git (Chacon & Straub 2014). Esta plataforma de la nube creada por Chris Wanstrath, PJ Hyett, Tom Preston-Werner y Scott Chacon en el 2008 facilita el control de versiones, el seguimiento de problemas y la revisión de código de muchos proyectos de código abierto.

2.2. Conceptos Bayesianos fundamentales

2.2.1. Estadística Bayesiana

Esta rama de la estadística tiene su base en el teorema de Bayes, el cual fue introducido por el matemático británico Thomas Bayes en 1763 y hace referencia a la actualización de la información sobre un evento A sabiendo que otro evento B ha sucedido, en este sentido este teorema resuelve el problema conocido como “la probabilidad inversa”. De acuerdo con Wakefield (2012), la diferencia entre las escuelas de estadística frecuentistas y Bayesianas proviene de las diferentes formas en que ven las cantidades de interés o parámetros, ya que en el enfoque frecuentista se toman como constantes fijas pero desconocidas, mientras que en el enfoque Bayesiano se consideran como cantidades o variables aleatorias.

2.2.2. Distribución a priori

En concordancia con lo expresado por Ashby (2006), una distribución a priori puede tener tres interpretaciones adecuadas: 1. distribución de frecuencias con base en datos previos, 2. representación normativa y objetiva sobre lo que se cree acerca de un parámetro y 3. una medida subjetiva de lo que un individuo o sujeto cree. Adicionalmente, se tiene que las distribuciones a priori se clasifican en propias e impropias y a su vez, en informativas y no informativas. Según lo expresado por Morales & Causil (2018) “una a priori propia es una distribución que asigna pesos no negativos y que suman o integran hasta uno, a todos los valores posibles del parámetro”; también afirman Morales & Causil (2018) que “una a priori es no informativa, cuando refleja una ignorancia total o un conocimiento muy limitado sobre

el parámetro de interés, mientras que en las a priori informativas se tiene conocimiento sobre dichos parámetros”.

2.2.3. Distribución a posteriori

Ya que la estadística Bayesiana tiene como objetivo primordial la realización de inferencias sobre el parámetro o parámetros de interés de las distribuciones que describen cierto fenómeno, es importante resaltar la forma en la que se puede escribir la distribución de probabilidad conjunta de θ , el parámetro o parámetros, y x , los datos de la muestra, esto teniendo en cuenta la notación y pasos realizados por Gelman et al. (2014). En este sentido, la distribución conjunta se verá expresada en el análisis Bayesiano como el producto entre la distribución a priori y la distribución muestral:

$$p(\theta, x) = p(\theta)p(x|\theta) \quad (2-1)$$

Teniendo en cuenta la expresión 2-1, la distribución a posteriori se puede encontrar haciendo uso de la regla de Bayes:

$$p(\theta|x) = \frac{p(\theta, x)}{p(x)} = \frac{p(\theta)p(x|\theta)}{p(x)} \quad (2-2)$$

Donde $p(x)$ en la ecuación 2-2 corresponde a la distribución marginal de X . No obstante, en ocasiones se prefiere trabajar con la forma aproximada de la densidad a posteriori:

$$p(\theta|x) \propto p(\theta)p(x|\theta) \quad (2-3)$$

Omitiendo así el cálculo de la constante que no depende del parámetro o parámetros.

2.2.4. Pruebas de hipótesis Bayesianas

Una prueba de hipótesis es un procedimiento formal que se asemeja al método científico, pues se basa en la observación de algún fenómeno natural, que conlleva a la formulación de una teoría y, finalmente, al análisis a partir de lo observado en una muestra aleatoria representativa (Wackerly et al. 2008); para el caso de la estadística Bayesiana este proceso no resulta igual que en la estadística clásica, ya que las hipótesis por lo general se encuentran dadas en función del(los) parámetro(s) de interés y el espacio paramétrico en el que se encuentran, de este modo se tiene que:

$$H_0 : \theta \in \Theta_0 \quad \text{vs.} \quad H_1 : \theta \in \Theta - \Theta_0$$

En correspondencia con Berger (1985), la tarea de decidir entre H_0 y H_1 en el análisis Bayesiano se limita a calcular las probabilidades a posteriori $p(H_0|x)$ y $p(H_1|x)$ y decidir entre ellas. En este sentido, autores como López & Jiménez (2010) exponen que una forma de decidir entre H_0 y H_1 es hacer uso del factor de Bayes ($B(x)$):

$$B(x) = \frac{O(H_0, H_1|x)}{O(H_0, H_1)}$$

Donde:

$$O(H_0, H_1|x) = \frac{P(H_0|x)}{P(H_1|x)}; \quad O(H_0, H_1) = \frac{P(H_0)}{P(H_1)}$$

Si $B(x)$ es mayor a 1 existe evidencia a favor de H_0 y, en caso de obtener un valor menor, la evidencia se encuentra a favor de H_1 .

2.2.5. Estadística Bayesiana no paramétrica

La estadística Bayesiana no paramétrica hace referencia al análisis Bayesiano realizado sobre modelos estadísticos cuyo espacio paramétrico no es finito dimensional (Lee 2011). Esta área de la estadística proporciona una herramienta útil en tanto provee flexibilidad y robustez a los modelos estadísticos, constituyéndose como una alternativa a las técnicas paramétricas (Johnson & de Carvalho 2015).

2.2.6. Proceso Dirichlet

Rossi (2014) afirma que un proceso Dirichlet (DP), introducido inicialmente por Ferguson (1973), es generalmente usado como parte de un modelo de mezcla y se puede definir de la siguiente manera:

Sea F_0 una distribución sobre el espacio paramétrico Θ y M un número real positivo, entonces para cualquier partición finita medible A_1, \dots, A_r de Θ , el vector $(F(A_1), \dots, F(A_r))$ es aleatorio siempre y cuando F sea una medida aleatoria de probabilidad. Se dice que F es un proceso Dirichlet con distribución base F_0 y parámetro de concentración M si:

$$(F(A_1), \dots, F(A_r)) \sim \text{Dir}(MF_0(A_1), \dots, MF_0(A_r))$$

Para cada partición finita medible A_1, \dots, A_r de Θ , es decir, cada subconjunto del espacio paramétrico acotado en intervalos de valores $[a, b]$.

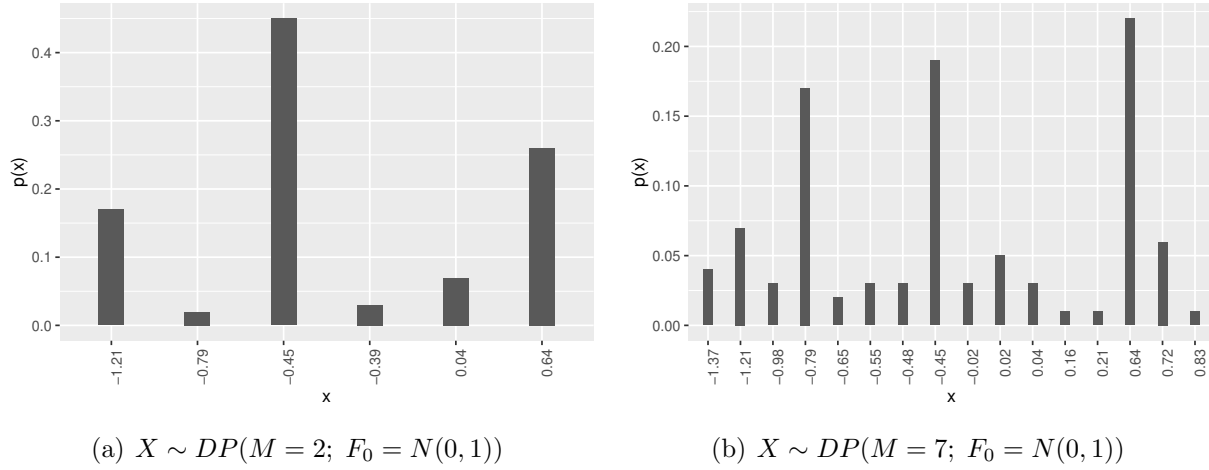


Figura 2-2: Realizaciones del proceso Dirichlet, con medida base Gaussiana estándar y cambios en el parámetro de concentración. **Fuente:** Elaboración propia.

En otras palabras, un proceso Dirichlet se puede pensar como una distribución cuyas realizaciones son en sí mismas distribuciones de probabilidad. Así mismo, se puede decir que la distribución base es intrínsecamente la distribución a priori del proceso, aquella que genera los valores esperados, mientras que el parámetro de concentración controla la variabilidad de las observaciones alrededor de la medida base. Para la muestra del efecto de estos parámetros, se expone la Figura 2-2, que considera en ambos casos como medida base una distribución Gaussiana estándar y un cambio en el parámetro de concentración. De esta manera, se observa que el dominio de valores para ambas funciones se encuentra acorde a la distribución normal típica, mientras que el parámetro de concentración acota el dominio cuando tiende a cero o lo dispersa a medida que va tomando valores crecientes.

2.2.7. Modelo de mezcla Dirichlet

Los modelos de mezcla son modelos jerárquicos que tienen la finalidad de describir las diferencias que puedan existir en los datos y que una única distribución no permite identificar. En otras palabras, buscan identificar subpoblaciones dentro de la población original y agruparlas en clústeres, que se definen como grupos con características semejantes entre sí pero que entre grupos son diferentes; los modelos de mezcla se clasifican en finitos e infinitos y su principal diferencia radica en la cantidad de componentes a considerar, lo cual frecuentemente genera inconvenientes debido a que, según lo expresado por Bouguila & Ziou (2008), en los modelos finitos el determinar el número apropiado de componentes de la mezcla que mejor describe el conjunto de datos es una labor bastante compleja. A su vez Bouguila & Ziou (2008) también proponen una solución al problema asumiendo un número infinito de componentes utilizando una estructura probabilística denominada **proceso Dirichlet**.

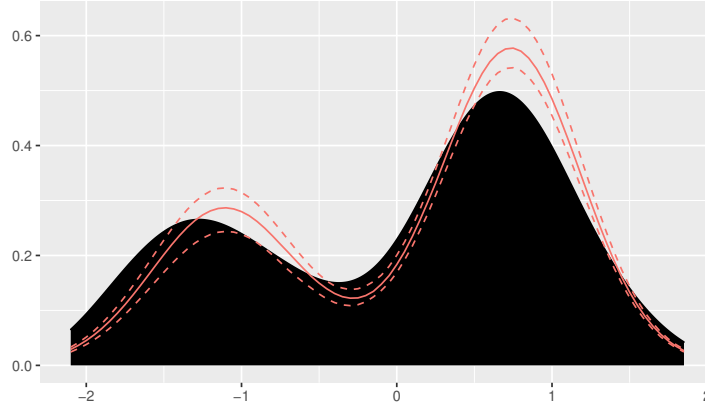


Figura 2-3: Modelo de mezcla Dirichlet con una kernel Gaussiana bivariada para el dataset *faithful*, usando el paquete *dirichletprocess*. **Fuente:** J. Ross & Markwick (2020).

Para este documento, se hará uso del modelo de mezcla Dirichlet (ver Figura 2-3), el cual se enmarca dentro de los modelos infinitos y se define de la siguiente forma:

Sea un conjunto de observaciones x_1, \dots, x_n modeladas a partir de un conjunto de variables latentes $\theta_1, \dots, \theta_n$, donde cada θ_i es independiente e idénticamente distribuida F , mientras que cada x_i tiene una distribución $G(\theta_i)$ parametrizada por θ_i , por lo cual se tiene entonces que:

$$\begin{aligned} x_i | \theta_i &\sim G(\theta_i) \\ \theta_i &\sim F \\ F | M, F_0 &\sim DP(M, F_0) \end{aligned}$$

2.2.8. Distribuciones a priori spike-slab

Dentro del contexto de la selección de variables desde un enfoque Bayesiano, las mezclas a priori del tipo *spike-slab* tienen un uso frecuente. De acuerdo con Malsiner-Walli & Wagner (2011), el componente *spike* tiende a despreciar los efectos pequeños concentrando la densidad en valores que son cercanos a cero; por el contrario, el componente *slab* no concentra la densidad alrededor de un punto específico, sino que se encuentra expandida sobre un rango plausible de valores para los coeficientes de regresión.

En este sentido, si se tiene un modelo de regresión lineal definido de la siguiente forma:

$$\mathbf{y} = \beta_0 + X\vec{\beta} + \varepsilon, \quad \varepsilon \sim N(\vec{0}, I\sigma^2) \quad (2-4)$$

Es posible asumir distribuciones *slope-slab* a priori para el vector de coeficientes $\vec{\beta}_{(d \times 1)}$, introduciendo en primera instancia un vector aleatorio $\vec{\delta} = (\delta_1, \delta_2, \dots, \delta_d)$, donde cada uno de sus elementos δ_j tomará el valor 1 si el coeficiente β_j se encuentra localizado dentro del componente *slab* y 0 si se encuentra localizado en el componente *slope*. Siguiendo la notación de Malsiner-Walli & Wagner (2011), estas distribuciones se pueden denotar de la forma que sigue:

$$p(\vec{\beta}|\delta) = p_{slab}(\vec{\beta}_\delta) \prod_{j:\delta_j=0} p_{slope}(\beta_j) \quad (2-5)$$

En la ecuación 2-5, p_{slab} hace referencia a la distribución *slab* multivariada, mientras que p_{slope} denota la distribución *slope* univariada.

2.3. Algoritmos para estimación a posteriori

2.3.1. Algoritmo MCMC

Estos algoritmos en la actualidad son muy conocidos e importantes en el campo de la estadística Bayesiana, particularmente en lo que respecta a la inferencia estadística, ya que combinan dos propiedades cruciales que son las cadenas de Markov y los métodos Monte Carlo. Las cadenas de Markov hacen alusión a una serie de eventos donde la probabilidad de ocurrencia de un evento se encuentra sujeta al inmediatamente anterior y los métodos estadísticos Monte Carlo son usados para tomar de una distribución de probabilidad muestras que permitan aproximar a la cantidad deseada (van Ravenzwaaij et al. 2018). Dicho de otra forma, los algoritmos MCMC son métodos de simulación estocástica que permiten obtener muestras de una distribución de una forma sencilla evitando el cálculo directo de las probabilidades posteriores.

2.3.2. Muestreador de Gibbs

Dentro del amplio campo de los métodos de Monte Carlo vía Cadenas de Markov (MCMC), el muestreador de Gibbs se destaca como un algoritmo para simular muestras de una función de densidad conjunta (Gelfand 2000). Este método, que es un caso especial del algoritmo de Metropolis-Hastings, fue propuesto por Stuart Alan Geman y Donald Jay Geman en 1984 y su aplicación se da comúnmente cuando la distribución conjunta es desconocida o resulta difícil tomar muestras aleatorias de ella, por lo que se procede a considerar las distribuciones condicionales. Este algoritmo permite abordar un gran grupo de problemas en la inferencia Bayesiana, ya que todas las distribuciones posteriores condicionales por lo general están disponibles o son fáciles de simular haciendo uso de distribuciones de probabilidad estándar (Albert 2009).

Una expresión en pseudocódigo para el muestreador de Gibbs multietápico es dada por Gelfand (2000), quien supone que θ es un vector aleatorio que puede escribirse como $\theta = (\theta_1, \dots, \theta_r)$; para cada una de dichas componentes se asume que las funciones de densidad condicionales son conocidas, permitiendo establecer los siguientes pasos para aplicar el muestreador de Gibbs:

1. $\theta_1^{(t+1)} \sim h(\theta_1 | \theta_2^{(t)}, \dots, \theta_r^{(t)});$
2. $\theta_2^{(t+1)} \sim h(\theta_2 | \theta_1^{(t+1)}, \theta_3^{(t)}, \dots, \theta_r^{(t)});$
- \vdots
- $r.$ $\theta_r^{(t+1)} \sim h(\theta_r | \theta_1^{(t+1)}, \dots, \theta_{r-1}^{(t+1)})$

2.3.3. Algoritmo de Metrópolis

En el ámbito Bayesiano, la distribución a posteriori $p(\theta|x)$ puede resultar difícil de calcular de forma exacta debido a la expresión de la sumatoria o integral que se encuentra en el denominador. De esta forma, Hoff (2009) expone que una alternativa es considerar un conjunto de valores $\theta^{(1)}, \dots, \theta^{(S)}$ que siguen una distribución empírica que pueda aproximarse a la posteriori de interés, lo cual comprende la idea básica del algoritmo de Metrópolis. Específicamente, este algoritmo se basa en el muestreo de un valor θ^* cercano al último valor del conjunto $\theta^{(s)}$, haciendo uso de una distribución simétrica $J(\theta^*|\theta^{(s)})$ que suele ser generalmente una distribución uniforme o normal, así:

- $J(\theta^*|\theta^{(s)}) = \text{uniforme}(\theta^{(s)} - \delta, \theta^{(s)} + \delta).$
- $J(\theta^*|\theta^{(s)}) = \text{normal}(\theta^{(s)}, \delta^2).$

Donde δ es un parámetro establecido de tal forma que el algoritmo sea eficiente.

Posteriormente, teniendo en cuenta tal valor muestreado, se procede a calcular la razón de aceptación:

$$r = \frac{p(\theta^*|x)}{p(\theta^{(s)}|x)} = \frac{p(x|\theta^*)p(\theta^*)}{p(x|\theta^{(s)})p(\theta^{(s)})} \quad (2-6)$$

De la ecuación 2-6 es posible determinar el parámetro puntual en la iteración $s + 1$, como indica Hoff (2009):

- $\theta^{(s+1)}$ tendrá asociado el valor θ^* con probabilidad $\min(r, 1)$.
- $\theta^{(s+1)}$ será $\theta^{(s)}$ con probabilidad $1 - \min(r, 1)$.

De igual manera, es equivalente establecer que $\theta^{(s+1)} = \theta^*$ si $u \sim \text{uniforme}(0, 1) < r$; de lo contrario, $\theta^{(s+1)} = \theta^{(s)}$.

2.3.4. Método slice sampling propuesto por Walker (2007)

Este método se consolida como una pieza clave dentro de la prueba de hipótesis Bayesiana no paramétrica para muestras pareadas propuesta por Pereira et al. (2020), ya que permite limitar la cantidad de componentes a estimar en un modelo de mezcla infinito dimensional a través de la introducción de un conjunto de variables latentes, las cuales no alteran la densidad original del modelo.

En este sentido, se muestrean las siguientes variables:

$$(\mu_j, \sigma_j^2, \nu_j), \quad j = 1, 2, \dots; \quad (d_i, u_i), \quad i = 1, 2, \dots, n. \quad (2-7)$$

Ahora, se enumeran los pasos que componen el algoritmo de Walker (2007).

Paso 1. Muestreo de los átomos o parámetros (μ_j, σ_j^2) :

$$f(\mu_j, \sigma_j^2) \propto p_0(\mu_j, \sigma_j^2) \cdot \prod_{d_i=j} N(y_i | \mu_j, \sigma_j^2) \quad (2-8)$$

La ecuación 2-8 corresponde a la actualización de los átomos a partir de la distribución a priori sobre los parámetros de cada clúster por la verosimilitud de los datos.

Paso 2. Simulación de ν_j de una $\text{Beta}(\nu_j | a_j, b_j)$, donde:

$$\begin{aligned} a_j &= 1 + \sum_{i=1}^n \mathbf{1}(d_i = j), \\ b_j &= M + \sum_{i=1}^n \mathbf{1}(d_i > j) \end{aligned} \quad (2-9)$$

Y los pesos w_j se calculan por medio de la representación *stick breaking*:

$$\begin{aligned} w_1 &= \nu_1 \\ w_2 &= \nu_2 \cdot (1 - \nu_1) \\ &\vdots \\ w_j &= \nu_j \cdot \prod_{j=1}^{\infty} (1 - \nu_{j-1}) \end{aligned} \quad (2-10)$$

En este paso se simulan los pesos para los infinitos parámetros a partir de una distribución beta conjugada con los parámetros ilustrados en la ecuación 2-9, con la finalidad de determinar cuánto aportan en peso los clústeres en la función de densidad de y .

Paso 3. Muestreo de n variables latentes:

$$f(u_i | \dots) \propto \mathbf{1}(0 < u_i < w_{d_i}) \quad (2-11)$$

Cada variable u_i hará posible el truncamiento dentro del modelo, evadiendo así el inconveniente que se tiene con el número infinito de clústeres en el modelo de mezcla Dirichlet (ver ecuación 2-11).

Paso 4. Muestreo del número de clústeres:

$$P(d_i = k | \dots) \propto \mathbf{1}(k : w_k > u_i) \cdot N(y_i | \mu_k, \sigma_k^2) \quad (2-12)$$

Con el paso final del algoritmo de Walker (2007) dado por la ecuación 2-12, el objetivo es encontrar la distribución de convergencia, luego de realizar un número significativo de iteraciones del muestreo. Se aclara que el elemento d_i , que corresponde a cada etiqueta de cluster para cada individuo, es una variable aleatoria en sí misma. Por otro lado, el valor k de esta expresión se considera como el conjunto de datos $\{1, 2, \dots, N\}$, en el que N es el máximo valor para N_i que se define como el valor entero l más grande para el cual se cumple la condición $w_l > u_i$.

Luego de obtener el slice sampling se procede a determinar la etiqueta del individuo a través del cálculo de la densidad en cada uno de los k clústeres hallados en el paso 4.

2.4. Comparación entre distribuciones

2.4.1. Función shift

Esta función, introducida por Doksum (1974) y Doksum & Sievers (1976), permite trazar la diferencia entre los cuantiles de dos distribuciones en función de los cuantiles de uno de los grupos, con el objetivo de cuantificar las diferencias existentes entre las dos muestras de estudio. De acuerdo con la notación de Doksum (1974), si se tienen dos muestras X_1, X_2, \dots, X_m y Y_1, Y_2, \dots, Y_n , cuyas funciones de distribución se denotan por $X \sim G_1$ y $Y \sim G_2$ respectivamente, es posible hallar una función *shift* o de desplazamiento $\Delta(\cdot)$, tal que $X + \Delta(X)$ tenga la misma distribución que Y , siempre y cuando G_1 sea absolutamente continua.

Adicionalmente, a diferencia de métodos analíticos clásicos de comparación de distribuciones como la prueba t de Student, la función *shift* genera un resultado gráfico que permite establecer en qué aspectos y qué tanto difieren las dos distribuciones entre sí, lo cual resulta muy conveniente cuando las distribuciones no son homogéneas entre todos los sujetos de estudio (que es lo más habitual en situaciones reales).

Para la ilustración de estas funciones se muestran dos situaciones hipotéticas en las que se comparan las distribuciones correspondientes a los resultados de un grupo de sujetos antes y después de una intervención en particular. En el primer caso, las distribuciones y las diferencias entre pares se muestran en la Figura 2-4. Como puede observarse, existen cambios marcados en la dispersión de los datos, no obstante, en términos de localización no existe una variación notable.

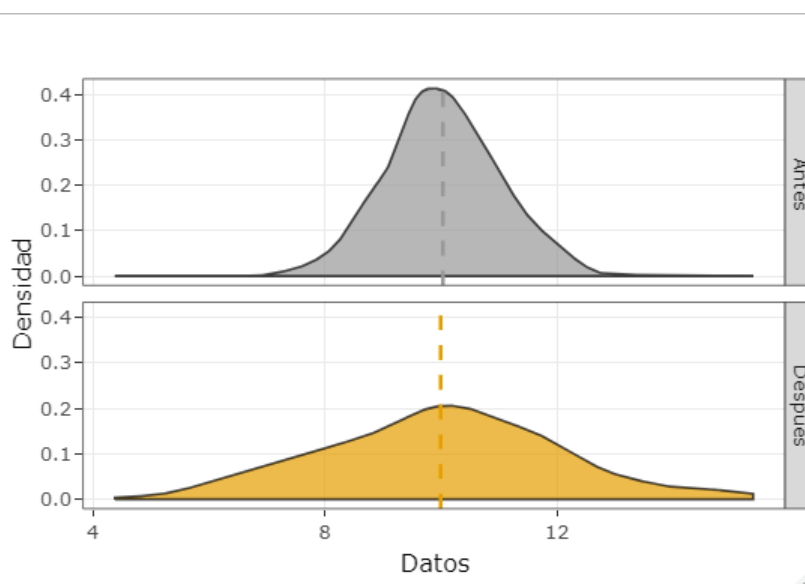


Figura 2-4: Distribuciones que difieren en dispersión pero no en valor medio. **Fuente:** Elaboración propia.

La función *shift* que resulta de la diferencia entre los cuantiles de ambas distribuciones se expone en la Figura 2-5.

Lo que se observa en este caso es una clara diferenciación entre las dos distribuciones, excepto aproximadamente en el intervalo (9.75, 10.25), en el cual se incluye el valor cero en el eje Y (diferencia entre los cuantiles de ambos grupos). Esto indica que, en el mencionado intervalo, no se diferencian las distribuciones de forma significativa, mientras que en el resto de puntos se puede evidenciar el efecto que ejerce la dispersión.

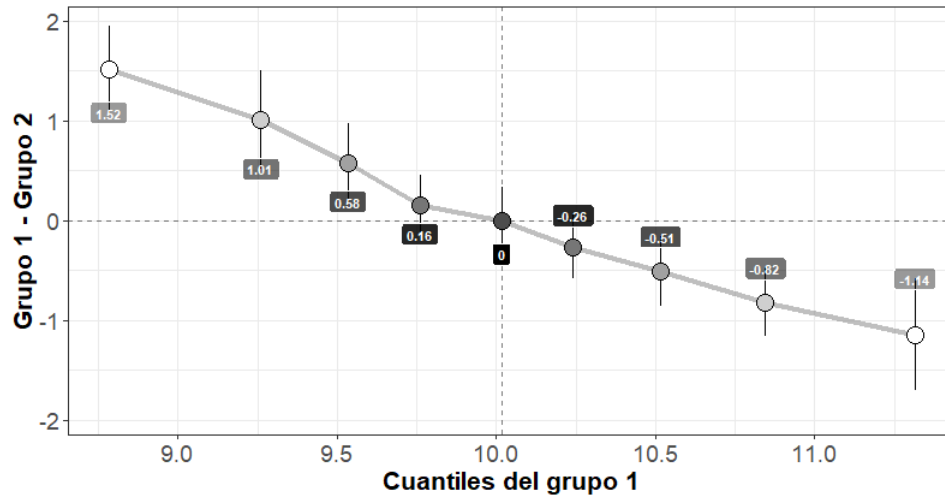


Figura 2-5: Función *shift* para las distribuciones que difieren en dispersión, usando el paquete *rogme* (Rousselet et al. 2017). **Fuente:** Elaboración propia.

En cuanto al segundo caso, se tienen observaciones pareadas que generan las distribuciones que se exponen en la Figura 2-6. En contraste con la situación anterior, la diferencia entre estas distribuciones se da fundamentalmente en la localización, tal y como se evidencia en la Figura 2-7.

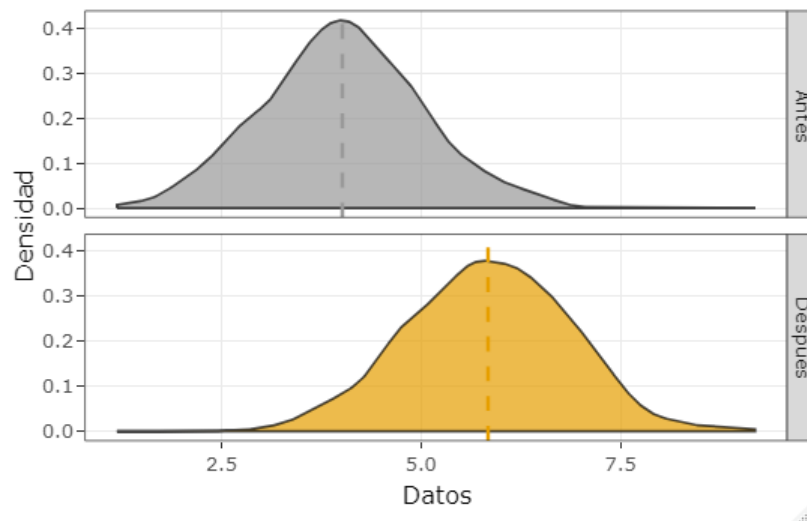


Figura 2-6: Distribuciones que difieren en valor medio pero no en dispersión. **Fuente:** Elaboración propia.

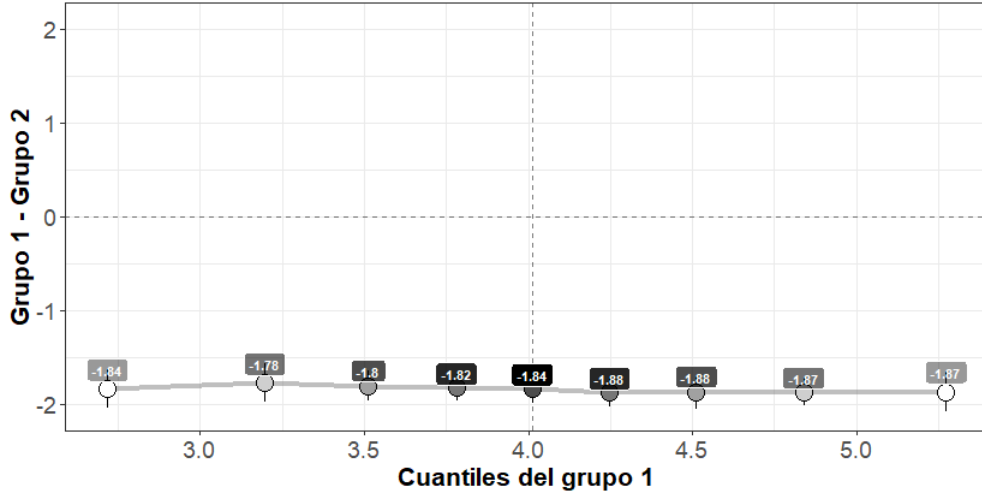


Figura 2-7: Función *shift* para las distribuciones que difieren en media, usando el paquete *rogme* (Rousselet et al. 2017). **Fuente:** Elaboración propia.

Debido a que la distribución que se presentó posterior a la intervención hipotética se encuentra desplazada a la derecha en comparación a la distribución previa, la función *shift* se manifiesta más bien constante y siempre con valores negativos en el eje de las ordenadas.

2.5. Prueba de hipótesis BNP para muestras pareadas

Esta prueba desarrollada por Pereira et al. (2020) permite la comparación de las muestras pareadas y tiene como ventajas la flexibilización de los supuestos paramétricos, el aprovechamiento de las correlaciones que puedan existir entre las muestras y la posibilidad de comparar las distribuciones en su totalidad antes y después de realizar una intervención o tratamiento. En este sentido, se cuenta con un conjunto de individuos $i = 1, 2, \dots, n$ medidos en 2 momentos de tiempo j , obteniendo así la medición Y_{ij} de los individuos o unidades experimentales, consolidando de este modo un vector bivariado $Y_i = (Y_{i1}, Y_{i2})'$ para cada uno de los sujetos de estudio, el cual sigue una distribución conjunta $G(\cdot)$. De acuerdo con el objetivo general de esta prueba, se pretende realizar una comparación de las distribuciones marginales $G_1(\cdot)$ y $G_2(\cdot)$, lo cual se expresa en notación estadística de la siguiente forma:

$$H_0 : G_1(\cdot) = G_2(\cdot) \quad vs. \quad H_1 : G_1(\cdot) \neq G_2(\cdot)$$

De este modo, al aplicar este procedimiento inferencial sobre la muestra de datos pareada, será posible establecer si existe una diferencia estadísticamente significativa entre ambas distribuciones.

Ahora bien, para que G se defina en toda regla como una función de distribución, es preciso que G sea absolutamente continua en el intervalo $[0, 1]$. En tal caso, es posible afirmar que $G(\cdot)$ cuenta con una función de densidad de probabilidad asociada $g(\cdot)$, definida por un modelo de mezcla Bayesiano no paramétrico que permite contrastar las subpoblaciones existentes dentro de la misma población, es decir, la formación natural de clústeres en la distribución. El modelo estadístico para Y_i está dado por el siguiente modelo jerárquico Bayesiano no paramétrico:

$$Y_i|F \stackrel{iid}{\sim} g(\cdot) := \int_{\Theta} K(\cdot|\theta) dF(\theta), \quad i = 1, 2, \dots, n$$

$$F|H_{\zeta} \sim DP(M, F_{0|H_{\zeta}}), \quad \zeta \in \{0, 1\} \quad (2-13)$$

$$H_{\zeta} \sim \pi_{\mathcal{M}}, \quad \mathcal{M} \in \{H_0, H_1\}$$

Donde en el primer nivel del modelo 2-13 se tiene que $K(\cdot|\theta)$ corresponde a un kernel continuo, que se asume como Gaussiano bivariado, y $F(\theta)$ es la a priori para el vector $\theta(\mu, \Sigma)$, la cual se asume sigue un proceso Dirichlet con parámetros M , llamado parámetro de precisión que indica la variabilidad de las realizaciones del proceso, y $F_{0|H_{\zeta}}$, que se conoce como medida base y hace referencia a una distribución *spike-slab* cuya actualización viene indexada por la hipótesis. Por último, se tiene en la jerarquía la expresión $\pi_{\mathcal{M}}$, que hace referencia a la distribución a priori no informativa de Jeffreys para proporciones, muy conveniente cuando no se tiene información sobre lo plausible que es cada hipótesis en la realidad.

Ahora, recordando que uno de los problemas principales de las técnicas tradicionales desde el enfoque de modelo mixto, es que no permiten la captura de correlaciones negativas de las observaciones (Zimmerman 1997), los autores Pereira et al. (2020) proponen una parametrización especial para el kernel, donde:

$$\mu = \begin{pmatrix} \beta_1 \\ \beta_1 + \beta_2 \end{pmatrix}; \quad \Sigma = \begin{pmatrix} \sigma_1^2 + \tau^2 & (1 - 2\gamma_1)(1 - 2\gamma_2)\tau^2 \\ (1 - 2\gamma_1)(1 - 2\gamma_2)\tau^2 & \sigma_1^2\sigma_2^2 + \tau^2 \end{pmatrix} \quad (2-14)$$

Esta parametrización (ver ecuación 2-14) es el resultado de considerar que usualmente las observaciones repetidas Y_{ij} , $i = 1, 2, \dots, n$, $j = 1, 2$, pueden ser expresadas en función de efectos fijos y aleatorios, donde los efectos aleatorios describen las diferencias entre los sujetos. En consecuencia, Y_{ij} puede expresarse como un modelo mixto:

$$\begin{aligned}
Y_{ij} &= \beta_1 + \beta_2 X_{ij} + Z_{ij} \delta_i + \epsilon_{ij}, \quad i = 1, 2, \dots, n, \quad j = 1, 2, \\
\epsilon_{i1} &\stackrel{iid}{\sim} N(0, \sigma_1^2), \\
\epsilon_{i2} &\stackrel{iid}{\sim} N(0, \sigma_1^2 \sigma_2^2), \\
\delta_i &\stackrel{iid}{\sim} N(0, \tau^2)
\end{aligned} \tag{2-15}$$

Donde Y_{ij} en la ecuación 2-15 es el valor que toma la variable de respuesta para el individuo i en el tiempo j ; β_1 y β_2 son los efectos fijos y a su vez, β_1 representa el valor medio de Y_1 y β_2 hace alusión a la diferencia media que presentan los individuos en el tiempo 2 (Y_2) con respecto al tiempo 1 (Y_1); X_{ij} corresponde a un indicador sujeto al tiempo en el que se encuentre el individuo i , donde toma el valor de 0 si se encuentra en el tiempo $j = 1$ y 1 cuando se encuentra en el tiempo $j = 2$; δ_i es un efecto aleatorio que representa la variabilidad de que un individuo se encuentre en el grupo j ; ϵ_{ij} hace referencia al error aleatorio que presenta cada uno de los sujetos dado el tiempo puntual en el que se esté observando la medición y Z_{ij} es una variable latente que toma los valores $\{-1, 1\}$ con probabilidades dadas por la siguiente función de masa:

$$P(Z_{ij} = z_{ij}) = \begin{cases} \gamma_j & \text{si } z_{ij} = -1 \\ 1 - \gamma_j & \text{si } z_{ij} = 1 \end{cases} \tag{2-16}$$

A partir de la cual se tiene que γ_j es un valor entre 0 y 1 que sigue una distribución $Beta(a, b)$. Por otro lado, $\frac{1}{\tau^2}$ es la precisión (parámetro del efecto aleatorio que permite determinar el grado de error que se tiene en las mediciones realizadas) y tiene asociada una distribución $Ga(a_0, b_0)$. En cuanto a los efectos y errores aleatorios ($\delta_i, \epsilon_{i1}, \epsilon_{i2}$) se puede decir que son independientes entre sí.

La parametrización en la ecuación 2-14 resulta ser útil en lo que respecta a la validación de la hipótesis sobre las diferencias existentes entre las distribuciones marginales del conjunto de datos pareados, puesto que tanto la distribución conjunta de los datos como las marginales presentan una estructura más simple y sencilla al momento de realizar el muestreo sobre la distribución a posteriori, tal y como se muestra a continuación:

$$\begin{aligned}
G(\cdot) &= \sum_{h \geq 1} w_h \Phi_2(\cdot | \mu_h, \Sigma_h), \\
G_1(\cdot) &= \sum_{h \geq 1} w_h \Phi(\cdot | \beta_{1h}, \sigma_{1h}^2 + \tau_h^2), \\
G_2(\cdot) &= \sum_{h \geq 1} w_h \Phi(\cdot | \beta_{1h} + \beta_{2h}, \sigma_{1h}^2 \sigma_{2h}^2 + \tau_h^2)
\end{aligned} \tag{2-17}$$

En donde $\Phi(\cdot)$ en la ecuación 2-17 representa la función de densidad Gaussiana acumulada y w_h hace referencia al peso asociado a cada uno de los clústeres. Además, esta parametrización permite capturar las correlaciones positivas y negativas que puedan existir entre las observaciones, como se ilustra de manera subsecuente:

$$-1 < Cor(Y_1, Y_2) < 1 \quad (2-18)$$

La comprensión de valores dentro del rango de -1 a 1 en la desigualdad 2-18 se debe explícitamente a la parametrización de Σ , ya que con esta se tienen tres escenarios posibles dependiendo del valor que pueda tomar la covarianza:

- Si $\gamma_1 \approx 0.5$ ó $\gamma_2 \approx 0.5$, entonces $(1 - 2\gamma_1)(1 - 2\gamma_2)\tau^2 \approx 0$.
- Si $\gamma_1 < 0.5$ y $\gamma_2 < 0.5$, entonces $(1 - 2\gamma_1)(1 - 2\gamma_2)\tau^2$ toma un valor positivo.
- Si $\gamma_1 < 0.5$ y $\gamma_2 > 0.5$ ó $\gamma_1 > 0.5$ y $\gamma_2 < 0.5$, entonces $(1 - 2\gamma_1)(1 - 2\gamma_2)\tau^2$ toma un valor negativo.

Por último, para configurar el test de hipótesis, Pereira et al. (2020) utilizan como medida base en el proceso Dirichlet una distribución *spike-slab* (ver ecuación 2-19), la cual según Ishwaran & Rao (2005) es útil para reducir la incertidumbre de los modelos, además de que resulta ser adecuada en este contexto para determinar las diferencias entre grupos, pues en presencia de la hipótesis alterna se tomará el componente *slab* de la densidad y, en caso contrario, los átomos del segundo instante tendrán asociada una distribución con $\beta_{2h} \rightarrow 0$ y $\sigma_{2h}^2 \rightarrow 1$, lo que quiere decir que la distribución toma la forma *spike*. En este sentido, la medida base se define como:

$$F_{0|H_\zeta} : N_2((\beta_{1h}, \beta_{2h})^T | \mu_0 = (0, 0)^T, \Sigma_0 = \text{diag}[\psi, \kappa(\mathbf{1}_{(\zeta=1)} + \nu_0 \mathbf{1}_{(\zeta=0)})]) \times Ga(1/\sigma_{1h}^2 | \epsilon, \epsilon) Ga(1/\sigma_{2h}^2 | s(\mathbf{1}_{(\zeta=1)} + \nu_0^{-1} \mathbf{1}_{(\zeta=0)}), s(\mathbf{1}_{(\zeta=1)} + \nu_0^{-1} \mathbf{1}_{(\zeta=0)})) \quad (2-19)$$

Expresión en la que los hiperparámetros κ y s siguen distribuciones Gamma:

$$\begin{aligned} \frac{1}{\kappa} &\sim Ga(a_2 = 5, b_2 = 1) \\ s &\sim Ga(a_3 = 5, b_3 = 50) \end{aligned} \quad (2-20)$$

Para el establecimiento de los valores con los que se inicializan los parámetros de κ y s de la ecuación 2-20, se considera lo expuesto por Ishwaran & Rao (2005), de tal forma que $Var(\beta_{2h})$ y $Var(1/\sigma_{2h}^2)$ tengan una distribución continua bimodal, con una *spike* en $\nu_0 = 0.1$ y asimetría positiva.

2.5.1. Algoritmo de la prueba BNP para muestras pareadas

El algoritmo para la implementación de esta prueba de hipótesis tiene su fundamento en el muestreador de Gibbs y consta de ocho pasos generales en los que se actualizan los átomos (parámetros), se hace uso del método *slice sampling* para la actualización de los clústeres y se actualiza el valor que toma la hipótesis de estudio. Debido a que esta prueba se enmarca en el campo Bayesiano, todos los pasos —y subpasos— que componen su algoritmo surgen a partir de los cálculos de las distribuciones a posteriori considerando el modelo 2-15.

Paso 1. Actualización de los átomos:

$$p(\theta_h | \dots) \propto f_{0|H_\zeta} \prod_{i:d_i=h} N_2(y_i | \theta_h)$$

Subpaso 1.1. Actualización de μ_h :

$$(\beta_1, \beta_2 | \dots) \sim N_2 \left[(nX^T \Sigma^{-1} X + \Sigma_0^{-1})^{-1} \left(X^T \Sigma^{-1} \sum_{i=1}^n Y_i + \Sigma_0^{-1} \mu_0 \right), (nX^T \Sigma^{-1} X + \Sigma_0^{-1})^{-1} \right]$$

Donde X es una matriz de diseño fijada de la manera que sigue:

$$X = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

La cual se dispone con el objetivo de referenciar únicamente a β_{1h} o a la suma de β_{1h} y β_{2h} .

Subpaso 1.2. Actualización de Σ_h :

La actualización de Σ_h , cuya expresión matricial incluye a σ_1^2 , σ_2^2 , τ^2 , γ_1 y γ_2 , requiere de la ejecución de los siguientes subpasos adicionales.

Subpaso 1.2.1. Actualización de σ_1^2 :

$$\frac{1}{\sigma_1^2} | \dots \sim Ga(n + \epsilon, \lambda_1);$$

$$\lambda_1 = \frac{\sum_{i=1}^n (y_{i1} - (\beta_1 + Z_{i1}\delta_i))^2}{2} + \frac{\sum_{i=1}^n (y_{i2} - (\beta_1 + \beta_2 + Z_{i2}\delta_i))^2}{2\sigma_2^2} + \epsilon$$

Subpaso 1.2.2. Actualización de σ_2^2 :

$$\frac{1}{\sigma_2^2} | \dots \sim Ga \left(\frac{n}{2} + s (\mathbf{1}_{(\zeta=1)} + \nu_0^{-1} \mathbf{1}_{(\zeta=0)}) , \lambda_2 \right);$$

$$\lambda_2 = \frac{\sum_{i=1}^n (y_{i2} - (\beta_1 + \beta_2 + Z_{i2}\delta_i))^2}{2\sigma_1^2} + s \left(\mathbf{1}_{(\zeta=1)} + \nu_0^{-1} \mathbf{1}_{(\zeta=0)} \right)$$

Subpaso 1.2.3. Muestreo de δ_i :

$$\delta_i | \dots \sim N \left[\frac{\frac{Z_{i1}(y_{i1} - \beta_1)}{\sigma_1^2} + \frac{Z_{i2}(y_{i2} - \beta_1 - \beta_2)}{\sigma_1^2 \sigma_2^2}}{\frac{1}{\sigma_1^2} + \frac{1}{\sigma_1^2 \sigma_2^2} + \frac{1}{\tau^2}}, \frac{1}{\frac{1}{\sigma_1^2} + \frac{1}{\sigma_1^2 \sigma_2^2} + \frac{1}{\tau^2}} \right]$$

Subpaso 1.2.4 Muestreo de $Z_{i1} \in \{-1, 1\}$ con probabilidades:

$$P(Z_{i1} = -1 | \dots) \propto \gamma_1 \times \exp \left\{ \frac{-1}{2\sigma_1^2} (y_{i1} - \beta_1 + \delta_i)^2 \right\}$$

$$P(Z_{i1} = 1 | \dots) \propto (1 - \gamma_1) \times \exp \left\{ \frac{-1}{2\sigma_1^2} (y_{i1} - \beta_1 - \delta_i)^2 \right\}$$

Subpaso 1.2.5 Muestreo de $Z_{i2} \in \{-1, 1\}$ con probabilidades:

$$P(Z_{i2} = -1 | \dots) \propto \gamma_2 \times \exp \left\{ \frac{-1}{2\sigma_1^2 \sigma_2^2} (y_{i2} - \beta_1 - \beta_2 + \delta_i)^2 \right\}$$

$$P(Z_{i2} = 1 | \dots) \propto (1 - \gamma_2) \times \exp \left\{ \frac{-1}{2\sigma_1^2 \sigma_2^2} (y_{i2} - \beta_1 - \beta_2 - \delta_i)^2 \right\}$$

Subpaso 1.2.6. Muestreo de γ_j :

$$\gamma_j | \dots \sim \text{Beta} \left(a + \sum_{i=1}^n \mathbf{1}_{\{Z_{ij}=-1\}}, b + n - \sum_{i=1}^n \mathbf{1}_{\{Z_{ij}=-1\}} \right)$$

Subpaso 1.2.7. Muestreo de τ^2 :

$$\frac{1}{\tau^2} | \dots \sim \text{Ga} \left(a_0 + \frac{n}{2}, b_0 + \frac{\sum_{i=1}^n \delta_i^2}{2} \right)$$

Paso 2. Actualización de los pesos por *stick breaking*:

$$P(\nu_h | \dots) \propto \text{Beta} \left(1 + \sum_{i=1}^n \mathbf{1}_{(d_i=h)}, M + \sum_{i=1}^n \mathbf{1}_{(d_i>h)} \right)$$

Paso 3. Muestreo de las variables latentes u :

$$P(u_i | \dots) \propto \mathbf{1}_{(0 < u_i < w_{d_i})}$$

Paso 4. Actualización de las etiquetas de *cluster*:

$$P(d_i = k | \dots) \propto \mathbf{1}_{(k:w_k > u_i)} N_2(y_i | \theta_k)$$

Paso 5. Actualización de la hipótesis ζ :

$$P(\zeta = 1 | \dots) = \frac{\pi \prod_{h=1}^N N(\beta_{2,h} | 0, k) Ga(1/\sigma_{2,h}^2 | s, s)}{\pi \prod_{h=1}^N N(\beta_{2,h} | 0, k) Ga(1/\sigma_{2,h}^2 | s, s) + (1 - \pi) \prod_{h=1}^N N(\beta_{2,h} | 0, \kappa \nu_0) Ga(1/\sigma_{2,h}^2 | s \nu_0^{-1}, s \nu_0^{-1})}$$

Paso 6. Actualización de la distribución a priori de la hipótesis:

$$P(\pi | \dots) \propto Beta\left(\frac{1}{2} + \mathbf{1}_{(\zeta=1)}, \frac{3}{2} - \mathbf{1}_{(\zeta=1)}\right)$$

Paso 7. Actualización de κ :

$$P\left(\frac{1}{\kappa} | \dots\right) \propto Ga\left(a_2 + \frac{N}{2}, b_2 + \frac{\sum_{h=1}^N \beta_{2,h}^2}{2(\mathbf{1}_{(\zeta=1)} + \nu_0 \mathbf{1}_{(\zeta=0)})}\right)$$

Paso 8. Actualización de s :

$$P(s | \dots) \propto s^{a_3-1} \exp \left\{ -b_3 s + N s (\mathbf{1}_{(\zeta=1)} + \nu_0^{-1} \mathbf{1}_{(\zeta=0)}) \log(s (\mathbf{1}_{(\zeta=1)} + \nu_0^{-1} \mathbf{1}_{(\zeta=0)})) \right. \\ \left. - N \log(\Gamma(s (\mathbf{1}_{(\zeta=1)} + \nu_0^{-1} \mathbf{1}_{(\zeta=0)}))) + s (\mathbf{1}_{(\zeta=1)} + \nu_0^{-1} \mathbf{1}_{(\zeta=0)}) \left(\sum_{h=1}^N \log(1/\sigma_{2,h}^2) - 1/\sigma_{2,h}^2 \right) \right\}$$

3 Metodología

En esta sección se presenta la metodología utilizada para realizar la creación del paquete estadístico en el *software* R para la prueba de hipótesis Bayesiana no paramétrica para muestras pareadas (de forma abreviada, prueba de hipótesis BNP para muestras pareadas). Inicialmente se ilustra el proceso utilizado para crear las funciones en el *software* R a partir del algoritmo de la prueba BNP para muestras pareadas (ver sección 2.5.1); posteriormente, se presentan los pasos que permitieron dar lugar a la versión final del paquete que estará disponible para cualquier usuario y, finalmente, se expone una descripción de los datos usados para mostrar el funcionamiento del paquete.

3.1. Creación de las funciones en el software R

En este apartado se presentan los pasos utilizados para realizar la configuración del algoritmo de la prueba BNP propuesta por Pereira et al. (2020) en el *software* R, la cual fue denominada como *BNP.test*. Cada uno de las figuras aquí mostradas corresponden a la versión final de la función que fue cargada al repositorio de GitHub.

Paso 0. Establecimiento de variables y valores iniciales.

```
# Definition of variables and initial values -----  
  
mu.0<-c(0,0)  
  
a2<-5  
b2<-1  
  
v0<-0.1  
  
epsilon<-0.01  
  
a3<-50  
b3<-5  
  
a0<-0.01  
b0<-0.01
```

Figura 3-1: Valores iniciales de la prueba en GitHub. **Fuente:** Elaboración propia.

En primer lugar, se definieron todas las variables y valores iniciales de los hiperparámetros que permitieron llevar a cabo la prueba de hipótesis en su totalidad, tal y como se muestra

parcialmente en la Figura 3-1. Además, a manera de inicialización, se asignaron de forma aleatoria los clústeres y se calcularon los parámetros con base en los datos ingresados por parte del usuario.

Ahora bien, los pasos 1 a 4 de la prueba BNP para muestras pareadas corresponden al método *slice sampling* desarrollado por Walker (2007), sin embargo, se programaron en R con algunas modificaciones estructurales para una actualización adecuada en cada iteración del algoritmo de Gibbs.

Paso 1. Actualización de los átomos.

Como primera medida, se ilustran los pasos del muestreador de Gibbs para realizar la inferencia a posteriori de los átomos (μ, Σ) . Todos los subpasos aquí expuestos se agruparon en una función que se itera según el número de simulaciones especificadas por el usuario.

Subpaso 1.1. Actualización de μ_h .

```
## Updating the mean of conditional distributions

sigma0<-diag(c(10,kappa),nrow=2)

product.0<-solve(sigma0) %>% mu.0

matrix_var<-pracma::pinv(n*t(X)%%pracma::pinv(matrix(c(variance1 +tau ,rep((1-2*gamma1)*(1-2*gamma2)*tau,2), tau +(variance1*variance2)),byrow=F,ncol=2))%*X + solve(sigma0))
matrix_mu<-matrix_var %%% (product.0 + t(X)%%pracma::pinv(matrix(c(variance1 +tau ,rep((1-2*gamma1)*(1-2*gamma2)*tau,2), tau +(variance1*variance2)),byrow=F,ncol=2))
%%matrix(c(sum.y1,sum.y2)))

betas<-MASS::mvrnorm(n = 1,matrix_mu,matrix_var)
Beta1<-betas[1]
Beta2<-betas[2]
```

Figura 3-2: Código en GitHub para la actualización de los valores medios de las distribuciones condicionales. **Fuente:** Elaboración propia.

Para la actualización de los valores medios (β_1, β_2) de las distribuciones condicionales se estimaron los parámetros asociados a la distribución Gaussiana bivariada, aquí denominados como *matrix_var* y *matrix_mu* (ver Figura 3-2). Con tales estimaciones, se obtuvieron de forma aleatoria los dos valores para cada β_j considerando la función *mvrnorm* de la librería *MASS*.

Subpaso 1.2. Actualización de δ_i .

Para cada uno de los sujetos de estudio, se actualizó el efecto aleatorio δ , partiendo de una distribución Gaussiana mediante la función de base *rnorm* en R, cuyos parámetros se pueden observar en la Figura 3-3 como *mean.delta* y *variance.delta*.

```
# Updating delta

numerator<-(z1*(y1-Beta1)/variance1)+(z2*(y2-Beta1-Beta2)/(variance1*variance2))
denominator<-(1/variance1)+(1/(variance2*variance1))+(1/tau)

mean.delta<-numerator/denominator

variance.delta<-1/denominator

delta<-rnorm(n,mean.delta, sqrt(variance.delta))
```

Figura 3-3: Código en GitHub para la actualización de los efectos aleatorios. **Fuente:** Elaboración propia.

Subpaso 1.3. Actualización de τ^2 .

```
# Updating tau

tau<-1/rgamma(1,shape=a0+(n/2), rate=(1/2)*(sum(delta^2)) + b0)
```

Figura 3-4: Código en GitHub para la actualización de τ^2 . **Fuente:** Elaboración propia.

El muestreo del valor de τ^2 en cada iteración se estableció como la inversa del valor aleatorio generado a partir de una distribución Gamma, como se muestra en la Figura 3-4.

Subpaso 1.4. Actualización de σ_1^2 .

```
# Updating sigma1

variance1<-1/rgamma(1,shape=n+epsilon, rate=(1/2)*sum((y1-(Beta1+z1*delta))^2)+(1/(2*variance2))*sum((y2-(Beta1+Beta2+z2*delta))^2)+epsilon)
```

Figura 3-5: Código en GitHub para la actualización de σ_1^2 . **Fuente:** Elaboración propia.

Uno de los subpasos más relevantes concierne a la actualización de σ_1^2 , que se define en el algoritmo como la inversa de una distribución Gamma, considerando los valores ya actualizados de β_j y δ_i (ver Figura 3-5).

Subpaso 1.5. Actualización de σ_2^2 .

Pese a que la actualización de σ_2^2 también se implementa en concordancia con la inversa de una distribución Gamma, fue necesario establecer previamente una declaración condicional (ver Figura 3-6), pues sus hiperparámetros se encuentran indexados según el valor que presente la hipótesis en cada iteración del algoritmo.


```

# Updating sigma2

if(hypothesis==1){

  param1.sigma2.alt<-s
  param2.sigma2.alt<-s

  param1.sigma2<-s/v0
  param2.sigma2<-s/v0

}else{

  param1.sigma2.alt<-s*v0
  param2.sigma2.alt<-s*v0

  param1.sigma2<-s
  param2.sigma2<-s

}

if(hypothesis==1){
  param1.final.sigma2<-param1.sigma2.alt
  param2.final.sigma2<-param2.sigma2.alt
}else{
  param1.final.sigma2<-param1.sigma2
  param2.final.sigma2<-param2.sigma2
}

alpha.sigma2<-param1.final.sigma2*(n/2)
beta.sigma2<-(1/(2*variance1))*sum((y2-(Beta1+Beta2+z2*delta))^2) + param2.final.sigma2

variance2<-1/rgamma(1,shape=alpha.sigma2, rate=beta.sigma2)

```

Figura 3-6: Código en GitHub para la actualización de σ_2^2 . **Fuente:** Elaboración propia.

Subpaso 1.6. Actualización de z_{i1} .

Debido a que z_{i1} es una variable dicotómica, fue programada en R mediante la función *sample*, de acuerdo con las probabilidades establecidas para cada valor como se muestran en la Figura 3-7. Así mismo, cabe resaltar que las expresiones de cada una de estas probabilidades fueron abordadas mediante la distribución Gaussiana, teniendo en cuenta que su estructura es proporcional a la expresión presentada en el algoritmo de la prueba BNP.

```

# Updating z1

prob.post.z1_neg<-gamma1*dnorm(y1,mean=Beta1-delta,sd=sqrt(variance1))
prob.post.z1_positivo<- (1-gamma1)*dnorm(y1,mean=Beta1+delta,sd=sqrt(variance1))
vec.prob_z1<-matrix(c(prob.post.z1_neg/(prob.post.z1_neg+prob.post.z1_positivo), prob.post.z1_positivo/(prob.post.z1_neg+prob.post.z1_positivo)),ncol=2)

k<-matrix(c(1:n),ncol=1)
z1<-apply(k,1, function(k) sample(c(-1,1),1,replace=F, prob=vec.prob_z1[k,]))

```

Figura 3-7: Código en GitHub para la actualización de z_{i1} . **Fuente:** Elaboración propia.

Subpaso 1.7. Actualización de z_{i2} .

La actualización de z_{i2} fue realizada tal y como en el subpaso anterior, como se observa en la Figura 3-8.

```
# Updating z2

prob.post.z2_neg<-gamma2*dnorm(y2,mean=Beta1+Beta2-delta,sd=sqrt(variance1*variance2))
prob.post.z2_positivo<-(1-gamma2)*dnorm(y2,mean=Beta1+Beta2+delta,sd=sqrt(variance1*variance2))

vec.prob_z2<-matrix(c(prob.post.z2_neg/(prob.post.z2_neg+prob.post.z2_positivo), prob.post.z2_positivo/(prob.post.z2_neg+prob.post.z2_positivo)),ncol=2)

z2<-apply(k,1, function(k) sample(c(-1,1),1,replace=F, prob=vec.prob_z2[k,]))
```

Figura 3-8: Código en GitHub para la actualización de z_{i2} . **Fuente:** Elaboración propia.

Subpaso 1.8. Actualización de γ_j .

```
# updating gamma.j

gamma1<-rbeta(1, shape1=a.gamma+ sum(z1==1), shape2=b.gamma-sum(z1==1)+n)

gamma2<-rbeta(1, shape1=a.gamma+ sum(z2==1), shape2=b.gamma-sum(z2==1)+n)
```

Figura 3-9: Código en GitHub para la actualización de γ_j . **Fuente:** Elaboración propia.

Para finalizar con la actualización de los átomos, se procedió con la extracción de dos valores aleatorios para γ_j a partir de distribuciones Beta (ver Figura 3-9), los cuales alimentarán los subpasos 1.6. y 1.7 en cada paso del algoritmo de Gibbs.

Paso 2. Actualización de los pesos por el proceso *stick breaking*.

El proceso denominado *stick breaking*, que permite la actualización de los pesos del modelo de mezcla Dirichlet, fue programado en R según lo mostrado en la Figura 3-10, haciendo uso de las funciones *apply*, para la extracción de valores aleatorios de una distribución Beta (*rbeta*) según el número de clústeres, y *cumprod*, para el cálculo de la suma acumulada de los pesos.

Este procedimiento se realizará para cada clúster por cada una de las iteraciones del algoritmo y, ya que es posible que la suma de los pesos sea inferior a 1 dada la naturaleza aleatoria del proceso, fue necesario asociar mediante una estructura condicional un peso y parámetros adicionales, siendo estos últimos asignados según la información a priori establecida en la prueba de hipótesis.

```

# Updating the weights by stick-breaking process

v.j<-apply(b,1,function(b) rbeta(1,1+sum(cluster.0==b),M + sum(cluster.0>b)))
prod.cum<-cumprod(1-v.j)
w<-v.j*c(1,prod.cum[-length(prod.cum)])

if(sum(w)<1){
  p.j<-c(w,1-sum(w))
}else{
  p.j<-w
}

if(ncol(parameters)<length(p.j)){

  sigma0<-diag(c(10,kappa),nrow=2)

  param1.final.sigma2<-s
  param2.final.sigma2<-s

  mu1<-MASS::mvrnorm(n = 1,mu.0,sigma0)
  variance1.j<-1/rgamma(1,shape=epsilon, rate=epsilon)
  variance2.j<-1/rgamma(1,shape=param1.final.sigma2, rate=param2.final.sigma2)
  tau.j<-1/rgamma(1,shape=a0, rate=b0)

  gamma1.j<-rbeta(1, shape1=a.gamma, shape2=b.gamma)
  gamma2.j<-rbeta(1, shape1=a.gamma, shape2=b.gamma)

  parameters<-cbind(parameters,c(mu1,variance1.j,variance2.j,gamma1.j,
                                gamma2.j,tau.j))
}

```

Figura 3-10: Proceso *stick breaking* para los pesos del modelo. **Fuente:** Elaboración propia.

Paso 3. Muestreo de la variable latente u .

En este paso, se asigna un valor aleatorio a partir de una distribución uniforme para cada uno de los individuos de la muestra (ver Figura 3-11). Esta actualización se encuentra condicionada a los pesos obtenidos por el método *stick breaking*, ya que los parámetros requeridos para la simulación de los valores aleatorios siempre van a estar entre 0 y el peso de cada clúster en el que se encuentre el individuo. Es importante resaltar que este paso y el siguiente permiten truncar los infinitos parámetros del modelo de mezcla Dirichlet asumido.

```

# Sampling the latent variable

u<-rep(NA,nrow(data.initial))
data1<-cbind(nrow(data.initial),cluster.0,u)
g<-sort(unique(cluster.0))
sort.clust<-matrix(g,ncol=1)

for (i in 1:length(sort.clust)){
  data1[data1[,2]==sort.clust[i],3]<-runif(sum(cluster.0==sort.clust[i]),0,p.j[i])
}

```

Figura 3-11: Simulación de la variable latente. **Fuente:** Elaboración propia.

Paso 4. Actualización de la densidad bivariada.

Para actualizar la etiqueta de cada individuo se calcula la densidad bivariada para cada clúster a partir de los átomos obtenidos en el paso 1 y se multiplica por una matriz de 1's y 0's con i filas correspondientes a cada individuo y j columnas correspondientes a la cantidad de clústeres. La determinación de los valores de la matriz se realiza a partir del paso anterior, en el que Walker (2007) propone la condición de establecer una variable indicadora que toma el valor de 1 en caso de que el peso del cluster en el que se encuentra asociado el individuo sea mayor al valor de la variable latente y 0 en caso contrario (ver Figura 3-12).

```
# Updating the bivariate density

x<-matrix(c(1:length(p.j)),ncol=1)
l<-apply(x,1,function(i) p.j[i] > u.j)*1
ll<-apply(x,1,function(i) l[,i]*i)
N<-apply(ll,1,max)
k<-1:max(N)

r<-matrix(k,ncol=1)
indicadoras<-apply(r,1,function(j) (p.j[j]>u.j)*1)

X<-matrix(c(1,1,0,1),ncol=2)

densities<-apply(r,1, function(j) mvtnorm::dmvnorm(sample.y, mean=t(X%*parameters[1:2,j]),
  sigma=matrix(c(parameters[3,j]+parameters[7,j],rep((1-2*parameters[5,j])*(1-2*parameters[5,j])*parameters[7,j],2),
    (parameters[3,j]*parameters[4,j]+parameters[7,j]),ncol=2)))
```

Figura 3-12: Obtención de las densidades bivariadas. **Fuente:** Elaboración propia.

Paso 5. Actualización del parámetro de masa M .

```
# Updating the Mass parameter

eta<-rbeta(1,M+1,nrow(data.initial))

tau.eta<-(a1 + ncol(parameters)-1)/(nrow(data.initial)*b1 - nrow(data.initial)*log(eta)+a1+ncol(parameters)-1)

u.M<-runif(1)
if(u.M<tau.eta){
  M<-rgamma(1,shape=a1+ncol(parameters), rate=b1 - log(eta))
}else{
  M<-rgamma(1,shape=a1+ncol(parameters)-1, rate=b1 - log(eta))
}
```

Figura 3-13: Muestreo del valor del parámetro de masa. **Fuente:** Elaboración propia.

El parámetro de masa o concentración M del proceso Dirichlet también debe ser actualizado, considerando particularmente la cantidad de clústeres al finalizar el algoritmo *slice-sampling* en cada iteración, los cuales pueden haber incrementado, decrecido o

mantenido. Adicionalmente, es importante mencionar que el valor de M es generado aleatoriamente de un modelo de mezcla conformado por dos densidades Gamma.

Paso 6. Actualización de la hipótesis ζ .

Para la actualización de la hipótesis recurrimos al teorema de Bayes, que en términos matemáticos se expresa como:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \quad (3-1)$$

Donde B es el evento del que se tiene conocimiento y A hace referencia al conjunto de posibles causas (excluyentes entre sí) que pueden producir el evento. En este sentido, $P(A|B)$ son las probabilidades a posteriori, $P(A)$ es la probabilidad a priori y $P(B|A)$ es la probabilidad de que ocurra B en cada una de las hipótesis de A . Para nuestro caso, se calculó la probabilidad a posteriori para la hipótesis alternativa H_1 y se recurrió al uso del logaritmo natural en las fórmulas por cuestiones de convergencia y facilidad a nivel operacional; considerando que ζ es una variable dicotómica se muestreó del vector $(0, 1)$ a través de la función *sample* en R teniendo en cuenta las probabilidades previamente calculadas (ver Figura 3-14).

```
# Updating the hypothesis

if(hypothesis==1){

  joint.1<-function(l) dnorm(l[2], mean = mu.0[2], sd = sqrt(kappa), log = TRUE)+
    dgamma(1/l[4],shape=s,rate=s,log = TRUE)

  log.post.zeta.1<-log(pi.0)+sum(apply(parameters,2,join.1))

  joint.0<-function(l) dnorm(l[2], mean = mu.0[2], sd = sqrt(kappa*v0), log = TRUE)+
    dgamma(1/l[4],shape=s*(1/v0),rate=s*(1/v0),log = TRUE)

  log.post.zeta.0<- log(1-pi.0)+sum(apply(parameters,2,join.0))
}

if(hypothesis==0){

  joint.1<-function(l) dnorm(l[2], mean = mu.0[2], sd = sqrt(kappa*(1/v0)), log = TRUE)+
    dgamma(1/l[4],shape=s*v0,rate=s*v0,log = TRUE)

  log.post.zeta.1<-log(pi.0)+sum(apply(parameters,2,join.1))

  joint.0<-function(l) dnorm(l[2], mean = mu.0[2], sd = sqrt(kappa), log = TRUE)+
    dgamma(1/l[4],shape=s,rate=s,log = TRUE)

  log.post.zeta.0<- log(1-pi.0)+sum(apply(parameters,2,join.0))
}

vec.prob.zeta<-c(exp(log.post.zeta.1)/(exp(log.post.zeta.1)+exp(log.post.zeta.0)),exp(log.post.zeta.0)/(exp(log.post.zeta.1)+exp(log.post.zeta.0)))
hypothesis<-sample(c(1,0),1,replace=F, prob=vec.prob.zeta)
```

Figura 3-14: Obtención del valor de la hipótesis. **Fuente:** Elaboración propia.

Paso 7. Actualización de los hiperparámetros π , κ y s .

Según lo expresado en el algoritmo de la prueba BNP para muestras pareadas, los hiperparámetros π , κ y s se actualizan dependiendo del valor que tome la hipótesis. Ante esto, se establecieron dos grandes estructuras condicionales que agruparan dichas actualizaciones, siendo en los dos primeros casos (π, κ) dadas por distribuciones Beta y Gamma, respectivamente, mientras que en el segundo caso (s) se estableció una estructura basada en el algoritmo de Metrópolis, en la cual se consideró una distribución Gaussiana truncada en correspondencia con el espacio paramétrico asociado a s , como se muestra en la Figura 3-15.

```
# Updating de Pi, Kappa and S

if(hypothesis==1){

  pi.0<-rbeta(1,(1/2)+1,(3/2)-1)

  kappa<-1/rgamma(1, a2+(N/2),b2+(sum(parameters[2,]))^2/(2))

  log.post<-function(x){
    (a3-1)*log(x) - x*b3 + ncol(parameters)*x*log(x) - ncol(parameters)*(lgamma(x))+
    (x-1)*sum(log(1/parameters[4,])) - x*(sum(1/parameters[4,]))
  }

  stars.s2<-truncnorm::rtruncnorm(1, a=0.001, b=Inf, mean = s, sd = 0.3)

  log.r<-log.post(stars.s2)-log(truncnorm::dtruncnorm(stars.s2, a=0.001, b=Inf, mean = s, sd = 0.3))-
  log.post(s)+log(truncnorm::dtruncnorm(s, a=0.001, b=Inf, mean = stars.s2, sd = 0.3))

  if(log(runif(1))< min(0,log.r)){
    s<-stars.s2
  }
}
```

Figura 3-15: Cálculo de los hiperparámetros π , κ y s . **Fuente:** Elaboración propia.

Todos estos pasos del algoritmo son ejecutados tantas veces como el usuario lo requiera, obteniendo una cantidad total de estimaciones de parámetros, hipótesis y densidades. Con el objetivo de eliminar la posible dependencia que existe en cada iteración, se realizó un proceso de quemado del primer 20 % de los resultados, así como un proceso de salto cada 8 observaciones. Posterior a esto, se extraen los parámetros, se calcula la media a posteriori que facilita la conclusión acerca de la prueba de hipótesis y se provee un gráfico interactivo con las densidades resultantes.

Adicionalmente, se programaron en R dos funciones complementarias a la prueba de hipótesis: a) *plotshift.function*, para la visualización de la función *shift*, y b) *contours.plot*, para la visualización de la distribución conjunta por medio de contornos en un gráfico bidimensional.

3.2. Creación del paquete en el software R

Si bien la creación de paquetes puede ser abordada desde distintos ángulos, para el desarrollo de este paquete se examinaron detalladamente los lineamientos de Wickham (2015). Es así como los pasos aquí expuestos contribuyen de manera conjunta al producto final que puede ser utilizado por los usuarios de R, el cual se ha denominado *BNPPairedSamples*.

3.2.1. Proyecto en R

En primer lugar, fue necesario crear un nuevo proyecto en la interfaz de RStudio (ver Figura 3-16). Este paso es fundamental para un desarrollo más organizado del paquete, pues en este proyecto se almacena toda la información relacionada con el paquete.

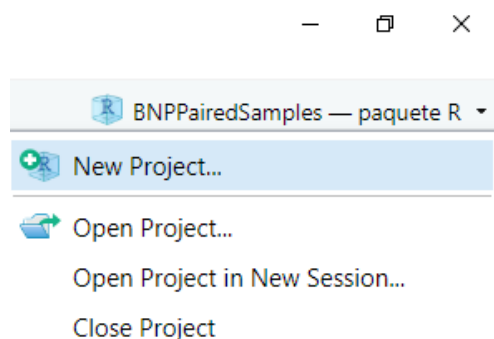


Figura 3-16: Creación de un nuevo proyecto en RStudio. **Fuente:** Elaboración propia.

3.2.2. Directorio de trabajo

Posterior a la creación de un nuevo proyecto, se muestran en RStudio los tres tipos de directorios que se pueden utilizar (ver Figura 3-17). En este caso, el proyecto se inició a partir de un nuevo directorio, sin embargo, se puede navegar a través de directorios de R ya existentes o clonar proyectos de repositorios como Git.

3.2.3. Tipo de proyecto

A continuación, se selecciona el tipo de proyecto a trabajar. En este caso en particular, se seleccionó la opción de *R Package*, como se muestra en la Figura 3-18.

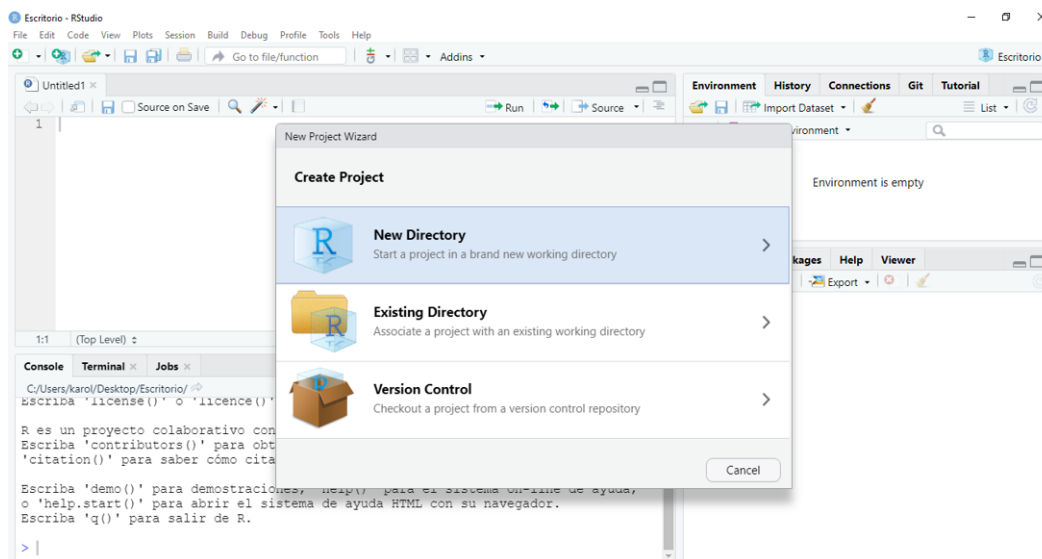


Figura 3-17: Creación de un nuevo directorio para el paquete. **Fuente:** Elaboración propia.

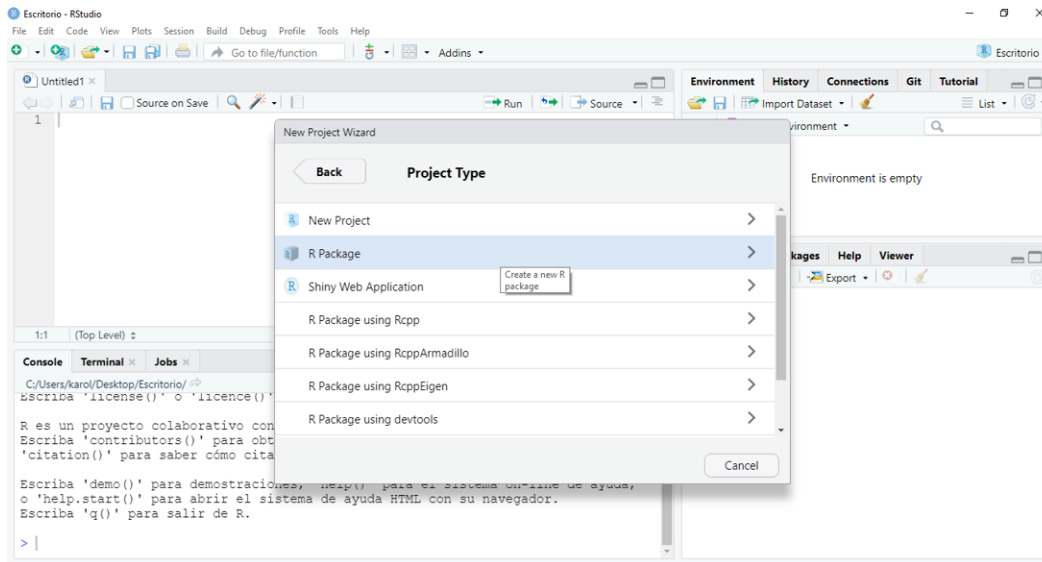


Figura 3-18: Elección del tipo de proyecto. **Fuente:** Elaboración propia.

3.2.4. Detalles del paquete

El siguiente paso hace alusión a la especificación de todos los detalles generales asociados al paquete (ver Figura 3-19), como el nombre con el que se define (*BNPPairedSamples*), la dirección exacta del directorio en el que se pretende alojar el paquete e información adicional

como la creación de un repositorio en Git. En principio, esta información genérica es suficiente para trabajar de forma activa en el paquete con todas sus funcionalidades.

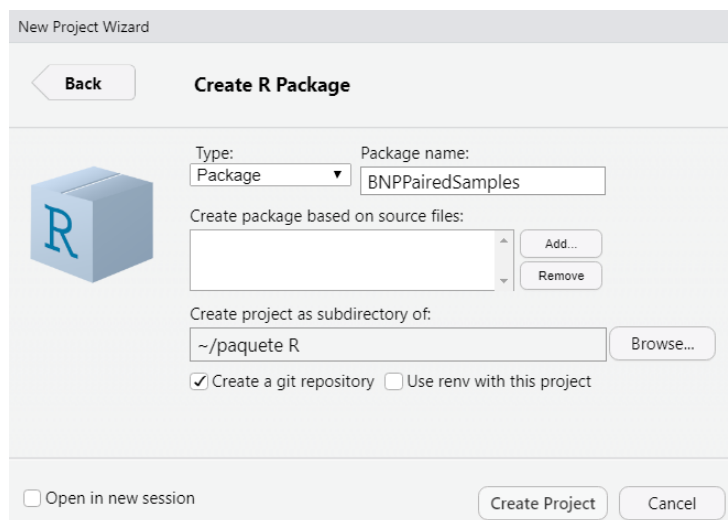


Figura 3-19: Detalles del paquete. **Fuente:** Elaboración propia.

3.2.5. Archivos esenciales del paquete

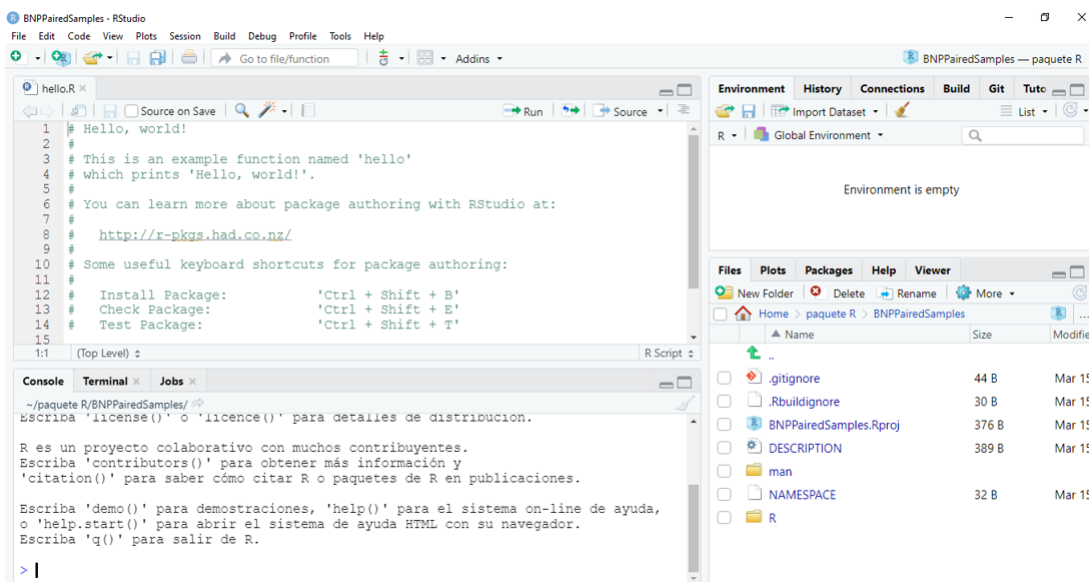


Figura 3-20: Generación de los archivos principales del paquete. **Fuente:** Elaboración propia.

La Figura 3-20 muestra todos los archivos esenciales (ubicados en el recuadro inferior derecho) que RStudio dispone de forma automática, basado en la información proporcionada hasta el momento. Este nuevo directorio contiene el proyecto establecido previamente, que destaca por su extensión *.Rproj*; los archivos *DESCRIPTION* y *NAMESPACE*; las carpetas denominadas *man* y *R* y, finalmente, los archivos *.gitignore* y *.Rbuildignore*, que permiten establecer qué archivos no se tendrán en cuenta al momento de añadir el paquete al repositorio local de Git o archivos de R que son confidenciales o no son necesarios para la construcción del paquete, respectivamente.

3.2.6. Estructuración del código en R y la documentación

El siguiente paso es consolidar todo el código en R de las funciones que se almacenarán en el paquete. En este caso se hizo uso de la librería *roxygen2*, que permite la escritura de todas las funciones en un solo *script* de R; así mismo, este archivo permite detallar la documentación asociada a cada una de estas funciones, que será visible cada que un usuario haga una llamada de ayuda por medio de *help()* o *?*, esto posterior a la instalación del paquete.

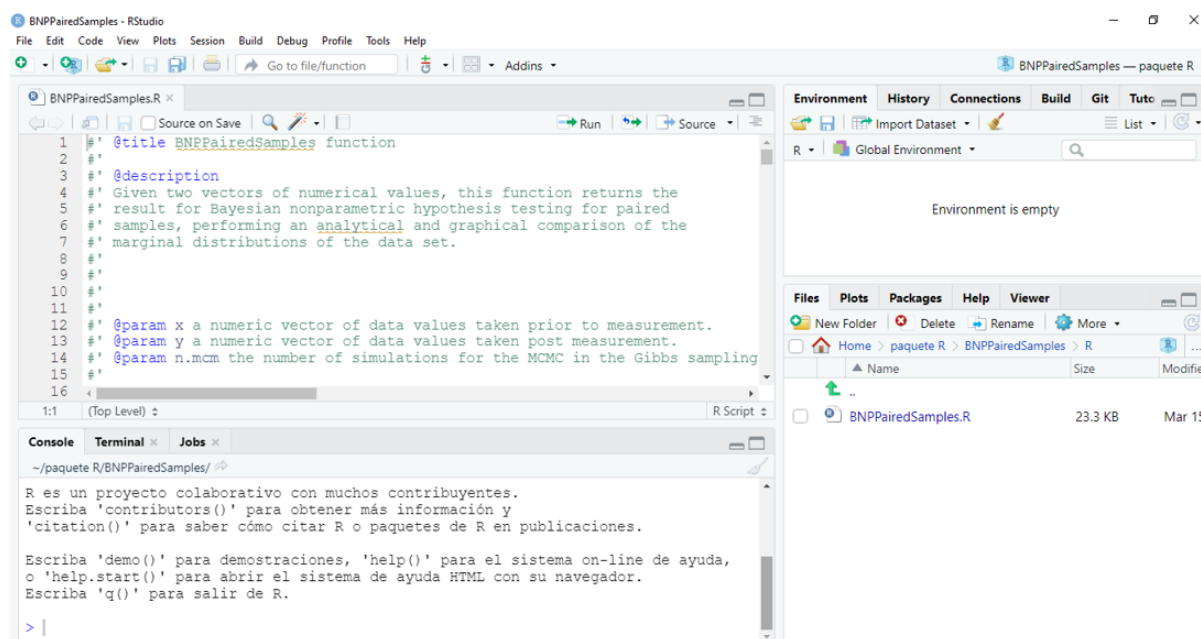


Figura 3-21: Archivo de código en R y documentación para las funciones *BNP.test*, *plotshift.function* y *contours.plot*. **Fuente:** Elaboración propia.

La Figura 3-21 muestra las primeras líneas de código de documentación del paquete *BNPPairedSamples*, que se compone de las tres funciones especificadas en la sección 3.1: 1. *BNP.test*, para la realización de la prueba BNP para muestras pareadas, brindando al usuario la probabilidad de obtener la hipótesis alterna y un gráfico con la media de las distribuciones

a posteriori; 2. *plotshift.function*, para graficar la función *shift*; y 3. *contours.plot*, para visualizar la distribución conjunta por medio del gráfico de contornos.

```
> library(usethis)
> library(roxygen2)
> library(devtools)
> devtools::document()
Updating BNPPairedSamples documentation
First time using roxygen2. Upgrading automatically...
Loading BNPPairedSamples
Writing NAMESPACE
Writing NAMESPACE
Writing BNP.test.Rd
Writing plotshift.function.Rd
Writing contours.plot.Rd
```

Figura 3-22: Ejecución de la función *document()*, haciendo uso de la librería *devtools*.
Fuente: Elaboración propia.

Finalmente, con todo el código y documentación estructurados, se procedió a realizar la carga de las librerías *usethis* y *devtools*, que permiten la ejecución de la función *document()* (ver Figura 3-22), con la finalidad de que se construyan los archivos en lenguaje Markdown de la documentación, los cuales se almacenan automáticamente en la carpeta *man*, como se muestra en la Figura 3-23.

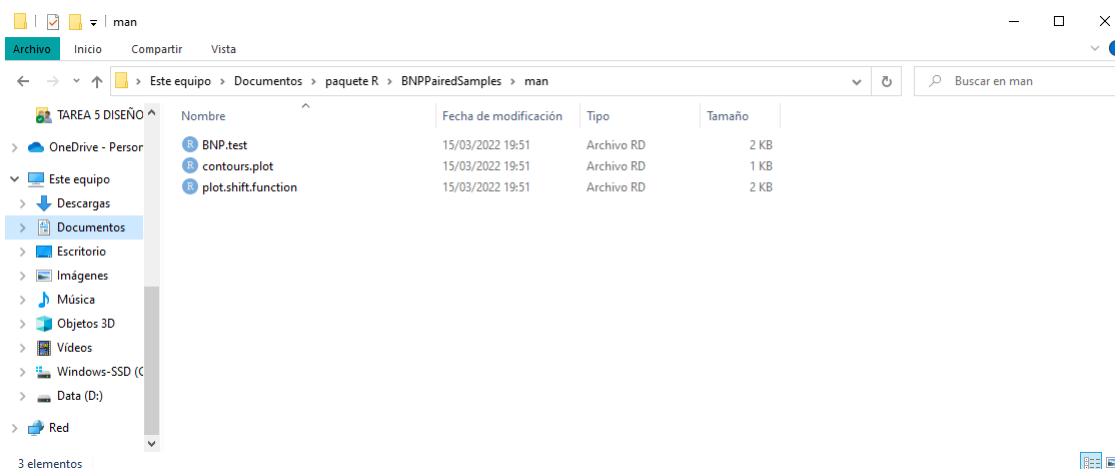


Figura 3-23: Archivos de documentación en la carpeta *man*. **Fuente:** Elaboración propia.

3.2.7. Archivo DESCRIPTION

El siguiente paso corresponde a la generación de la descripción del paquete a partir del archivo *DESCRIPTION* (ver Figura 3-24).



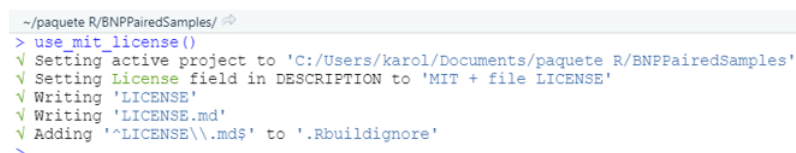
```

1 Package: BNPPairedSamples
2 Type: Package
3 Title: Bayesian Nonparametric Hypothesis Testing for Paired Samples
4 Version: 0.1.0
5 Authors@R: c(person(given = "Kevin",
6                     family = "Ortiz González",
7                     role = c("aut", "cre"),
8                     email = "kevin.ortiz@correounivalle.edu.co"),
9             person(given = "Karol Michelle",
10                  family = "Sandoval Burbano",
11                  role = c("aut")),
12             person(given = "Luz Adriana",
13                  family = "Pereira",
14                  role = c("aut")))
15 Description: This package performs hypothesis testing for paired samples within the frame
16 License: MIT + file LICENSE
17 Depends:
18   R (>= 2.10)
19 Imports:
20   MASS,
21   mvtnorm,
22   truncnorm,
23   pracma,
24   ggplot2,
25   plotly,
26   tidyr,
27   dplyr
28
29.5

```

Figura 3-24: Archivo *DESCRIPTION* del paquete. **Fuente:** Elaboración propia.

Para el caso particular de la licencia, fue necesario hacer uso del comando `use_mit_license()`, tal y como se expone en la Figura 3-25.



```

~/paquete R/BNPPairedSamples/
> use_mit_license()
√ Setting active project to 'C:/Users/karol/Documents/paquete R/BNPPairedSamples'
√ Setting License field in DESCRIPTION to 'MIT + file LICENSE'
√ Writing 'LICENSE'
√ Writing 'LICENSE.md'
√ Adding '^LICENSE\\.md$' to '.Rbuildignore'
>

```

Figura 3-25: Establecimiento de la licencia MIT. **Fuente:** Elaboración propia.

3.2.8. Carga de datos

Como en este caso se realizó la carga de conjuntos de datos simulados y reales, estos últimos provenientes de las librerías *datasets* y *datarium*, fue necesario almacenarlos en el paquete para que puedan ser usados de forma libre para probar el funcionamiento del paquete. La Figura 3-26 muestra la función que permite cargar estos datos en bruto, así como la visualización de la carpeta que se crea y en la que se almacena esta información.

```
> usethis::use_data(airquality, internal=FALSE)
√ Creating 'data/'
√ Saving 'airquality' to 'data/airquality.rda'
* Document your data (see 'https://r-pkgs.org/data.html')
```

(a) Ejecución de la función *use_data*.

Este equipo > Escritorio > BNPPairedSamples > data			
Nombre	Fecha de modificación	Tipo	Tamaño
airquality	15/03/2022 22:37	Archivo RDA	2 KB

(b) Visualización de la carpeta *data*.**Figura 3-26:** Carga de datos internos en el paquete. **Fuente:** Elaboración propia.

3.2.9. Viñetas de las funciones

Al momento de crear un paquete en R, no solo resulta relevante la información de ayuda básica que se encuentra alojada en la carpeta *man*, sino también la construcción de las viñetas. De acuerdo con Wickham (2015), estas son guías más largas que permiten contextualizar al usuario sobre el problema al que está dirigido el paquete, detallando sus funciones y la manera de usarlas con el objetivo de brindar una solución.

```
> usethis::use_vignette("BNP-test-vignette")
√ Writing 'vignettes/BNP-test-vignette.Rmd'
* Modify 'vignettes/BNP-test-vignette.Rmd'
```

(a) Ejecución de la función *use_vignette*.

```
1 ---
2 title: "Using the BNP Paired Samples Package"
3 author:
4 - "Kevin Ortiz Gonzalez"
5 - "Karol Michelle Sandoval"
6 date: "April 2022"
7 output: rmarkdown::html_vignette
8 vignette: >
9   %\VignetteIndexEntry{Using the BNP Paired Samples Package}
10  %\VignetteEngine{knitr::rmarkdown}
11  %\VignetteEncoding{UTF-8}
12 ---
```

(b) Información inicial de la viñeta.

Figura 3-27: Creación de las viñetas del paquete. **Fuente:** Elaboración propia.

La Figura 3-27 muestra, en primer lugar, la manera en la que se crea una viñeta usando la librería *usethis*; en segunda instancia, se observan las primeras líneas de código en R Markdown, formato que permite ejecutar texto y código en R. En este sentido, para el paquete *BNPPairedSamples* se estableció una viñeta para explicar al usuario las tres funciones que lo componen, de una forma mucho más específica y práctica. Como se verá más adelante, esta viñeta se puede ver directamente desde un repositorio bajo el nombre de *BNP-test-vignette* o se carga al momento de realizar la instalación del paquete, ante lo cual se genera en formato HTML.

3.2.10. Archivo README

El archivo *README* brinda a los usuarios un panorama general del uso del paquete, definiendo los casos en los que es de utilidad y presentando códigos genéricos en R para una adecuada instalación y ejecución de las funciones. En la Figura 3-28 es posible observar el código con el que se realiza la creación de este archivo, así como las primeras líneas del archivo por defecto. Ahora, vale la pena destacar que este archivo en R Markdown, si bien se encuentra disponible para descarga y visualización desde el repositorio, este no se carga al momento de realizar la instalación del paquete.

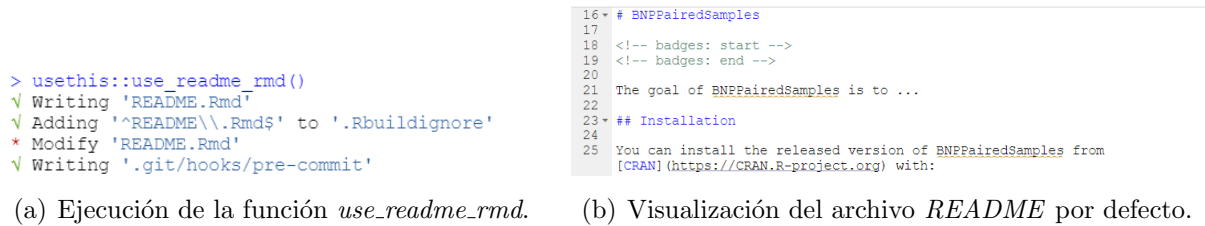


Figura 3-28: Archivo *README* del paquete. **Fuente:** Elaboración propia.

3.2.11. Carpeta de pruebas

La carpeta de pruebas, que se origina a partir del uso de la función `use_testthat()` del paquete `testthat` (ver Figura 3-29), permite formalizar unidades de prueba con el objetivo de establecer estándares propios de calidad del paquete (Wickham 2015). Aunque estas pruebas son establecidas por los mismos creadores del paquete, resulta ser una buena práctica conforme el paquete va actualizándose a lo largo del tiempo, pues se trata de pruebas que deben ser consistentes y generar los mismos resultados, independientemente de las características de forma o fondo que se hayan cambiado en versiones más recientes. En el caso del paquete *BNPPairedSamples*, se programaron cinco pruebas para la función *BNP.test* y una para las funciones *plotshift.function* y *contours.plot*.

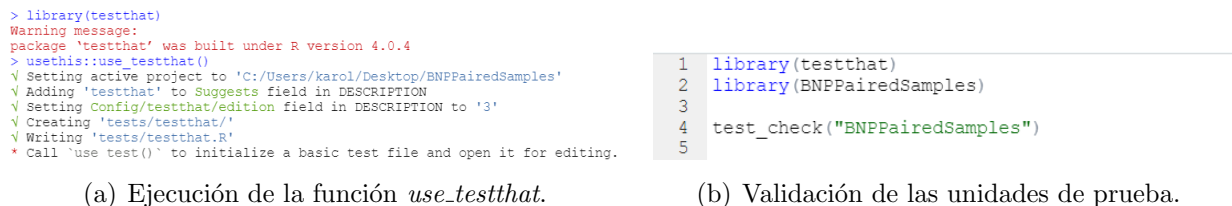


Figura 3-29: Construcción de las unidades de prueba. **Fuente:** Elaboración propia.

3.2.12. Git - GitHub

Con la finalidad de poder compartir con la comunidad el paquete desarrollado a nivel local en el *software* R, es importante recurrir a un repositorio remoto de dominio público para su divulgación. Para este caso, se hizo uso de GitHub como repositorio remoto y de Git, a través de su interfaz de Git Bash, para el control de versiones y sincronización entre los repositorios local y remoto, teniendo como referencia lo expresado por Wickham (2015).

Configuración inicial de Git (local)

Antes de comenzar a realizar el seguimiento de los archivos del paquete, es necesario realizar un proceso de configuración inicial, en el cual se establece el nombre y correo electrónico del

usuario local. Cabe destacar que el signo \$ presente en los códigos, representa un comando a ejecutar de la terminal de Git Bash.

1. Para configurar el nombre en Git, se usa el siguiente comando:



Figura 3-30: Configuración del nombre en Git. **Fuente:** Elaboración propia.

2. Posteriormente, para establecer un correo electrónico asociado a Git, se ejecuta el siguiente comando:



Figura 3-31: Configuración del correo electrónico en Git. **Fuente:** Elaboración propia.

3. Para asegurarse de que toda la configuración se haya aplicado correctamente, se ejecuta el siguiente comando que muestra la lista completa de configuraciones de Git:



Figura 3-32: Comprobación de la configuración en Git. **Fuente:** Elaboración propia.

La Figura 3-33 representa la salida para la verificación de los campos iniciales, entre los que se incluyen el nombre y el email.

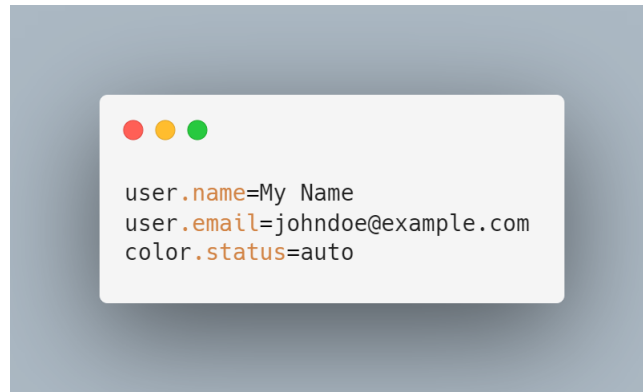


Figura 3-33: Salida de la configuración en Git. **Fuente:** Elaboración propia.

Creación del repositorio remoto en GitHub

Un repositorio remoto en GitHub es indispensable para que los demás usuarios accedan al paquete y puedan hacer uso de él. Para ello, se crea un repositorio público de la siguiente forma:

1. Hacer clic en *New repository*, como paso inicial:

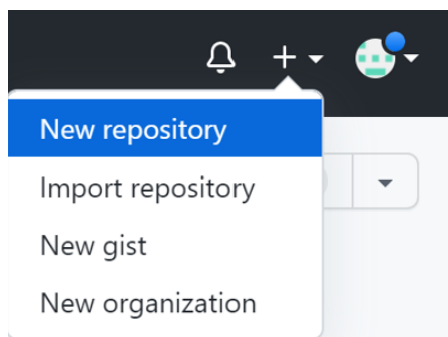




Figura 3-34: Creación de un nuevo repositorio en GitHub. **Fuente:** Elaboración propia.

2. En la pantalla siguiente, se debe establecer un nombre para el repositorio (que también será la URL), una breve descripción, verificar que el repositorio sea público e inicializar la opción *Add a README file* para agregar un archivo Markdown correspondiente a la documentación para GitHub, al igual que una licencia adecuada para el proyecto.

Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)


Owner * Repository name *

 kevortiz10 / BNPPairedSamples. 

Great repository names are short and memorable. Need inspiration? How about [psychic-enigma?](#)

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)


This will set  **main** as the default branch. Change the default name in your [settings](#).

Figura 3-35: Especificaciones del repositorio de GitHub. **Fuente:** Elaboración propia.

Inicialización del repositorio local y seguimiento de archivos con Git

Para realizar el *tracking* o seguimiento a los archivos del paquete, se navega hasta la carpeta donde están ubicados los archivos del proyecto y una vez allí, se abre la consola Git Bash en dicha ruta:

1. Se inicializa el repositorio Git:



Figura 3-36: Inicialización del repositorio en Git. **Fuente:** Elaboración propia.

2. Para visualizar el estado de los archivos se ejecuta el comando:



Figura 3-37: Visualización del estado de archivos en Git. **Fuente:** Elaboración propia.

3. Con el objetivo de realizar un seguimiento a todos los archivos del paquete, se necesita ejecutar el siguiente comando, en el cual el “.” hace referencia a todos los archivos presentes en la ruta en la que se encuentran alojados los archivos del paquete:



Figura 3-38: Incorporación de archivos del repositorio local (Git) al repositorio remoto (GitHub). **Fuente:** Elaboración propia.

4. Una vez ejecutado el anterior comando, todos los archivos tendrán seguimiento por parte de Git, en el momento en que sean modificados de alguna forma. Por cada nuevo archivo(s) que se agregue en dicha ruta, se tendrá que realizar nuevamente el comando anterior, a menos de que no se desee hacer seguimiento a dicho archivo. Cuando se haya completado alguna tarea y los cambios realizados sean verificados, se debe ejecutar un *commit*, que se puede ver como un punto en la historia del paquete, con el cual se pueden llevar a cabo múltiples acciones útiles para el desarrollo del proyecto. La ejecución de un *commit* se ilustra en la Figura 3-39.



Figura 3-39: Ejecución de un *commit* para la consolidación de cambios en el proyecto. **Fuente:** Elaboración propia.

Cómo comando adicional, para evitar la ejecución del comando del paso 3, para todos los archivos que hayan sido modificados (no aplica para nuevos archivos), se puede ejecutar:



Figura 3-40: Alternativa para la ejecución de un *commit*. **Fuente:** Elaboración propia.

Estos *commits* están disponibles por ahora, de forma local, en la base de datos de Git.

5. Existen diversas formas de ver los *commits* existentes del proyecto, pero una de las más simplificadas a nivel visual es la ejecución del comando de la Figura 3-41.



Figura 3-41: Visualización de los *commits* del proyecto. **Fuente:** Elaboración propia.

Este comando permite ver todos los *commits* en una sola línea, siendo visibles sus mensajes, su identificador único (código hash) y la posición del *HEAD* (posición actual del repositorio local).

Añadir repositorio remoto a Git

Para poder subir los archivos, con toda su historia (incluidos los *commits*), se debe agregar mediante un comando muy simple el repositorio remoto de GitHub.

1. Para añadir el repositorio remoto de GitHub al Git local, se debe ejecutar:

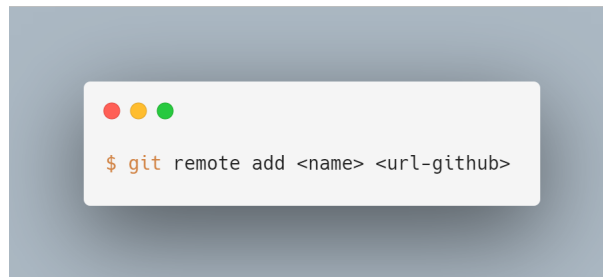


Figura 3-42: Adición del repositorio de GitHub al Git local. **Fuente:** Elaboración propia.

Se reemplaza `< name >` por un nombre que identifique dicho repositorio remoto, generalmente se usa *origin*, aunque puede ser cualquiera. De igual forma, se reemplaza `< url – GitHub >` por el enlace HTTPS (o SSH si es el caso), al cual se puede acceder haciendo clic en el botón *Code* en el repositorio en GitHub.

2. Para verificar que se ha añadido correctamente el repositorio remoto, se ejecuta el siguiente código y se realiza la respectiva verificación:



Figura 3-43: Verificación de integración del repositorio de GitHub al repositorio Git local. **Fuente:** Elaboración propia.

Subir el paquete a GitHub

El repositorio remoto en GitHub contiene dos archivos, que corresponden al archivo de documentación propia y la licencia. El primer paso es traer dichos archivos al repositorio local para evitar conflictos en Git.

1. Para traer los cambios del repositorio remoto al local, se ejecuta:

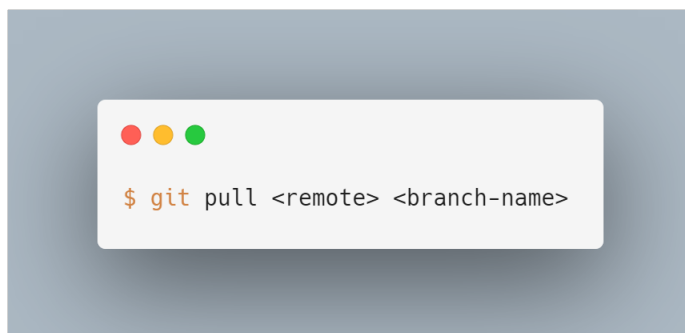


Figura 3-44: Inclusión de cambios del repositorio remoto al local. **Fuente:** Elaboración propia.

El fragmento `< remote >` se reemplaza por el nombre que identifica al repositorio remoto. Este comando, si no produce ningún error, trae los cambios existentes en el repositorio de GitHub. `< branch - name >` corresponde al nombre de la rama local.

2. Una vez se ha actualizado el repositorio local, se procede a subir todos los cambios de Git a remoto. Para ello, se ejecuta:

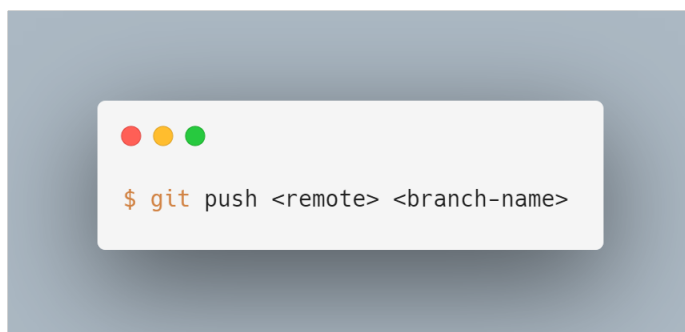


Figura 3-45: Comando para cargar cambios del repositorio local a GitHub. **Fuente:** Elaboración propia.

Este código funciona de igual forma que el *pull* y una vez se han realizado todos los pasos anteriores, se podrá continuar ejecutando los mismos comandos para los desarrollos posteriores.

3.3. Archivos de datos

Para este caso se trabajará con cuatro conjuntos de datos: dos de ellos corresponden a una simulación estadística de mezclas normales y los dos restantes son datos reales que se encuentran disponibles dentro de paquetes en el software R.

3.3.1. Datos 1 - Simulación de una mezcla normal bivariada

En este primer conjunto de datos se realizó la simulación estadística de una mezcla normal bivariada con vectores de medias y matrices de varianzas y covarianza dadas por:

$$Y \sim 0.5 \cdot N_2 \left[\mu = (0 \quad -3)^T, \Sigma = \begin{pmatrix} 1 & 0.8 \\ 0.8 & 1 \end{pmatrix} \right] + 0.5 \cdot N_2 \left[\mu = (0 \quad 3)^T, \Sigma = \begin{pmatrix} 1 & 0.8 \\ 0.8 & 1 \end{pmatrix} \right]$$

3.3.2. Datos 2 - Calidad del aire

Este archivo de datos se encuentra alojado en el paquete *Datasets* del *software* R, el cual tiene el nombre de *airquality* y contiene datos de mediciones diarias de la calidad del aire en la ciudad de Nueva York en el año 1973, en los meses de mayo a septiembre. Para este caso se trabajará únicamente con la columna *wind* que corresponde a datos de la velocidad promedio del viento en millas por hora en el Aeropuerto La Guardia en el horario de 7:00 a.m a 10:00 a.m. Estos datos son longitudinales, ya que se realiza la medición de las mismas variables en diferentes periodos de tiempo en la misma localidad, por lo cual, se tomarán dos periodos de tiempo (el mes de mayo y el mes de agosto) que podrían pensarse como pareados y, de este modo, poder identificar si existen diferencias en la velocidad del viento en las dos estaciones que se estarían evaluando (mayo corresponde al final de la primavera y agosto corresponde al final del verano). Estos datos fueron obtenidos del Servicio Meteorológico Nacional de Nueva York (R-Core-Team 1993).

3.3.3. Datos 3 - Pérdida de peso por dieta

Este conjunto de datos, bajo el nombre de *weightloss* pertenece al paquete *Datarium* del *software* R y contiene información obtenida por un investigador sobre los efectos que tienen las dietas y el ejercicio en el peso de un grupo de hombres sedentarios. Se contaba con un total de 4 ensayos: sin dieta y sin ejercicios, sólo dieta, sólo ejercicios y dieta y ejercicios combinados; cada ensayo tuvo una duración de 9 semanas y un periodo de lavado para eliminar los efectos que pudiera tener el ensayo previamente realizado, también se contó con un total de 3 mediciones del peso, una al inicio del ensayo (t_1), una a la mitad del ensayo (t_2) y la tercera medición se realizó al finalizar el ensayo (t_3). Estos datos son longitudinales pero se pensaron como pareados al realizar la selección de dos tiempos, el inicial (t_1) y el final (t_3); además, se seleccionó el ensayo 4 que corresponde a la combinación entre dieta y ejercicio (Kassambara 2019).

3.3.4. Datos 4 - Simulación de una mezcla normal bivariada

Para este último conjunto de datos se realizó la simulación estadística de una mezcla normal bivariada con vectores de medias y matrices de varianzas y covarianza dadas por:

$$Y \sim 0.5 \cdot N_2 \left[\mu = \begin{pmatrix} 0 & 0 \end{pmatrix}^T, \Sigma = \begin{pmatrix} 1 & -0.7 \\ -0.7 & 1 \end{pmatrix} \right] + 0.5 \cdot N_2 \left[\mu = \begin{pmatrix} 0 & 0.1 \end{pmatrix}^T, \Sigma = \begin{pmatrix} 1 & -0.7 \\ -0.7 & 1 \end{pmatrix} \right]$$

4 Resultados

En este apartado se presenta el paquete *BNPPairedSamples*, el repositorio desde el que se puede consultar e instalar y el uso de las funciones que lo componen considerando los cuatro conjuntos de datos pareados expuestos en la sección 3.3. Los resultados más relevantes aquí mostrados surgen de la aplicación de la función asociada a la prueba de hipótesis Bayesiana no paramétrica para muestras pareadas, no obstante, estos se complementan a partir de las dos funciones adicionales que fueron integradas al paquete, alusivas a la función *shift* y al gráfico de contornos.

4.1. Paquete BNPPairedSamples

El paquete *BNPPairedSamples* se encuentra alojado de forma pública a través del repositorio remoto de GitHub, el cual se puede consultar en la URL <https://github.com/kevortiz10/BNPPairedSamples>. Por medio de este enlace, es posible visualizar todos los archivos específicos que se han utilizado para la construcción del paquete (ver sección 3.2), entre los que destacan el código en R de las funciones desarrolladas, la documentación en la carpeta *man* y el archivo *README* que provee una breve visualización del funcionamiento del paquete.

Con su publicación en GitHub, el paquete puede ser instalado en RStudio por medio de la función *install_github* del paquete *remotes*, así:

```
remotes :: install_github("kevortiz10/BNPPairedSamples")
```

Siguiendo las instrucciones de instalación del paquete, es posible entonces proceder con la ejecución de cada una de las funciones, cuyo código de aplicación en R y documentación se encuentran especificados en el archivo *README* de GitHub, en la página de ayuda propia para cada función en RStudio utilizando *help()* o *?* y, de manera más detallada, en las viñetas, las cuales se pueden consultar tanto en el repositorio remoto como posterior a la instalación del paquete, esto último en caso de que se especifique su construcción con *remotes* :: *install_github*("kevortiz10/BNPPairedSamples", *build_vignette* = *TRUE*).

4.2. Ilustración del funcionamiento del paquete BNPPairedSamples

Con el paquete fijado de forma pública en GitHub, es conveniente entonces ilustrar el desempeño de las funciones allí alojadas haciendo uso de distintas fuentes de información. Como se mostró en la sección 3.3, dos de las fuentes de datos utilizadas engloban situaciones reales, mientras que las dos restantes corresponden a simulaciones, lo que nos permite visualizar el desempeño de las funciones en ambos contextos.

4.2.1. Simulación estadística #1 - Mezcla Gaussiana bivariada

Con este primer conjunto de datos simulados de una mezcla distribucional Gaussiana, se tenía el propósito de observar el desempeño de la prueba BNP para muestras pareadas, en el caso en que las distribuciones marginales difieren en gran medida: la distribución previa cuenta con un solo clúster que la asemeja a una distribución normal univariada, mientras que la distribución posterior simulada se compone de dos clústeres. Con la finalidad de identificar inicialmente si era requerido hacer uso de la prueba BNP se comprobó la normalidad de las observaciones a través de la prueba de *Shapiro-Wilk* y se obtuvo que para la distribución marginal G_1 se cumplió el supuesto (valor-p = 0.85) y para la distribución marginal G_2 no se cumplió este supuesto (valor-p < 0.001), hecho que propicia el uso de la prueba BNP para muestras pareadas propuesta por Pereira et al. (2020).

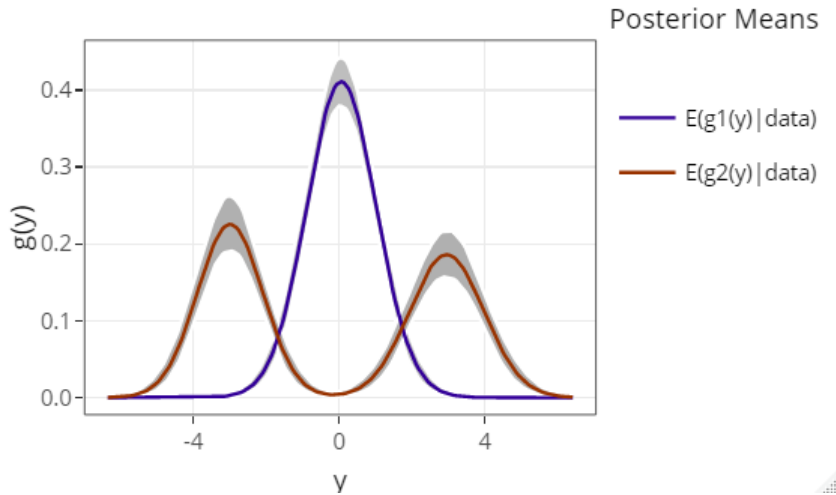


Figura 4-1: Distribuciones marginales de la simulación de una mezcla Gaussiana bivariada #1, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

La Figura 4-1 permite observar que las distribuciones marginales presentaron diferencias a nivel de forma y escala, puesto que se puede visualizar la formación de dos clústeres en

la distribución a posteriori de los datos del segundo momento, mientras que en el primer momento tan sólo hay presencia de un clúster.

```
There are significant differences between marginal distributions G1 and G2.  
[1] "Posterior probability for H1: 1"
```

Figura 4-2: Probabilidad a posteriori para H_1 considerando la mezcla Gaussiana bivariada #1, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

En este orden de ideas, se puede observar que la probabilidad a posteriori para H_1 obtenida a partir de la prueba de hipótesis (ver Figura 4-2), permitió confirmar a nivel analítico que existen diferencias entre las dos distribuciones marginales.

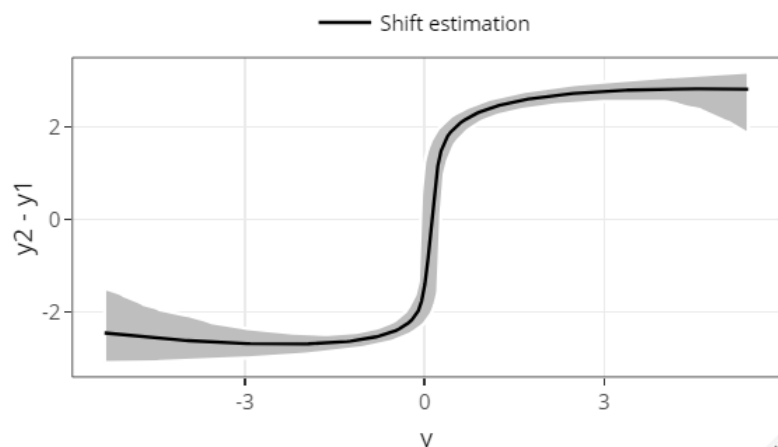


Figura 4-3: Función *shift* para la mezcla Gaussiana bivariada #1, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

Posterior a la identificación de diferencias, se procedió con la realización de la función *shift* para determinar en qué rango de valores se encontraban. Para este caso, se observa en la Figura 4-3 que en la cola izquierda de las distribuciones las diferencias se encuentran a favor de la distribución en el momento 1, es decir que los valores de la distribución se encuentran más a la derecha, mientras que para la cola derecha se observan las diferencias a favor de la distribución del momento 2.

En cuanto a los contornos, se destaca que tienen la finalidad de ilustrar la distribución bivariada, además de determinar la correlación que existe entre las distribuciones. En este sentido, se observa en la Figura 4-4 que la distribución conjunta está conformada por dos clústeres y que en las proximidades a 0 la densidad es 0; también se puede destacar el comportamiento elevado que presentan los contornos, ya que estarían expresando una leve

correlación positiva entre las distribuciones, que de acuerdo con el coeficiente de correlación de Spearman es de 0.4.

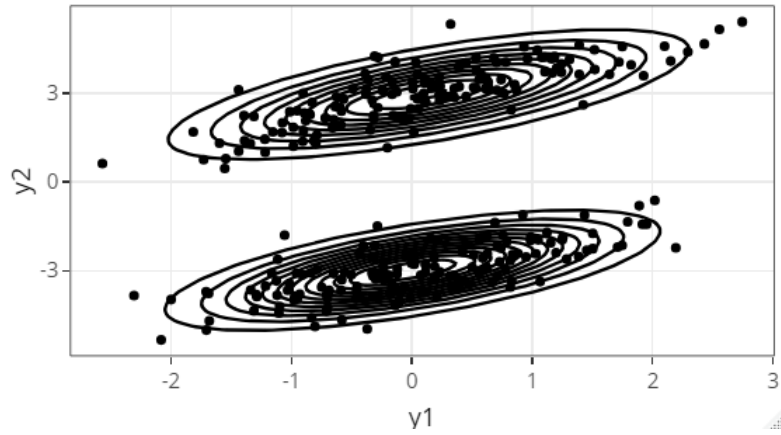


Figura 4-4: Gráfico de contornos para la mezcla Gaussiana bivariada #1, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

4.2.2. Airquality - Paquete datasets

Para este conjunto de datos, que cuenta con un total de 31 registros por mes, se tiene la finalidad de identificar si existen diferencias en la velocidad promedio del viento en la ciudad de Nueva York para los meses de mayo y agosto que corresponden al final de la primavera y al final del verano, respectivamente. En este sentido, se procedió inicialmente con la validación de la normalidad de las observaciones con el propósito de identificar a que prueba recurrir.

Para este caso, a partir de la prueba de *Shapiro-Wilk* se cumplió el supuesto de normalidad para las distribuciones marginales G_1 y G_2 (valor-p de 0.47 para G_1 y valor-p de 0.94 para G_2) y por tal razón, sería adecuado hacer uso de la prueba t de Student para muestras pareadas, sin embargo, por razones de comparabilidad se decidió también hacer uso de la prueba BNP.

Para la prueba t de Student se obtuvo un valor-p de 0.005, que bajo un nivel de significancia de 0.05 indica que existen diferencias entre las distribuciones marginales G_1 y G_2 , es decir, que hay diferencias entre las velocidades promedio del viento en los meses de mayo y agosto. Es importante destacar que la prueba BNP, a diferencia de la prueba t de Student, permite visualizar en qué parte de la distribución se encuentran tales diferencias.

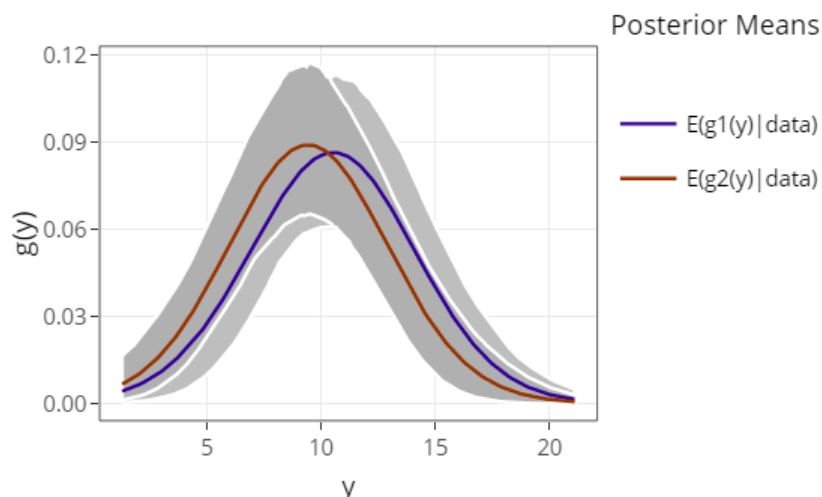


Figura 4-5: Distribuciones marginales del conjunto de datos *airquality*, donde $g_1(y)$ y $g_2(y)$ corresponden a la velocidad promedio del viento en la ciudad de Nueva York para los meses de mayo y agosto de 1973 respectivamente, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

Por otro lado, la Figura 4-5 corresponde a las distribuciones marginales que resultaron posterior a la aplicación de la prueba BNP para muestras pareadas. En ella, es posible observar que no existen diferencias a nivel de forma y escala, pero sí a nivel de localización, lo que quiere decir que la velocidad promedio del viento es mayor en la primavera.

```
There are significant differences between marginal distributions G1 and G2.
[1] "Posterior probability for H1: 0.539"
```

Figura 4-6: Probabilidad a posteriori para H_1 considerando el conjunto de datos *airquality*, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

Teniendo en cuenta la interpretación visual obtenida previamente, se observó que la probabilidad a posteriori es de 0.539 (ver Figura 4-6), lo cual expresa la existencia de leves diferencias que podrían ser estudiadas con mayor detalle a partir de la función *shift*.

Se observa entonces en la Figura 4-7 que la estimación puntual de la función *shift* se encuentra por debajo de cero, lo cual quiere decir que la velocidad promedio en el mes de agosto resulta ser mayor que la obtenida en el mes de mayo, sin embargo, el intervalo de credibilidad (región de color gris) siempre contiene el valor cero, que indica que no existen diferencias entre las distribuciones de estudio.

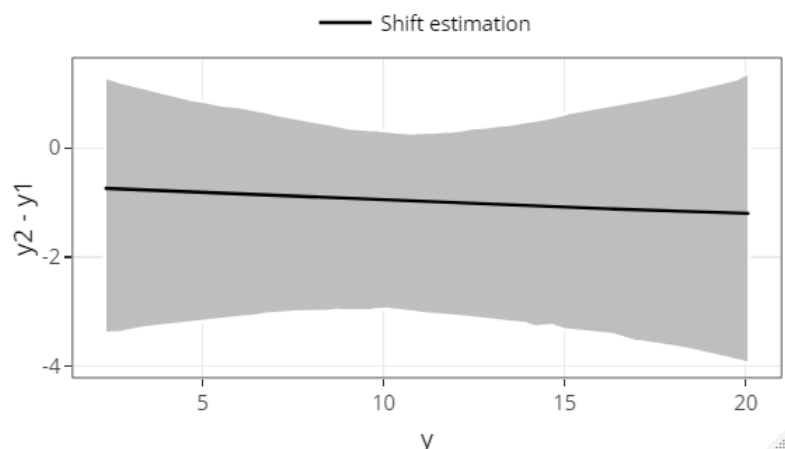


Figura 4-7: Función *shift* para el conjunto de datos *airquality*, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

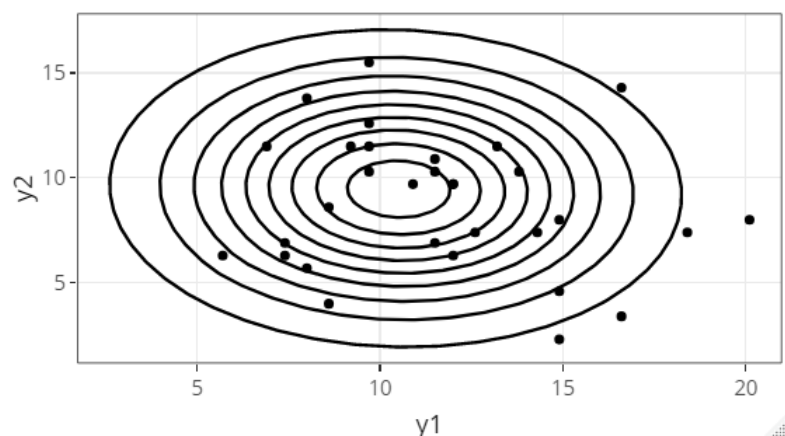


Figura 4-8: Gráfico de contornos para el conjunto de datos *airquality*, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

Por último, se presentan en la Figura 4-8 los contornos estimados para la distribución conjunta $g(y_1, y_2)$, los cuales sugieren poca correlación (-0.17, según el coeficiente de correlación de Pearson) por su forma e inclinación.

4.2.3. Weightloss - Paquete datarium

Para este conjunto de datos que cuenta con un total de 12 registros por tiempo de medición del peso, se tiene como objetivo identificar la efectividad del ensayo cuatro que incluye la

combinación entre dieta y ejercicio, en la pérdida de peso de un grupo de hombres sedentarios. Al igual que en el ejemplo anterior, se realizó la validación del supuesto de normalidad a partir de la prueba *Shapiro-Wilk* y se observó el cumplimiento del supuesto para las distribuciones marginales G_1 y G_2 con valores-p de 0.24 y 0.46, respectivamente. En este sentido, se aplicaron las pruebas t de Student y BNP por razones de comparabilidad.

Al ser aplicada la prueba t de Student se obtuvo un valor-p < 0.001 , que indica la presencia de diferencias entre los pesos de los hombres sedentarios antes y después del tratamiento de dieta y ejercicio.

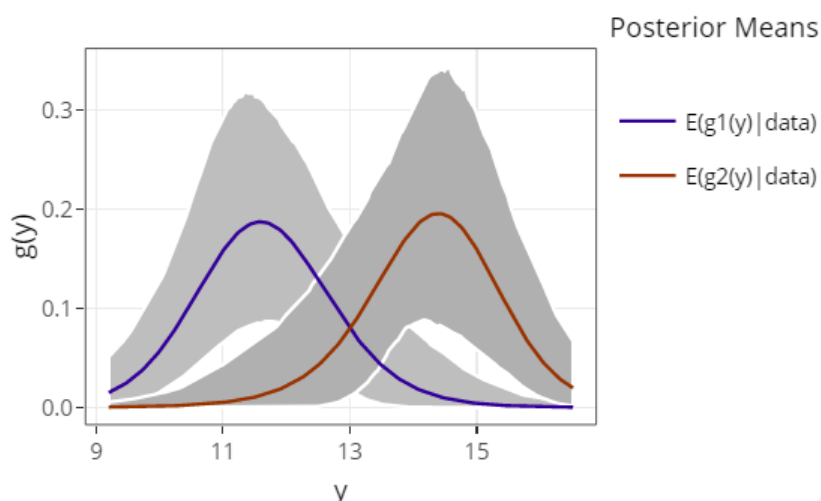


Figura 4-9: Distribuciones marginales del conjunto de datos *weightloss*, donde $g_1(y)$ y $g_2(y)$ hacen referencia a las puntuaciones de pérdida de peso de los hombres antes y después del ensayo, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

Para el caso de los datos de pérdida de peso, se observan diferencias a nivel de localización, ya que la puntuación promedio de pérdida de peso de los individuos antes de comenzar el ensayo tiene el valor de 11.39333, mientras que la puntuación promedio al finalizar el ensayo es de 14.655, lo que indicaría que la intervención sí fue efectiva, pues la puntuación promedio en la pérdida de peso posterior al tratamiento fue superior.

```
There are significant differences between marginal distributions G1 and G2.
[1] "Posterior probability for H1: 0.997"
```

Figura 4-10: Probabilidad a posteriori para H_1 considerando el conjunto de datos *weightloss*, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

Teniendo en cuenta la Figura 4-9 de las distribuciones marginales, se confirma a través de la probabilidad a posteriori obtenida de la prueba de hipótesis que existen diferencias en la puntuación de pérdida de peso de los hombres sedentarios al iniciar y finalizar el ensayo. De este modo, comprobamos que los resultados obtenidos a través de ambas pruebas a nivel analítico resultan en la misma conclusión.

Por otro lado, en la Figura 4-11 se muestra la función *shift* asociada a estos datos de pérdida de peso, a partir de la cual se observa que hay diferencia en todo el espectro de valores y , al estar en la parte positiva del eje y , esto indica que la puntuación de pérdida de peso de los individuos al finalizar el ensayo resultan ser mayores con respecto a los valores obtenidos al inicio del ensayo.

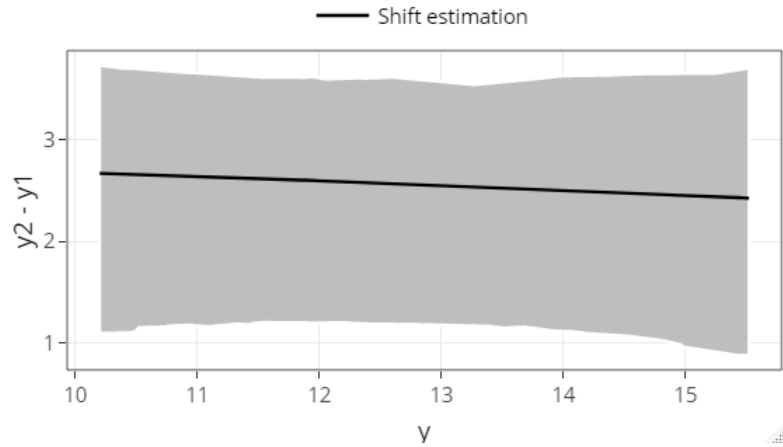


Figura 4-11: Función *shift* para el conjunto de datos *weightloss*, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

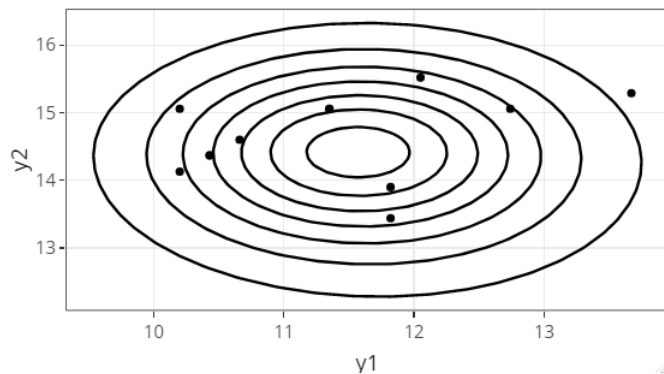


Figura 4-12: Gráfico de contornos para el conjunto de datos *weightloss*, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

En cuanto a los contornos, su estimación sugiere una baja correlación entre las puntuaciones de pérdida de peso de los hombres (0.32, de acuerdo con el coeficiente de correlación de Pearson), considerando la forma que se presenta en la Figura 4-12.

4.2.4. Simulación estadística #2 - Mezcla Gaussiana bivariada

Nuevamente, se realizó la simulación de una mezcla de distribuciones normales con la finalidad de poder ilustrar la presencia de no diferencias entre las distribuciones marginales y cómo la prueba identificaría dicho comportamiento. En este sentido, validamos al igual que en los ejemplos anteriores la normalidad a través de la prueba *Shapiro-Wilk* y se obtuvo un valor-p de 0.32 para la distribución marginal G_1 y un valor-p de 0.19 para la distribución marginal G_2 , lo cual indica que en ambos casos se cumplió el supuesto de normalidad y por tal razón, resultó adecuado hacer uso de la prueba t de Student. Posteriormente, se procedió con la prueba BNP para ilustrar su funcionamiento en estos casos.

A partir de la prueba t de Student se obtuvo un valor-p de 0.06, que bajo un nivel de significancia de 0.05 indica que la evidencia muestral no permite determinar la presencia de diferencias entre las distribuciones marginales G_1 y G_2 .

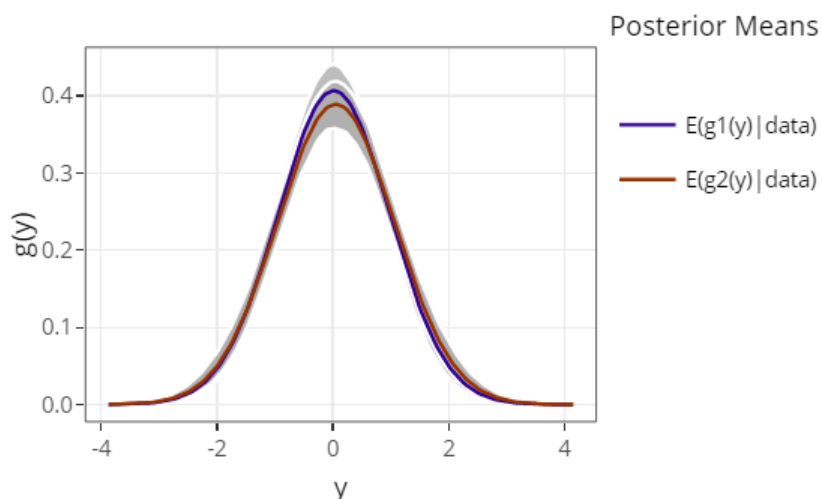


Figura 4-13: Distribuciones marginales para la mezcla Gaussiana bivariada #2, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

A partir de la Figura 4-13, es posible observar que las distribuciones marginales no presentan diferencias muy notorias a nivel de escala, forma y localización. Claramente, este comportamiento tan similar entre pares de datos dependientes es muy poco realista, sin embargo, se expone como prueba de la baja probabilidad a posteriori que se obtiene mediante la prueba BNP en estos casos, tal y como se muestra en la Figura 4-14.

There are no significant differences between marginal distributions G1 and G2.
 [1] "Posterior probability for H1: 0.199"

Figura 4-14: Probabilidad a posteriori para H_1 considerando la mezcla Gaussiana bivariada #2, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

Así, con base en la visualización de la gráfica de las distribuciones marginales y la probabilidad a posteriori obtenida a partir de la prueba BNP con un valor de 0.199, se confirma que no existen diferencias entre las distribuciones marginales g_1 y g_2 . En este orden de ideas, se comprueba que tanto la prueba t de Student como la prueba BNP presentaron resultados similares en cuanto a la conclusión de no presencia de diferencias entre las distribuciones marginales.

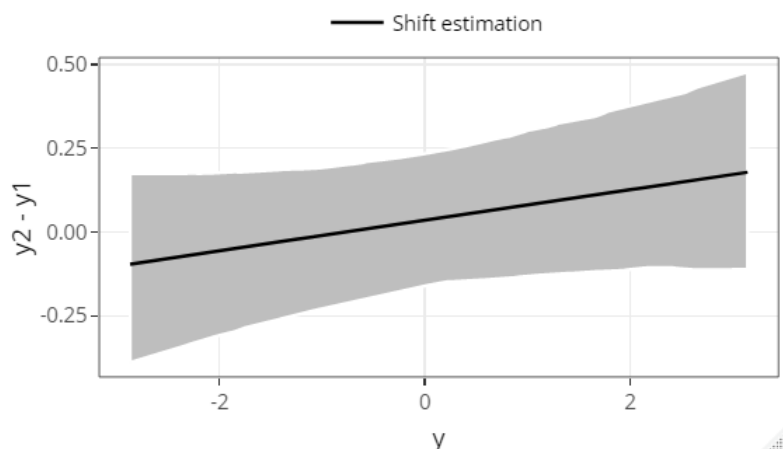


Figura 4-15: Función *shift* para la mezcla Gaussiana bivariada #2, usando el paquete *BNPPairedSamples*. **Fuente:** Elaboración propia.

Teniendo en cuenta que no se presentaron diferencias entre las distribuciones marginales, no resulta necesaria la obtención de la función *shift*, sin embargo, se ilustra la forma en la que se comporta en estos casos puntuales. Se observa entonces que el intervalo de credibilidad contiene al cero y que las colas presentan pequeñas diferencias entre las marginales (ver Figura 4-15).

5 Conclusiones y recomendaciones

5.1. Conclusiones

La prueba de hipótesis Bayesiana no paramétrica para muestras pareadas se consolida como una alternativa válida ante pruebas de hipótesis ya existentes para datos relacionados, como lo expusieron Pereira et al. (2020) en su desarrollo teórico. La creación del paquete estadístico BNPPairedSamples en el software R permite la implementación de esta prueba de hipótesis de forma práctica, contando adicionalmente con dos funciones que complementan los resultados que se muestran a partir de la ejecución de la prueba.

La configuración de este paquete contribuye entonces a una fácil aplicación de la prueba de hipótesis en cuestión, por parte de las comunidades académica y científica, cuyo interés se centra en una comparación robusta de datos pareados. Así mismo, su configuración en R de forma pública y en idioma inglés, permite que una mayor cantidad de personas puedan acceder a las funcionalidades provistas en este paquete. Adicionalmente, la estructura de las tres funciones aquí dispuestas y la documentación asociada a cada una de ellas fueron desarrolladas de tal forma que cualquier usuario que cuente con conocimientos teórico estadísticos y de programación variados, pueda hacer uso del paquete y obtener los resultados de manera simple y sencilla.

Por otro lado, se presentaron una serie de ejemplos a partir de datos simulados y reales que proveen al usuario una familiarización con los resultados que se pueden obtener mediante el uso de las funciones, los cuales se pueden visualizar mediante el archivo *README* del repositorio de GitHub y las viñetas alojadas dentro del paquete. La inclusión de gráficos interactivos a través de la librería *ggplotly* del paquete *plotly*, contribuyen a que el usuario tenga una mejor experiencia al hacer uso del paquete, pues aumenta la precisión y la profundidad del análisis que se esté realizando.

5.2. Limitaciones

La principal limitante del paquete corresponde a los tiempos de ejecución, que de acuerdo con las características particulares de cada equipo de cómputo, tales como el procesador y la memoria RAM, serán más o menos cortos para la visualización de los resultados de la prueba. Además, la cantidad de simulaciones elegidas por el usuario juega un papel importante en la ejecución de las tres funciones.

5.3. Recomendaciones

Se recomienda a nivel general la búsqueda de alternativas que permitan optimizar el tiempo de ejecución de la función *shift*, particularmente la búsqueda de raíces a partir de la función *uniroot*, disponible en el paquete *stats* de R. Adicionalmente, se recomienda implementar en la función *plotshift.function* del paquete *BNPPairedSamples* la opción al usuario de seleccionar el tipo de función *shift* que se desee ajustar: la versión clásica propuesta por Doksum & Sievers (1976) o la versión robusta propuesta por Wilcox (1995), la cual resulta ser más potente, puesto que calcula intervalos de credibilidad de las diferencias entre deciles con una estimación Bootstrap.

Bibliografía

- Albert, J. (2009), *Bayesian Computation with R*, 2 edn, Springer New York.
- Ashby, D. (2006), ‘Bayesian statistics in medicine: A 25 year review’, *Statistics in Medicine* **25**, 3589–3631.
- Benavoli, A., Corani, G., Mangili, F., Zaffalon, M. & Ruggeri, F. (2014), ‘A bayesian wilcoxon signed-rank test based on the dirichlet process’, *Proceedings of the 31st International Conference on Machine Learning* **32**, 1026–1034.
- Berger, J. O. (1985), *Statistical Decision Theory and Bayesian Analysis*, Springer New York, chapter Bayesian Analysis, pp. 118–307.
- Boehmke, B. (2016), *Data Wrangling with R*, Springer.
- Bouguila, N. & Ziou, D. (2008), ‘A dirichlet process mixture of dirichlet distributions for classification and prediction’, *IEEE Workshop on Machine Learning for Signal Processing* p. 297–302.
- Canale, A. (2017), ‘msbp: An r package to perform bayesian nonparametric inference using multiscale bernstein polynomials mixtures’, *Journal of Statistical Software* **78**(6), 1–19.
- Carrasco, J., García, S., Rueda, M. & Herrera, F. (2017), *Hybrid Artificial Intelligent Systems*, Springer Science + Business Media, chapter rNPBST: An R Package Covering Non-parametric and Bayesian Statistical Tests, pp. 281–292.
- Chacon, S. & Straub, B. (2014), *Pro Git*, 2 edn, Apress.
- Chambers, J. M. (2008), *Software for Data Analysis: Programming with R*, Springer.
- de Leeuw, J. (2011), *International Encyclopedia of Statistical Science*, Springer, Berlin, Heidelberg, chapter Statistical Software: An Overview.
- Doksum, K. (1974), ‘Empirical probability plots and statistical inference for nonlinear models in the two-sample case’, *The Annals of Statistics* **2**(2), 267–277.
- Doksum, K. A. & Sievers, G. (1976), ‘Plotting with confidence: graphical comparisons of two populations’, *Biometrika* **63**, 421–434.

- Ferguson, T. S. (1973), ‘A bayesian analysis of some nonparametric problems’, *The Annals of Statistics* **1**(2), 209–230.
- Filippi, S. & Holmes, C. C. (2017), ‘A bayesian nonparametric approach to quantifying dependence between random variables’, *Bayesian Analysis* **12**(1), 919–938.
- Filippi, S., Holmes, C. C. & Nieto-Barajas, L. E. (2016), ‘Scalable bayesian nonparametric measures for exploring pairwise dependence via dirichlet process mixtures’, *Electronic Journal of Statistics* **10**(1), 3338–3354.
- Gelfand, A. E. (2000), ‘Gibbs sampling’, *Journal of the American Statistical Association* **95**(452), 1300–1304.
- Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A. & Rubin, D. B. (2014), *Bayesian Data Analysis*, 3 edn, Taylor & Francis Group.
- Genolini, C., Dubois, E. & Furió, D. (2019), *Human-Computer Interaction – INTERACT 2019*, Springer International Publishing, Cham.
- Girón, F. J., Martínez, M. L. & Moreno, E. (2003), ‘Bayesian analysis of matched pairs’, *Journal of Statistical Planning and Inference* **113**(1), 49–66.
- Hoff, P. D. (2009), *A First Course in Bayesian Statistical Methods*, Springer.
- Ihaka, R. (1998), ‘R: Past and future history’, *The University of Auckland: Statistics Department* pp. 1–5.
- Ishwaran, H. & Rao, J. S. (2005), ‘Spike and slab variable selection: Frequentist and bayesian strategies’, *The Annals of Statistics* **33**(2), 730–773.
- J. Ross, G. & Markwick, D. (2020), *dirichletprocess: Build Dirichlet Process Objects for Bayesian Modelling*. R package version 0.4.0.
URL: <https://CRAN.R-project.org/package=dirichletprocess>
- Jank, W. (2011), *Business Analytics for Managers*, Springer New York, Nueva York, NY.
- Jara, A., Hanson, T. E., Quintana, F. A., Muller, P. & Rosner, G. L. (2011), ‘Dppackage: Bayesian semi- and nonparametric modeling in r’, *Journal of Statistical Software* **40**(5), 1–30.
- Johnson, W. O. & de Carvalho, M. (2015), *Bayesian Nonparametric Biostatistics*, Springer International Publishing, Cham, pp. 15–54.
URL: https://doi.org/10.1007/978-3-319-19518-6_2
- Kassambara, A. (2019), ‘Datarium: Data bank for statistical analysis and visualization’, *The Comprehensive R Archive Network*.

- Kryptos, A.-M., Klugkist, I. & Engelhard, I. M. (2017), ‘Bayesian hypothesis testing for human threat conditioning research: an introduction and the condir r package’, *European Journal of Psychotraumatology* **8**, 1–9.
- Kuhn, T. (2008), ‘Design dictionary’, *Board of International Research in Design* p. 369.
- Lee, J. (2011), ‘Bayesian nonparametric statistics’, *International Encyclopedia of Statistical Science* pp. 99–101.
- Leisch, F. (2009), ‘Creating r packages: A tutorial’, *R Development Core Team* pp. 1–19.
- López, N. E. A. & Jiménez, J. d. C. (2010), ‘Contraste de hipótesis: Clásico vs bayesiano’, *Revista digital Matemática, Educación e Internet* **11**(1), 1–13.
- Malsiner-Walli, G. & Wagner, H. (2011), ‘Comparing spike and slab priors for bayesian variable selection’, *Austrian Journal of Statistics* **40**(4), 241–264.
- Morales, J. C. C. & Causil, C. J. B. (2018), *Introducción a la Estadística Bayesiana*, Instituto Tecnológico Metropolitano, Medellín, Colombia.
- Ohri, A. (2014), *R for Cloud Computing*, Springer.
- Pereira, L. A., Taylor-Rodríguez, D. & Gutiérrez, L. (2020), ‘A bayesian nonparametric testing procedure for paired samples’, *Biometrics* **76**(1), 1–14.
- R-Core-Team (1993), ‘The r datasets package’, *The Comprehensive R Archive Network* .
- Romero, M. (2013), ‘Comparación de medias en grupos apareados o dependientes’, *Revista Enfermería del Trabajo* **3**(3), 118–123.
- Rossi, P. E. (2014), *Bayesian Non- and Semi-parametric Methods and Applications*, Princeton University Press.
- Rousselet, G. A., Pernet, C. R. & Wilcox, R. R. (2017), ‘Beyond differences in means: robust graphical methods to compare two groups in neuroscience’, *European Journal of Neuroscience* **46**(2), 1738–1748.
- Soriano, J. (2015), Bayesian Methods for Two-Sample Comparison, PhD thesis, Universidad Duke.
- Sprent, P. & Smeeton, N. C. (2001), *Applied Nonparametric Statistical Methods*, 3 edn, Chapman & Hall.
- Student (1908), ‘The probable error of a mean’, *Biometrika* **6**(1), 1–25.
- Sueur, J. (2018), *Sound Analysis and Synthesis with R*, Springer.

- van Ravenzwaaij, D., Cassey, P. & Brown, S. D. (2018), ‘A simple introduction to markov chain monte-carlo sampling’, *Psychonomic Bulletin Review* **76**, 143–154.
- Wackerly, D. D., Mendenhall, W. & Scheaffer, R. L. (2008), *Estadística matemática con aplicaciones*, 7 edn, Cengage Learning.
- Wakefield, J. (2012), *Bayesian and Frequentist Regression Methods*, Springer New York.
- Walker, S. G. (2007), ‘Sampling the dirichlet mixture model with slices’, *Communications in Statistics - Simulation and Computation* **36**(1), 45–54.
- Wickham, H. (2015), *R Packages: Organize, Test, Document, and Share Your Code*, 1 edn, O’Reilly Media.
- Wilcox, R. R. (1995), ‘Comparing two independent groups via multiple quantiles’, *Journal of the Royal Statistical Society. Series D (The Statistician)* **44**(1), 91–99.
- Wilcoxon, F. (1945), ‘Individual comparisons by ranking method’, *Biometrics Bulletin* **1**(6), 80–83.
- Wiley, M. & Wiley, J. F. (2016), *Advanced R*, Apress, Berkeley, CA.
- Xia, Y., Sun, J. & Chen, D.-G. (2018), *Statistical Analysis of Microbiome Data with R*, Springer, Singapore, chapter Introduction to R, Rstudio and ggplot2, pp. 77–127.
- Zimmerman, D. W. (1997), ‘The probable error of a mean’, *Journal of Educational and Behavioral Statistics* **22**(1), 349–360.