FACULTAD DE CIENCIAS AGRARIAS
UNIVERSIDAD NACIONAL DE ROSARIO

Paquete de R y aplicación Web para el análisis de datos provenientes de ensayos multiambientales

JULIA ANGELINI

TRABAJO FINAL PARA OPTAR AL TÍTULO DE ESPECIALISTA EN BIOINFORMÁTICA

DIRECTOR: Dr. Gerardo Cervigni
CO-DIRECTOR: Mgs. Marcos Prunello

AÑO: 2021

Paquete de R y aplicación Web para el análisis de datos provenientes de ensayos multiambientales

Julia Angelini

Licenciada en Estadística – Universidad Nacional de Rosario

Este Trabajo Final es presentado como parte de los requisitos para optar al grado académico de Especialista en **Bioinformática**, de la Universidad Nacional de Rosario y no ha sido previamente presentada para la obtención de otro título en ésta u otra Universidad. El mismo contiene los resultados obtenidos en investigaciones llevadas a cabo en **el Centro de Estudios Fotosintéticos y Bioquímicos (CEFOBI)**, durante el período comprendido entre **los años 2017 y 2021**, bajo la dirección del **Dr. Gerardo Cervigni y Mgs. Marcos Prunello**.

Nombre y firma del autor

Nombre y firma del Director

Nombre y firma del Co - Director

Defendida: _____ de 20____.

Agradecimientos

En este trabajo final, directa o indirectamente, participaron muchas personas a las que les quiero agradecer.

En primer lugar al Dr. Gerardo Cervigni por confiar en mí y permitirme explorar el mundo de la Bioinformática durante mi tesis doctoral, para que hoy sea parte de mis conocimientos. Al Mgs. Marcos Prunello por acompañarme en el desarrollo del trabajo final, por su dedicación y sus consejos.

Todo esto nunca hubiera sido posible sin el apoyo y el cariño de mis padres, de mi hermano, de Otto, de Segundo y Kalita. Siempre estuvieron a mi lado, las palabras nunca serán suficientes para agradecerles.

A mis compañeras Jor y Lu, por su ayuda y por compartir excelentes momentos.

A Gaby y Euge mis compañeras de CEFOBI, gracias a ustedes este camino ha sido más fácil!

A mis amigos, por estar siempre presentes.

Muchas gracias a todos!

Abreviaturas y Símbolos

EMA: ensayos multiambientales.

IGA: interacción genotipo ambiente.

NCOI: interacción sin cambio de rango, del inglés *no crossover interaction*

COI: interacción con cambio de rango, del inglés *crossover interaction*

ANOVA: análisis de la variancia, del inglés *analysis of variance*

AMMI: modelo de los efectos principales aditivos y interacción multiplicativa, del inglés *Additive Main effects and Multiplicative Interaction*

ACP: análisis de componentes principales

SREG: modelo de regresión por sitio, del inlés *Site Regression model*

DVS: descomposición de valores singulares

GNU: *General Public Licence*

CRAN: *Comprehensive R Archve Network*

EM: maximización de la esperanza, del inglés *Expectation-Maximization*

Resumen

Las variedades mejoradas de cultivos vegetales son el resultado del trabajo de desarrollo genético llevado a cabo en los programas de fitomejoramiento, los cuales se extienden a lo largo de varios años y requieren cuantiosas inversiones. En etapas avanzadas, los ensayos multiambientales (EMA), que comprenden experimentos en múltiples ambientes, son herramientas fundamentales para incrementar la productividad y rentabilidad de los cultivos. La vigencia comercial de las variedades puede extenderse durante varias décadas, por lo que su elección es crítica para que el productor evite pérdidas económicas por malas campañas y el suministro al mercado sea constante. Consecuentemente, un análisis adecuado de la información de los EMA es indispensable para asegurar el éxito del programa de mejoramiento de cultivos. Actualmente, R es uno de los lenguajes de programación más utilizados para el análisis de datos debido a su distribución como software libre y a la gran variedad de herramientas que ofrece. Sin embargo, los mejoradores que no están familiarizados con la programación tienden a utilizar otros tipos de programas que responden a instrucciones por menú en lugar de escribir lineas de código, a pesar de los costos económicos derivados del pago de sus licencias. Mientras que, aquellos que sí tienen afinidad con el uso de código para el análisis de datos se enfrentan con dificultades a la hora de identificar las herramientas apropiadas entre el gran número de instrumentos disponibles. Por lo tanto, en este trabajo se presenta el desarrollo de dos herramientas informáticas para asistir en el análisis de datos provenientes de EMA. Por un lado, se creó un nuevo paquete de R que incluye metodología recientemente publicada que no se encuentra disponible en el software y al mismo tiempo reúne todas aquellas de mayor utilidad, de modo que aquellos usuarios que posean un manejo del lenguaje puedan simplificar su tarea. Por otro lado, se confeccionó una interfaz gráfica de usuario mediante una aplicación web Shiny que permite realizar los principales análisis implementados en el paquete sin necesidad de programar y se encuentra publicada en internet en el servidor de CONICET para su libre acceso.

Palabras Clave: análisis estadístico, ensayos multiambientales, interfaz gráfica, lenguaje R, programación

Abstract

Crop improvement is the result of genetic development which requires several years and large investments. In advanced stages of breeding programs, multi-environment trials (MET), which consist of evaluating different cultivars in multiple environments, are essential tools to increase crop productivity. Since varieties remain on market for decades, their choice is essential to avoid economic losses due to bad seasons and to ensure a constant supply. Consequently, an adequate analysis of MET data is essential to guarantee the success of a breeding program. Currently, R is one of the most widely used programming language for data analysis due to its distribution as free software and the wide variety of tools it offers. However, breeders who are unfamiliar with programming tend to use other types of programs that respond to menu prompts instead of writing lines of code, despite the financial costs of their licenses. Whereas, those who have an affinity with the use of code for data analysis face difficulties in identifying the right tools from the large number of instruments available. Therefore, in this work two tools are develop for MET data analysis. On one hand, a new R package that includes new methodology not available in the software and at the same time brings together all those most useful was created to facilitate the users task. On the other hand, a graphical user interface was created using a Shiny web application that allows the main analyzes implemented in the package to be carried out without the need for programming and is published on CONICET server for free access.

Keywords: multi-environment trials, programming, statistical analysis, user interfaz, R languaje

Índice general

Capítulos **Página**

1. Introducción **1**

2. Objetivos **5**

2.1. Objetivo general 5

2.2. Objetivos específicos 5

3. Métodos **6**

3.1. Métodos estadísticos 6

3.1.1. Modelo AMMI y SREG 6

3.1.2. Modelo AMMI robusto 8

3.1.3. Métodos de imputación 8

3.2. Paquete de R 9

3.2.1. Creación del paquete 10

3.2.2. Archivos de código 11

3.2.3. Documentación 12

3.2.4. Editar el archivo DESCRIPTION 12

3.2.5. Testeos 14

3.2.6. Compilación e instalación 16

3.2.7. Algunos elementos complementarios 16

3.2.7.1. Viñetas 17

3.2.7.2. Agregar datasets 17

3.2.7.3. Archivo README 17

3.2.7.4. Archivo NEWS 19

3.2.7.5. Crear una página web 19
3.2.7.6. Publicación 19
3.3. Shiny APP 20
3.3.1. Estructura de Shiny APP 20
3.3.2. Desarrollo de Shiny APP 21
3.3.3. Compartiendo una Shiny Web App 22

4. Resultados **23**
4.1. Paquete de R *geneticae* 23
4.1.1. Conjuntos de datos en geneticae 23
4.1.2. Modelo AMMI 24
4.1.3. Modelo de Regersión por Sitio 27
4.1.4. Métodos de imputación 36
4.2. Geneticae Shiny Web App 37
4.2.1. Preparación de un archivo de datos para la aplicación Geneticae . . 37
4.2.2. Análisis descriptivo 39
4.2.3. Modelo de regresión por sitio 41
4.2.4. modelo AMMI 44
4.2.5. Ayuda 44

5. Conclusiones **45**

Bibliografía **46**

Índice de figuras

1.1. Representación gráfica de tipos de IGA: (A)IGA crossover, (B) IGA no crossover y (C) no IGA 2

3.1. Chequeo de disponibilidad del nombre elegido 10
3.2. Creación del paquete geneticae 11
3.3. Archivo DESCRIPTION de geneticae 14
3.4. Resultado de correr los tests creados para geneticae 15
3.5. Análisis de cobertura de los tests de geneticae 16
3.6. Fragmento de README de geneticae 18
3.7. Archivo NEWS de geneticae 19

4.1. Biplot GE obtenido del modelo AMMI clásico basado en los datos de rendimiento de trigo de invierno obtenidos en Ontario en 1993. El 71,66 % de la variabilidad de la IGA se explica por los dos primeros términos multiplicativos. Los cultivares se muestran en letras minúsculas y los ambientes en mayúsculas. 26
4.2. Biplot GGE basado en datos de rendimiento de trigo de invierno obtenido de Ontario en 1993. El método de escala utilizado es la partición simétrica de valores singulares (opción por defecto). El 78 % de la variabilidad de G + GE se explica por los dos primeros términos multiplicativos. Los cultivares se muestran en minúsculas y los entornos en mayúsculas. 29
4.3. A: Ranking de cultivares en el ambiente OA93. B: Ranking de ambientes para cultivar Kat, basado en datos de rendimiento de trigo de invierno obtenido de Ontario en 1993. El método de escala utilizado es la partición simétrica de valores singulares (opción por defecto). El 78 % de la variabilidad de G + GE se explica por los dos primeros términos multiplicativos. Los cultivares se muestran en minúsculas y los entornos en mayúsculas. . . 30

4.4. comparación de los cultivares Kat y Cas. El método de escala utilizado es la partición simétrica de valores singulares (por defecto). El 78 % de la variabilidad de G + GE se explica por los dos primeros términos multiplicativos. Los cultivares se muestran en minúsculas y los entornos en mayúsculas. . . 31

4.5. Vista poligonal del biplot GGE, que muestra qué cultivares presentaron mayor rendimiento en cada ambiente/mega-ambiente. El método de escala utilizado es la partición simétrica de valores singulares (por defecto). El 78 % de la variabilidad de G + GE se explica por los dos primeros términos multiplicativos. Los cultivares se muestran en minúsculas y los entornos en mayúsculas. 32

4.6. A: Evaluación de los cultivares con base en el rendimiento promedio y la estabilidad y B: Clasificación de genotipos con respecto al genotipo ideal, basado en el escalado centrado en los genotipos. 34

4.7. A: Relación entre ambientes y B: Clasificación de ambientes con respecto al ambiente ideal, basado en el escalado centrado en los genotipos. 36

4.8. (A) Plrv dataset (B) yanwinterwheat dataset 38

4.9. Cargando el conjunto de datos *yanwinterwheat* en Geneticae APP 39

4.10. Diagrama de caja de (A) genotipos y (B) ambientes para el conjunto de datos *yanwinterwheat*. 40

4.11. Gráfico de correlación (A) y matriz (B) entre genotipos yanwinterwheat dataset 40

4.12. Gráfico de interacción para (A) ambientes a través de genotipos y (B) genotipos a través de entornos del conjunto de datos de *yanwinterwheat*. . 41

4.13. Boxplot de genotipos a través de los ambientes para el conjunto de datos Plrv 43

4.14. AMMI 44

Capítulo 1

Introducción

A lo largo de la historia de la agricultura, el hombre ha desarrollado el mejoramiento vegetal en forma sistemática y lo ha convertido en un instrumento esencial para incrementar la producción agrícola en términos de cantidad, calidad y diversidad. Las variedades mejoradas son el resultado del trabajo llevado a cabo en los programas de fitomejoramiento, los cuales se extienden a lo largo de varios años y requieren cuantiosas inversiones. En etapas avanzadas de estos programas, comúnmente se llevan a cabo ensayos multiambientales (EMA) de comparación de rendimientos, donde un conjunto de variedades se evalúan en múltiples ambientes. Estos son esenciales debido a la presencia de interaccion genotipo - ambiente (IGA) la cual es inevitable debido a las variaciones en las condiciones climáticas y de suelo de los distintos ambientes analizados. La IGA es considerada casi unánimemente por los fitomejoradores como el principal factor que limita la selección de cultivares superiores y, en general, afecta la eficiencia de los programas de mejoramiento (Crossa et al., 1990; Cruz Medina, 1992; Kang y Magari, 1996). Cuando los ambientes son muy diferentes, la IGA usualmente gana importancia porque cambia el rango de las líneas de mejoramiento. Gauch y Zobel (1997) explicaron que si no hubiera interacción, una sola variedad o híbrido rendirían al máximo en todo el mundo, además los materiales podrían evaluarse en un solo lugar y proporcionarían resultados universales.

Peto (1982) ha distinguido las interacciones cuantitativas, conocidas también como sin cambio de rango o *no crossover* (NCOI), de las cualitativas, denominadas a su vez como con cambio de rango o *crossover* (COI) (Cornelius et al., 1996). Cuando dos genotipos G_1 e G_2 tienen una respuesta diferencial en dos ambientes, se dice que la IGA es del tipo COI si hay cambios en el orden de los genotipos según su rendimiento (Figura 1.1(A)) y del tipo NCOI si su ordenamiento permanece sin cambios (Figura 1.1(B)). Por otro lado, se dice que la IGA es inexistente cuando los genotipos responden de manera similar en ambos ambientes (Figura 1.1(C)).

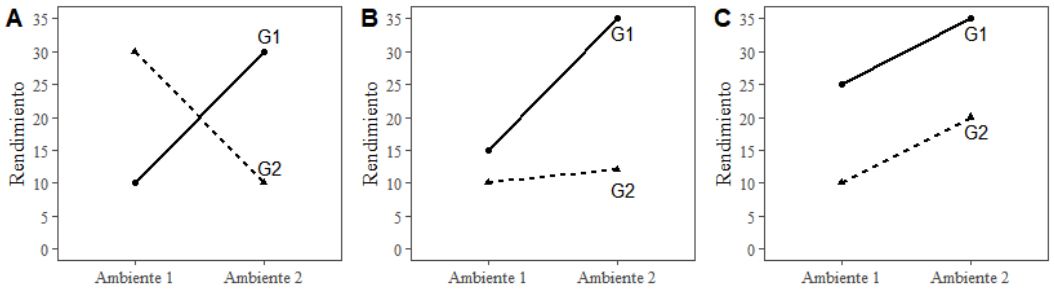

Figura 1.1: Representación gráfica de tipos de IGA: (A)IGA crossover, (B) IGA no crossover y (C) no IGA

Entre las implicancias negativas de la IGA en los programas de mejoramiento se encuentra el impacto negativo sobre la heredabilidad, cuanto menor sea la heredabilidad de un caracter, mayor será la dificultad para mejorarlo. Conceptos importantes tales como regiones ecológicas, ecotipos, mega-ambientes, adaptación especifica y estabilidad se pueden analizar a partir de la IGA (Yan y Hunt, 2001).

Un análisis adecuado de la información de los EMA es indispensable para el éxito del programa de mejoramiento genético de los cultivos. El rendimiento medio en los ambientes es un indicador suficiente del rendimiento genotípico sólo en ausencia de IGA (Yan y Kang, 2003). Sin embargo, la aparición de IGA es inevitable y no basta con la comparación de las medias de los genotipos, sino que se debe recurrir a una metodología estadística más aporopiada. Las más difundidas para analizar los datos provenientes de EMA se basan en modificaciones de los modelos de regresión, análisis de variancia (ANOVA) y técnicas de análisis multivariado.

Particularmente, para el estudio de la IGA y los análisis que de ella se derivan, dos modelos multiplicativos han aumentado su popularidad entre los fitomejoradores como herramientas de análisis gráfico: el modelo de los efectos principales aditivos e interacción multiplicativa (*Additive Main effects and Multiplicative Interaction*, AMMI) (Kempton, 1984; Gauch, 1988) y el de regresión por sitio (*Site Regression model*, SREG) (Cornelius et al., 1996; Gauch y Zobel, 1997). Estos modelos se ajustan en dos etapas. Primero, se realiza un ANOVA para obtener estimaciones de los efectos principales aditivos de ambientes y genotipos en AMMI y sólo de los ambientes en SREG. En segundo lugar, los residuos del ANOVA se ordenan en una matriz con genotipos en las filas y ambientes en las columnas y se aplica una descomposición de valores singulares (DVS), representando los patrones de GEI presentes en los residuos en AMMI, y de genotipo (G) y GEI en SREG. El resultado de los dos primeros términos multiplicativos de la DVS a menudo se presentan en un biplot llamado GE para el modelo AMMI y GGE para SREG (Cornelius

et al., 2002). Sin embargo, estos modelos no siempre son lo suficientemente eficiente para analizar la estructura de datos MET de programas de mejoramiento vegetal. Por un lado, tiene serias limitaciones frente a información faltante y, apesar de que los MET están diseñados para que todos los genotipos se evalúen en todos los ambientes, la presencia de valores perdidos es muy común. Esto ocurre, por ejemplo, debido a errores de medición o destrucción de plantas por presencia de animales, inundaciones o problemas durante la cosecha, además de la dinámica propia de la evaluaciones en las que se incorporan y se descartan genotipos debido a su mal desempeño (Hill y Rosenberg, 1985). Numerosas metodologías de imputación se han estado desarrollando en los ultimos años para solventar esta limitación (Arciniegas-Alarcón et al., 2010, 2014). Por otro lado, ambos modelos son sensibles a la presencia de observaciónes atipicas, lo cual es una regla más que una excepción cuando se consideran datos reales. Para superar esta fragilidad, recientemente distintas metodologias robustas se han desarrollado para el modelo AMMI (Rodrigues et al., 2016).

En este contexto, el análisis de datos provenientes de EMA requiere metodología estadística cuyas rutinas informáticas no se encuentran disponibles en programas comerciales debido a su reciente desarrollo o bien se deben utilizar varios de ellos para cumplir un único objetivo. Esto último genera el inconveniente de tener que disponer de todos los programas necesarios para los distintos análisis, atender los requerimientos de formatos de datos usados por cada uno de ellos, y comprender los diversos tipos de salidas en las que se presentan los resultados obtenidos. Además, los costos de las licencias algunos programas pueden resultar muy elevados.

Ante estas dificultades, una alternativa para el análisis es el empleo de algún lenguaje de programación de distribución libre y gratuita, que le confiera al analista la flexibilidad necesaria para cumplir con su objetivo. En este contexto, R es uno de los lenguajes de programación desarrollados para el análisis de datos de mayor uso en la actualidad. R es un software de uso libre y distribuido bajo los términos de la *General Public Licence* (GNU). Este programa se descarga de un repositorio mantenido por *The R Foundation for Statistical Computing* conocido como CRAN (*Comprehensive R Archive Network*), en el cual también se encuentran disponibles miles de paquetes adicionales que consisten en conjuntos de funciones desarrolladas con fines específicos que se distribuyen con un protocolo determinado, garantizando su correcto funcionamiento. Cualquier desarrollador puede producir su propio paquete y publicarlo en CRAN, siempre que cumpla con los requisitos establecidos y pase correctamente por los procedimientos de control. Además, hay paquetes que pueden obtenerse de otros repositorios como Github, Bioconductor, rOpenSci, entre otros. R es propicio para el análsis de datos de EMA puesto que se ha desarrollado metodología específica para este entorno computacional.

A pesar de estas ventajas, el análisis de datos de EMA en R presenta algunos desafíos. Por un lado, existen numerosos paquetes con funcionalidad afín que hay que identificar cómo combinar adecuadamente. Por otro lado, el software puede resultar dificultoso para aquellos analistas no familiarizados con la programación. Atendiendo a estas dos necesidades, se crea un paquete que incluya metodología recientemente publicada y reuna las funciones más útiles a fin de solventar la primera de ellas. Para la segunda, se crea una aplicación web Shiny de libre acceso mediante conexión a internet que permita realizar los principales análisis implementados en el paquete sin necesidad de escribir lineas de código.

Capítulo 2

Objetivos

2.1. Objetivo general

Desarrollar un paquete de R para el análisis de datos provenientes de EMA y una interfaz gráfica de usuario para el mismo a través de la aplicación web Shiny.

2.2. Objetivos específicos

- Mostrar un flujo de trabajo reproducible para la construcción de paquetes de R.
- Programar e incluir en el paquete metodología para el análisis de datos provenientes de EMA recientemente publicada y no disponible en R.
- Añadir en el paquete de R funciones ya existentes con modificaciones o agregados para favorecer su uso.
- Desarrollar una aplicación web shiny que sirva como interfaz gráfica de usuario para el paquete.
- Publicar el paquete y la aplicación web para su libre uso.

Capítulo 3

Métodos

3.1. Métodos estadísticos

Este capítulo tiene como objetivo focalizar al lector en los aspectos fundamentales sobre los que se apoya este trabajo. Se compone de tres secciones, en la primera se presenta la metodología estadística que se incluirá en el paquete de R. En la segunda y tercera sección, se presenta un flujo de trabajo reproducible para el desarrollo del paquete y la aplicación web Shiny, respectivamente.

3.1.1. Modelo AMMI y SREG

El modelo AMMI (Zobel et al. 1988) y SREG (Cornelius et al., 1996; Crossa y Cornelius, 1997; Crossa et al., 2002) son modelos lineales-bilineales que difieren en las componentes aditivas que se eliminan de la ecuación y se incorporan en forma multiplicativa. Mientras que AMMI considera los ambientes (A) y el genotipo (G) como aditivos, SREG solo A. Por lo tanto, en el primero la IGA representa la parte multiplicativa y en el segundo G e IGA en forma conjunta.

Los parámetros multiplicativos, tanto en el modelo AMMI como en el SREG, se estiman por medio de la Descomposición en Valores Singulares (DVS) de la matriz que contiene los residuos para cada combiánción de genotipo y ambiente luego de ajustar por mínimos cuadrados el modelo de efectos principales.

Las ecuaciones de los distintos modelos son:

AMMI: $y_{ij} = \mu + G_i + A_j + \sum_{k=1}^{K} \lambda_k \alpha_{ik} \gamma_{jk}$

SREG: $y_{ij} = \mu + A_j + \sum_{k=1}^{K} \lambda_k \alpha_{ik} \gamma_{jk}$

donde

- y_{ij} es el caracter fenotípico evaluado (rendimiento o cualquier otro caracter de interes) del i-ésimo genotipo en el j-ésimo ambiente,
- μ es la media general,
- G_i es el efecto del i-ésimo genotipo con $i = 1, ..., g$,
- A_j es el efecto del j-ésimo ambiente con $j = 1, ..., a$,
- $\sum_{k=1}^{K} \lambda_k \alpha_{ik} \gamma_{jk}$ es la sumatoria de términos multiplicativas utilizadas para modelar la IGA en AMMI o G e IGA en SREG. Siendo, K el número de términos multiplicativos retenidos en el modelo con $K \leq min(g-1, a-1)$ en AMMI y $K \leq min(g, a-1)$ en SREG; λ_k el k-ésimo valor singular y α_{ik} y γ_{jk} el elemento del autovector asociado con el i-ésimo genotipo y el j-ésimo ambiente para el k-ésimo término multiplicativo. Generalmente los dos primeros términos multiplicativos ($K = 2$) son suficientes para explicar los patrones de la IGA en AMMI y de G e IGA en SREG; la variabilidad remanente se interpreta como ruido aleatorio.

El resultado de los dos primeros términos multiplicativos de la SVD se presenta a menudo en un biplot llamado GE (*Genotipe-Environment*) (**CITA**) en AMMI y GGE en SREG (Yan et al., 2000) y representan una aproximación de dos rangos de los efectos multiplicativos. Dado que para seleccionar cultivares, el efecto de G e IGA debe considerarse simultáneamente, el modelo SREG resulta superior a AMMI para visualizar patrones en datos EMA. Un biplot GGE que explica suficiente variabilidad debida a G e IGA de un conjunto de datos EMA permite, entre otras cosas, la visualización de tres aspectos importantes:

(i) las relaciones entre los genotipos y ambientes representadas por el patrón *which-won-where*, que facilitan la investigación del megaambiente (Gauch y Zobel, 1997);

(ii) las interrelaciones entre los ambientes de prueba que permiten la identificación de mejores para la evaluación de cultivares (Cooper et al., 1997) y de aquellos redundantes que pueden ser descartados (Yan y Rajcan, 2002);

(iii) las interrelaciones entre genotipos que posibilita la comparación entre ellos y la clasificación de los mismos considerando tanto en el rendimiento medio como la estabilidad (Yan et al., 2001).

En un biplot, el i-ésimo genotipo se muestra como un punto definido por todos los valores $g_{ik} = \lambda_k^s \gamma_{ik}$, y el j-ésimo ambiente por $e_{kj} = \lambda_k^{1-s} \delta_{jk}$ ($k = 1, 2$ para un biplot bidimensional), donde s es el factor de partición. Cuando $s = 1$ se denomina escalamiento centrado en los genotipos, centrado en los ambientes si $s = 0$ y simétrica cuando $s = 0{,}5$. El último factor es el utilizado en el biplot GE y el más utilizado en GGE, aunque dependiendo de los intereses de la investigación, se pueden construir numerosos biplots derivados de SREG. Independientemente del factor de partición utilizado, los biplots GGE revelan el mismo patrón *which-won-where*. Sin embargo, difieren en diversos aspectos. El

primero muestra la interrelación entre genotipos con mayor precisión que cualquier otro método, el centrado en el ambiente es el más informativo de las interrelaciones entre los ambientes, mientras que el simétrico permite visualizar la magnitud relativa tanto de la variación de los genotipos como de los ambientes.

3.1.2. Modelo AMMI robusto

El modelo AMMI, en su forma estándar, asume que no hay valores atípicos en el conjunto de datos. Sin embargo, la presencia de *outliers* es más una regla que una excepción cuando se consideran datos agronómicos debido características inherentes a los genotipos que se evalúan, errores de medición o el efecto inesperado de plagas o enfermedades que pueden afecta el rendimiento de algunos genotipos.

Rodrigues et al. (2016) proponen una generalización robusta del modelo AMMI, que resulta de ajustar la regresión robusta basada en el estimador M-Huber (Huber, 1981) y luego utilizar un procedimiento DVS / PCA robusto. Consideraron varios métodos de DVS / ACP dando lugar a un total de cinco modelos robustos llamados: R-AMMI, H-AMMI, G-AMMI, L-AMMI, PP-AMMI.

El empleo de la versión robusta del modelo AMMI puede ser extremadamente útil debido a que una mala representación de genotipos y ambientes puede resultar en un mala decisión con respecto a qué genotipos seleccionar para un conjunto dado de ambientes (Gauch1997, Yan et al. 2000). A su vez, la elección de los genotipos incorrectos pueden provocar grandes pérdidas en términos de rendimiento. Los biplots obtenidos de los modelos robustos mantienen las características e interpretación estándar del modelo AMMI clásico (Rodrigues et al., 2016).

3.1.3. Métodos de imputación

Una limitación importante que presentan los modelos multiplicativos descriptos previamente es que requieren que el fenotipo de todas las combinaciones de genotipos y ambientes se encuentre registrado, es decir no admiten valores perdidos. Aunque los EMA están diseñados para que todos los genotipos se evalúen en todos los ambientes, la presencia de valores faltantes es muy común debido a errores de medición o pérdidas de plantas por animales, inundaciones o problemas durante la cosecha, además de la dinámica propia de la evaluaciones en las que se incorporan y se descartan genotipos debido a su pobre desempeño (Hill y Rosenberg, 1985).

Se han propuesto numerosas metodologías para superar el problema de valores ausentes en el conjunto de datos, entre las cuales se encuentran:

- EM-AMMI: Gauch y Zobel (1990) desarrollaron un procedimiento iterativo que utiliza el algoritmo de maximización de la esperanza (EM, del inglés *Expectation-Maximization*) incorporando el modelo AMMI.
- EM-SVD: Perry (2009) propone un método de imputación que combina el algoritmo EM con DVS.
- EM-PCA: Josse y Husson (2013) proponen imputar los valores faltantes de un conjunto de datos mediante un ACP.
- Gabriel Eigen: Arciniegas-Alarcón et al. (2010) presentan un método de imputación que combina regresión y aproximación de rango inferior usando DVS.
- WGabriel Eigen: Arciniegas-Alarcón et al. (2014) plantean una extensión ponderada del método Gabriel Eigen.

3.2. Paquete de R

Un paquete de R consisten en conjuntos de funciones desarrolladas con fines específicos que se distribuyen con un protocolo determinado, garantizando su correcto funcionamiento. Para la creación del mismo se deben seguir ciertas convenciones, existiendo elementos obligatorios y otros opcionales. Entre los primeros se encuentran:

- Archivo DESCRIPTION: describe el contenido del paquete y establece cómo se va a relacionar con otros.
- Carpeta R: contiene el o los archivos de código de R con las funciones del paquete.
- Carpeta man: incluye archivos con la documentación del paquete, funciones y datasets.
- Archivo NAMESPACE: declara las funciones del paquete que se ponen a disposición de los usuarios y de qué funciones de otros paquetes hace uso.

Los elementos opcionales que se pueden agregar son por ejemplo:

- Carpeta data: contiene objetos de R que contienen datos.
- Carpeta vignettes: contiene los tutoriales que muestran ejemplos de uso del paquete, generalmente escritos en Rmarkdown.
- Carpeta tests: incluye código que permiten someter al paquete a diversos controles.

Para la creación del paquete, se deben instalar y cargar en la sesión de trabajo los paquetes: *devtools*, *usethis*, *roxygen2*, *usestthat*, *knitr*, *available*. Además, en caso de utilizar el Windows se debe descargar e instalar Rtools .

3.2.1. Creación del paquete

En primer lugar se debe elegir el nombre del paquete, el cual debe cumplir con ciertas reglas: solo puede contener letras, números o puntos; tener al menos dos caracteres y empezar con una letra y no terminar con un punto. Una vez elegido el nombre, se debe chequear si el mismo está disponible en los repositorios *GitHub*, *CRAN* y *Bioconductor*, donde se alojan los paquetes. Para ello, se utiliza el paquete *available*, que además indicará si el nombre elegido tiene algún significado especial que podemos desconocer (revisa las webs de Wikipedia, Wiktionary y Urban Dictionary) (Figura 3.1).

```
# Cargar la libreria devtools
library(available)
# Crear el paquete geneticae
available("geneticae")
```

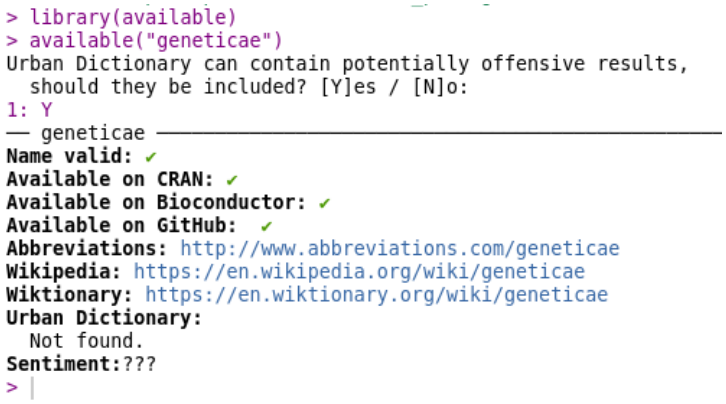

Figura 3.1: Chequeo de disponibilidad del nombre elegido

Para la creación del paquete se utilizán *devtools* y *usethis* que incluyen funciones que simplifican la tarea. La función create_package("nombre_paquete") generará una carpeta con el nombre provisto (Figura 3.2). Si nose especifica una ubicación entonces se creará en el directorio actual.

```
# Cargar la libreria devtools
library(devtools)
# Crear el paquete geneticae
create_package("/home/julia-fedora/Escritorio/geneticae")
```

```
> # Cargar la libreria devtools
> library(devtools)
Loading required package: usethis
> # Crear el paquete geneticae
> create_package("/home/julia-fedora/Escritorio/geneticae")
✓ Creating '/home/julia-fedora/Escritorio/geneticae/'
✓ Setting active project to '/home/julia-fedora/Escritorio/geneticae'
✓ Creating 'R/'
✓ Writing 'DESCRIPTION'
Package: geneticae
Title: What the Package Does (One Line, Title Case)
Version: 0.0.0.9000
Authors@R (parsed):
    * First Last <first.last@example.com> [aut, cre] (YOUR-ORCID-ID)
Description: What the package does (one paragraph).
License: `use_mit_license()`, `use_gpl3_license()` or friends to
    pick a license
Encoding: UTF-8
LazyData: true
Roxygen: list(markdown = TRUE)
RoxygenNote: 7.1.1
✓ Writing 'NAMESPACE'
✓ Writing 'geneticae.Rproj'
✓ Adding '.Rproj.user' to '.gitignore'
✓ Adding '^geneticae\\.Rproj$', '^\\.Rproj\\.user$' to '.Rbuildignore'
✓ Opening '/home/julia-fedora/Escritorio/geneticae/' in new RStudio session
✓ Setting active project to '<no active project>'
>
```

Figura 3.2: Creación del paquete geneticae

3.2.2. Archivos de código

Una vez creada la estructura del paquete se deben programar las funciones que el mismo contendrá. Cada una de ellas debe ser guardada en un archivo de extensión .R, en el subdirectorio R/. Para ello, se utiliza la función use_r() la cual crea un script ubicado en la carpeta R/, donde el código de interés será agregado.

A medida que se va desarrollando el paquete, con funciones internas y otras que se exportan, con algunas que se relacionan entre si y que a su vez dependen de otros paquetes, se deben ir realizando pruebas para asegurarse que los creados códigos realizan lo que realmente se desea. Para ello, la función load_all() simula el proceso de construcción, instalación y carga del paquete, permitiendo probar la función de manera interactiva.

Muy frecuentemente se utilizan funciones que se encuentran disponibles en otros paquetes. La función use_package() agrega el paquete de interés a la sección Imports del archivo DESCRIPTION, y luego para llamar a las mismas dentro de una función se debe utilizar @importFrom paquete función. Alternativamente, si se utilizan repetidamente muchas funciones de otro paquete, es posible importarlas todas utilizando @import paquete. Sin embargo, esta es la solución menos recomendada porque hace que el código sea más difícil de leer, y si tiene muchos paquetes, aumenta la posibilidad de que entren en conflicto nombres de funciones.

3.2.3. Documentación

Uno de los aspectos más importantes del paquete es la documentación donde se describe cómo se usa cada función, para qué sirven los argumentos, aclarar qué tipo de resultado devuelve, proveer ejemplos para el uso, etc. El paquete *roxygen2*, provee pautas para escribir comentarios con un formato especial que incluyan toda la información requerida justo antes de la definición de la función. El código y la documentación son adyacentes, de modo que cuando el código se modifique le exigirá que actualice la documentación.

El flujo de trabajo para crear la documentación con el paquete *roxygen2* es el siguiente:

- Agregar comentarios a los archivos .R. Estos deben comenzar con #', para distinguirlo de los comentarios regulares, y preceden a una función. La primera oración se convierte en el título y el segundo párrafo es una descripción de la función. Para el resto de los campos de la documentación, se utilizan de etiquetas que comienzan con @, siendo las más importantes a incluir:
  - @param: se detalla para qué sirve cada parámetro de la función.
  - @return: para explicar qué objeto devuelve la función.
  - @details: agregar cualquier aclaración que se considere necesaria.
  - @examples: incluir ejemplos de uso de la función.
  - @export: indicar que esta función tiene que estar disponible cuando alguien cargue el paquete con library(). No es necesario exportar funciones auxiliares de utilidad interna.
- Ejecutar devtools::document() para convertir los comentarios de roxygen en archivos .Rd que compondrán el manual y que deben ir guardados en la carpeta man.

Roxygen2 permite utilizar la descripción de los parámetros de otras funciones usando @inheritParams. Esta documentará los parámetros que no están documentados en la función actual, pero que si lo están en la función fuente. La fuente puede ser una función en el paquete actual, vía @inheritParams function, u otro paquete, vía @inheritParams package::function. Además *Roxygen2* permite incluir referencias utilizando @references. En caso de importar paquetes, como se indicó en la sección anterior, se deben declarar usando @importFrom o @import, previo a la definición de la función.

3.2.4. Editar el archivo DESCRIPTION

El archivo DESCRIPTION provee toda la metadata sobre el paquete que se esta creando. En este archivo hay algunos campos que tienen que estar presentes de forma obligatoria y otros que son opcionales. Los elementos obligatorios son:

- Package: nombre del paquete
- Title: título del paquete (hasta 65 caracteres, Escrito De Esta Forma).
- Version: número de la versión actual del paquete (por ejemplo, 0.2.1)
- Author, Maintainer o Authors@R: quiénes han participado en el paquete.
- Description: un párrafo que describa el paquete.
- License: nombre de la licencia bajo la cual se distribuye el paquete. Si se pretende que cualquiera lo puede usar, entonces se debe recurrir a los tipos mas comunes de licencia para código abierto: CC0, MIT o GPL. Como se muestra en la Figura 3.3, el paquete geneticae se encuentra bajo la licencia GPL-3. Para esto, se utilizó la función use_gpl3_license() del paquete usethis, la cual agrega la información al archivo DESCRIPTION y además crea un archivo LICENSE al directorio del paquete.

En cambio, los elementos no obligatorios:

- Date: fecha de publicación de esta versión del paquete.
- Imports, Depends, Suggests: es muy común que las funciones desarrolladas necesiten hacer uso de algunas que pertenecen a otros paquetes. Estos serán indicados en los campos Imports, Depends, Suggests del archivo DESCRIPTION. Como se muestra en la Figura 3.3, en el campo Imports se indica que geneticae necesita los paquetes: stats, GGEBiplots, ggforce, ggplot2, etc. Mientras que los listados en Suggest indica que se podría hacer uso de los mismos, aunque no son indispensables. Por último, el paquete geneticae se puede utilizar en versiones de R iguales o superiores a la 2.12, como se establece en Depends.
- URL: dirección de la página web del paquete.
- BugReports: dirección donde los usuarios pueden enviar avisos con los problemas que encuentren al utilizar el paquete.

El archivo DESCRIPTION del paquete geneticae se muestra en la Figura 3.3

```
Package: geneticae
Type: Package
Title: Statistical analysis tool for agricultural research
Version: 0.0.9000
Date: 13-04-2019
Author: Julia Angelini - Marcos Prunello - Gerardo Cervigni
Maintainer: Julia Angelini <jangelini_93@hotmail.com>
Description: Original idea was presented in the thesis to obtain the degree of
Bioinformatics, National University Rosario (UNR), Rosario Argentina. Some
experimental data for the examples come from INTA San Pedro and others research.
Geneticae offers extensive statistical analysis tool for agricultural and plant
breeding experiments, which can also be useful for other purposes.
License: GPL
Encoding: UTF-8
LazyData: true
NeedsCompilation: no
Repository: CRAN
RoxygenNote: 7.1.1
Imports: stats,
         GGEBiplots,
         ggforce,
         ggplot2,
         scales,
         MASS,
         pcaMethods,
         rrcov,
         dplyr,
         bcv,
         missMDA,
         calibrate,
         graphics,
         agridat,
         reshape2,
         matrixStats,
         tidyr,
         prettydoc
Suggests:
    knitr,
    rmarkdown,
    testthat (>= 2.1.0)
VignetteBuilder: knitr
Depends: R (>= 2.12.0)
URL: https://github.com/jangelini/geneticae
BugReports: https://github.com/jangelini/geneticae/issues

```

Figura 3.3: Archivo DESCRIPTION de geneticae

3.2.5. Testeos

Las pruebas resultan fundamentales en el desarrollo de paquetes, asegura que el código haga lo que realmente se desea. Existen pruebas informales como aquellas realizadas con la función load_all() que permite que las funciones creadas estén disponible rápidamente para uso interactivo. Sin embargo, las pruebas interactivas pueden convertirse en scripts reproducibles, los cuales resultan superiores debido a que se indica explícitamente cómo debería comportarse el código, provocando que los errores solucionados no vuelvan a ocurrir. Para ello, se utiliza la función use_testthat() del paquete *testthat* (Wickham,2011). Esta agrega testthat al campo Suggests en el archivo DESCRIPTION, crea un directorio tests/ para alojar cualquier tipo de unidad de testeo, una subcarpeta testthat donde se ubicaran los testeos escritos bajo este sistema y además, crea un archivo testthat.R, que

se encarga de la ejecución de todos los testeos.

Los testeos se organizan en tres niveles:

- Archivo de tests: uno por cada archivo .R en la carpeta R/.
- Ejecutar pruebas automáticamente cada vez que algo cambie con la función auto-test(). Estas son útiles cuando las pruebas se ejecutan con frecuencia. Si se modifica un archivo de prueba, probará ese archivo; si se modifica un archivo de código, volverá a cargar ese archivo y volverá a ejecutar todas las pruebas.
- Expectation: es el nivel más desagregado, corre cierto código y se compara el resultado obtenido con el esperado.

La función use_test(), creará los archivos de prueba cuyo nombre tienen que ser test-nombre_archivo_de_codigo.R y los ubicará en la carpeta test/testthat. Una vez escritos estos archivos, podemos evaluar los resultados de los testeos con devtools::test(). Ante cada error encontrado, nos detenemos para corregirlo y repetimos este proceso hasta que todas las unidades de testeo pasen la prueba. En la Figura 3.4 se muestra el resultado de correr los test creados para el paquete geneticae.

```
> devtools::test()
Loading geneticae

Attaching package: 'testthat'

The following object is masked from 'package:devtools':

    test_file

Testing geneticae
✓ |  OK F W S | Context
✓ |   6       | GGE_Model [0.1 s]
✓ |   1       | GGE_Plot
✓ |   4       | impute [0.3 s]
✓ |   3       | r_AMMI [0.2 s]

══ Results ═══════════════════════════════════════════════════════
Duration: 0.7 s

OK:       14
Failed:   0
Warnings: 0
Skipped:  0
>
```

Figura 3.4: Resultado de correr los tests creados para geneticae

Una medida de la calidad de un paquete está dada por el porcentaje de su código que es evaluado durante los testeos. El paquete covr permite hacer ese cálculo, además de mostrar interactivamente qué partes del código fueron evaluadas y cuáles no. Por un lado puede evaluarse en cada archivo .R mediante devtools::test_coverage_file(), o bien, la cobertura total usando devtools::test_coverage(). El paquete geneticae tiene un porcentaje total de cobertura de los test igual a 16.83 % (Figura 3.5).

geneticae coverage - 16.83%

Files | Source

File	Lines	Relevant	Covered	Missed	Hits / Line	Coverage
R/GGE_Plot.R	446	233	0	233	0	0.00%
R/W_GabrielEigen.R	264	131	0	131	0	0.00%
R/GabrielEigen.R	145	66	0	66	0	0.00%
R/rAMMI.R	229	121	2	119	0	1.65%
R/impute.R	188	49	37	12	1	75.51%
R/GGE_Model.R	107	27	21	6	4	77.78%
R/EM_AMMI.R	151	59	49	10	6	83.05%

Figura 3.5: Análisis de cobertura de los tests de geneticae

ampliar los test para tener mas cobertura

3.2.6. Compilación e instalación

La función check() o R CMD check ejecutado en el shell, es utilizado para verificar que un paquete R esta en pleno funcionamiento. La misma verificará que no haya errores de sintaxis o no se generen warnings. Está compuesto por más de 50 chequeos individuales entre los cuales se encuentran: la estructura del paquete, el archivo descripción, namespace, el código de R, los datos, la documentación, entre otros. La diferencia con los testeos realizados mediante use_testthat(), es que la última evalúa si las funciones desarrolladan realizan lo deseado, resultando propios de cada paquete. En cambio, lo realizado por R CMD check es común para todos los paquetes.

Se aconseja realizar verificaciones completas de que todo funciona a medida que se van incorporando funciones ya que si se incorporan muchas y luego se verifican será difícil identificar y resolver los problemas. Una vez que se desarrollaron todos los elementos necesarios para el paquete y no se detectan errores, advertencias o notas, se ejecuta la función install(), con el objetivo de instalar el paquete en la biblioteca.

3.2.7. Algunos elementos complementarios

Existen algunas componentes que no son obligatorias a la hora de desarrollar un paquete, pero que ayudan a la comprensión y difusión del mismo.

3.2.7.1. Viñetas

Una viñeta es un tipo especial de documentación que puede agregarse al paquete para dar más detalles y ejemplos sobre el uso del mismo. En ella se brinda es una descripción el problema que el paquete está diseñado para resolver y muestra al lector cómo resolverlo. Se diferencian de las páginas de ayuda en que su adición es opcional y no sigue una estructura fija, dándole la libertad al autor de enseñar de la forma que más le guste cómo usar su paquete.

Muchos de los paquetes existentes tienen viñetas la cuales se pueden encontrar utilizando la función browseVignettes("packagename") si el mismo se encuentra instalado, sino deben consultarse en su página de CRAN, por ejemplo para el paquete *dplyr*: `http://cran.r-project.org/web/packages/dplyr`. Cada viñeta proporciona el archivo fuente original, una página HTML o PDF y un archivo de código R.

Las Viñetas se pueden construir de diversas formas, en este trabajo se utiliza se utiliza usethis::use_vignette("tutorial_del_paquete"). La misma crea un directorio vignettes/, agrega las dependencias necesarias a DESCRIPTION y crea el archivo para redactar la viñeta.

3.2.7.2. Agregar datasets

A menudo es útil incluir datos en un paquete a fin de proporcionar ejemplos de las funciones incluidas en él. Esto es posible realizarlo con la función usethis::use_data() que crea un archivo .RData y lo almacena en el directorio data/. Notar que el archivo DESCRIPTION contiene el campo LazyData: true, lo cual genera que los conjuntos de datos no ocupen memoria hasta que sean usados.

Los objetos en la carpeta data siempre se exportan, por lo cual hay que agregar documentación para los mismos. A diferencia de las funciones que son documentadas directamente, para los objetos en data/, se debe crear un archivo y guardarlo en el directorio R/. Esto se puede hacer con roxygen en cualquier Rscript de la carpeta R, aunque se acostumbra juntar toda la documentación para todos los datasets en un único archivo llamado data.R.

3.2.7.3. Archivo README

Un archivo README es una forma de documentación de software que contiene información acerca de otros archivos en una carpeta. Usualmente es un archivo de texto plano

que permite describir brevemente por qué y para qué alguien tendría que usar el paquete, a la vez que indicar cómo conseguirlo o instalarlo. Se diferencian de la viñeta ya que sólo presenta una descripción breve del paquete.

Para generar el README con R Markdown se utiliza la función use_readme_rmd() la cual crea un archivo de Rmarkdown con una plantilla donde se escribirá el mismo y será además agregado a .Rbuildignore. Luego, al compilarlo con knitr se obtendrá un archivo README.md, que será la cara visible del paquete si, por ejemplo, en GitHub.

badges y logo

Las insignias o badges son unos íconos que señalan distintas características del paquete, como su nivel de maduración, el nivel de cobertura en el testeo, cantidad de descargas, número de versión, resultado de los controles de CRAN, etc. Son visualmente muy atractivas y se colocan el en archivo README. El paquete usethis trae un conjunto de funciones que generan automáticamente el código a incluir en el README.Rmd para agregar los badges: use_badge(badge_name, href, src), use_cran_badge(), use_bioc_badge(), use_lifecycle_badge(stage), use_binder_badge(sturlpath = NULL) .

Muchos de los paquetes disponibles disponen de un logo con forma hexagonal, que generalmente termina en forma de sticker: los hexStickers. Estos permiten terminar de darle identidad a tu paquete y hacerlo más vistoso. Para crearlos existe el paquete HexSticker. Una vez creado el logo, le podemos pasar su ubicación a la función use_logo(), que se encargará de darle el tamaño adecuado, guardarlo en la carpeta man del paquete y producir el código de Markdown para incluirlo en el README. La Figura 3.6 presenta un fragmento GitHub del paquete geneticae, donde se muestra parte del contenido del archivo README, badges y logo.

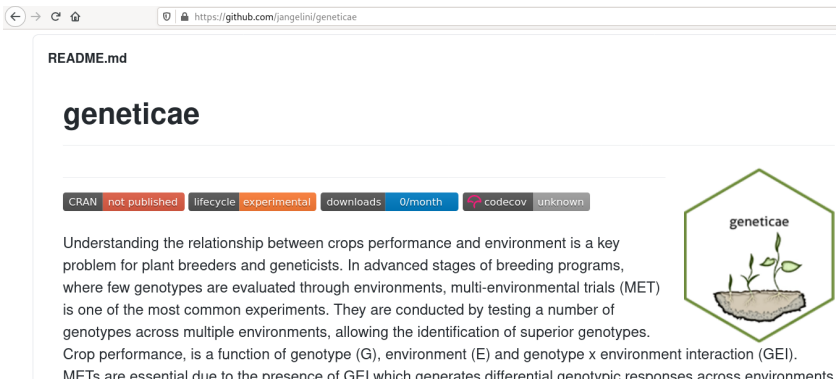

Figura 3.6: Fragmento de README de geneticae

3.2.7.4. Archivo NEWS

Mientras que el README apunta a ser leído por nuevos usuarios, el archivo NEWS es para aquellos que ya usan el paquete. Este archivo se encarga de contar los cambios presentes en cada versión nueva del paquete que publicamos. Se sugiere usar Markdown para escribir este archivo y colocar un título principal para cada versión, seguido por títulos secundarios que describen lo realizado (cambios principales, bugs arreglados, etc.). Si se trata de cambios impulsados por otras personas, por ejemplo, a través de sugerencias hechas en GitHub, se los menciona. Una buena práctica es ir escribiendo este archivo cada vez que se realiza algo nuevo en el paquete. La función que nos permite crear este archivo automáticamente es usethis::use_news_md(). Hasta el momento, del paquete geneticae solamente se cuenta con la versión de desarrollo (Figura 3.7)

```
1  # geneticae 0.0.9000
2
3  * This is the development version of the package.
4
5
```

Figura 3.7: Archivo NEWS de geneticae

3.2.7.5. Crear una página web

Para mayor visualización del paquete es posible crear una página web [1]. El paquete pkgdown está diseñado para la creación de un sitio web de manera rápida y sencilla. Utiliza todo lo creado hasta el momento y lo convierte automáticamente en una pagina web mediante la función pkgdown::build_site().

No aparecen los badges en la pag web

3.2.7.6. Publicación

Por último, para que otros usuarios puedan utilizar el paquete es necesario compartirlo en alguno de los repositorios más populares de paquetes de R, entre los cuales se encuentran: CRAN, Bioconductor, GitHub, rOpenSci, R-Forge y RForge. El paquete *geneticae* se encuentra en GitHub, por lo tanto, para instalar el mismo se debe utilizar devtools::install_github("jangelini/geneticae").

[1] Para visitar la página web de geneticae debe dirigirse a `https://.....`

3.3. Shiny APP

Shiny es un paquete R que permite construir aplicaciones web directamente desde RStudio sin necesidad de conocer en profundidad los lenguajes HTML / CSS / JavaScript . Estas aplicaciones constituyen una interfaz gráfica entre el usuario y R, que permiten realizar un análisis a través de un navegador web sin necesidad de programar.

Una característica importante de las aplicaciones web creadas mediante Shiny es que son dinámicas e interactivas. Para que shiny funcione correctamente, es necesario tener instalado R 3.0.2 o cualquier versión posterior.

3.3.1. Estructura de Shiny APP

Las aplicaciones están compuestas por la interfaz de usuario, ui (*user interfaz*), sección server y la función shinyApp().

Interfaz del usuario

La interfaz del usuario (user interface o ui, por sus siglas en inglés) controla el diseño de la aplicación, recibe los inputs y muestra los outputs en el navegador. En general, definir las características de la interfaz puede no resultar tan sencillo ya que muchas de sus herramientas están vinculadas a otros lenguajes de programación, por ejemplo HTML, CSS o JavaScript. Sin embargo, las funciones del paquete shiny facilitan la tarea sin necesidad de conocer en profundidad estos lenguajes.

Server

En la sección server se escribe el código de R que le indica a la app qué debe hacer y cómo debe funcionar, incluyendo la lectura y manipulación de datos, el armado de gráficos, el ajuste de modelos, etc. Para esto, se define una función que debe tener dos argumentos: input y output. Los mismos son listas que almacenan elementos de entrada (datos u opciones elegidas por el usuario a través de la ui) y elementos de salida para mostrar en la app (resultados, tablas, gráficos, mapas, etc.), respectivamente.

Ejecución

Por último, se llama a la función shinyApp(), cuyos dos argumentos principales son ui y server, es decir, cada uno de los elementos definidos anteriormente. Ejecutar esta función da como resultado el lanzamiento de la aplicación, la cual podremos utilizar dentro de RStudio o usando nuestro navegador (Google Chrome, Mozilla Firefox, Microsoft Edge, etc.). Es importante destacar que, al seguir estos pasos, la aplicación sólo funcionará mientras la sesión de RStudio desde la cual se lanzó siga vigente.

3.3.2. Desarrollo de Shiny APP

Una forma de desarrollar una aplicación es a partir de un nuevo directorio con un sólo archivo llamado app.R, como se muestra a continuación.

```
library(shiny)
ui<- ...
server<- ...
shinyApp(ui = ui, server = server)
```

En este archivo se carga el paquete shiny, se define la interfaz de usuario, la función server y por último, se ejecuta función que permite construir e iniciar una aplicación. Al ejecutar la aplicación la misma aparecerá, de manera predeterminada, en una ventana emergente. Sin embargo, otras dos opciones se pueden configurar desde el menú desplegable de *Run App*. Una de ellas es la ejecución en el panel del visor que permite verla al mismo tiempo que ejecuta el código. La segunda opción es ejecutar en un navegador externo mostrando la aplicación como la mayoría de los usuarios la verán. Dado que la sesión de R estará monitoreando la aplicación y ejecutando las ordenes dadas por el usuario, no se podrá ejecutar ningún comando.

En cualquier lenguaje de programación tener el código duplicado genera un desperdicio computacional y, lo que es más importante, aumenta la dificultad de mantener o depurar el código. Cuando se programa en R, se utilizan dos técnicas para lidiar con el código duplicado: guardar un valor usando una variable o utilizar una función para almacenar un cálculo. Ninguno de estos enfoques son apropiados en una Shiny APP, sino que se utilizan expresiones reactivas. Una expresión reactiva tiene una diferencia importante con una variable: sólo se ejecuta la primera vez que se llama y luego almacena en caché el resultado de la misma hasta que necesite actualizarse. La programación reactiva es un estilo de programación que enfatiza valores que cambian con el tiempo, y cálculos y acciones que dependen de esos valores. Esto es importante para las aplicaciones Shiny porque son interactivas: los usuarios cambian los inputs, lo que hace que la lógica se ejecute en el servidor que finalmente resultan en actualización de los outputs/resultados.

Entre los problemas que pueden surgir al crear una Shiny app se encuentran los errores inesperados, no se obtiene ningún error pero el valor obtenido es incorrecto, o bien todos los resultados son correctos, pero no se actualizan cuando se deben. Una vez localizada la fuente del error, la herramienta más poderosa es el depurador interactivo, éste detiene la ejecución y brinda una consola interactiva donde puede se ejecutar cualquier código para descubrir el error. Para iniciar el mismo, se puede agregar la función browser() en el código fuente, o bien agregar un punto de interrupción RStudio haciendo clic a la izquierda del número de línea.

Al modificar la aplicación, se la ejecuta para poder ver los cambios realizados, por lo tanto resulta esencial reducir la velocidad de iteración. La primera forma acelerar el proceso consiste en escribir el código, utilizar el atajo del teclado Cmd/Ctrl+ Shift+ Enter en lugar del botón "Ejecutar aplicación", experimentar interactivamente con la aplicación y cerrar la aplicación, repitiendo este proceso al realizar cualquier cambio. Otra forma de reducir aún más la velocidad de iteración es activar la recarga automática (options(shiny.autoreload = TRUE)) y luego ejecutar la aplicación en un trabajo en segundo plano. Con este flujo de trabajo cuando se guarde un archivo, su aplicación se reiniciará: no es necesario cerrarla y reiniciarla, lo cual conduce a un flujo de trabajo aún más rápido. La principal desventaja de esta técnica es que debido a que la aplicación se ejecuta en un proceso separado, es considerablemente más difícil de depurar.

3.3.3. Compartiendo una Shiny Web App

Una vez creada la aplicación se la publica para su libre uso. En este caso la Shiny Web App encuentra disponible en el servidor de CONICET `www.cefobi.com`. Además el proyecto se encuentra en GitHub `https://github.com/jangelini/shinyAPP_geneticae`.

Capítulo 4

Resultados

En esta sección se muestran ejemplos de uso tanto del paquete geneticae como de Geneticae Shiny Web APP.

4.1. Paquete de R *geneticae*

Para instalar la versión del paquete publicada en CRAN: install.packages("geneticae"), mientras que la versión en desarrollo se debe instalar desde el repositorio de Github: devtools::install_github("jangelini/geneticae").

Una vez instalado el paquete, se debe cargar en la sesion de R mediante el comando: library(geneticae).

Información detallada sobre las funciones del paquete geneticae se puede obtener mediante help(package = "geneticae"). La ayuda para una función, por ejemplo imputation(), en una sesión R se puede obtener usando *?imputation* o help(imputation). La función browseVignettes("geneticae") permite obtener la viñeta del paquete, es decir una descripción el problema que está diseñado para resolver así como ejemplos de aplicación del mismo.

Además, se encuentra disponible una página web (http://....) que contiene una breve descripción de la utilidad del paquete, las funciones que se incluyen en él, un tutorial de uso, un enlace de acceso a la shiny app, entre otra información.

4.1.1. Conjuntos de datos en geneticae

El paquete geneticae proporciona dos conjuntos de datos que pueden utilizarse para ilustrar la metodología incluida para analizar los datos provenientes de EMA.

- *yan.winterwheat dataset* (Wright, 2018): se cuenta con información sobre el rendimiento de 18 variedades de trigo de invierno cultivadas en nueve ambientes en Ontario en 1993. A pesar de que el experimento contaba con cuatro bloques o réplicas en cada ambiente, sólo el rendimiento medio para cada combinación de variedad y ambiente se encuentra disponible.

```
data(yan.winterwheat)
head(yanwinterwheat)
```

```
##   gen  env yield
## 1 Ann BH93 4.460
## 2 Ari BH93 4.417
## 3 Aug BH93 4.669
## 4 Cas BH93 4.732
## 5 Del BH93 4.390
## 6 Dia BH93 5.178
```

- *plrv dataset* (CITA AGRICOLAE): se registró información sobre el rendimiento, el peso de planta y de la parcela de 28 genotipos en 6 localidades de Perú con el fin de estudiar la resistencia a PLRV (*Patato Leaf Roll Virus*) causante del enrollamiento de la hoja. Cada clon fue evaluado tres veces en cada ambiente.

```
data(plrv)
head(plrv)
```

```
##   Genotype Locality Rep WeightPlant WeightPlot    Yield
## 1   102.18     Ayac   1   0.5100000       5.10 18.88889
## 2   104.22     Ayac   1   0.3450000       2.76 12.77778
## 3   121.31     Ayac   1   0.5425000       4.34 20.09259
## 4   141.28     Ayac   1   0.9888889       8.90 36.62551
## 5   157.26     Ayac   1   0.6250000       5.00 23.14815
## 6    163.9     Ayac   1   0.5120000       2.56 18.96296
```

En las siguiente subsecciones se muestran las herramientas de análisis incluidas en el paquete utilizando el conjunto de datos *yan.winterwheat*.

4.1.2. Modelo AMMI

Para visualizar el efecto de IGA se utiliza el biplot GE obtenido del modelo AMMI. Este gráfico es posible obtenerlo utilizando la función rAMMI(). Esta función requiere datos en formato largo, es decir, cada fila corresponde a una observación y cada columna a una variable (genotipo, ambiente, repetición (si existe) y fenotipo observado). Si cada

genotipo ha sido evaluado más de una vez en cada ambiente, la media fenotípica para cada combinación de genotipo y ambiente se calcula internamente y luego se estima el modelo. Las variables adicionales que no se utilizarán en el análisis pueden estar presentes en el conjunto de datos. No se permiten valores perdidos pero se pueden imputar como se indicará más adelante.

El biplot clásico para el conjunto de datos *yan.winterwheat* se muestra en la figura 4.1 junto con la sentencia utlizada para obtener el mismo. El primer argumento es el conjunto de datos de entrada, luego se indican los nombres de las columnas en las cuales se encuentra la información necesaria para aplicar la técnica y además el biplot que se desea obtener que por defecto es el derivado del modelo AMMI clásico. Opcionalmente, el porcentaje de IGA explicado por el biplot se puede agregar como una nota al pie con el argumento $footnote = T$ así como un título con $titles = T$.

En este ejemplo, BH93, KE93 y OA93 son los ambientes que más contribuyen a la interacción ya que sus vectores son los de mayor magnitud. Los cultivares m12 y Kat presentan patrones de interacción similares (sus marcadores están próximos entre sí) y son muy diferentes de Ann y Aug, por ejemplo. La cercanía entre el cultivar Dia y el ambiente BH93 indica una fuerte asociación positiva entre ellos, lo que significa que BH93 es un ambiente extremadamente favorable para ese genotipo. Como los marcadores OA93 y Luc son opuestos, este entorno es considerablemente desfavorable para ese genotipo. Por último, Cas y Reb están cerca del origen, lo que significa que se adaptan por igual a todos los entornos. Por último, se observa que los genotipos Cas y Reb están próximos al origen, lo que quiere decir que se adaptan en igual medida a todos los ambientes.

```
rAMMI(yan.winterwheat, genotype = "gen", environment = "env",
      response = ""yield", type = "AMMI", footnote = F, titles = F)
```

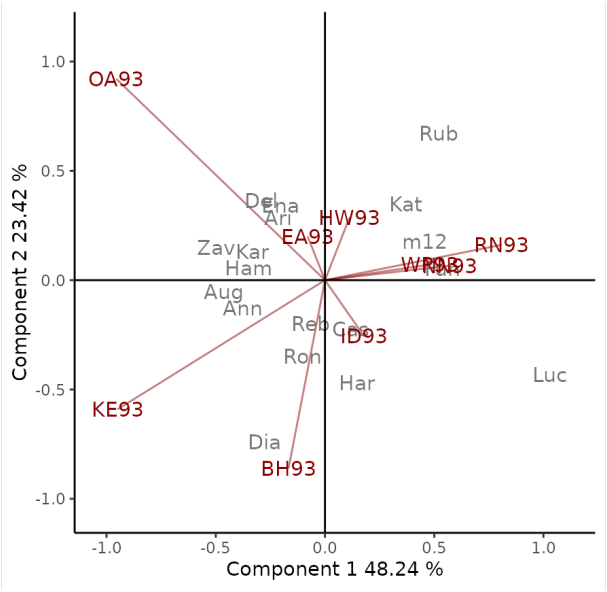

Figura 4.1: Biplot GE obtenido del modelo AMMI clásico basado en los datos de rendimiento de trigo de invierno obtenidos en Ontario en 1993. El 71,66 % de la variabilidad de la IGA se explica por los dos primeros términos multiplicativos. Los cultivares se muestran en letras minúsculas y los ambientes en mayúsculas.

El modelo AMMI, en su forma estándar, asume que no hay valores atípicos presentes en los datos. Para superar el problema de la contaminación de datos con *outliers*, Rodrigues et al. (2016) propuso cinco modelos AMMI robustos, los cuales no se encuentran disponible en R hasta el momento.

Todos los biplots robustos propuestos por Rodrigues et al. (2016) se puede obtener usando rAMMI(), indicando en el argumento *type* cuál de ellos se desea ajustar: "rAMMI", "hAMMI", "gAMMI", "lAMMI", "ppAMMI".

Dado que el conjunto de datos de muestra *yan.winterwheat* no presenta valores atípicos, las conclusiones obtenidas con biplots robustos no diferirán de las obtenidas con el biplot clásico (Rodrigues et al., 2016). Por lo tanto, no se presenta ninguna interpretación de los biplots robustos.

4.1.3. Modelo de Regersión por Sitio

Para visualizar conjuntamente el efecto de G e IGA Yan et al. (2000) propuso el biplot GGE mediante el cual se pueden abordar diversos aspectos relacionados con la evaluación de genotipos y ambientes. Para obtener dicho biplot en primer lugar se debe ajustar el modelo SREG mediante la función GGEmodel().

La función GGEmodel() propuesta en este paquete es un *wrapper* de GGEModel() del paquete GGEBiplots (Dumble, 2017). Como en el caso de rAMMI(), los datos deben presentarse en un formato largo y se permiten repeticiones o variables adicionales en el conjunto de datos. El rasgo fenotípico para cada combinación de genotipo y ambiente debe estar registrado.

La sentencia utilizada para ajustar el modelo GGE en el conjunto de datos *yan.winterwheat* se muestra a continuación. El primer argumento de la misma consiste en el nombre del conjunto de datos y en los siguientes indican los nombres que reciben las columnas que contienen la información de los genotipos, ambientes y del rasgo fenotípico de interés. Por defecto, la función considera que no hay réplicas en el conjunto de datos, sin embargo, si existieran en el parámetro *rep* se debe indicar el nombre de la columna con dicha información. Otros argumentos de dicha función son el método de centrado, SVD y escalado. Por defecto los datos se centran utilizando la opción *centering="tester"* lo cual resulta en el modelo SREG, otro valor dará lugar a un modelo diferente. La elección del método de SVD no altera las relaciones o interacciones relativas entre los genotipos y los ambientes, aunque la apariencia del biplot será diferente (Yan 2002). El centrado respecto a los genotipos (*SVP="row"*) muestra la interrelación entre genotipos con mayor precisión, el de los ambientes (*SVP="column"*) es el más informativo de las interrelaciones entre los ambientes, mientras que el simétrico (*SVP="symmetrical"*) permite visualizar la magnitud relativa tanto de la variación de los genotipos como de los ambientes, por lo que se utiliza por defecto. Por último, se indica que los datos no se deben escalar con el parámetro *scaling="none"*.

```
GGE1 <- GGEmodel(yan.winterwheat, genotype = "gen", environment = "env",
                 response = "yield", rep = NULL, centering = "tester",
                 scaling = "none", SVP = "symmetrical")
```

La salida de GGEmodel () es una lista con los siguientes elementos:

- coordgenotype: coordenadas para los genotipos en cada componente.
- coordenviroment: coordenadas para los ambientes en cada componente.
- eigenvalues: vector de autovalores para cada componente.
- vartotal: varianza general.

- varexpl: porcentaje de varianza explicado por cada componente.
- labelgen: nombres de genotipos.
- labelenv: nombres de entorno.
- axes: etiquetas de eje.
- Data: datos de entrada escalados y centrados.
- centering: método de centrado.
- scaling: método de escala.
- SVP: método SVP.

Utilizando la salida de GGEmodel(), la función GGEPlot() crea numerosas vistas de biplots de GGE que permiten dar respuesta a distintos objetivos de los fitomejoradores. En estos gráficos los cultivares se muestran en minúsculas y los ambientes en mayúsculas. El método de centrado, escalado y SVD se muestran en una nota al pie junto con el porcentaje de G + IGA explicado por los dos ejes al agregar el argumento *footnote* = *T* y un título con *titles* = *T*.

Comparaciones simples utilizando GGE biplot

El biplot básico se obtiene con el parámetro *type* = *"Biplot"* (Figura 4.2). En este ejemplo, el 78 % de la variabilidad de G + GE se explica por los dos primeros términos multiplicativos. Los ángulos entre los marcadores de genotipos y entre los vectores ambientales son utilizados para interpretar el gráfico. Así, por ejemplo, Kat tiene un rendimiento por debajo de la media en todos los ambientes debido a su ángulo superior a 90° con todos ellos. Por otro lado, Fun presenta un rendimiento superior a la media en todas las localidades excepto OA93 y KE93, como lo indican los ángulos agudos. La longitud de los vectores ambientales es una medida de la capacidad del ambiente para discriminar entre cultivos.

GGEPlot(GGE1, type = "Biplot", footnote = F, titles = F)

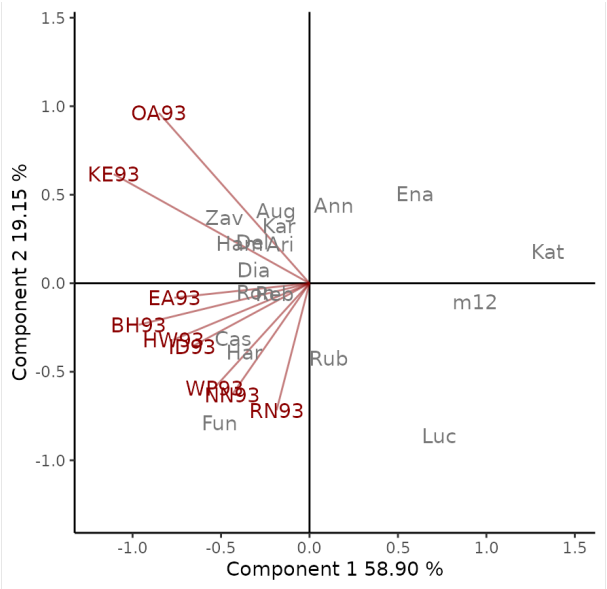

Figura 4.2: Biplot GGE basado en datos de rendimiento de trigo de invierno obtenido de Ontario en 1993. El método de escala utilizado es la partición simétrica de valores singulares (opción por defecto). El 78 % de la variabilidad de G + GE se explica por los dos primeros términos multiplicativos. Los cultivares se muestran en minúsculas y los entornos en mayúsculas.

Los mejoradores quieren identificar los cultivares más adaptados a su área, es decir a un ambiente particular, por ejemplo OA93. Para esto, Cornelius et al. (2002) sugieren constituir un eje del ambiente de interés (OA93), trazando una recta que una el identificador del ambiente y el origen de coordenadas, que puede denominarse eje OA93. Los genotipos se clasifican en función del rendimiento en dicho ambiente de acuerdo con sus proyecciones, en la dirección indicada por el eje OA93 (Figura 4.3). Para ello, se indica la opción *Selected Environment* en el argumento *type* de la función y además el ambiente a evaluar en el argumento *selectedE*. Se observa en este caso que el cultivar de mayor rendimiento fue es Zav seguido por Aug, Ham, y así sucesivamente hasta llegar al genotipo Luc, que es el de menor rendimiento en ese ambiente. El eje perpendicular al del ambiente de interés, separa los genotipos con rendimiento mayor al promedio, de Zav a Cas, de aquellos con valores inferior a la media, de Ema a Luc, en OA93.

En forma similar, el ambiente más adecuado para un cultivar es posible determinarlo graficando una línea que una el origen de coordenadas y el marcador del genotipo de interés, por ejemplo Kat, como se muestra en la figura 4.3 (Cornelius et al., 2002). Los ambientes se clasifican a lo largo del eje del genotipo en la dirección indicada por la flecha.

Para obtener este gráfico la opción *Selected Genotype* debe indicarse en el argumento *type*, y el genotipo de interés en *selectedG*. El eje perpendicular al del genotipo separa los ambientes en los que Kat presentó un rendimiento por debajo de su promedio, en todos los ambientes estudiados. En este ejemplo, Kat presentó un desempeño por debajo de la media en todos los entornos estudiados.

```
# Ranking de cultivares en el ambiente OA93
GGEPlot(GGE1, type = "Selected Environment", selectedE = "OA93", footnote = F, titles
    = F)

# Ranking de ambientes para cultivar Kat
GGEPlot(GGE1, type = "Selected Genotype", selectedG = "Kat", footnote = F, titles = F)
```

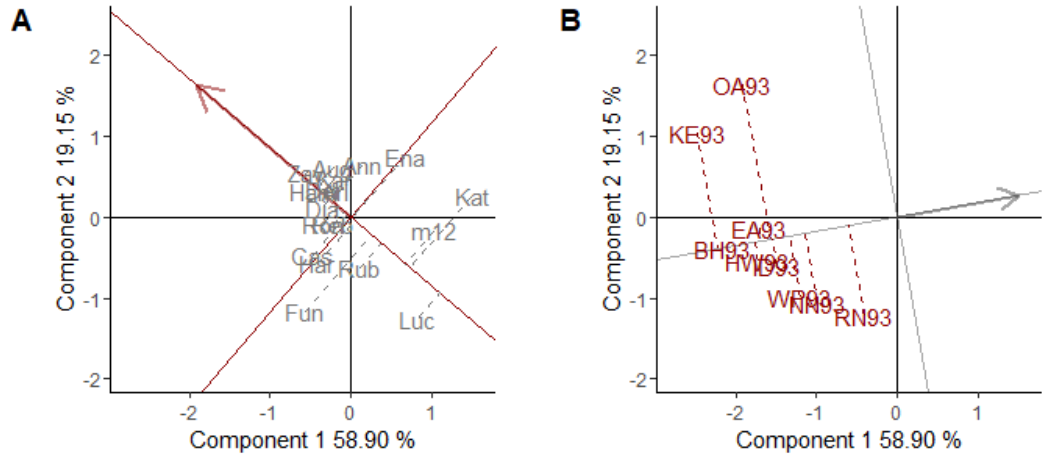

Figura 4.3: A: Ranking de cultivares en el ambiente OA93. B: Ranking de ambientes para cultivar Kat, basado en datos de rendimiento de trigo de invierno obtenido de Ontario en 1993. El método de escala utilizado es la partición simétrica de valores singulares (opción por defecto). El 78 % de la variabilidad de G + GE se explica por los dos primeros términos multiplicativos. Los cultivares se muestran en minúsculas y los entornos en mayúsculas.

También es posible comparar dos cultivares, es decir, Kat y Cas, vinculándolos con una línea y una perpendicular a la anterior (figura 4.4). Este biplot te obtiene colocando *Comparison of Genotype* en el argumento *type* y los genotipos a comparar en *selectedG1* y *selectedG2*. Se observa que Cas fue más rendidor que Kat en todos los ambientes, ya que todos están en el mismo lado de la línea perpendicular que Cas.

GGEPlot(GGE1, type = "Comparison of Genotype", selectedG1 = "Kat", selectedG2 = "Cas", footnote = F, titles = F)

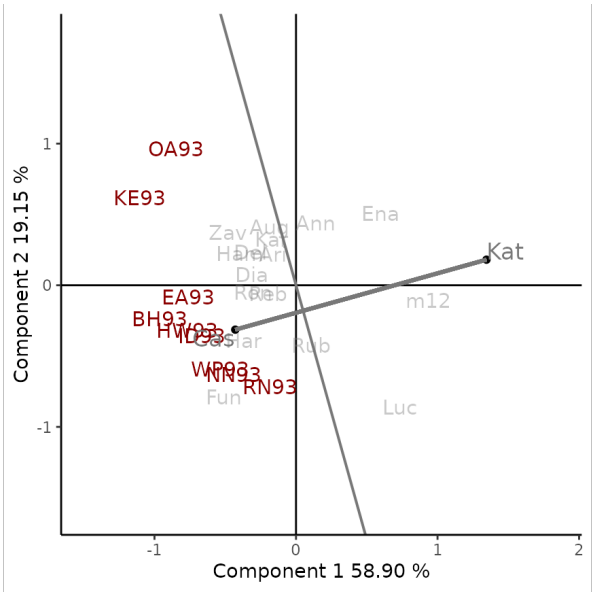

Figura 4.4: comparación de los cultivares Kat y Cas. El método de escala utilizado es la partición simétrica de valores singulares (por defecto). El 78 % de la variabilidad de G + GE se explica por los dos primeros términos multiplicativos. Los cultivares se muestran en minúsculas y los entornos en mayúsculas.

Identificación de mega-ambientes con GGE biplot

La vista poligonal del biplot GGE, obtenida al indicar *Which Won Where/What* en el argumento *type*, proporciona un medio eficaz de visualización del patrón "quíen ganó dónde"de un conjunto de datos EMA (Figura 4.5). El polígono se obtiene uniendo los cultivares (fun, zav, ena, kat y luc) que se encuentran más alejados del origen de coordenadas, de modo que todos los restantes se encuentren contenidos en el polígono. La distancia de los cultivares respecto del origen de coordenadas, en sus respectivas direcciones, es una medida de la capacidad de respuesta a los ambientes. Los ubicados en los vértices son los más alejados, por lo tanto son los cultivares que más responden, mientras que los que se encuentran en el origen de coordenadas no responden en absoluto a los ambientes estudiados.

Las perpendiculares a los lados del polígono dividen al biplot en mega-ambientes, siendo el cultivar de mayor rendimiento en todos los ambientes que se encuentran en él aquel que se encuentra en el vértice de dicho sector. Por un lado, se observa que OA93 y KE93 conforman un mega-ambiente y que Zav es el mejor cultivar. Otro está formado por el resto de los ambientes, al cual llamaremos ME1 en futuros análisis, siendo Fun el que

se encuentra en el vértice. En el sector con ena, kat y luc en los vértices del polígono no se observó ningún ambiente, lo cual indica que estos cultivares fueron los menos rendidores en algunos o todos los ambientes considerados.

GGEPlot(GGE1, type = "Which Won Where/What", footnote = F, titles = F)

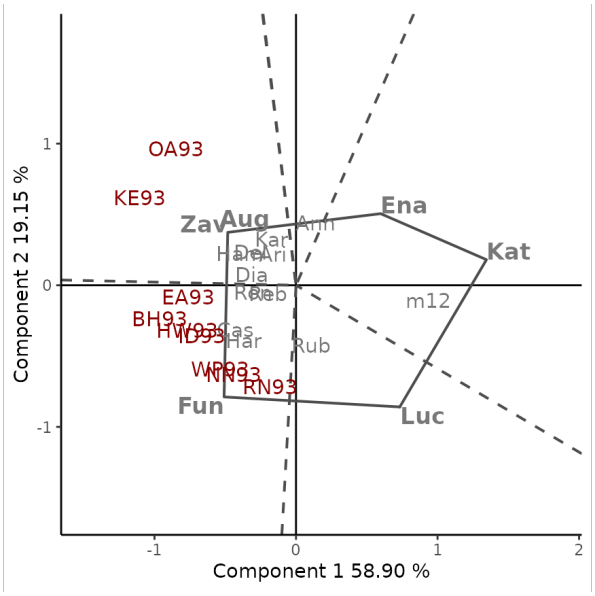

Figura 4.5: Vista poligonal del biplot GGE, que muestra qué cultivares presentaron mayor rendimiento en cada ambiente/mega-ambiente. El método de escala utilizado es la partición simétrica de valores singulares (por defecto). El 78 % de la variabilidad de G + GE se explica por los dos primeros términos multiplicativos. Los cultivares se muestran en minúsculas y los entornos en mayúsculas.

Evaluación de los cultivos dentro de un mega-ambientes con GGE biplot

Una vez identificado los mega-ambientes, el siguiente paso es seleccionar cultivares dentro de cada uno de ellos. De acuerdo con la figura 4.5, zav es el mejor cultivar para los ambientes en uno de los mega-ambiente y fun para el otro. Sin embargo, los fitomejoradores no seleccionarán un único cultivar en cada mega-ambiente, sino que es necesario evaluar todos los cultivares con el fin de conocer su desempeño (rendimiento y estabilidad). Esto se debe a que la unidad de ambos ejes para los genotipos es la unidad original de los datos.

El biplot GGE, particularmente enfocando la SVD en los genotipos, es decir utilizando el argumento *SVP="row"* en la función GGEmodel(), proporciona un medio superior para visualizar tanto el rendimiento medio como la estabilidad de los genotipos (Figura 4.6). Además, dado que los ambientes no son de interés cuando se evaluan los cultivares dentro

de un mega-ambiente, se indica con el argumento *sizeEnv = 0* de la función GGEPlot() que no los muestre en el gráfico.

La visualización del rendimiento medio y la estabilidad de los genotipos se logra dibujando una coordenada ambiental promedio (AEC). Por ejemplo, la Figura 4.6 muestra el AEC para el megaambiente ME1 compuesto por los entornos BH93, EA93, HW93, ID93, NN93, RN93, WP93. Mientras que la abscisa representa el efecto de G la ordenada el de la IGA, que es una medida de la variabilidad o inestabilidad, asociada con cada genotipo. Una mayor proyección sobre la ordenada AEC, independientemente de la dirección, significa mayor inestabilidad. Fun fue claramente el cultivar de mayor rendimiento, en promedio, en este megaambiente, seguido por Cas y Har, y Kat fue el más pobre. Mientras que Rub y Dia son más variables y menos estables que otros cultivares, por el contrario, Cas, Zav, Reb, Del, Ari y Kar, fueron más estables.

La Figura 4.6 compara los cultivares con el "ideal" que es el más rendidor y con estabilidad absoluta. Este cultivar ideal se usa como referencia, ya que rara vez existe. La distancia entre cultivares y el ideal se puede utilizar como medida de conveniencia. Los círculos concéntricos ayudan a visualizar estas distancias. En el ejemplo, para el ME1, Fun es el más cercano al cultivo ideal, y por tanto el más deseable, seguido de Cas y Hay, que a su vez son seguidos por Rum, Ham, Rub, Zav, Del y Reb, etc.

FALTAEL SIGNO $ EN LA PRIMER LINEA DEL CODIGO QUE SIGUE...PERO ME TIRA ERROR O PONERLO CON FILTER???

```
ME1 <- yan.winterwheat[yan.winterwheat env %in% c("BH93", "EA93","HW93", "ID93"
    ,"NN93", "RN93", "WP93"), ]

# Modelo SREG enfocando SVD en los genotipos
GGE_Gpartition <- GGEmodel(ME1, genotype = "gen", environment = "env", response =
    "yield", SVP = "row")

# Visualizacion del rendimiento medio y la estabilidad
GGEPlot(GGE_Gpartition, type = "Mean vs. Stability", footnote = F, titles = F, sizeEnv =
    0)

# Ranking de los genotipos respecto a uno ideal
GGEPlot(GGE_Gpartition, type = "Ranking Genotypes", footnote = F, titles = F, sizeEnv
    = 0)
```

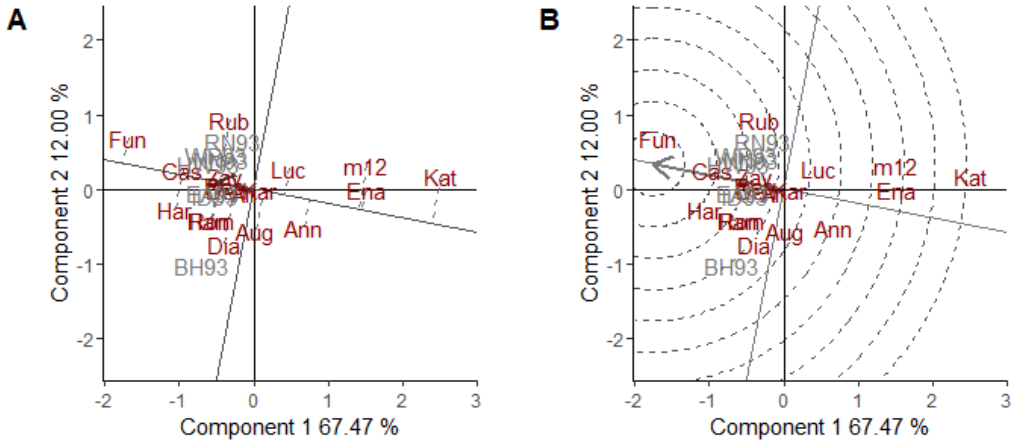

Figura 4.6: A: Evaluación de los cultivares con base en el rendimiento promedio y la estabilidad y B: Clasificación de genotipos con respecto al genotipo ideal, basado en el escalado centrado en los genotipos.

Evaluación de los ambientes con GGE biplot

A pesar de que el objetivo principal de los EMA es seleccionar cultivares también es posible evaluar los ambientes, enfocando la SVD en los ambientes al ajustar el modelo SREG (*SVP = "column"* en la función GGEmodel()). Esto incluye varios aspectos: (i) evaluar si la región objetivo pertenece a uno o más megaambientes; (ii) identificar mejores entornos de prueba; (iii) detectar ambientes redundantes que no brindan información adicional sobre cultivares; y (iv) determinar los ambientes que se pueden utilizar para la selección indirecta.

En la figura 4.7 los ambientes están conectados con el origen de coordenadas a través de vectores, permitiendo comprender las interrelaciones entre ellos. Esta visualización del biplot GGE se obtiene indicando *Relationship Among Environments* (Figura 4.7). El coeficiente de correlación entre dos ambientes es aproximadamente el coseno del ángulo entre sus vectores. En este ejemplo se considera la relación entre los ambientes de ME1. El ángulo entre los vectores para los entornos NN93 y WP93 es de aproximadamente 10° entre sus vectores; por lo tanto, están estrechamente relacionados; mientras que RN93 y OA93 presentan una correlación negativa débil ya que el ángulo es levemente mayor a 90°. El coseno de los ángulos no se traduce precisamente en coeficientes de correlación, ya que el biplot no explica toda la variabilidad en el conjunto de datos. Sin embargo, son lo suficientemente informativos como para comprender la interrelación entre los entornos de prueba.

Si algunos de los ambientes tienen ángulos pequeños y, por lo tanto, están altamente correlacionados, la información sobre los genotipos obtenidos de estos ambientes debe ser similar. Si esta similitud es repetible a través de los años, estos ambientes son redundantes y por lo tanto, uno solo debería ser suficiente. Obtener la misma o mejor información utilizando menos ambientes reducirá el costo y aumentará la eficiencia de producción.

La capacidad de discriminación así como la representatividad respecto del ambiente objetivo, son medidas fundamentales para un ambiente. Si no tiene capacidad de discriminación, no proporciona información sobre los cultivares y, por lo tanto, carece de utilidad. A su vez, si no es representativo no sólo que carece de utilidad sino que también puede proporcionar información sesgada sobre los cultivares evaluados. Para visualizar estas medidas, se define un ambiente promedio (AEC mencionado anteriormente) y el ambiente ideal como el centro de un conjunto de círculos concéntricos (Figura 4.7). Para obtener este biplot se debe indicar *Ranking Environments* en el argumento *type* de GGEPlot() (Figura 4.7). El ángulo entre el vector de un ambiente y el eje proporciona una medida de la representatividad. Por lo tanto, EA93 e ID93 son los más representativos, mientras que RN93 y BH93 son los menos representativos del ambiente promedio, cuando se analiza ME1 4.7. Por otro lado, para ser discriminativo debe estar cercano al ambiente ideal. HW93 es el ambiente más cercano al ideal y, por lo tanto, es el más deseable del ME1, seguido por EA93 e ID93. Por el contrario, RN93 y BH93 fueron los ambientes de prueba menos deseables de ME1.

```
# Modelo SREG enfocando SVD en los ambientes
GGE_Epartition <- GGEmodel(ME1, genotype="gen", environment="env", response="
    yield", SVP="column")

# Relacion entre ambientes
GGEPlot(GGE_Epartition, type = "Relationship Among Environments", footnote = F,
    titles = F)

# Clasificacion de ambientes con respecto al ambiente ideal
GGEPlot(GGE_Epartition, type = "Ranking Environments", footnote = F, titles = F)
```

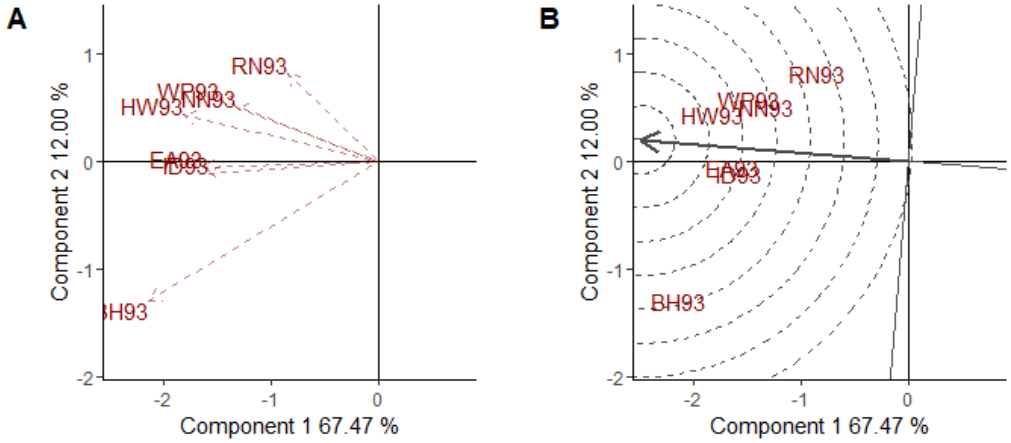

Figura 4.7: A: Relación entre ambientes y B: Clasificación de ambientes con respecto al ambiente ideal, basado en el escalado centrado en los genotipos.

4.1.4. Métodos de imputación

Una limitación importante de los modelos presentados anteriormente es que requieren una que el conjunto de datos este completo, es decir que todos los genotipos sean evaluados en todos los ambientes. Por lo tanto, en el paquete se incluyen una serie de metodologías recientemente publicadas, algunas de las cuales no se encuentran disponible en R, para superar el problema de las observaciones perdidas. Entre los métodos incluidos se encuentran: "EM-AMMI", "EM-SVD", "Gabriel","WGabrielz "EM-PCA", los cuales se indican en la opción *type* de la función imputation(). El formato requerido para el conjunto de datos de entrada es análogo al indicado en las otras funciones incluidas en el paquete.

Para presentar un ejemplo, se eliminan algunas observaciones del yan.winterwheat completo:

```
# Generando datos faltantes
yan.winterwheat [1,3] <- NA
yan.winterwheat [3,3] <- NA
yan.winterwheat [2,3] <- NA
```

La imputación de valores perdidos con el método "EM-AMMI"se puede realizar de la siguiente manera:

```
imputation(yanwinterwheat, PC.nb = 2, genotype = "gen", environment = "env", response
    = "yield", type = "EM-AMMI")
```

El resultado es la matriz con datos imputados en aquellas celdas vacías.

4.2. Geneticae Shiny Web App

El objetivo de Geneticae Shiny Web APP es proporcionar una interfaz gráfica de usuario para el paquete geneticae de R descripto anteriormente, de modo que pueda ser utilizado por fitomejoradores y analistas sin experiencia previa en programación R.

Es un software interactivo, no comercial y de código abierto, que ofrece una alternativa gratuita al software comercial disponible para analizar MET. Se encuentra disponible en un servidor gratiuto `https://geneticae.shinyapps.io/geneticae-shiny-web-app/` el cual tiene límite en el tiempo de uso, sin embargo, la APP será ubicada en el servidor de CONICET. Además, se puede acceder a la misma desde la página web del instituto CEFOBI de CONICET `https://www.cefobi-conicet.gov.ar/bases-de-datos-y-programas/`.

4.2.1. Preparación de un archivo de datos para la aplicación Geneticae

La APP requiere datos en formato .csv, delimitados por comas o punto y coma, y permite que los nombres de las columnas se encuentren en la primera fila del archivo (*heading*). Se requieren datos en formato largo, es decir, cada fila corresponde a una observación y cada columna a una variable (genotipo, ambiente, repetición (si existe) y fenotipo observado). Si cada genotipo ha sido evaluado más de una vez en cada ambiente, la media fenotípica requerida por el modelo SREG y AMMI para cada combinación de genotipo y ambiente se calcula internamente y luego se estima el modelo. Las variables adicionales que no se utilizarán en el análisis pueden estar presentes en el conjunto de datos. No se permiten valores perdidos.

Dos conjuntos de datos están disponibles en la APP:

- *plrv* (de Mendiburu, 2020): estudio sobre la resistencia al PLRV (Patato Leaf Roll Virus) que provoca el enrollamiento de las hojas. Se experimentaron 28 genotipos en 6 localidades del Perú. Cada clon se evaluó tres veces en cada ambiente y se registró el rendimiento, el peso de la planta y la parcela.
- *yanwinterwheat* (Wright, 2020): rendimiento de 18 variedades de trigo de invierno cultivadas en nueve ambientes en Ontario en 1993. Aunque en el experimento se realizaron cuatro bloques o réplicas en cada ambiente, solo se dispuso de la media del rendimiento para cada combinación de variedad y ambiente. Este conjunto de datos se utilizará para ilustrar la metodología incluida en Geneticae APP para analizar datos MET.

Ellos se están disponibles en la pestaña *Data − > Example datasets* y se pueden descargar en formato .csv (Figura). El conjunto de datos *yanwinterwheat* no tiene repeticiones, mientras que *plrv* sí.

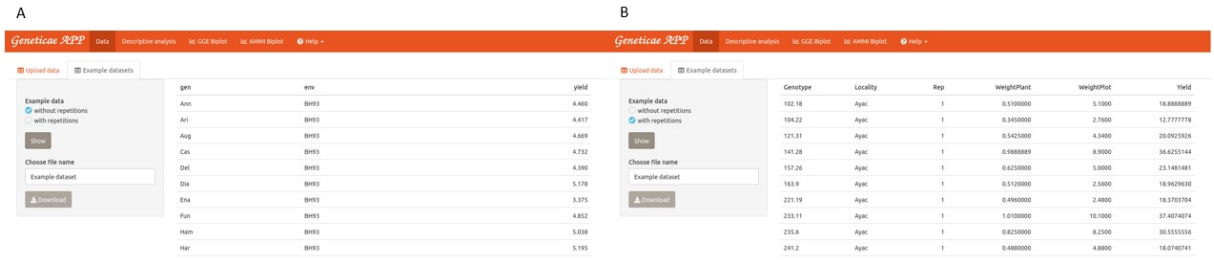

Figura 4.8: (A) Plrv dataset (B) yanwinterwheat dataset

Cargando un conjunto de datos en la APP

El conjunto de datos que se analizará debe cargarse en la pestaña *Data − > Upload data*. Por ejemplo, para importar el conjunto de datos *yanwinterwheat*, se debe cargar el archivo .csv, lo que indica que está delimitado por comas, que la primera fila contiene los nombres de cada variable (*heading*) y los nombres de las columnas que continen la información del genotipo, ambiente y rasgo fenotípico (gen, env y rendimiento en este caso). Si hay repeticiones disponibles, se debe especificar el nombre de la columna con dicha información; de lo contrario, no indique nada.

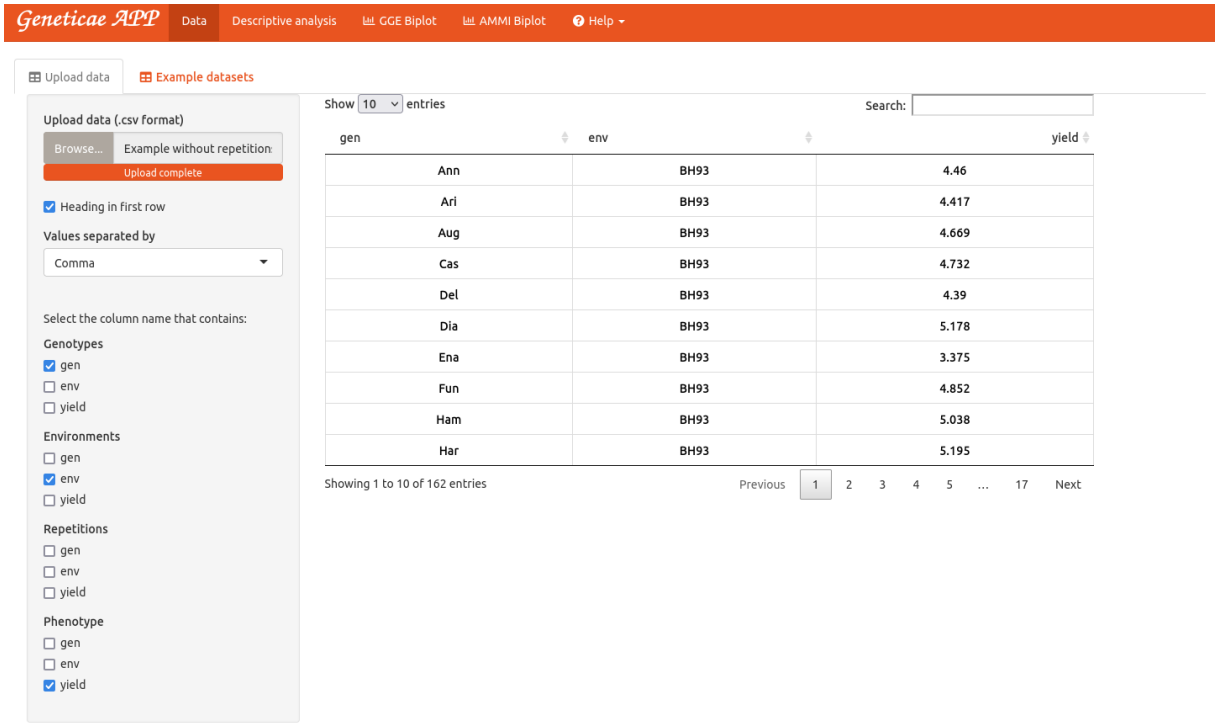

Figura 4.9: Cargando el conjunto de datos *yanwinterwheat* en Geneticae APP

4.2.2. Análisis descriptivo

Cualquier estudio debe comenzar con un análisis descriptivo del conjunto de datos. La pestaña *Descriptive Analysis* proporciona algunas herramientas para esto. Se pueden obtener diagramas de caja, diagramas y matriz de correlación así como también gráficos de interacción.

Una gráfica de caja que compara el rasgo cuantitativo entre ambientes o genotipos (Figura 4.10) puede ser de interés. Las medidas de resumen utilizadas para su construcción se muestran de forma interactiva moviendo el mouse dentro del panel de la figura. Además, se puede descargar como un archivo interactivo (.html) así como un archivo .png, haciendo clic en el botón Descargar o en la cámara que aparece en la figura, respectivamente (Figura 4.10). El usuario puede personalizar algunos aspectos del gráfico, como el color del cuadro y los nombres de los ejes.

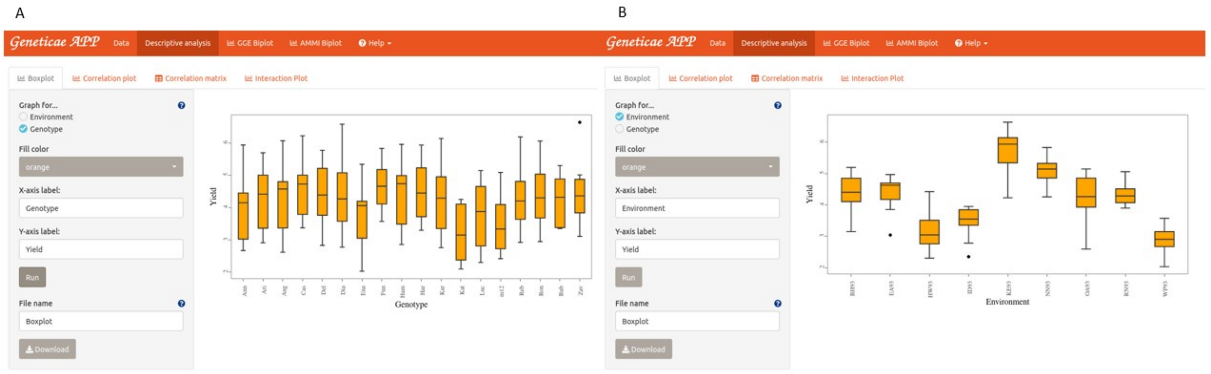

Figura 4.10: Diagrama de caja de (A) genotipos y (B) ambientes para el conjunto de datos *yanwinterwheat*.

Los coeficientes de correlación de Pearson o Spearman entre genotipos se pueden mostrar como un gráfico o una matriz (Figura 4.11). Las correlaciones positivas se muestran en azul y las negativas en rojo, mientras que la intensidad del color y el tamaño del círculo son proporcionales a los coeficientes de correlación. La gráfica de correlación se puede descargar en formato .png. Se observan altas correlaciones entre el rendimiento de los genotipos estudiados.

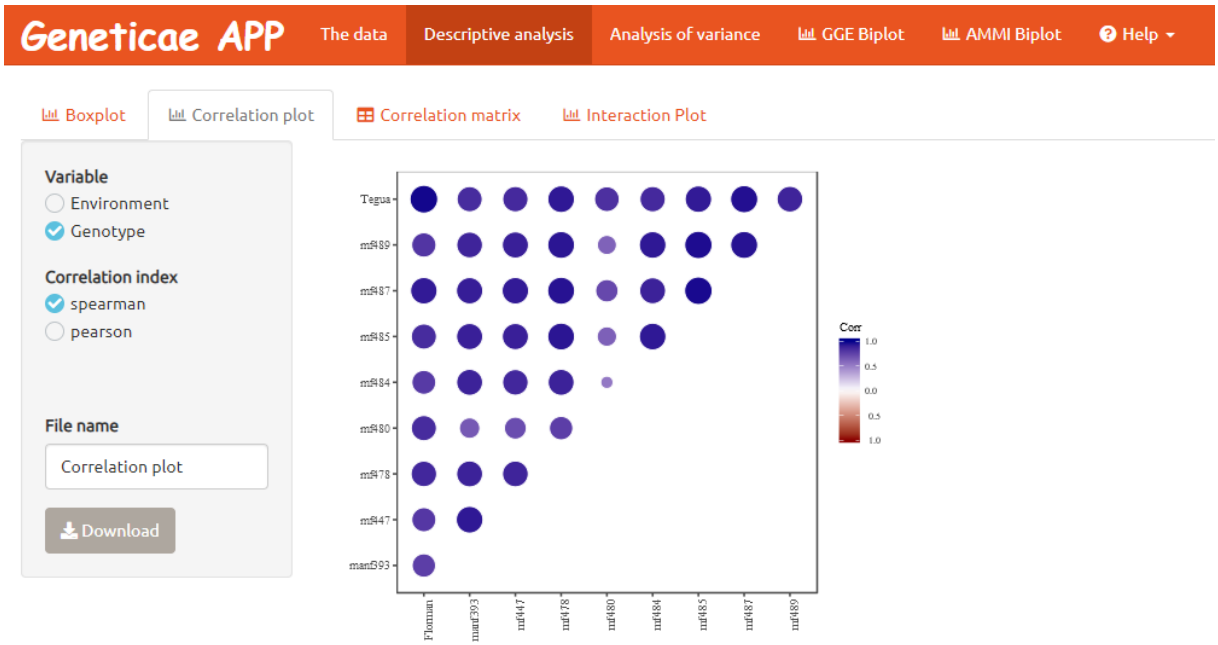

Figura 4.11: Gráfico de correlación (A) y matriz (B) entre genotipos yanwinterwheat dataset

Dado que IGA genera respuestas genotípicas diferenciales en diferentes ambientes, lo que complica selección de los mejores cultivares, una gráfico de interacción puede ser de interés. El cambio en el efecto genotípico a través de los ambientes se muestra en la Figura 4.12, mientras que el cambio en el efecto ambiental a través de los genotipos en la Figura 4.12. Del mismo modo que el diagrama de caja es un gráfico interactivo, por lo que es posible descargarla en formatos .HTML o .png con el botón Descargar o haciendo clic en la cámara, respectivamente. Además, el usuario puede personalizar los nombres de los ejes. En este ejemplo se pueden ver inconsistencias en el desempeño de genotipos en diferentes ambientes.

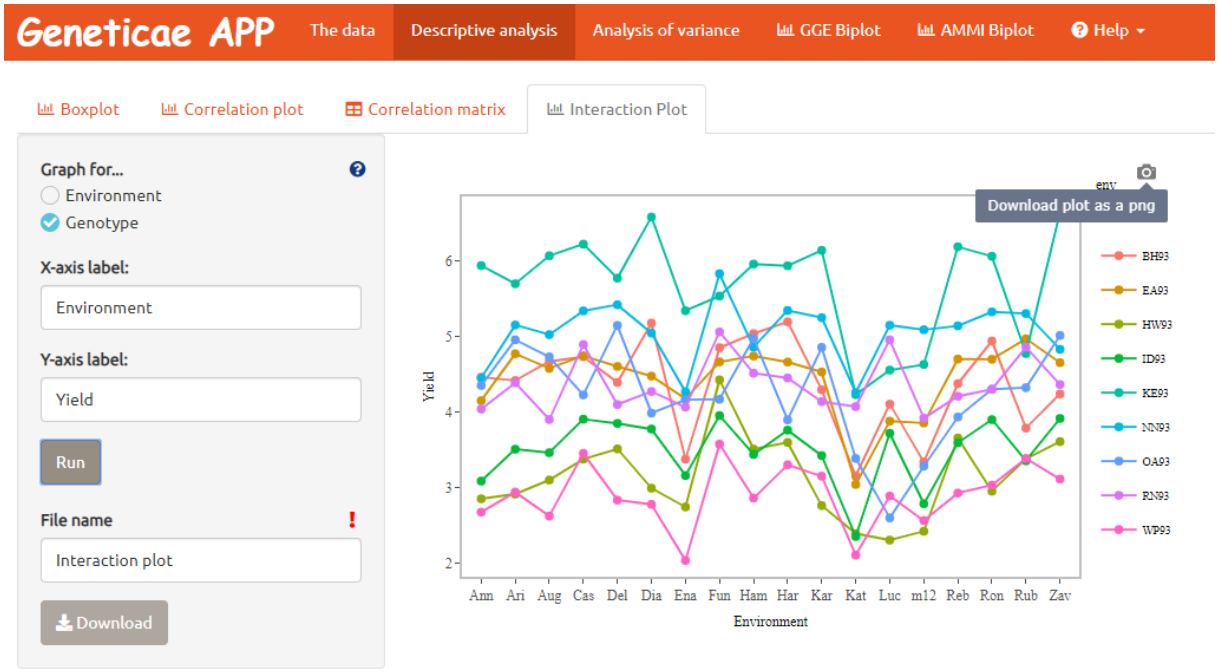

Figura 4.12: Gráfico de interacción para (A) ambientes a través de genotipos y (B) genotipos a través de entornos del conjunto de datos de *yanwinterwheat*.

4.2.3. Modelo de regresión por sitio

Geneticae Shiny Web App permite realizar todos los biplots GGE mostrados en el paquete geneticae.Ciertos atributos estilísticos de dichos gráficos se pueden personalizar y además pueden ser descargados.

La pestaña *GGE Biplot* genera las vistas de biplots GGE presentadas anteriormente, en las que los cultivares se muestran en minúsculas y los entornos en mayúsculas. Dado que el modelo requiere una única observación para cada combinación de genotipo y ambiente,

si hay repeticiones, el valor promedio fenotípico se calcula automáticamente antes de ajustar el modelo. No se permiten valores perdidos.

Se debe seleccionar el método SVD, sin embargo, esta elección no altera las relaciones o interacciones relativas entre genotipos y ambientes, aunque la apariencia del biplot será diferente (Yan, 2002). La opción simétrica permite la comparación tanto de genotipos como de ambientes (opción por *default*); *Genotype-Focused* muestra la interrelación entre genotipos con mayor precisión que cualquier otro método, y *Environment-Focused* es el que más informa sobre las interrelaciones entre ambientes. Una nota a pie de página que indica que el método de centrado, que será *tester-center* para obtener el bioplot GGE, que no se aplica ninguna escala a los datos, el método SVD seleccionado por el usuario y el porcentaje de variación G + GEI explicado por los dos ejes puede ser agregado. A su vez, el título del gráfico, los ejes y los nombres de los ejes se pueden configurar para que aparezcan o no. Por último, el usuario puede personalizar el color y el tamaño del marcador de genotipo y entornos.

El biplot básico, aquel que muestra los cultivares más adecuados para un ambiente particular (OA93), los ambientes más adecuados para un genotipo (Kat), la comparación de dos genotipos (Cas y Kat) y la vista del poligono se pueden obtener como se indica en la figura 4.13, donde el escalado es el simétrico (*SVP type − > symmetrical*) y las opciones de *plot type* son *Biplot, Selected Environment, Selected Genotype, Comparison of Genotype* y *Which Won Where/What*, respectivamente. Al indicar *Selected Environment* el ambiente de interés se debe especificar, de igual modo cuando se utiliza *Selected Genotype* y *Comparison of Genotype* se debe señalar cuál es el genotipo a analizar.

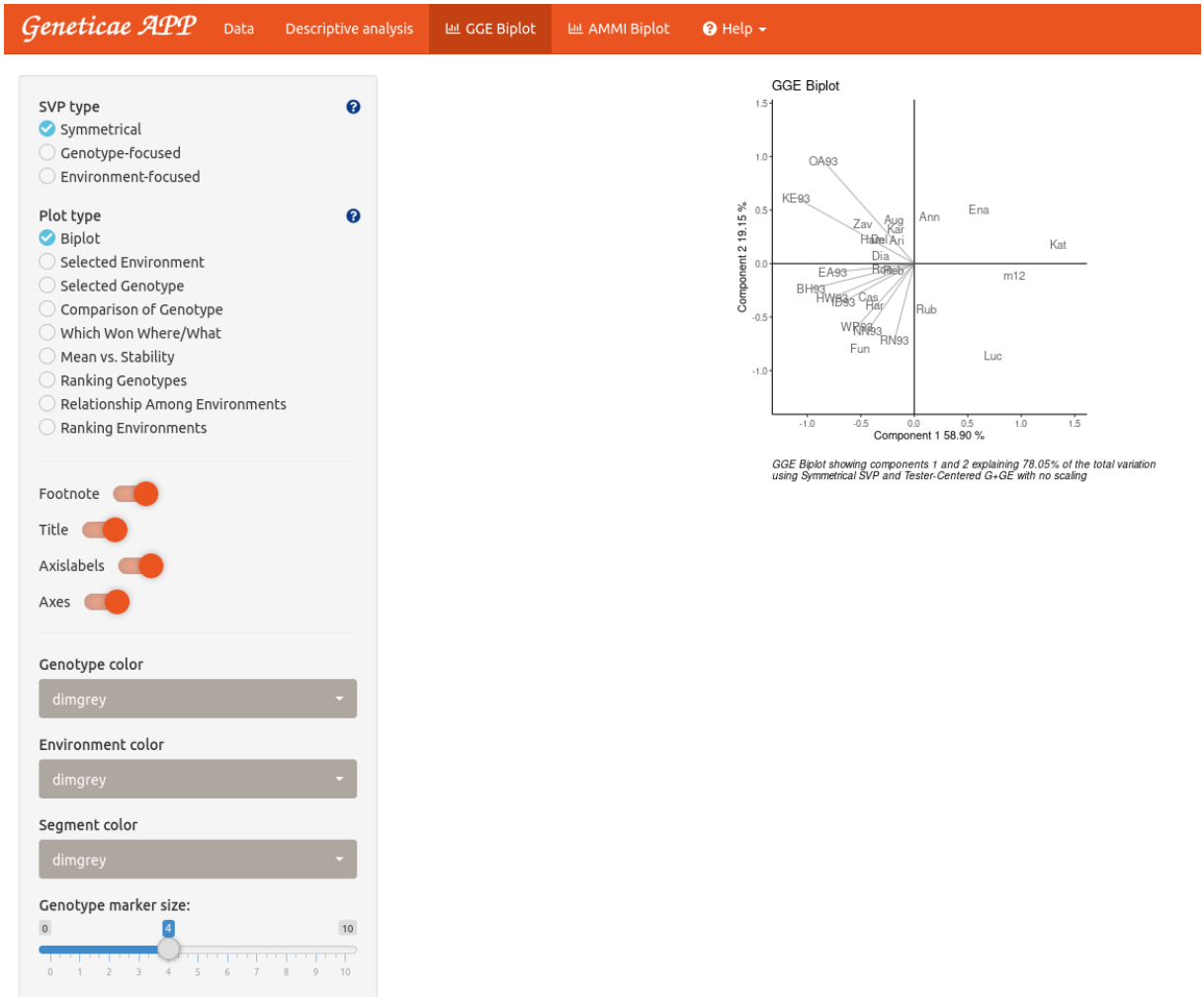

Figura 4.13: Boxplot de genotipos a través de los ambientes para el conjunto de datos Plrv

La selección de cultivares dentro de cada megaambiente se realiza con el escalado centrado en los genotypos (*SVP type − > genotype-focused*), y los tipos de gráficos *Mean vs. Stability* que permite la visualización de la media y estabilidad de genotipos y *Ranking Genotypes* que compara las cultivares con el "ideal". Dado que estos análisis son propios de cada megaambiente, al indicar alguno de esos tipos de gráficos se tendrá que señalar cuales son los ambientes que forman el megaambiente de interés.

Por último para el análisis de los ambientes de cada megaambiente se utiliza el escalado centrado en los ambientes (*SVP type − > environment-focused*). Para comprender las interrelaciones entre ellos el tipo de gráfico *Relationship Among Environments* se debe seleccionar y para visualizar la dapacidad de discriminación y representatividad *Ranking Environments*. Dado que estos análisis son propios de cada megaambiente, al indicar alguno de esos tipos de gráficos se tendrá que señalar cuales son los ambientes que forman

el megaambiente de interés.

4.2.4. modelo AMMI

La pestaña *AMMI Biplot* crea el biplot GE, en el que los cultivares se muestran en minúsculas y los entornos en mayúsculas. Dado que las alternativas clásica y robusta requieren una única observación para cada combinación de genotipo y ambiente, si hay repeticiones, el valor promedio fenotípico se calcula automáticamente antes de ajustar el modelo. No se permiten valores perdidos. Al igual que en el biplot de GGE, una nota a pie de página que indica que el porcentaje de variación de IGA explicado por los dos ejes, el gráfico de título, los ejes y los nombres de los ejes se pueden configurar para que aparezcan o no. Además, el color y tamaño del marcador de genotipos y ambientes pueden ser personalizados por el usuario. Los biplots pueden ser descargados.

Por ejemplo, para obtener el biplot GE derivado del modelo AMMI clásico se debe indicar AMMI en *plot type* (Figura ...)., en caso de contar con *outliers* alguna de las alternativas robustas (rAMMI, hAMMI, gAMMI, lAMMI o ppAMMI) se debe especificar.

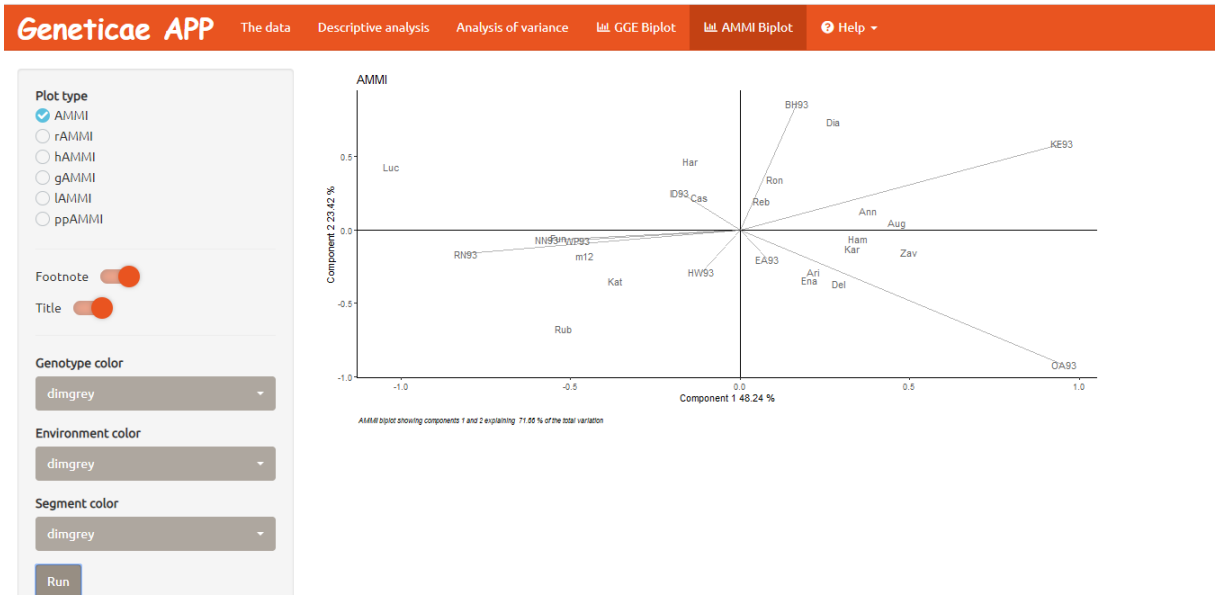

Figura 4.14: AMMI

4.2.5. Ayuda

En la pestaña *Help* se presenta información general, un tutorial y un video sobre cómo utilizar la APP.

Capítulo 5

Conclusiones

Bibliografía

S. Arciniegas-Alarcón, M. García-Peña, C.T.S. Dias, y W.J. Krzanowski. An alternative-methodology for imputing missing data in trials with genotype-by-environment interaction. *Biometrical Letters*, 47:1–47, 2010.

S. Arciniegas-Alarcón, M. García-Peña, W.J. Krzanowski, y C.T.S. Dias. An alternative-methodology for imputing missing data in trials with genotype-byenvironment interaction: somenew aspects. *Biometrical Letters*, 51:75–88, 2014.

R.E. Cooper, M.and Stucker, I.H. DeLacy, y B.D. Harch. Wheat breeding nurseries, target environments, and indirect selection for grain yield. *Crop Science*, 37:1168–1176, 1997.

P.L. Cornelius, J. Crossa, y M.S. Seyedsadr. *Genotype by Environment Interaction*, cap. Statistical test and estimators of multiplicative models for genotype-by-environment interaccition., págs. 199–234. CRC Press, Boca Raton, 1996.

P.L. Cornelius, J. Crossa, y M.S. Seyedsadr. *Quantitive Genetics, Genomics and Plant Breeding.*, cap. Biplot analysis of Multi-environment trial data., págs. 199–234. CABI Publishing, Wallingford, 2002.

J. Crossa y P. L. Cornelius. Sites regresion and shifted multiplicative model clustering of cultivar trial sites under heterogeneity of error variances. *Crop Science*, 37:406–415, 1997.

J. Crossa, P.L. Cornelius, y W. Yan. Biplots of linear-bilinear models for studying crossover genotipe–environment interaction. *Crop Science*, 42:619–633, 2002.

J. Crossa, H.G. Gauch, y R.W. Zobel. Additive main effects and multiplicative interaction analysis of two international maize cultivar trials. *Crop Science*, 30:493–500, 1990.

R. Cruz Medina. Some exact conditional tests for the multiplicative models to explain genotype-environment interaction. *Heredity*, 69:128—-132, 1992.

S. Dumble. *GGEBiplots: GGE Biplots with 'ggplot2'*, 2017. URL `https://CRAN.R-project.org/package=GGEBiplots`. R package version 0.1.1.

H. G. Gauch. Model selection and validation for yield trials. *Theoretical and Applied Genetics*, 80:153–160, 1988.

H.G. Gauch y R.W. Zobel. Identifying mega-environments and targeting genotypes. *Crop Science*, 37:311—-326, 1997.

Jr. R.R. Hill y J.L. Rosenberg. Models for combining data from germplasm evaluation trials. *Crop Science*, 25:467–470, 1985.

M.S. Kang y R. Magari. *Genotype by Environment Interaction*, cap. New Developments in Selecting for Phenotypic Stability in Crop Breeding., págs. 201–213. Elsevier, New York, 1996.

R. A. Kempton. The use of biplot in interpreting variety by environment interactions. *Journal of Agricultural Science*, 122:335–342, 1984.

P.C. Rodrigues, A. Monteiro, , y V.M. Lourenço. A robust ammi model for the analysis of genotype-by-environment data. *Bioinformatics*, 32:58–66, 2016.

W. Yan, P.L. Cornelius, J. Crossa, y L.A. Hunt. Two types of gge biplots for analyzing multi-environment trial data. *Crop Science*, 41:656–663, 2001.

W. Yan y L. A. Hunt. Genetic and environment causes of genotype by environment interaction for winter wheat yield in ontario. *Crop Science*, 41:19–25, 2001.

W. Yan, L. A. Hunt, Q. Sheng, y Z. Szlavnics. Cultivar evaluation and mega-environment investigation based on the GGE biplot. *Crop Science*, 40:597—-605, 2000.

W. Yan y M. Kang. *GGE Biplot Analysis: A Graphical Tool for Breeders, Geneticists.* CRC Press, 2003.

W. Yan y I. Rajcan. Biplot evaluation of test sites and traitrelations of soybean in ontario. *Crop Science*, 42:11—-20, 2002.